DESARROLLO DE UN SISTEMA PARA LA DETECCIÓN DE MOVIMIENTOS SÍSMICOS USANDO REDES NEURONALES ARTIFICIALES

JULIÁN DARÍO MIRANDA CALLE
CIRO ALBERTO GAMBOA ENTRALGO

UNIVERSIDAD PONTIFICIA BOLIVARIANA
ESCUELA DE INGENIERÍA
FACULTAD DE INGENIERÍA DE SISTEMAS E INFORMÁTICA
FACULTAD DE INGENIERÍA ELECTRÓNICA
BUCARAMANGA
2018

DESARROLLO DE UN SISTEMA PARA LA DETECCIÓN DE MOVIMIENTOS SÍSMICOS USANDO REDES NEURONALES ARTIFICIALES

JULIÁN DARÍO MIRANDA CALLE
CIRO ALBERTO GAMBOA ENTRALGO

Trabajo de grado para optar al título de Ingeniero de Sistemas e Informática e Ingeniero Electrónico

Directores
ANGÉLICA FLÓREZ ABRIL, MSc.
MIGUEL ALFONSO ALTUVE PAREDES, PhD.

UNIVERSIDAD PONTIFICIA BOLIVARIANA
ESCUELA DE INGENIERÍA
FACULTAD DE INGENIERÍA DE SISTEMAS E INFORMÁTICA
FACULTAD DE INGENIERÍA ELECTRÓNICA
BUCARAMANGA
2018

Nota de aceptación:

_____
_____
_____
_____
_____
_____

_____
Firma del presidente del jurado

_____
Firma del jurado

_____
Firma del jurado

Bucaramanga, 12 de octubre de 2018

DEDICATORIA

A Dios por la fortaleza y perseverancia con la que me ha permitido caminar por el sendero de la vida, colmándome de bendiciones a cada instante. A mis padres por el apoyo incondicional, por el amor con el que se han dedicado a esta extraordinaria labor de ser padres. Dios me ha bendecido con su compañía y me ha dado la fortuna de tenerlos a mi lado en todo momento. A ustedes, todo el cariño y el triunfo que hasta hoy he conseguido y que alcanzaré a su lado. Los amo mucho mis queridos padres.

A mis abuelos, gracias por la afectuosa preocupación con la que se interesan por las trivialidades de mi día a día, que acaban siendo aventuras interminables en el camino hacia el éxito. Ese éxito que consigo en este maravilloso y efímero instante, se lo dedico a ustedes, mis amados abuelitos. Dios los bendiga y me permita tenerlos conmigo por siempre.

A Daniela, mi princesa, gracias por tu apoyo incondicional. Desde que llegaste a mi vida has llenado de felicidad mi mundo y te has convertido en una parte muy importante de mi hermosa familia. Por cada noche de espera paciente, cada mañana llena de ilusión y cada segundo de compañía y amor; por estar a mi lado en todo momento y apoyarme cuando no encontraba salida a las adversidades, este triunfo que hoy alcanzo te lo dedico a ti. Por este y muchos más triunfos a tu lado. Dios bendiga tu camino a donde vayas y me permita compartirlo contigo por muchos años, décadas y la eternidad.

A mi hermano, te comparto la felicidad de alcanzar esta anhelada meta y te agradezco por las muchas experiencias que hemos pasado juntos. De una u otra forma, esas experiencias han formado mi carácter y me han hecho ser una persona diferente.

A toda mi gran familia, que sin su apoyo y afecto sería una victoria con un sabor agridulce, pues su amor y compañía le dan la calidez que la hace sorprendente y aún más excepcional.

Para aquellos profesores que se han cansado de mis preguntas, pero que para cada una de ellas ha existido una respuesta; para aquellos que han tenido el respeto y la humildad de considerarme su amigo, y para quienes han sabido darse a entender con mucho profesionalismo, gracias. La entrega y la pasión son la base de la perseverancia y el éxito, cualidades que cada uno de ustedes ha sabido exhibir y enseñar en diferentes formas.

A todos ustedes, gracias desde lo más profundo de mi corazón.

Julián Darío Miranda Calle

DEDICATORIA

A la vida y al destino, que me han puesto en el camino a las personas y condiciones indicadas.

A mis padres que, con todo su amor y apoyo, me han impulsado a soñar en grande y a luchar por culminar lo que me propongo. A pesar de los altibajos, siempre han confiado en mis capacidades y se han esforzado en ayudarme a ser siempre mejor. A mis hermanos que, aunque están usualmente lejos, siempre mantienen un estado de atención y apoyo hacia mí. A mi familia, que me ha dado aliento y me ha tenido siempre en este sendero con paciencia, dedicación y cariño.

A los colegas del equipo Demain, por su continuo apoyo y motivación para proponer y desarrollar ideas geniales. Nuestro trabajo empieza a mostrar resultados importantes y tenemos muchos más retos emocionantes por asumir juntos. La visión compartida y sinergia que nos caracteriza siempre me ha impulsado a ir más allá.

A mis grandes amigos Kamilo Peña y Yandy Guerrero, con los que siempre hemos reído a carcajadas sin importar las inclemencias del camino. A los compañeros de las facultades de Ingeniería Eléctrica y Electrónica e Ingeniería de Sistemas e Informática, con quienes he compartido este arduo pero hermoso proceso.

A los docentes que influenciaron mi proceso de formación, desde el Colegio Centro de Orientación Infantil, el Colegio Balbino García, el Servicio Nacional de Aprendizaje y la Universidad Pontificia Bolivariana. Su disposición a enseñar y a formar personas íntegras ha marcado mi forma de caminar (y a veces de correr) por la vida.

Con la paciencia, el trabajo en equipo, la perseverancia y el amor que he encontrado en las personas que me rodean, puedo culminar con éxito esta aventura y ofrecerle algo útil y apasionante al mundo.

Gracias.

Ciro Alberto Gamboa Entralgo

AGRADECIMIENTOS

A la Universidad Pontificia Bolivariana por el apoyo durante el desarrollo de nuestro proyecto, facilitándonos espacios y materiales en todo momento.

A la MSc. Angélica Flórez Abril por el acompañamiento continuo durante el desarrollo del proyecto. Por incentivarnos a creer en lo imposible y apoyar nuestras ideas que, aunque algunas eran muy cuerdas y otras muy locas, todas fueron parte fundamental para alcanzar nuestros objetivos. Gracias por las horas extra dedicadas a ayudarnos a solucionar situaciones adversas, a creer plena y ciegamente en nosotros y a creer en nuestra visión que en ningún momento se desdibujó producto de su apoyo constante.

Al PhD. Miguel Alfonso Altuve Paredes por invertir parte de su tiempo en apoyar nuestros objetivos y alimentar nuestra sed de conocimiento con sus valiosos aportes teóricos y prácticos en el área de Machine Learning para el enriquecimiento de nuestro trabajo.

Al Ing. Nelson Fernando Monroy por sus aportes, consejos e ideas que permitieron mejorar nuestro trabajo. Por todos los segundos, minutos y hasta horas invertidas en explicaciones en temas que parecían vagos al principio, pero que con su esfuerzo fueron muy importantes para construir una base sólida de lo que con orgullo hemos alcanzado.

A la coordinadora de la Red Sismológica Nacional de Colombia MSc. Viviana Dionicio Lozano, a la Ing. Ruth Emilse Bolaños y al personal del área sismológica, analistas, electrónicos y de sistemas por su atenta disponibilidad para atender nuestras dudas.

Al ingeniero Edwar Zambrano por compartir sus conocimientos y experiencia en el contexto sismológico, escuchando nuestras ideas y aportando conocimientos que enriquecieron el crecimiento de nuestro trabajo.

A Álix Andrea Angarita Castillo por sus valiosos aportes y trabajo en la materialización del proyecto que empezó como una vaga idea de clase y hoy se convierte en una meta cumplida.

Al plantel docente de las facultades de Ingeniería de Sistemas e informática e Ingeniería Eléctrica y Electrónica por ser un pilar fundamental en nuestra formación académica desde el primer día que ingresamos a la Universidad, apoyándonos e incentivándonos a llevar a cabo nuestras ideas. Gracias por la tan amable colaboración, solidaridad y respeto con la que desinteresadamente fueron parte de nuestro proyecto de vida, y que hoy nos hacen ser mejores profesionales.

Al Centro de Tecnologías de Información y Comunicación (CTIC) y a la Dirección de Investigaciones y Transferencia (DIT) por la colaboración continúa prestada para el desarrollo del trabajo.

A nuestras familias que nos brindaron un apoyo incondicional a lo largo del desarrollo de este proyecto. Gracias por incentivarnos cada día y hacer de nosotros mejores personas.

CONTENIDO

LISTA DE FIGURAS

Pág.

LISTA DE TABLAS

Pág.

LISTA DE ANEXOS

Pág.

RESUMEN GENERAL DE TRABAJO DE GRADO

TITULO:	DESARROLLO DE UN SISTEMA PARA LA DETECCIÓN DE MOVIMIENTOS SÍSMICOS USANDO REDES NEURONALES ARTIFICIALES
AUTOR(ES):	Julián Darío Miranda Calle Ciro Alberto Gamboa Entralgo
PROGRAMA:	Facultad de Ingeniería de Sistemas e Informática Facultad de Ingeniería Electrónica
DIRECTOR(ES):	MSc. Angélica Flórez Abril PhD. Miguel Alfonso Altuve Paredes

RESUMEN

Este proyecto plantea el desarrollo de un sistema para la detección de movimientos sísmicos mediante el uso de redes neuronales artificiales. Se inicia con una contextualización en el ámbito sismológico y de aprendizaje automático que incluye características y atributos de las señales sísmicas y características de los procesos de clasificación, complementando los conceptos con una revisión de los antecedentes, evidenciando la implementación de técnicas de aprendizaje de máquina en la clasificación y detección sísmica. Posteriormente, se define la población muestral que comprende eventos sísmicos históricos de la Red Sismológica Nacional de Colombia (RSNC), dentro de la cual se extrae una muestra que es usada para el desarrollo modular del sistema. Este desarrollo está enmarcado en una metodología Ágil Scrum y de Prototipado. El flujo modular inicia con el análisis de las estaciones de mayor relevancia con el fin de reducir el costo computacional y de procesamiento, seguido del preprocesamiento de los datos (filtrado, normalización y re-muestreo), la extracción de atributos, y las etapas de entrenamiento, validación y prueba del clasificador binario desarrollado. Los atributos fueron extraídos sobre ventanas de 200 muestras de 5.144 eventos de las estaciones BRR, RUS, PAM y PTB de la RSNC con epicentro en Santander, en un periodo comprendido entre el 2015 y el 2017. Al unificar los módulos del último prototipo funcional y ejecutar las pruebas de validación cruzada al clasificador, se obtiene un modelo generalizable que clasifica los eventos sísmicos con un 99.21% de exactitud.

PALABRAS CLAVE:

Redes Neuronales Artificiales, Clasificación de eventos sísmicos, sismología en Santander.

V° B° DIRECTOR DE TRABAJO DE GRADO

GENERAL SUMMARY OF WORK OF GRADE

TITLE: DEVELOPMENT SEISMIC EVENT CLASSIFICATION SYSTEM USING ARTIFICIAL NEURAL NETWORKS

AUTHOR(S): Julián Darío Miranda Calle
Ciro Alberto Gamboa Entralgo

FACULTY: Facultad de Ingeniería de Sistemas e Informática
Facultad de Ingeniería Electrónica

DIRECTOR(S): MSc. Angélica Flórez Abril
PhD. Miguel Alfonso Altuve Paredes

ABSTRACT

This project proposes the development of a system for the detection of seismic movements using artificial neural networks. It begins with a contextualization in the seismological and Machine Learning environment. This includes characteristics and attributes of the seismic signals and characteristics of the classification processes. The concepts are complemented with a review of the background, showing the implementation of Machine Learning techniques for seismic classification and detection. Subsequently, the sample population that includes historical seismic events of the National Seismological Network of Colombia (RSNC) is defined, within which a sample that is used for the modular development of the system is extracted. This development is framed in an Agile Scrum and Prototyping methodology. The modular flow begins with the analysis of the most relevant stations to reduce the computational and processing costs. Then a preprocessing of the data (filtering, normalization and re-sampling) is carried out, followed by the extraction of attributes, and the training, validation and testing stages. The attributes DOP, RV2T, entropy, kurtosis and asymmetry were extracted on 200 samples windows of 5144 events from the RSNC stations: BRR, RUS, PAM and PTB with epicenter in Santander, between 2015 and 2017. By performing the cross-validation tests to the classifier, a generalizable model that classifies the seismic events with a 99.21% accuracy is obtained. These results agree with what was reported by Ibs-Von Seht, et. al., who obtained a 97% accuracy in the classification with data between 2005 and 2007 from Indonesia, and Hasan, et. al., who obtained 90% accuracy in the classification with data between 2013 and 2014 from Morocco.

KEYWORDS:

Artificial Neural Networks, Seismic event classification, seismology in Santander.

V° B° DIRECTOR OF GRADUATE WORK

INTRODUCCIÓN

Hace unos 4.540 millones de años se produjo una acumulación de nebulosa, una masa de gas y polvo en forma de disco en el sistema solar que dio origen al planeta Tierra, un cuerpo en estado líquido que se fue enfriando hasta formar una corteza terrestre sólida, en la que se formaron rocas y placas continentales[1]. El supercontinente conocido como Pangea, formado por estas rocas y placas, empezó a separarse lentamente debido a la transición magmática de las capas internas.

Ese proceso de traslación de las capas continentales que la Tierra sufría en la era primitiva se mantiene actualmente, produciendo en la corteza terrestre movimientos verticales y horizontales que en promedio representan un desplazamiento de 100 micrómetros anuales[2]. Esto se debe a que dicha corteza se encuentra compuesta por placas (trozos de litosfera) que conforman los fondos marinos y las superficies continentales. El movimiento de estas placas es el producto de las presiones internas de las corrientes del manto terrestre y las diferencias de densidad y temperatura, lo que causa roces y choques con placas contiguas, produciendo roturas, elevaciones, plegamientos montañosos o hundimientos de una placa bajo la otra (subducción)[3]. Estos desplazamientos internos se perciben como movimientos superficiales en las capas de la litosfera que son imperceptibles en la mayoría de los casos, pero que, en ocasiones, producto de la energía acumulada, resultan en vibraciones perceptibles que se transforman en sismos de baja y de gran magnitud.

---

[1] DALRYMPLE, Brent. The Age of the Earth. California: Stanford University Press. ISBN 0-8047-1569-6. 492 pp. 1991.

[2] TARBUCK, Edward; LUTGENS, Frederick. Ciencias de la Tierra, una introducción a la Geología Física. Universidad Autónoma de Madrid. Pearson, Prentice Hall, 8va Ed. ISBN: 978-84-832-2690-2. 736 pp. 2005.

[3] *def*. Subducción: Proceso de hundimiento de una porción oceánica componente de la placa de litósfera respectiva, en un límite convergente. Tomado de: ESTRADA, Luis. Apuntes de Sismología. Universidad Nacional de Tucumán UNT, México. Facultad de Ciencias exactas y tecnología, Departamento de Geodesia y Topografía. 2012. 31 pp.

Al ocurrir, los sismos liberan ondas sísmicas que afectan directa o indirectamente a la infraestructura y población de la superficie terrestre, generando pérdidas económicas, ambientales y humanas. También son la causa de deslizamientos, inundaciones, movimientos mareomotrices, epidemias, fuego, entre otros. Según los datos publicados en el periódico El Tiempo en abril de 2016, un estudio de la Asociación Colombiana de Ingeniería Sísmica (AIS) concluye que el 87% de la población colombiana se encuentra en un nivel de riesgo sísmico considerable; de este porcentaje, el 40% se encuentra en zonas de amenaza alta y el 47% en zonas de amenaza media[4]. Adicionalmente, la falla de Santa Marta – Bucaramanga atraviesa el territorio colombiano, convirtiendo a la región de la Mesa de los Santos, departamento de Santander, en el segundo nido sísmico más activo del mundo[5].

En consecuencia, se hace necesario planificar, prevenir y mitigar los efectos de los desastres naturales de categoría sísmica para la preservación de la vida humana, de las estructuras comerciales y viviendas, entre otros. Aunque Colombia cuenta con el sistema de la RSNC que permite la visualización y el monitoreo de los eventos sísmicos, existe una carencia de un sistema robusto que, adicional al monitoreo, ayude en las labores de identificación oportuna de este tipo de eventos, tal que puedan ser desplegadas las acciones de reacción y mitigación de los efectos producidos por estos eventos en zonas de alto riesgo sísmico, como lo es el departamento de Santander.

Teniendo en cuenta la alta sismicidad del departamento y la carencia de un sistema integrado de monitoreo y alerta temprana ante el riesgo y ocurrencia de eventos sísmicos en Colombia, la Universidad Pontificia Bolivariana seccional Bucaramanga (UPB) en alianza con la Universidad Francisco de Paula Santander de Cúcuta ha

[4] CORREAL, Juan Francisco. ¿Cuán vulnerable es Colombia ante un sismo? El Tiempo, Colombia. Disponible en: http://www.eltiempo.com/archivo/documento/CMS-16571309. 2016.

[5] ZARIFI, Zoya; HAVSKOV, Jens: HANYGA, Andrezej. An insight into the Bucaramanga nest. Tectonophysics. 1-13 pp. 2007.

planteado el proyecto de investigación “Desarrollo de un sistema de monitoreo y alerta de movimientos sísmicos – Tellurico” en el que se definen las etapas de detección, localización, distribución de las intensidades y alerta temprana de eventos sísmicos. En el proyecto ha planteado que, para que exista una alerta temprana, los datos deben ser procesados tal que los eventos puedan ser anticipados antes de que cause un impacto sobre la superficie, situación que no puede ser resuelta adecuadamente con los algoritmos estandarizados que actualmente implementa la RSNC.

Ante esta situación, se plantea este proyecto, en el cual se desarrolla un prototipo de un sistema de detección de movimientos sísmicos, haciendo uso de planteamientos estadísticos, procesamiento de señales y redes neuronales artificiales.

Este documento presenta un sistema de clasificación de eventos sísmicos fuera de línea, iniciando con una contextualización sismológica y de aprendizaje automático que se enfoca en las técnicas de clasificación de este tipo de eventos, detallando los antecedentes de aquellas técnicas llevadas a cabo para la solución de problemas de carácter sismológico. En la segunda parte se describe la metodología y el procedimiento ejecutados para el desarrollo del sistema, partiendo de la población y la muestra sísmica escogidas, de los instrumentos y herramientas utilizados para el diseño, desarrollo e implementación modular de los componentes del sistema. Con los módulos integrados, se prosigue a ejecutar las pruebas de validación del clasificador para tener una noción cuantitativa y cualitativa del desempeño del sistema. Finalmente, se presentan conclusiones y recomendaciones futuras a partir del trabajo realizado con el sistema.

OBJETIVOS

Objetivo General

Desarrollar un sistema para la detección de movimientos sísmicos con los datos registrados por los sismogramas de la Red Sismológica Nacional de Colombia, mediante el uso de redes neuronales artificiales y clasificación de segmentos de señales sísmicas.

Objetivos Específicos

- Analizar los patrones existentes en las señales sísmicas en el proceso de detección de sismos, tal que permita la caracterización de los datos de entrada y salida requeridos para el proceso de detección.
- Seleccionar las señales sísmicas apropiadas con el fin de reducir la carga computacional.
- Identificar los atributos de tiempo, frecuencia y linealidad de las señales sísmicas a través del análisis de los patrones existentes en las mismas en el proceso de detección de sismos.
- Diseñar la arquitectura del sistema a través del modelado de las interacciones de los diversos componentes, mediante el uso de diagramación UML de acuerdo con los requerimientos definidos.
- Validar el algoritmo de clasificación de segmentos de señales sísmicas, mediante la descripción de su arquitectura y pruebas de evaluación de desempeño.
- Implementar el sistema diseñado para la detección de eventos sísmicos, integrando los módulos desarrollados.

1. MARCO TEÓRICO

Esta sección se divide en cuatro temáticas de estudio, consideradas relevantes para el desarrollo del proyecto: sismología, aprendizaje automático, complejidad temporal y metodología ágil de desarrollo de software Scrum.

1.1. SISMOLOGÍA

Los sismos son movimientos que se producen por vibraciones repentinas causadas por la relajación y reactivación de la energía acumulada por la deformación de la litósfera, que se propaga a través de las demás capas geológicas. Los sismos también pueden ser causados por[6]: eventos volcánicos, hundimiento, deslizamiento y explosiones atómicas.

Cuando un sismo ocurre, el foco o hipocentro se sitúa siempre por debajo de la superficie, con una profundidad no superior a los 700 kilómetros[7]. El epicentro es la proyección del foco a nivel superficial donde el sismo alcanza su mayor intensidad. La falla es una zona de liberación repentina de energía en la que las rocas son sometidas a gran presión constante y por la que son generadas las ondas sísmicas.

1.1.1. Magnitudes de medición sismológica

La magnitud permite comparar un sismo con otro con base en una medida instrumental que indica la energía liberada durante la ruptura[8]. En sismología suele medirse la aceleración máxima del suelo compuesta por la aceleración horizontal y la aceleración vertical.

---

[6] DÁVILA MADRID, Ramón. Notas Introductorias en Sismología. Posgrado en ciencias de la Tierra, Centro de Geociencias, Universidad Autónoma de México. 2011. 36 pp.

[7] SERVICIO GEOLÓGICO MEXICANO. Causas, Características e Impactos de los Sismos. Secretaría de Economía de México. 2013.

[8] SARRIA MOLINA, Alberto. Ingeniería sísmica. Ediciones Uniandes, 2ª Edicion. 1995.

La Aceleración Máxima del Suelo (PGA) se expresa como la composición de la aceleración horizontal (PHA o PHGA) y la aceleración vertical (PVA o PVGA) del movimiento. La PHA es expresada en términos del movimiento norte-sur (H1) y el movimiento este-oeste (H2). El valor de la PGA resultante puede obtenerse al escoger el valor más alto entre las tres componentes, promediar estos valores o calcular la magnitud del vector que los conforman.

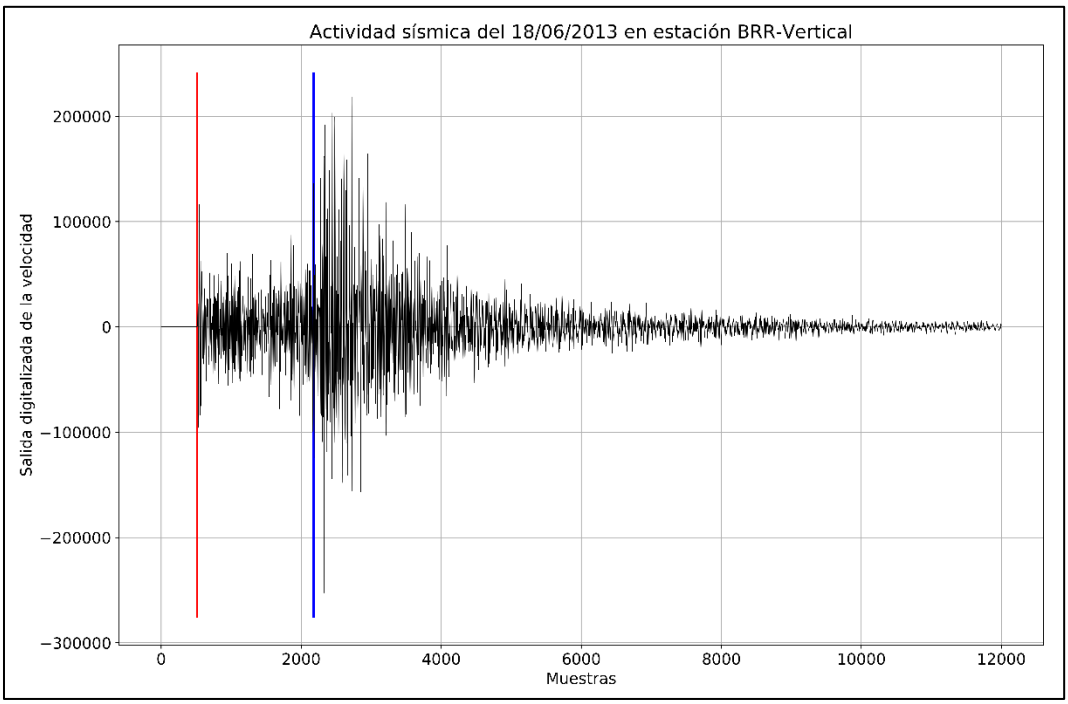
1.1.2. Ondas sísmicas

Cuando una fuerza incide sobre un material, éste tiende a la deformación. Cuando el límite elástico es alcanzado y se sobrepasa, la fuerza se propaga a través del material y el medio en el que se encuentra en forma de ondas elásticas[9]. Estas ondas se clasifican en: ondas de cuerpo y ondas de superficie.

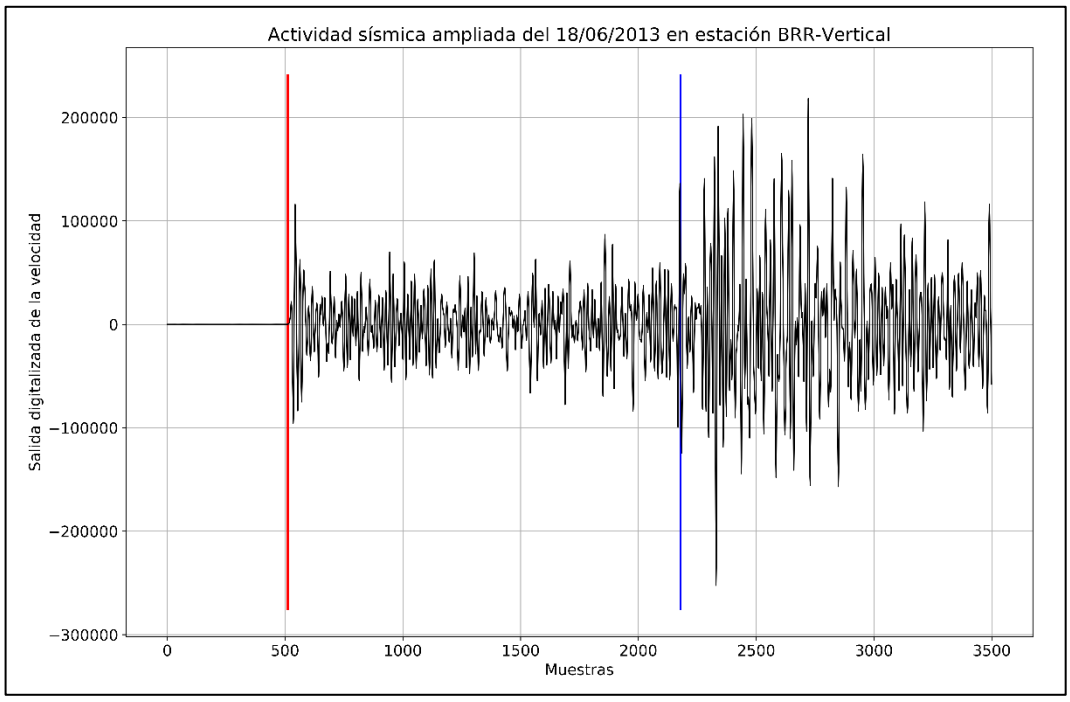
- Ondas de cuerpo: se propagan por el interior y se dividen en ondas P o compresionales y ondas S o cortantes. Estas primeras involucran refracciones y compresiones de los materiales que atraviesan. El movimiento de las partículas del medio es paralelo a la dirección de propagación de la onda. Cuando un sismo se presenta, son las primeras ondas detectadas debido a su frecuencia, tal como lo muestra la Figura 1 con una marcación de color rojo.

  Las ondas S o secundarias arriban detrás de las ondas P y causan deformación cortante al medio que atraviesan. El movimiento de las partículas del medio es perpendicular a la dirección de propagación de la onda. Al ocurrir un sismo, ambas ondas son liberadas, lo que causa un movimiento usualmente de la roca en las capas al interior de la Tierra. Esto significa que en la superficie son ligeramente apreciables o, en la mayoría de las ocasiones, apenas perceptibles o imperceptibles, tal como lo muestra la Figura 1 con una marcación de color azul.

[9] SÁNCHEZ, Francisco. Los Terremotos y sus Causas. Instituto Andaluz de Geofísica y Prevención de Desastres Sísmicos. España. 2007. 24 pp.



(a)



(b)

Figura 1. Señal registrada por la estación de Barranca BRR en su componente vertical del evento sísmico ocurrido el 18 de junio de 2013 con epicentro en el departamento de Santander: (a) registro completo del evento, (b) acercamiento a la porción de interés. Señal adaptada y procesada a partir de los registros públicos sismológicos de la Red Sismológica Nacional de Colombia.

- Ondas superficiales: son el resultado de la interacción entre las ondas de cuerpo y los estratos superficiales de la tierra. Estas ondas viajan a lo largo de la superficie terrestre con amplitudes que decrecen exponencialmente con la profundidad[10] y son las últimas ondas en ser detectadas, pero su impacto, al ser superficiales, puede ser mayor.

1.1.3. Escalas sismológicas

Existen dos tipos de escalas para la clasificación de movimientos telúricos: escalas de intensidad y escalas de magnitud. La escala sismológica de Mercalli, la escala sismológica de Richter y la escala sismológica de magnitud de momento son las más usadas en la actualidad para la clasificación de movimientos sísmicos. A continuación, se presentan estas tres escalas.

1.1.3.1. Escala sismológica de Mercalli (MM)

Conocida como la Escala de Mercalli Modificada (MM), esta escala ha pasado por cuatro modificaciones desde su planteamiento por el vulcanólogo italiano Giuseppe Mercalli. Inicialmente, la escala fue propuesta por el italiano en 1884 para la evaluación de un sismo según su impacto percibido a nivel visual en las estructuras y edificaciones. Con diez niveles, Mercalli clasificaba los sismos y los databa conforme identificaba su magnitud. Adolfo Cancani amplió los niveles de visualización del impacto del sismo a doce, modificación que fue acogida por el geofísico August Sieberg para su posterior reformulación.

Actualmente, la Escala sismológica de Mercalli permite la evaluación de la intensidad de los sismos en doce grados de medición, mediante la visualización y valoración de los efectos y daños causados a las estructuras involucradas. Los niveles de intensidad propuestos en la escala se encuentran detallados en la Tabla 1, donde la medida *g* corresponde a la aceleración de la gravedad.

[10] KRAMER L. Steven. Geotechnical Earthquake Engineering. Prentice Hall, USA, 1996.

Tabla 1. Escala sismológica de Mercalli.

Int.	Aceleración		Grado	Descripción
	(Gal)	(g)		
I	< 0.5	< 0.0005	Muy débil	Imperceptible
II	0.5 – 2.5	0.0005–0.0025	Débil	Levemente perceptible
III	2.5 – 6.0	0.0025–0.0061	Leve	Perceptible por personas en pisos altos
IV	6.0 – 10	0.0061–0.0100	Moderado	Perceptible por personas en edificios
V	10 – 20	0.0100–0.0200	Poco fuerte	Perturbaciones en edificaciones
VI	20 – 35	0.0200–0.0350	Fuerte	Daños leves en viviendas
VII	35 – 60	0.0350–0.0611	Muy fuerte	Daños leves en estructuras y edificios
VIII	60 – 100	0.0611–0.1019	Destructivo	Daños en estructuras y edificaciones
IX	100 – 250	0.1019–0.2549	Muy destructivo	Desplazamiento de estructuras
X	250 – 500	0.2549–0.5098	Desastroso	Daños graves en edificaciones
XI	> 500	> 0.5098	Muy desastroso	Destrucción de puentes y mampostería
XII	N/R	N/R	Catastrófico	Destrucción total de edificaciones

En la Tabla 1 puede notarse un aumento progresivo en intervalos de aceleración que se encuentra relacionado con la percepción de destrucción del sismo. Esta aceleración no representa una magnitud relevante para el estudio sismológico, debido a que se trata de una aproximación al cambio de velocidad durante el movimiento telúrico y la escala está enfocada en la representación del daño estructural en términos prácticos visibles.

Para tener una idea de la aceleración sísmica que puede presentarse durante un movimiento telúrico, existe una correlación entre las magnitudes de aceleración máxima y velocidad máxima del suelo, conforme se han expresado los niveles de intensidad de la Escala Sismológica de Mercalli Modificada. Esta correlación es presentada en la Tabla 2.

Tabla 2. Correlación de la aceleración máxima (PGA) y la velocidad máxima (PGV) del suelo con la escala sismológica de Mercalli.

Int. de Mercalli	PGA (g)	PGV (cm/s)	Percepción	Potencial de daño
I	< 0.0017	< 0.1	No apreciable	Ninguno
II-III	0.0017 – 0.014	0.1 – 1.1	Muy leve	Ninguno
IV	0.014 – 0.039	1.1 – 3.4	Leve	Ninguno
V	0.039 – 0.092	3.4 – 8.1	Moderado	Muy leve
VI	0.092 – 0.18	8.1 – 16	Fuerte	Leve
VII	0.18 – 0.34	16 – 31	Muy fuerte	Moderado
VIII	0.34 – 0.65	31 – 60	Severo	Moderado a fuerte
IX	0.65 – 1.24	60 – 116	Violento	Fuerte
X+	> 1.24	>116	Extremo	Muy fuerte

1.1.3.2. Escala sismológica de Richter

La Escala sismológica de Richter fue propuesta por Charles Francis Richter y Beno Gutenmerg en el año 1935 como parte de una investigación del Instituto de Tecnología de California Caltech en movimientos sísmicos al sur de California. La Escala de Richter se enfoca en la búsqueda de la representación cuantitativa de la magnitud del impacto de un sismo. La primera magnitud establecida en la escala es el 0. Desde el 0 hasta la magnitud representada por el 3.0, se clasifican sismos cuyo desplazamiento promedio puede variar y solaparse entre los niveles de medición de este intervalo. La Escala sismológica de Richter se muestra en la Tabla 3.

Tabla 3. Escala logarítmica de Richter.

Magnitud	Descripción	Intens. Mercalli	Efectos	Frecuencia de ocurrencia
1.0 – 1.9	Micro	I	Microsismos registrados en sismógrafos sensibles.	Continua, muchos millones por año.
2.0 – 2.9	Menor	I a II	Sin daño en edificaciones, sentido por pocas personas.	Más de un millón por año.
3.0 – 3.9	Menor	III a IV	Vibración imperceptible de puertas, sentida por pocas personas.	Más de 100.000 por año.
4.0 – 4.9	Leve	IV a VI	Vibración perceptible que no causa efecto infraestructural.	Entre 10.000 y 15.000 por año.
5.0 – 5.9	Moderado	VI a VII	Daño leve en edificaciones, sentido por casi todas las personas.	Entre 1.000 y 1.500 por año.
6.0 – 6.9	Fuerte	VII a X	Daño moderado en edificaciones, sentido por todas las personas.	Entre 100 y 150 por año.
7.0 – 7.9	Fuerte	VIII o más	Daño a la mayoría de edificaciones, sentido hasta en un radio de 250 Km.	Entre 10 a 20 por año.
8.0 – 8.9	Muy fuerte	VIII o más	Daño estructural a edificios, sentido en gran radio de expansión.	Uno al año.
9.0 o más	Demasiado fuerte	VIII o más	Daño severo en estructuras.	Uno cada 10 a 50 años.

La magnitud de la escala es determinada por el logaritmo de la relación entre la amplitud de las ondas registradas por el sismograma, teniendo en cuenta un margen de medición del sismógrafo que lo registra.

1.1.4. Atributos de señales sismológicas

Las señales sismológicas, como toda señal física, presentan atributos y características que pueden ser extraídas mediante su procesamiento. A continuación, se describen estos atributos.

1.1.4.1. Polarización

La polarización de una onda hace referencia a la orientación de los vectores de campo de la onda. En el caso de ondas electromagnéticas, por ejemplo, la polarización corresponde a la orientación de los vectores de campo eléctrico y magnético[11]. En el caso de las ondas sísmicas, existen cuatro tipos de ondas que determinan el vector de campo sísmico: las ondas compresionales, u ondas P, las ondas secundarias, u ondas S, las ondas de Love y las ondas de Rayleigh. Estas dos últimas permiten obtener información de la estructura del suelo debido a sus propiedades de dispersión[12].

La polarización en las ondas sísmicas, al igual que en cualquier tipo de onda presente en espacio-tiempo, es elíptica. Sin embargo, debido a las propiedades de las dos primeras ondas, ondas P y ondas S, el radio axial de polarización es infinito y la elipse se ve reducida a una línea. De esta forma, ambas ondas están polarizadas linealmente[13], aunque las ondas P estén polarizadas longitudinalmente[14] y las ondas S estén polarizadas horizontal y verticalmente.

---

[11] HUM, Sean Victor. Wave Polarization. ECE422: radio and Microwave Wireless Systems, The Edward S. Roger Sr. Department of Electrical & Computer Engineering, University of Toronto, Canada. 4 pp.

[12] HOBIGER, Manuel. Polarization of surface waves: characterization, inversion and application to seismic hazard assessment. Earth Sciences. Université de Grenoble. NNT: 2011GRENU005. 309 pp.

[13] PERELBERG, Azik; HORNBOSTEL, Scott. Applications of seismic polarization analysis. Geophysics Journal, Vol. 59, No. 1. ISBN: 0926-9851. pp. 119-130. 1994.

[14] SHEARER; Peter. Introduction to Seismology. Cambridge University Press. ISBN 978-0-521-88210-1. 2009.

Una técnica utilizada para calcular la dirección de la polarización de la onda sísmica, obviando el modelo de velocidades, es haciendo uso de la matriz de covarianza, que permite determinar la varianza de cada una de las componentes y la covarianza de las mismas entre sí. El vector propio del eje mayor determina la dirección, que será paralela a la dirección de las ondas P y perpendicular a la dirección de las ondas S. La matriz de covarianzas C se describe de la siguiente forma:

$$C = \begin{bmatrix} Cov(x,x) & Cov(x,y) & Cov(x,z) \\ Cov(y,x) & Cov(y,y) & Cov(y,z) \\ Cov(z,x) & Cov(z,y) & Cov(z,z) \end{bmatrix} \qquad \text{(Ecuación 1)}$$

En la matriz de covarianzas, cada $Cov(C_1, C_2)$ corresponde a la covarianza que existe entre la componente C_1 y la componente C_2. Cuando se evalúan las mismas componentes contrastadas ($C_1 = C_2$), se está hallando la varianza de la componente.

La covarianza entre las componentes se halla como[15]:

$$Cov(C_1, C_2) = \frac{\sum_{i=1}^{n}(C_{1i} - \overline{C_1})(C_{2i} - \overline{C_2})}{n-1} \qquad \text{(Ecuación 2)}$$

Donde $\overline{C_1}$ y $\overline{C_2}$ denotan la media para cada uno de los datos en las componentes C_1 y C_2. Sin embargo, según lo han manifestado Kaur, et al.[16], es matemáticamente complejo identificar la llegada de la onda P basando el análisis en la dirección de polarización que se consigue con el vector propio del mayor eje. Por ende, se

[15] ZHANG, Yuli; WU, Huaiyu; CHENG, Lei. Some new deformation formulas about variance and covariance. Proceedings of 4th International Conference on Modelling, Identification and Control (June 2012 - ICMIC2012). pp. 987–992.

[16] KAUR, Komalpreet; WADHAWA, Manish; PARK, E.K. Detection and Identification of Seismic P-Waves using Artificial Neural Networks. The 2013 International Joint Conference on Neural Networks, Dallas, Texas, United States of America. DOI: 10.1109/IJCNN.2013.6707117. 2013.

considera un parámetro que es independiente de la fuente: el Grado de Polarización (DOP).

El Grado de Polarización es una medida usada para describir la porción de onda que se encuentra polarizada y es muy sensible a características de espacio-tiempo del campo de onda[17]. El DOP depende en su mayoría la estructura terrestre, obviando la condición de la fuente.

Para calcular el DOP debe tenerse en cuenta que las tres componentes de cada estación estén sometidas al mismo nivel de ruido, presenten la misma frecuencia, la misma escala y el mismo ancho de banda. En caso de que estos parámetros varíen entre componentes, el DOP puede tener un sesgo creciente según la variabilidad de los parámetros.

El DOP está definido sobre la matriz de covarianzas como[18]:

$$F(t) = \frac{(\lambda_1 - \lambda_2)^2 + (\lambda_2 - \lambda_3)^2 + (\lambda_3 - \lambda_1)^2}{2(\lambda_1 + \lambda_2 + \lambda_3)} \qquad \text{(Ecuación 3)}$$

Donde cada λ_i son los valores propios de la matriz de covarianzas en un tiempo t de la ventana de señal correspondiente. La descomposición por valores propios de esta matriz de covarianzas, se hace teniendo en cuenta la función característica:

$$det(C - \lambda I) = 0 \qquad \text{(Ecuación 4)}$$

[17] GALPERIN, E. I. The Polarization of Seismic Waves and its Potential for Studying the Rocks Surrounding the Borehole. ISBN: 978-94-009-5195-2. DOI: https://doi.org/10.1007/978-94-009-5195-2_12.

[18] KAUR, Komalpreet; WADHAWA, Manish; PARK, E.K. Detection and Identification of Seismic P-Waves using Artificial Neural Networks. The 2013 International Joint Conference on Neural Networks, Dallas, Texas, United States of America. DOI: 10.1109/IJCNN.2013.6707117. 2013.

Que resulta en la ecuación general:

$$\lambda^3 - tr(C) + (C_{11} + C_{22} + C_{33})\lambda - det(C) = 0 \qquad \text{(Ecuación 5)}$$

Donde $tr(C)$ es la traza de la matriz de covarianzas (sumatoria de la diagonal de la matriz), cada C_{ij} es el cofactor de la sub-matriz en c_{ij} y $\det(C)$ corresponde al determinante de la matriz de covarianzas. Las tres raíces de la Ecuación 5 son los tres valores propios de la matriz de covarianzas y son los utilizados para el cálculo del DOP.

1.1.4.2. Radio de Potencia Vertical contra Potencia Total (RV2T)

El Radio de Potencia Vertical contra Potencia Total (RV2T) es una medida de comparación entre la potencia del componente vertical en la onda sísmica contra la potencia total de la onda, pues es esta componente la que se ve mayormente alterada por el movimiento sísmico. Es así como las ondas P tienen la mayoría de contenido sísmico en su componente vertical. El RV2T es calculado de la siguiente forma[19]:

$$RV2T = \frac{\sum_{t=1}^{N} x^2(t)}{\sum_{t=1}^{N}\left(x^2(t) + y^2(t) + z^2(t)\right)} \qquad \text{(Ecuación 6)}$$

Donde $x(t), y(t)$ y $z(t)$ son las amplitudes de las tres componentes espaciales registradas en el tiempo t y N es la cantidad de muestras de la ventana de evaluación.

[19] KAUR, Komalpreet; WADHAWA, Manish; PARK, E.K. Detection and Identification of Seismic P-Waves using Artificial Neural Networks. The 2013 International Joint Conference on Neural Networks, Dallas, Texas, United States of America. DOI: 10.1109/IJCNN.2013.6707117. 2013.

1.1.4.3. Entropía de Shannon

Le entropía de Shannon o entropía de la información describe la cantidad de información que presenta un evento singular, concepto que está ligado a la incertidumbre que se tiene del evento, asociada con una distribución de probabilidad simétrica[20].

Por ejemplo, si se tiene una caja negra con lápices de 4 colores distintos sin que ninguno predomine, la incertidumbre sobre la escogencia de uno de estos colores es máxima, pues no se tiene información sobre el contenido de la caja. En este caso, la entropía es muy alta o máxima, pues el evento presenta una cantidad de certeza muy baja y mucha información que puede ser recopilada. Por otro lado, si se conoce que en la caja hay más de un lapicero azul que de otro color, la incertidumbre sobre la salida disminuye, la igual que la entropía, pues hay la certeza sobre el evento es ligeramente mayor y la cantidad de información qué recopilar relacionada con el evento es menor.

La entropía de Shannon está definida como la suma del producto entre todas las posibles salidas x_i del evento X y el logaritmo de la probabilidad de ocurrencia de x_i:

$$H(X) = \sum_{i=1}^{n} p(x_i) log_2 \left(\frac{1}{p(x_i)}\right) = -\sum_{i=1}^{n} p(x_i) log_2 \big(p(x_i)\big) \quad \text{(Ecuación 7)}$$

Donde $p(x_i) = Pr(X = x_i)$ es la probabilidad que ocurrencia de i en el evento X.

[20] SCHNEIDER, Thomas. Information theory primer with an appendix on logarithms, National Cancer Institute. DOI: http://dx.doi.org/10.13140/2.1.2607.2000. 2007.

1.1.4.4. Momentos Centrales Estadísticos

En la teoría de la probabilidad, un momento central es una medida cuantitativa específica de una distribución de probabilidad de una variable aleatoria, alrededor de la media de dicha variable, debido al esparcimiento y simetría de una distribución sobre la media.

El *n-ésimo* momento central de una variable aleatoria alrededor de la media puede ser calculado como[21]:

$$\mu_n = E[(X - E[X])^n] \qquad \text{(Ecuación 8)}$$

Donde X es el conjunto de datos de la variable aleatoria y $\mathrm{E[X]}$ es la media de la variable. Los primeros cinco momentos centrales son:

- $\mu_0 = 0$ correspondiente al momento cero.
- μ_1 correspondiente al primer momento central o a l valor esperado de media.
- μ_2 que corresponde al segundo momento central llamado varianza σ^2 donde σ es la desviación estándar.
- μ_3 que corresponde al tercer momento central llamado asimetría.
- μ_4 que corresponde al cuarto momento central llamado kurtosis.

La asimetría indica el grado de simetría que tiene la distribución de la variable aleatoria, asociándola con una distribución normal de la misma varianza[22], si la distribución de la variable es unimodal. Si la distribución tiende a la derecha y la cola izquierda es más larga, es decir, la mayoría de los datos están ubicados al costado derecho, se dice que es una distribución con asimetría negativa o izquierda. Si la

[21] WATKINS, Joseph. Moments and Generating Function. Department of Mathematics, University of Arizona. 2009.

[22] SATO, Michikazu. Some remarks on the mean, median, mode and skewness. Australian Journal of Statistics 39(2), 219-224. DIO: https://doi.org/10.1111/j.1467-842X.1997.tb00537.x. 1997.

distribución tiende hacia la izquierda y la cola derecha es más larga, se dice que es una distribución con asimetría positiva o derecha. En caso de que la distribución de la variable esté perfectamente solapada con la distribución normal, se dice que se trata de una distribución perfectamente simétrica.

La asimetría es calculada como:

$$\gamma_1 = \frac{\mathrm{E}[(X-\mu_1)^3]}{(\mathrm{E}[(X-\mu_1)^2])^{3/2}} = \frac{\mu_3}{\sigma^3} = \frac{\sum_{i=1}^{n}(x_i-\bar{x})^3}{n\cdot\sigma^3} \qquad \text{(Ecuación 9)}$$

Una aproximación del valor poblacional de asimetría para cálculos con muestras singulares puede ser calculado usando la Ecuación 13 en términos de los momentos centrales m_2 y m_3 de la forma:

$$G_1 = \frac{\sqrt{n(n-1)}}{n-1}\frac{m_3}{(m_2)^{3/2}} \qquad \text{(Ecuación 10)}$$

Los intervalos de asimetría son los siguientes:

- Alta asimetría: asimetría menor que -1 o mayor a 1.
- Asimetría moderada: asimetría entre -1 y -0.5 o entre 0.5 y 1.
- Simetría aproximada: asimetría entre -0.5 y 0.5.

La kurtosis, por otro lado, es el cuarto momento central estadístico e indica qué tan apuntada o plana es la distribución de la variable aleatoria asociándola con una distribución normal de la misma varianza[23]. Los valores que más influencian la medida de kurtosis son los que se encuentran en las colas de la distribución. Las distribuciones con pico más plano se denominan distribuciones platicúrticas; cuando

[23] LIANG, Zhiqiang; WEI, Jianming; ZHAO, Junyu; LIU, Haitao; LI, Baoqing; SHEN, Jie; ZHENG, Chunlei. The Statistical Meaning of Kurtosis and Its New Application to Identification of Persons Based on Seismic Signals. 8(8): 5106–5119. DOI: 10.3390/s8085106. 2008.

son muy apuntadas se denominan leptocúrticas y cuando tienen una tendencia a la distribución normal, se denominan mesocúrticas.

La kurtosis está definida como la cuarta potencia de la diferencia entre cada valor x_i del conjunto de la variable aleatoria X y la media de la variable, según Karl Pearson[24]:

$$Kurt[X] = \frac{\mathrm{E}[(X-\mu_1)^4]}{(\mathrm{E}[(X-\mu_1)^2])^2} = \frac{\mu_4}{\sigma^4} = \frac{\sum_{i=1}^{n}(x_i-\bar{x})^4}{n\cdot\sigma^4} \qquad \text{(Ecuación 11)}$$

Donde la varianza σ^2 es:

$$\sigma^2 = \frac{1}{n}\sum_{i=1}^{n}(x_i-\bar{x})^2 \qquad \text{(Ecuación 12)}$$

Una aproximación del valor poblacional de kurtosis para cálculos con muestras singulares puede ser calculado usando la Ecuación 13 en términos de los momentos centrales, designados con m_2 y m_4, de la forma:

$$G_2 = \frac{(n^2-1)}{(n-2)(n-3)}\frac{m_4}{(m_2)^2} - 3(n-1)^2 \qquad \text{(Ecuación 13)}$$

Para efectos de su cálculo computacional, usualmente suele sustraerse un factor correctivo de 3 a la kurtosis con el fin de hacerla comparable con la asimetría en la distribución normal. A esto se le llama Kurtosis de Fisher[25].

[24] PEARSON, K. Das Fehlergesetz und seine Verallgemeinerungen durch Fechner und Pearson. A Rejoinder. Biometrika. 1905;4:169–212.

[25] RICHARD, A; GROENEVELD, G. M. Measuring skewness and kurtosis. The Statistician. 1984;33:391–399.

1.1.4.5. Dimensión de correlación

En la teoría matemática, una dimensión es una medida cuantitativa del espacio real o abstracto que ocupa un conjunto de datos. Existen varios tipos de dimensiones entre los que se encuentra la dimensión topológica, la dimensión fractal y la dimensión de información, entro otros. La dimensión topológica o dimensión de Čech-Lebesgue[26] es una representación entera del espacio ocupado por el conjunto de datos, por ejemplo, la dimensión topológica de un punto es cero, la de una línea es uno, la de una superficie es dos y la de un volumen es tres.

Sin embargo, cada una de estas representaciones puede tener un sinnúmero de derivaciones en las que no puede definirse con claridad una dimensión topológica de cantidad entera. Por ejemplo, una hoja de papel es una superficie que tiene una dimensión topológica de dos. Cuando se juntan varias hojas de papel, aunque la superficie no cambia, existe un engrosamiento de esta que podría interpretarse como un volumen, pero que no alcanza a determinarse como tal y en consecuencia persiste la dimensión topológica de dos. Para marcar una diferencia específica con valores cuantitativos no enteros de la dimensión de un conjunto de datos se usa la dimensión fractal[27]. Así, cuando un conjunto de datos tenga una dimensión fractal mayor a su dimensión topológica, se considerará como fractal.

En el caso en que un conjunto de datos represente una dinámica no lineal y tienda a evolucionar de una forma específica, éste puede representarse en el espacio de fase o espacio fásico. Cuando la dinámica tiene una tendencia en su evolución, se

[26] COORNAERT, Michel. Topological Dimension and Dynamical Systems. University of Strasbourg, Strasbourg, France. Springer Editorial. 3-66 pp. ISBN 978-3-319-19793-7. 2015.

[27] MANDELBROT, B. How long is the coast of Britain Statistical self-similarity and fractional dimension. Science Review, No. 156, 636–638 pp. 1967.

dice que presenta un atractor[28], un conjunto de valores numéricos que pueden representar un punto, una curva o incluso una estructura fractal. Por ejemplo, un péndulo que se encuentra en oscilación por una fuerza externa que provoca este movimiento, al final tenderá a permanecer en equilibrio en su posición inicial. En este caso sería un atractor clásico de punto fijo[29].

La dimensión de correlación (D2, CD o v) es una medida cuantitativa de la dimensión fractal de la dimensionalidad del espacio ocupado por el conjunto de datos de una variable aleatoria o, dicho de otra forma, del estado caótico de la geometría del conjunto de datos[30], con respecto a su atractor. La dimensión de correlación se define usando la integral de correlación:

$$C(\varepsilon) = \lim_{N\to\infty} \frac{g}{N^2} \qquad \text{(Ecuación 14)}$$

Donde *N* es la cantidad de datos de un conjunto multidimensional y *g* es el número total de parejas de datos que tienen una distancia entre ellos menor a un valor ε definido. Una aproximación al valor de la dimensión de correlación es: $C(\varepsilon) \sim \varepsilon^{D2}$, donde D2 es el valor requerido. Después de una reorganización logarítmica de la expresión, este valor se puede aproximar a:

$$D2 \sim \frac{\log(C(\varepsilon))}{\log(\varepsilon)} \qquad \text{(Ecuación 15)}$$

[28] LEUNG, Henry. Chaotic Signal Processing. University of Calgary, Calgary, Alberta, Canada. 152 p. ISBN 978-1-61197-325-9. 2014.

[29] MANDELBROT, B. How long is the coast of Britain Statistical self-similarity and fractional dimension. Science Review, No. 156, 636–638 pp. 1967.

[30] CROSS, Michael. Chapter 9: Dimensions, in Physics 161: Introduction to Chaos. California Institute of Technology. 9 pp. 2000.

En la práctica de las señales físicas, la dimensión de correlación sirve para diferenciar una dinámica de ruido y una dinámica no lineal, pues el espacio de trayectorias de esta segunda dinámica usualmente evoluciona hacia un conjunto de coordenadas particulares que identifican un atractor; mientras que el espacio de trayectorias de la dinámica de ruido no tiende a un atractor particular[31].

1.1.5. Servicio Geológico Colombiano

El Servicio Geológico Colombiano (SGC), anteriormente llamado Instituto Colombiano de Geología y Minería (INGEOMINAS) y derivado del Sistema Nacional de Gestión del Riesgo de Desastres, es una agencia fundada en 1918 que está encargada de la realización de estudios sobre el manejo de los recursos hídricos y del subsuelo, de las amenazas de origen geológico presentes y los peligros directos e indirectos que pueden materializarse sobre la población de Colombia[32]. Esta entidad hace parte del Ministerio de Minas y Energía, y tiene la potestad de la administración independiente de los recursos administrativos, técnicos, financieros y de patrimonio.

Dentro de los servicios prestados por el SGC están:

- Sismológicos, por parte de la Red Sismológica Nacional de Colombia (RSNC).
- Vulcanológicos, por parte de los Observatorios Vulcanológicos y Sismológicos (OVS) ubicados en las ciudades de Pasto, Manizales y Popayán.
- Geológicos
- Seguridad nuclear

[31] BOON, Mei Ying; HENRY, Bruce; SUTTLE, Catherine; DAIN, Stephen. The correlation dimension: A useful objective measure of the transient visual evoked potential? Journal of Vision January, Vol. 8, No. 6. DOI: 10.1167/8.1.6. 2008.

[32] SERVICIO GEOLÓGICO COLOMBIANO. Redes de estaciones, Instrumentación. [Última consulta: 25 de agosto de 2018]. Disponible en: seisan.sgc.gov.co/RSNC/index.php/red-de-estaciones/instrumentación.

La Red Sismológica Nacional de Colombia (RSNC) es una división adscrita al SGC desde 1993 encargada de[33]: la caracterización de la estructura terrestre, el monitoreo del comportamiento sísmico, la evaluación de la amenaza sísmica, la formulación de planes de gestión del riesgo y el suministro de información de eventos sísmicos nacionales e internacionales actuales e históricos que impacten al país, con el fin de mitigar los daños a la población y sus bienes.

A la fecha, la RSNC cuenta con un conjunto de 85 estaciones sismológicas nacionales que cubren las zonas de mayor actividad sísmica: la zona andina, el borde de los Llanos, la Costa Pacífica y la Costa Atlántica y 10 estaciones sismológicas internacionales ubicadas en Venezuela y Ecuador, entre otros países vecinos, y una red de 97 acelerógrafos de la Red de Acelerógrafos de Colombia. Las señales de las estaciones se reciben vía satélite en la central de Bogotá, donde son analizadas.

Las estaciones de la RSNC pueden identificar, a groso modo, tres tipos de eventos sísmicos[34]:

- Locales: cuando el epicentro es determinado por estaciones aledañas en la misma región.
- Regionales: cuando el epicentro es determinado por estaciones que se encuentran en otras regiones del país, ajenas a la región de ubicación del epicentro.

[33] SISTEMA DE INFORMACIÓN EN GESTIÓN DEL RIESGO DE DESASTRES. Red Sismológica Nacional de Colombia. [Última consulta: 25 de agosto de 2018]. Disponible en: http://www.redriesgos.gov.co/red-sismologica-nacional/. 2018.

[34] SERVICIO GEOLÓGICO COLOMBIANO. Redes de estaciones, Instrumentación. [Última consulta: 25 de agosto de 2018]. Disponible en: seisan.sgc.gov.co/RSNC/index.php/red-de-estaciones/instrumentación.

- Tele-sismos: cuando se identifica un epicentro internacional o muy alejado de las regiones y las delimitaciones regionales del país.

1.2. APRENDIZAJE AUTOMÁTICO

Para resolver un problema mediante métodos computacionales, se requiere una secuencia de instrucciones que transformen un conjunto de variables de entrada conocidas, en un grupo de variables de salida deseadas. Para procesos en los cuales no se conoce su comportamiento en su totalidad, se construye una aproximación. Ésta puede no explicar la totalidad de los eventos consultados, pero permite la detección de patrones que sirven como indicios para su identificación y clasificación. Este es el nicho del *Aprendizaje Automático*.

En general, el aprendizaje automático es el proceso de programación de una unidad computacional para optimizar los criterios de rendimiento haciendo uso de muestras de información recolectadas de experiencias pasadas[35]. Un modelo de aprendizaje puede ser *supervisado* o *no supervisado*. En el aprendizaje *supervisado* se le indica al algoritmo qué predecir mediante un conjunto de categorías o etiquetas. Los algoritmos de *clasificación* y *regresión* son ejemplos de aprendizaje *supervisado.* En *clasificación* se intenta predecir la categoría a la que corresponde una observación mientras que, en *regresión*, se intenta predecir un valor numérico.

Por otro lado, en el aprendizaje *no supervisado* no existen categorías adjuntas a los datos de entrada. Los algoritmos de *clustering, estimación de densidad* y *reducción de dimensionalidad* son ejemplos de aprendizaje *no supervisado.* En *clustering* se intenta agrupar datos con características similares, mientras que en *estimación de densidad* se pretende encontrar valores estadísticos que describan al conjunto de datos. En cuanto a la *reducción de dimensionalidad*, se busca reducir la cantidad de

[35] NILSSON; N. Introduction to Machine Learning; Department of Computer Science, Standford University; Standford, CA 94305; 1st Edition; 188 pp. 2005.

características de los datos para reducir el costo computacional de su procesamiento y posibilitar su representación más fácilmente[36].

1.2.1. Redes Neuronales Artificiales (ANN)

Las redes neuronales artificiales (ANN: Artificial Neural Networks) se plantean como una alternativa computacional para la toma de decisiones dentro del dominio del *Aprendizaje Automático*. Esta técnica busca emular la capacidad de aprendizaje natural de los seres vivos, la cual es atribuida al sistema neuronal de su cerebro.

1.2.1.1. Composición de las Redes Neuronales Artificiales

La unidad básica de procesamiento de una red neuronal artificial es la *neurona*. Las redes neuronales funcionan mediante la interacción de conjuntos de *neuronas* con características diferentes. Dichas características son explicadas a continuación.

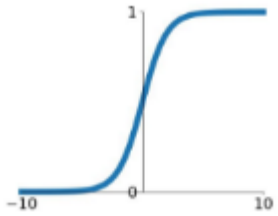
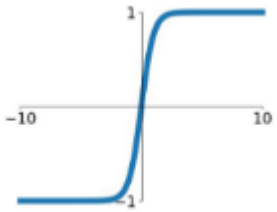
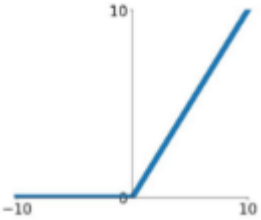
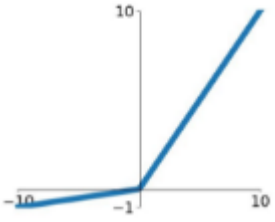
- Función de propagación: cada neurona tiene una serie de entradas provenientes de otras neuronas. Estas señales de entrada son atenuadas o amplificadas por un factor de peso y son operadas en conjunto por una función de propagación que es comúnmente una suma ponderada.

- Función de activación: es la función de umbral encargada de determinar la acción de una neurona, dependiendo del valor de entrada proveniente de la función de propagación. Si la función de activación es lineal, se dice que la neurona es lineal; de la misma forma ocurre cuando la función de activación es

[36] HARRINGTON, Peter, Machine Learning in Action. Manning publications Co, United States of America. ISBN: 978-16-172-9018-3. 382 pp. 2012.

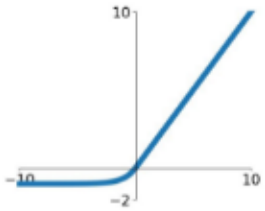
no lineal, diciendo entonces que la neurona es no lineal [37]. En la Tabla 4, se muestran algunas de las funciones de activación más comunes, usadas en redes neuronales artificiales.

Tabla 4. Funciones de activación comunes[38].

Funciones de activación		
Nombre	Función	Gráfica
Sigmoid	$\sigma(x) = \frac{1}{1 + e^{-x}}$	
Tanh	$\sigma(x) = tanh(x)$	
ReLU	$\sigma(x) = max(0, x)$	
Leaky ReLU	$\sigma(x) = max(0.1x, x)$	

[37] KRIESEL, David. A Brief Introduction to Neural Networks. In: https://doi.org/10.1016/0893-6080(94)90051-5. 244 pp. 2005.

[38] JADON, Shruti. Introduction to Different Activation Functions for Deep Learning. Available at: https://medium.com/@shrutijadon10104776/survey-on-activation-functions-for-deep-learning-9689331ba092. 2018

Funciones de activación		
Nombre	Función	Gráfica
ELU	$\sigma(x) = \begin{cases} x & x \geq 0 \\ \alpha(e^x - 1) & x < 0 \end{cases}$	

- Función de salida: es la función encargada de calcular el valor de salida de la neurona para ser transferido como entrada a otras neuronas. Con respecto a los valores de salida, se obtienen dos tipos principales de neuronas: binarias y reales. Las neuronas binarias son aquellas que proveen valores de salida dentro del intervalo {0, 1} o {-1, 1}, mientras que las neuronas reales generan una salida dentro del intervalo [0, 1] o [-1, 1][39].

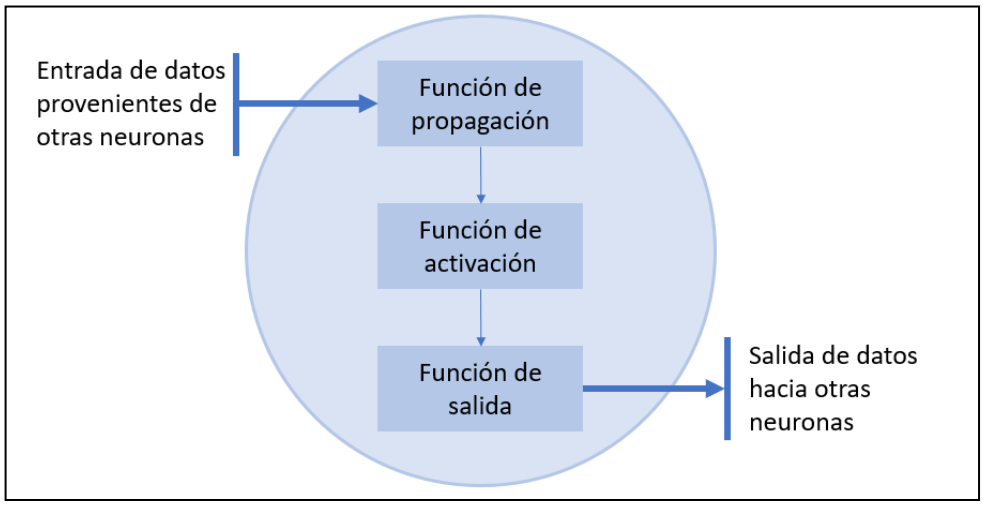


Figura 2. Estructura de una neurona artificial. Adaptado de KRIESEL, David. A Brief Introduction to Neural Networks. In: https://doi.org/10.1016/0893-6080(94)90051-5. 244 pp. 2005.

El modelo neuronal ilustrado en la Figura 2, es usado en la mayoría de las redes neuronales, modificando únicamente la función de activación. Tomando como

[39] MATICH, Damián. Redes Neuronales: Conceptos Básicos y Aplicaciones. Universidad Tecnológica Nacional, Rosario, Argentina. 55 pp. 2001.

referencia la estructura de la neurona artificial y su capacidad de interacción, las redes neuronales artificiales se distinguen por los aspectos de topología y mecanismos de aprendizaje.

La topología de una red neuronal está ligada con el algoritmo de aprendizaje usado para entrenar la red. Los factores que definen la topología de la red son las capas y la naturaleza de las conexiones entre las neuronas. En una red neuronal, una capa es un conjunto de nodos (pueden ser neuronas o fuentes de datos) con características similares. Pueden ser capas de entrada, capas ocultas o capas de salida. Mientras que las conexiones pueden ser unidireccionales (*feedforward)* o recurrentes (con al menos un lazo de realimentación o *feedback*). Teniendo en cuenta estos parámetros, existen tres clases de topologías fundamentales:

- Redes monocapa unidireccionales: en esta topología los datos fluyen en un único sentido y sólo se tienen dos capas: de entrada y de salida. Se le denomina monocapa porque sólo se consideran las capas que contienen neuronas; la capa de entrada es un conjunto de fuentes de información. Una representación gráfica de este tipo de topología se puede apreciar en la Figura 3(a).

- Redes multicapa unidireccionales: en esta topología los datos fluyen en un único sentido, pero a diferencia de las redes monocapa unidireccionales, existen una o más capas *ocultas.* La función de las capas *ocultas* es realizar un procesamiento parcial tomando los datos de la capa de entrada o de otra capa oculta y generar una salida que sirva como entrada para la capa de salida u otra capa oculta. Una representación gráfica de este tipo de topología se puede apreciar en la Figura 3(b).

- Redes recurrentes: en esta topología, existe al menos un lazo de realimentación que comunique las salidas de la red neuronal con sus entradas. Las redes neuronales recurrentes pueden ser monocapa o multicapa, además en sus lazos de realimentación añaden el uso de retrasos unitarios, lo que resulta en un

comportamiento no lineal dinámico, asumiendo que la red neuronal contiene elementos no lineales. Una representación gráfica de una red recurrente monocapa se puede apreciar en la Figura 3(c)[40].

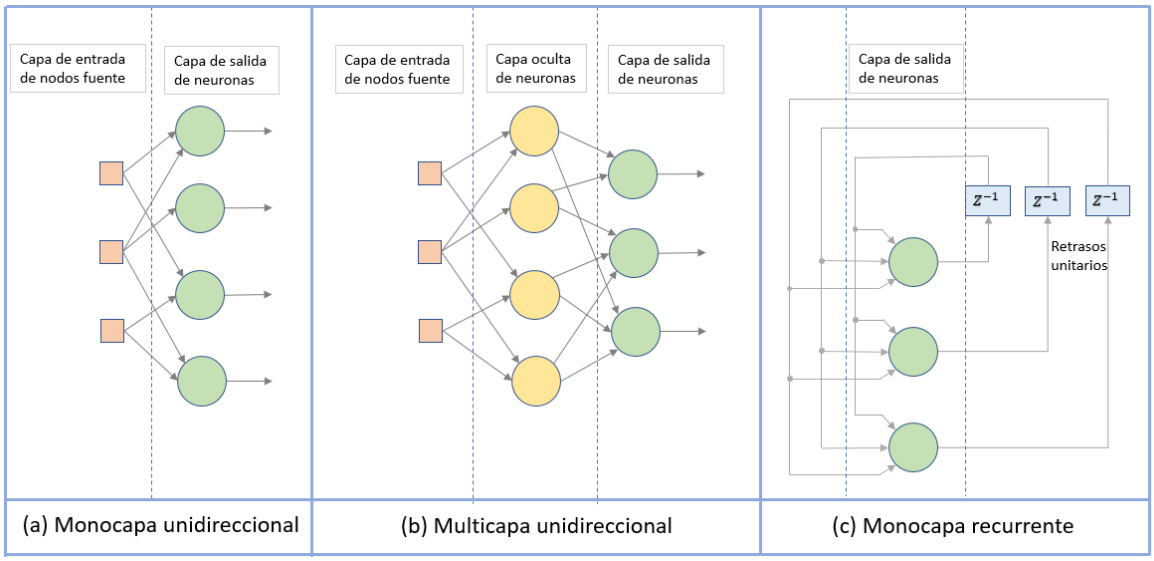


Figura 3. Topologías de red neuronal. Adaptado de HAYKIN, Simon. Neural Networks and Learning Machines. Pearson, Prentice Hall. Hamilton, Ontario, Canada. 3rd ed. ISBN: 978-0-13-147139-9. 938 pp. 2009.

1.2.1.2. Funciones y algoritmos de entrenamiento

La capacidad de aprendizaje, mediante un proceso de entrenamiento y la aplicación de la experiencia adquirida en este proceso le otorga a las ANN, la capacidad de responder apropiadamente a situaciones o datos a los que no había sido expuesta anteriormente[41].

[40] HAYKIN, Simon. Neural Networks and Learning Machines. Pearson, Prentice Hall. Hamilton, Ontario, Canada. 3rd ed. ISBN: 978-0-13-147139-9. 938 pp. 2009

[41] RUIZ, Carlo Alberto; BASUALDO, Marta Susana. Redes Neuronales: conceptos básicos y aplicaciones. Universidad Tecnológica Nacional, Facultad Regional Rosario, Argentina. 55 pp. 2001.

Los sistemas de aprendizaje cambian sus características para poder adaptarse al problema que se está afrontando y conseguir la generalización de la "comprensión" del problema. Las ANN pueden aprender mediante el desarrollo de nuevas conexiones, del cambio de la ponderación de sus conexiones, la creación de nuevas neuronas y el cambio los valores de umbral en la función de activación, entre otros.

Estos mecanismos se enmarcan en el paradigma de aprendizaje selecto para la red neuronal, la cual puede usar aprendizaje supervisado, aprendizaje no supervisado o aprendizaje reforzado. Este último involucra lazos de realimentación a la red (topología de red recurrente), para indicar si los resultados parciales son o no correctos[42]. En el aprendizaje no supervisado sólo se tiene el conjunto de atributos de entrada sin etiquetas. Con dicho conjunto, la red neuronal generar clases de atributos con características similares. En el aprendizaje supervisado, se tiene un conjunto de atributos de entrada, etiquetados con la clase a la que pertenecen, lo que posibilita hacer comparaciones durante el proceso de entrenamiento para estipular el error asociado a los valores predichos, con respecto a los valores reales.

El esquema de entrenamiento para el aprendizaje supervisado se expresa de forma general en los siguientes pasos:

1. Se ingresan los atributos (activación de las neuronas de entrada).
2. Se propaga frontalmente la entrada a través de la red (feedforward), generando la salida.
3. Se compara la salida deseada con la salida actual, generando el error asociado.
4. Se calculan las correcciones correspondientes a partir del error asociado.
5. Se aplican las correcciones, modificando los pesos de las neuronas (Backpropagation).

[42] KRIESEL, David. A Brief Introduction to Neural Networks. In: https://doi.org/10.1016/0893-6080(94)90051-5. 244 pp. 2005.

El proceso descrito se realiza de forma iterativa. Si se trata de un aprendizaje *offline*, se realiza la actualización de los pesos tras analizar un conjunto o *batch* de observaciones; si es *online*, los pesos se actualizan después de cada observación. Cuando la red neuronal ha sido expuesta a todo el conjunto de entrenamiento, se dice que ha pasado por un *epoch*[43].

Existen tres conjuntos de datos con los que se ejecutan las fases de entrenamiento, validación y prueba: el *training set* es el conjunto de datos usados como referencias para seleccionar los pesos asociados a las neuronas y crear las conexiones entre las mismas; el *validation set* corresponde al conjunto de datos usados durante el proceso de entrenamiento para contrastar el desempeño por cada ciclo de aprendizaje, clasificando datos desconocidos; y el *test set* es el conjunto de datos desconocidos para la red neuronal, con los que se prueba el desempeño.

Curva de aprendizaje y medición del error

Cuando la tarea de la ANN es clasificar, el error que se asocia a su capacidad para distinguir entre clases es una función que relaciona la salida que provee la red, con la salida real provista (aprendizaje supervisado).

La curva de aprendizaje refleja el cambio del error a medida que la red neuronal es entrenada. Mediante el análisis de esta curva, es posible determinar qué tantos *epochs* requiere la ANN para llegar a un valor aceptable de error. El cambio en el error depende de la arquitectura de la red, del algoritmo de entrenamiento, de las características del conjunto de entrenamiento y de sus hiperparámetros (parámetros que deben configurarse manualmente, para estructurar el modelo de la red neuronal artificial).

[43] KRIESEL, David. A Brief Introduction to Neural Networks. In: https://doi.org/10.1016/0893-6080(94)90051-5. 244 pp. 2005.

Mediante el análisis de la curva de aprendizaje, es posible detectar, entre otros, dos problemas comunes de las redes neuronales: el *underfitting* y el *overfitting.* Cuando una ANN presenta *underfitting* (ver Figura 4), se dice que, durante su proceso de entrenamiento, no logró abstraer las características necesarias para representar de forma general el conjunto de datos expuesto y su error se mantiene alto. Esto normalmente ocurre debido al hecho de entrenar la ANN con un conjunto reducido de datos, con pocas iteraciones de entrenamiento o una arquitectura de red muy simple para datos ampliamente distribuidos.

Sin embargo, dependiendo de la naturaleza de los datos y de la arquitectura de la red, es común que el error se mantenga estable después de cierta cantidad de *epochs*. Esto representa un riesgo para la capacidad de generalización de la red neuronal, denominado *overfitting* o sobreajuste. Cuando una red neuronal presenta este problema, se presume que "memoriza" el conjunto de entrenamiento, por lo que presenta un desempeño drásticamente inferior para datos de prueba distintos.

En la Figura 4 se puede apreciar un ejemplo de una función con *underfitting* (primera gráfica), seguida de una función con una buena aproximación (*good fit*, segunda gráfica) y por último una función con *overfitting* (tercera gráfica).

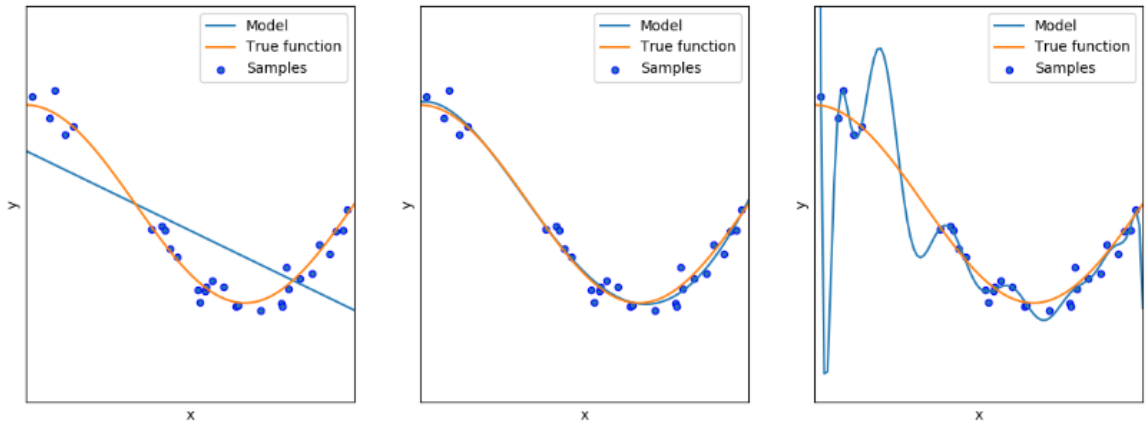


Figura 4. Comparación de los tipos de ajustes que pueden presentarse: a la izquierda underfitting, en la zona del medio good fit y en la zona rececha overfitting. Tomado de: SCIKIT-LEARN PROJECT.Underfitting vs. Overfitting. Scikit-learn 0.19.2 online documentation Available at: http://scikit-learn.org/stable/auto_examples/model_selection/plot_underfitting_overfitting.html. 2017.

El método de cálculo del error mostrado en las curvas de entrenamiento, también denominado *loss function* o *cost function*, influye significativamente en el proceso general de entrenamiento de la red neuronal. Seleccionar la función de error para el entrenamiento de la ANN es una tarea de suma relevancia, debido a que dicha función afecta el proceso de aprendizaje general de la red neuronal.

Optimización de la función de error

El objetivo, en términos prácticos, del entrenamiento de una red neuronal, es proporcionar la capacidad de distinguir los patrones presentes en sus atributos de entrada y asociarlos a una salida específica. Para hacer esto de forma eficiente, es necesario reducir progresivamente el error calculado en cada *batch* (número de muestras que son propagadas a través de la red). Con este fin, son usados comúnmente algoritmos de optimización basados en gradientes.

La optimización de una función consiste en encontrar los valores máximos o mínimos dentro del dominio de la función, como se muestra en la Figura 5. En este proceso, el gradiente es un vector que indica la dirección que se debe seguir para encontrar los valores extremos. En redes neuronales, esto se traduce en la selección de pesos adecuada para encontrar el mínimo (idealmente global) en la función de error.

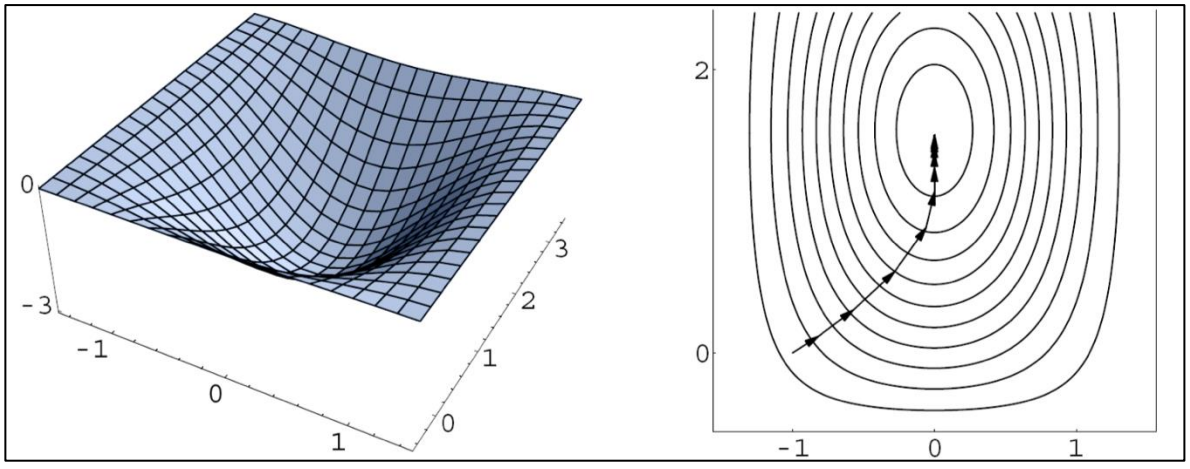


Figura 5. Optimización de una función de dos dimensiones. Tomado de: KRIESEL, David. A Brief Introduction to Neural Networks. In: https://doi.org/10.1016/0893-6080(94)90051-5. p. 62. 2005.

En el proceso de optimización de la función de error, existe un factor denominado *learning rate* o tasa de aprendizaje. Los pesos de las neuronas cambian de forma proporcional a este factor, por lo que se dice que el *learning rate* ayuda a controlar la velocidad de entrenamiento, influenciando los resultados de la función de error.

Los algoritmos conocidos popularmente como *gradient descent*, son algoritmos que recorren la función objetivo en pequeños pasos, empezando en una ubicación inicial (normalmente aleatoria) hacia la dirección estipulada por el gradiente[44]. Este tipo de algoritmos se consolidan como el mecanismo más común para optimizar redes neuronales artificiales.

Existen tres grandes variaciones de *gradient descent* aplicado a redes neuronales artificiales, dependiendo de la cantidad de observaciones que se usan para calcular el gradiente de la función objetivo:

- *Batch gradient descent* (BGD*)*: Con este método se calcula el gradiente de la función de error con todo el *training set*. Esto quiere decir que se procesan todas las observaciones para generar sólo una actualización a los pesos de la red neuronal. Este proceso puede ser considerablemente costoso en términos de complejidad temporal y espacial. Mediante *batch gradient descent* no es posible realizar aprendizaje en línea.
- *Stochastic gradient descent* (SGD): Mediante este método se actualizan los pesos de la red neuronal con cada observación. Por ello, es posible realizar aprendizaje en línea con SGD. Al actualizar los pesos con cada observación, la varianza de los cambios propuestos hace que la función objetivo fluctúe considerablemente.

[44] KRIESEL, David. A Brief Introduction to Neural Networks. In: https://doi.org/10.1016/0893-6080(94)90051-5. 244 pp. 2005.

- *Mini-batch gradient descent*: Este método toma lo mejor de BGD y SGD, al realizar actualizaciones a los pesos tomando grupos de observaciones o *mini-batches*. Esto reduce la varianza de la actualización de parámetros y conduce a la ubicación de los mínimos de forma más estable. Este algoritmo de optimización es el más usado para entrenar redes neuronales y comúnmente se usa el término SGD, también cuando se emplea *mini-batch gradient descent*[45].

Existen diversos algoritmos de optimización de la función de error que usan gradiente para encontrar los mínimos. La convergencia de estos algoritmos para problemas similares difiere en tiempo, debido a los arreglos matemáticos de cada uno de ellos. Algunos de estos algoritmos hacen uso del *learning rate* e inclusive lo modifican a medida que pasan las iteraciones del entrenamiento.

Entre los algoritmos *gradient descent* más usados actualmente en redes neuronales se encuentran:

- *Adagrad*: se trata de un optimizador que adapta el *learning rate* dependiendo de la frecuencia de actualización de los parámetros de la red neuronal. Mientras más actualizaciones recibe un parámetro, más pequeños se tornan estos cambios[46].
- *Adadelta*: este optimizador está basado en *Adagrad*, añadiendo la característica de usar una ventana movible de actualizaciones del gradiente usado para ajustar el *learning rate*, en vez de acumular todos los gradientes

[45] RUDER, Sebastian. An overview of gradient descent optimization algorithms. Cornell University Library. 14 pp. 2017.

[46] DUCHI, John; HAZAN, Elad; SINGER, Yoram. Adaptive Subgradient Methods for Online Learning and Stochastic Optimization. Journal of Machine Learning Research Vol. 12. 2121-2159 pp. 2011.

pasados. Esto permite continuar con el aprendizaje, inclusive después de haber realizado múltiples actualizaciones[47].

- *RMSprop*: de forma similar a *Adadelta*, este algoritmo de optimización intenta resolver el problema de la rápida disminución de los *learning rates* de *Adagrad*. La ventana movible de *RMSprop* mantiene un promedio del gradiente al cuadrado por cada peso[48].
- *Adam*: al igual que *RMSprop* y *Adadelta*, este algoritmo conserva un registro del promedio de los gradientes pasados en forma cuadrática. Además de ello, conserva el promedio de los gradientes pasados en su valor original. Con el contraste de este factor adicional, se consiguen mejoras relativas en la actualización de los pesos por iteración[49].

1.2.1.3. Optimización de hiperparámetros

Como se pudo apreciar en la sección anterior, existen múltiples parámetros que influyen en el modelado de una red neuronal artificial. Los hiperparámetros, deben ser configurados manualmente antes de que el proceso de aprendizaje dé inicio y esta configuración está directamente relacionada con el desempeño general de la red. Entre los hiperparámetros que se detallan con más frecuencia en los modelos de redes neuronales artificiales, se encuentran:

- Cantidad de capas ocultas
- Cantidad de neuronas en las capas ocultas

[47] ZEILER, Matthew. ADADELTA: AN ADAPTIVE LEARNING RATE METHOD. Cornell University Library. 6 pp. 2012.

[48] HINTON, Geoffrey; SRIVASTAVA, Nitish; SWERSKY, Kevin. Overview of mini-batch gradient descent. In: Neural Networks for Machine Learning. Computer Science, University of Toronto. 31 pp. 2018.

[49] KINGMA, Diederik; LEI BA, Jimmy. Adam: a method for stochastic optimization. International Conference on Learning Representations ICLR. Cornell University Library. 15 pp. 2015.

- Función de activación de las neuronas
- *Batch size*: número de muestras que son propagadas a través de la red.
- Cantidad de *epochs*: consiste en la cantidad de ciclos de entrenamiento completos con el conjunto de datos de entrenamiento.
- Función de error
- *Learning rate* inicial: consiste en la tasa de aprendizaje inicial de la red.
- Algoritmo de optimización

Debido a que la configuración de los hiperparámetros afecta directamente el desempeño de una red neuronal existen, entre otras técnicas, tres que son muy utilizadas para encontrar aquellos valores para los cuales se presenta el mejor desempeño de la red. Estas técnicas son: *Manual Search*, *Grid Search* y *Random Search*.

Manual Search

La selección de hiperparámetros puede realizarse manualmente o mediante métodos de selección algorítmicos. En términos generales, para realizar el procedimiento de búsqueda de los valores adecuados de los hiperparámetros asociados a un modelo, es necesario considerar el rango de búsqueda asociado a cada uno de ellos. En el caso del *batch size,* cantidad de *epochs* o *learning rate*, se tiene un conjunto de valores numéricos posibles, mientras que la función de error o el algoritmo de optimización, corresponden a una elección categórica.

Cuando se tiene una búsqueda manual de hiperparámetros, se generan algunos modelos de red neuronal siguiendo todo el proceso de entrenamiento de forma regular. Tras obtener los modelos de prueba, se comparan las métricas características de cada uno de ellos, seleccionando al final el modelo con mejor desempeño.

Grid Search

El proceso de optimización de hiperparámetros usando *grid search* o búsqueda en rejilla, consiste en generar tantos modelos como combinaciones entre hiperparámetros se consideren, generalmente haciendo uso de computación en paralelo.

Al igual que con una búsqueda manual, los hiperparámetros son seleccionados de acuerdo con las métricas de desempeño asociadas a cada modelo. En la implementación de *grid search,* cada modelo asociado a cada combinación puede ser entrenado de forma independiente. Esto posibilita el entrenamiento en paralelo, asociando múltiples unidades de procesamiento, lo que disminuye el tiempo de espera en obtener un resultado[50].

Random Search

En la búsqueda aleatoria o *random search*, se generan modelos en donde los hiperparámetros que los conforman adquieren un valor aleatorio seleccionado dentro de una distribución de posibles valores para cada hiperparámetro. Debido a que el impacto en el desempeño de la red neuronal por parte de cada hiperparámetro varía, no todos se consideran relevantes para optimizar[51].

Otras técnicas más elaboradas y robustas para la optimización de hiperparámetros son: optimización Bayesiana, optimización basada en el gradiente y optimización evolutiva.

---

[50] BENGIO, Yoshua. Practical Recommendations for Gradient-Based Training of Deep Architectures. Cornell University Library. 33 pp. 2012.

[51] BERGSTRA, James; BENGIO, Yoshua. Random Search for Hyper-Parameter Optimization. Journal of Machine Learning Research, Vol. 13. ISSN: 1532-4435. 281-305 pp. 2012.

1.2.1.4. Ventajas de las Redes Neuronales Artificiales

El uso de las redes neuronales artificiales se encuentra en constante expansión en la actualidad, debido a su eficiencia para afrontar problemas complejos en diferentes áreas de estudio, sumado a que las herramientas computacionales modernas permiten una implementación viable de esta técnica. Ya que las redes neuronales artificiales cuentan con la capacidad de aprender a partir de la experiencia e identificar patrones, entre otros, se pueden establecer algunas ventajas con respecto a su uso, como las siguientes:

- *Aprendizaje adaptativo:* capacidad de realizar tareas basándose en criterios adquiridos a partir de un entrenamiento.
- *Auto-organización:* las ANN pueden abstraer una representación de la información y organizarse de acuerdo a ella mediante una etapa de aprendizaje.
- *Tolerancia a fallos:* la composición modular de las ANN permite que, a pesar de sufrir daños parciales, algunas de las capacidades de la red se puedan conservar.
- *Operación en tiempo real:* las operaciones y cálculos por neurona pueden hacerse en paralelo, para ello se requiere un hardware con múltiples núcleos de procesamiento[52].
- *Representación de sistemas no lineales:* dependiendo del número de capas ocultas y de neuronas por capa, las ANN pueden aproximar *no linealidades* a un conjunto de operadores *lineales* que representen los datos[53].

[52] MATICH, Damián. Redes Neuronales: Conceptos Básicos y Aplicaciones. Universidad Tecnológica Nacional, Rosario, Argentina. 55 pp. 2001.

[53] FERNANDEZ, Benito; PARLOS, A.G.; TSAI, W. K. Nonlinear dynamic system identification using artificial neural networks (ANNs). IJCNN International Joint Conference on Neural Networks. DOI: 10.1109/IJCNN.1990.137706. 1990.

1.2.1.5. Evaluación de los modelos de clasificación

La evaluación de los modelos de aprendizaje automático es de vital importancia, para conocer el desempeño de los algoritmos implementados y tomar decisiones coherentes en cuanto a la viabilidad y aplicabilidad de estos, en tareas específicas. Existen métricas distintas para evaluar los algoritmos de *regresión, clasificación, clustering*, etc. A continuación, se presentan las métricas más comunes, aplicadas a modelos de clasificación:

- Matriz de confusión: muestra la cantidad de predicciones correctas y falsas para cada clase contemplada por el clasificador. A modo de ejemplo, se desea hacer una clasificación binaria (dos clases y una neurona de salida) de un evento en: sismo (clase positiva) o ruido (clase negativa). Según esta clasificación, pueden presentarse los siguientes cuatro escenarios:

    1. El clasificador clasifica correctamente un evento sísmico como sismo en lo que se denomina un verdadero positivo (TP).
    2. El clasificador clasifica correctamente una ventana de ruido como ruido en lo que se denomina un verdadero negativo (TN).
    3. El clasificador clasifica erróneamente una ventana de ruido como sismo en lo que se denomina un falso positivo (FP).
    4. El clasificador clasifica erróneamente un evento sísmico como ruido en lo que se denomina un falso negativo (FN).

La matriz de confusión para el ejemplo planteado anteriormente, desde un punto de vista general, se muestra en la Tabla 5, en la que se denotan 2.000 casos de los cuales 977 son verdaderos positivos, 993 verdaderos negativos, 18 falsos positivos y 12 falsos negativos.

Tabla 5. Matriz de confusión general del ejemplo de eventos sísmicos.

		Valor real	
		Positivo	Negativo
Valor predicho	Positivo	977	18
	Negativo	12	993

- *Accuracy* (exactitud): esta métrica mide qué tan cercana está la predicción del clasificador frente al valor esperado. Es la relación entre el número de predicciones correctas y el número total de predicciones:

$$\mathrm{ACC} = \frac{TP + TN}{TP + TN + FP + FN} \qquad \text{(Ecuación 16)}$$

- Precisión*:* hace referencia a la cantidad de observaciones relevantes selectas con respecto al total de observaciones selectas para una clase. También es comúnmente denominada como *valor predictivo positivo* (PPV) y se calcula como:

$$\text{Precisión (PPV)} = \frac{TP}{TP + FP} \qquad \text{(Ecuación 17)}$$

- Sensibilidad o Recall*:* también llamado *exhaustividad* o *Tasa de Verdaderos Positivos* (TPR)*,* hace referencia a la cantidad de observaciones relevantes selectas con respecto al total de observaciones relevantes y se calcula como:

$$\mathrm{TPR} = \frac{TP}{TP + FN} \qquad \text{(Ecuación 18)}$$

- Especificidad*:* también llamado *Tasa de Verdaderos Negativos* (TNR), es un indicador de la capacidad del clasificador para identificar casos negativos. Hace referencia a la cantidad de observaciones negativas selectas con respecto al total de observaciones negativas, calculado como:

$$\mathrm{TNR} = \frac{TN}{TN + FP} \qquad \text{(Ecuación 19)}$$

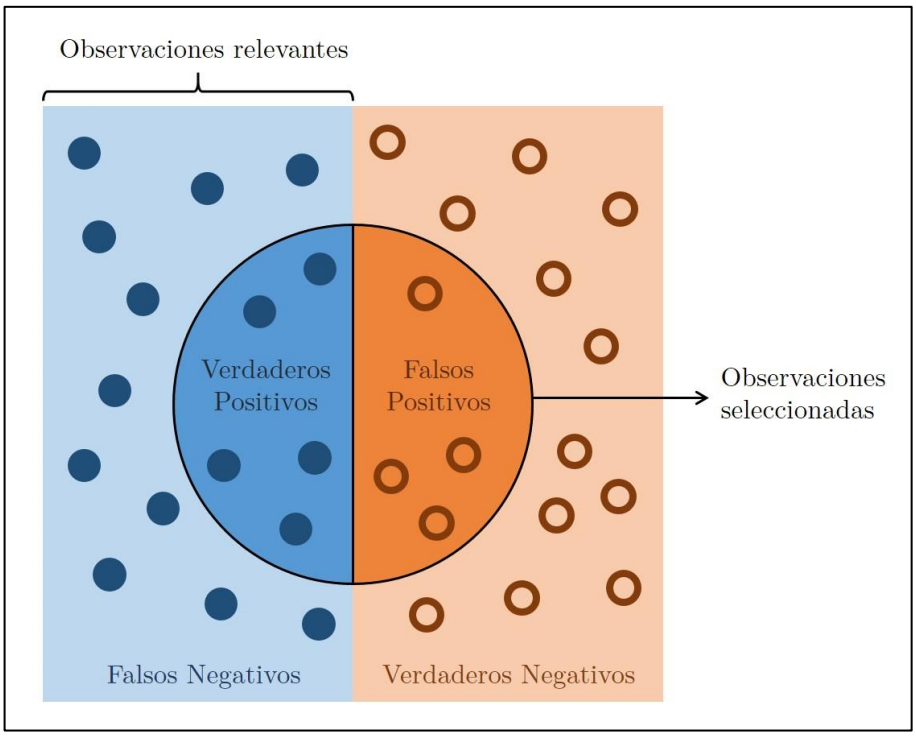


Figura 6. Representación de la clasificación. Adaptado de: WALBER, Matt. Precision and recall [Digital image]. Retrieved from Wikimedia Commons website: https://commons.wikimedia.org/wiki/File:Precisionrecall.svg. (2014).

- F-Score: también llamado medida-F, es una medida del desempeño que relaciona la cantidad de elementos seleccionados relevantes (precisión), con la cantidad de elementos relevantes seleccionados (exhaustividad), mediante su ponderación como media armónica:

$$\mathrm{F}_{\beta} = (1+\beta) \cdot \frac{Precisión \cdot Recall}{(\beta^2 \cdot Precisión) + Recall} \qquad \text{(Ecuación 20)}$$

  Valores del número real β superiores a 1.0 representan una asignación de un peso superior de la exhaustividad sobre la precisión. Para valores inferiores a 1.0, el peso de la ponderación es superior para la precisión, sobre la exhaustividad. Comúnmente se da un peso igual a estas dos variables en lo que se denomina el F1-score o simplemente F1.

Existen otras métricas complementarias a las mencionadas anteriormente, enfocadas en las características de la clasificación con respecto a los resultados negativos, se destacan el *Valor Predictivo Negativo* (NPV*)*, la *Tasa de Falsos Positivos* (FPR) y la *Tasa de Falsos Negativos* (FNR), entre otras[54].

1.2.1.6. Generalización de los modelos de clasificación

Con el fin de desarrollar modelos de clasificación generalizables, es decir, que puedan ser aplicados de forma general sobre diversos datos de entrada y no exista un sesgo o un sobreajuste del modelo, debe tenerse en cuenta la repetibilidad (capacidad de una prueba de ser repetida con iguales resultados) que puede lograrse aplicando dos técnicas: la validación cruzada por *k-fold* y la validación cruzada por *Monte Carlo*.

Repetibilidad

La repetibilidad o reproducibilidad se refiere a la capacidad que tiene una prueba o evento de ser repetida, de tal forma que se obtenga el mismo resultado, bajo las mismas condiciones. Algunos de los factores que pueden afectar la repetibilidad del entrenamiento o la prueba de un clasificador y hacer que el algoritmo de clasificación sea no determinístico[55] son:

[54] AMAZON WEB SERVICES. Evaluating ML Models. *Amazon Machine Learning Developer Guide.* Disponible en: https://docs.aws.amazon.com/machine-learning/latest/dg/evaluating_models.html. 2018.

[55] *def*. Algoritmo no determinístico: es un algoritmo que exhibe un comportamiento distinto cada vez que es ejecutado, incluso para la misma entrada. Esto puede deberse a múltiples razones, entre las que pueden encontrarse las condiciones de carrera y la aleatoriedad del modelo del algoritmo. Fuente: CORMEN, Thomas. Introduction to Algorithms. MIT Press, 3rd Edition. ISBN: 978-0-262-03384-8. 2009.

- Aleatorización de los pesos de las neuronas de entrada, ocultas o de salida: durante la especificación del modelo de la red neuronal, los pesos pueden variar si se utilizan semillas pseudo-aleatorias, lo que hace que la propagación del error sea distinta y por ende el resultado del clasificador.
- Intercambio del conjunto de entrada: si la red neuronal ha sido entrenada para un conjunto de datos de entrada específico, cualquier variación en el conjunto, sea inserción, eliminación o simplemente intercambio de observaciones, causa una variación en la salida del clasificador. Debe tenerse en cuenta también que estos cambios afectan el desempeño y que no es la cantidad de observaciones la variable que más relevancia tiene, pues puede existir un sobreajuste del clasificador que resulte en una disminución de su desempeño.
- Cambios en los algoritmos y librerías del clasificador: las actualizaciones en las librerías pueden afectar el desempeño del clasificador y disminuir la repetibilidad del mismo, pues el entorno del clasificador cambia, ya que los procesos que conllevan al resultado han sido variados.
- Pérdida de cifras decimales: para el cálculo de los pesos se tienen en cuenta cifras significativas que pueden variar y perderse en cada iteración, lo que hace que el error se vaya propagando y variando el resultado cada vez que es ejecutado de nuevo el procedimiento.
- Ejecución en multi-hilo: para abreviar el tiempo de ejecución del entrenamiento de un clasificador, se puede repartir una o varias tareas diferentes hilos. Esto puede causar errores en el cálculo de las variables, producto de condiciones de carrera que conlleven a la sobre-escritura de las mismas.

Validación Cruzada (CV)

La validación cruzada o *cross-validation* es una prueba imparcial que permite garantizar que los resultados estadísticos de un clasificador sean independientes de la porción de datos de entrada en el entrenamiento y la prueba, siempre y cuando las porciones sean extraídas del mismo conjunto general y se evite el sesgo de otras

fuentes, sean instrumentales o humanas[56]. Esta prueba consiste en registrar las métricas de desempeño cada vez que se hace una clasificación usando una porción distinta de los datos. El resultado es expresado como la media aritmética de las métricas encontradas en cada iteración de la prueba.

Existen dos tipos de validación cruzada: la validación exhaustiva, en la que se prueba haciendo un porcionamiento del conjunto de datos en subconjuntos de entrenamiento y validación de todas las formas posibles; y la no exhaustiva, en la que se dejan combinaciones de porciones del conjunto de datos sin validar y se hace una validación con particiones específicas. La validación cruzada exhaustiva es costosa computacionalmente tanto en recursos como en tiempo, razón por la cual suele abordarse una validación cruzada no exhaustiva. Dentro de esta última se encuentra la validación cruzada: por *k* iteraciones, por sub-muestreo aleatorio, aleatoria y dejando uno fuera.

En esta primera, la validación cruzada por *k* iteraciones, el conjunto de datos es partido en porciones de datos mutualmente excluyentes o disjuntos (que no hay solapamiento) que se denominan *folds*. Una validación en la que la cantidad de particiones finitas *k* está definida, se denomina *k-fold Cross Validation*. Por ejemplo, considérese el diagrama mostrado en la Figura 7, que corresponde a un *k-fold Cross Validation*, en el que el coeficiente *k* es 10. Esto significa que el conjunto de datos de entrada es partido en 10 subconjuntos, o *folds*, y estos a su vez son divididos en los conjuntos de entrenamiento, en color azul, y prueba, en color amarillo.

Para cada entrenamiento y prueba con cada *fold*, se obtiene una métrica del desempeño del clasificador. Estas métricas son promediadas y al finalizar el proceso de validación cruzada, se obtiene una métrica general promedio del

[56] EFRON, Bradley; TIBSHIRANI, Robert. "Improvements on cross-validation: The .632 + Bootstrap Method". Journal of the American Statistical Association. Vol. 92 (438). 548–560 pp. DOI: 10.2307/2965703. JSTOR 2965703. MR 1467848. 1997.

desempeño para diversas porciones del conjunto de entrada, aumentando la repetibilidad del clasificador y haciéndolo más generalizable. El error de la validación con *k* iteraciones es el promedio del error tras cada iteración[57].

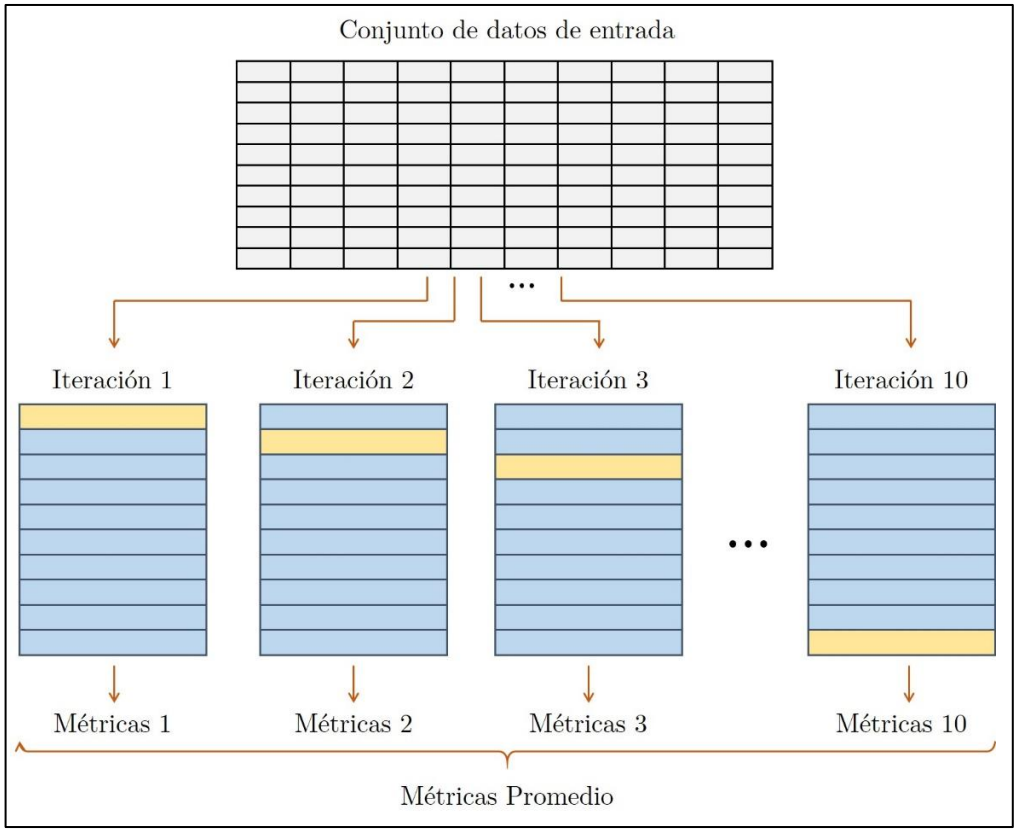


Figura 7. Esquema de 10-*fold Cross Validation* con un clasificador, en el que los subconjuntos de entrenamiento están en color azul y los de validación en color amarillo. Adaptado de: CASTELLANO, Pablo. K-fold Cross Validation [Digital image]. Retrieved from Wikimedia Commons website: https://commons.wikimedia.org/wiki/ File:Esquema_castellà.jpg. (2014).

A diferencia del *k-fold Cross Validation*, la validación por sub-muestreo aleatorio o Validación Cruzada por Monte Carlo (MCCV), separa el conjunto de datos en sub-conjuntos aleatorios para el entrenamiento y la validación, siendo los conjuntos mutualmente no excluyentes, razón por la cual puede presentarse repetición de algunas muestras en los distintos sub-conjuntos de datos separados. Las métricas

[57] VARMA, Sudhir; SIMON, Richard (2006). "Bias in error estimation when using cross-validation for model selection". BMC Bioinformatics. Vol. 7. DIO: 10.1186/1471-2105-7-91. 2006.

obtenidas tras cada aplicación de la validación sobre el sub-conjunto de datos son promediadas para obtener métricas generales como resultado.

La ventaja de esta técnica sobre las anteriores es que las porciones de entrenamiento y validación no dependen de un número de iteraciones, sin embargo, puede que algunas observaciones nunca sean seleccionadas y otras pueden ser seleccionadas más de una vez, en lo que se denomina el solapamiento. Esto conlleva a que los resultados obtenidos tras varias repeticiones sean diferentes. Por estas razones, la validación cruzada por *k-fold* presenta menor sesgo y alta varianza, mientras que la validación cruzada por Monte Carlo presenta alto sesgo y menor varianza.

Dietterich[58], 1998, en "*Approximate Statistical Tests for Comparing Supervised Classification Learning Algorithms"*, propone un test *t* de validación cruzada 5x2, en la que se hacen 5 iteraciones de validación cruzada por Monte Carlo sobre 2-*fold Cross Validation* (5x2 CV *Test*), combinando las dos técnicas de validación cruzada. Grellier[59], 2018, hace una prueba de 5x2 CV *Test* frente a un 10-*fold* CV, analizando los resultados siguiendo una distribución *t* de 5 grados de libertad. El resultado obtenido por el 5x2 CV *Test* es ligeramente superior al del 10-*fold* CV.

1.3. COMPLEJIDAD TEMPORAL

La complejidad temporal describe el comportamiento en tiempo de un algoritmo, relacionado con su gasto computacional, desde que empieza a ejecutar una tarea, hasta que cumple su objetivo. En otras palabras, identifica el tiempo que le toma al algoritmo la ejecución de una tarea, a medida que la entrada es variada. El tiempo puede medirse sustrayendo la fecha y hora en la que la tarea se empieza a ejecutar,

[58] DIETTERICH, Thomas. Approximate Statistical Tests for Comparing Supervised Classification Learning Algorithms. The MIT Press, Vol. 10, No. 7. ISSN: 0899-7667.1895-1923 pp. 1998.

[59] GRELLIER, Oliver. Parameter tuning : 5 x 2-fold CV statistical test. [Last query: 25 August, 2018]. Available at: https://www.kaggle.com/ogrellier/parameter-tuning-5-x-2-fold-cv-statistical-test. 2018.

a la fecha y la hora en la que termina. Sin embargo, deben considerarse tres casos de entrada:

- Un primer caso en el que se define una entrada con la que se tenga certeza de que el algoritmo va a tener su mejor comportamiento (mejor caso).
- Un caso promedio en el que se define una entrada tal que el algoritmo mantenga un comportamiento regular al visto normalmente (caso promedio).
- Un caso extremo en el que se defina una entrada con la que se tenga certeza de que el algoritmo va a tener su peor comportamiento (peor caso).

La curva de complejidad temporal para una cantidad de elementos de entrada contra el tiempo que tarda en ejecutar la tarea con estos elementos suele expresarse como una aproximación del comportamiento de una función conocida que esté ligeramente por encima o por debajo de la curva resultante de complejidad. Entre las funciones conocidas, se encuentran:

- La función constante $f(n) = k$
- La función lineal $f(n) = k \cdot n$
- La función cuadrática $f(n) = k \cdot n^2$
- La función cúbica $f(n) = k \cdot n^3$
- La función logarítmica $f(n) = log(n)$
- La función poli-logarítmica $f(n) = (log(n))^k$
- La función polinómica $f(n) = n^k$
- La función exponencial $f(n) = k^n$
- La función factorial $f(n) = n!$

Por ejemplo, considérese un algoritmo que tiene como objetivo el ordenamiento de un conjunto finito de datos numéricos que va cambiando en tamaño desde 2 elementos, hasta *n* elementos. Para cada conjunto de elementos se registra el tiempo en milisegundos que tarda el algoritmo en ejecutar el ordenamiento mostrado en el Algoritmo 1, lo que puede ser descrito como una función que relaciona la cantidad de elementos *n* contra tiempo *t* de forma generalizada con la notación *Big*

O: $O(n^2)$. Esta notación permite describir el comportamiento de una función cuando la variable dependiente (tiempo) tiende a un comportamiento conocido.

Algoritmo 1. Pseudocódigo del ordenamiento por el método de la Burbuja

Entrada: conjunto de elementos a ordenar S
Salida: conjunto de elementos ordenados S

```
1.  i ← 1
2.  ordenado ← no
3.  mientras (i < n) y (ordenado = no) hacer
4.      i ← i + 1
5.      ordenado ← sí
6.      para j ← 0 hasta n – i hacer
7.          si S(j) > S(j+1) entonces
8.              ordenado ← no
9.              temp ← S(j)
10.             S(j+1) ← temp
11.         fin si
12.     fin para
13. fin mientras
```

La complejidad temporal puede depender de diversos factores, entre los que se encuentran: la eficiencia del algoritmo, el lenguaje de programación en el que se encuentra implementado, el compilador usado para convertir el código escrito, el sistema operativo, la cantidad de memoria disponible para atender los requerimientos y la cantidad de recursos computacionales adicionales disponibles frente a los requeridos. Así, para hacer dos o más algoritmos comparables en su comportamiento en tiempo, debe garantizarse que las características del entorno en el que se encuentran los algoritmos sean las mismas.

Para comparar dos algoritmos por su eficiencia, se pueden comparar las funciones de complejidad temporal y evidenciar cuál es la función inferior y cuál es la superior. Por ejemplo, considérese el caso en el que deba desarrollarse un algoritmo para hacer la suma de los primeros *n* enteros positivos. Para dar solución a esta situación, el problema puede abordarse desde tres puntos de vista: el primero consiste hacer una suma unitaria por número (ver Algoritmo 2), el segundo consiste

en hacer una suma directa de cada número (ver Algoritmo 3) y el tercero en hacer uso de la ecuación con la que puede abreviarse la sumatoria. El primero algoritmo presenta una complejidad temporal de $O(n^2)$, el segundo de $O(n^2)$ y el tercero de $O(1)$, haciendo que los tres algoritmos puedan ser comparados en su demanda computacional en tiempo y pueda escogerse el más eficiente de ellos.

Algoritmo 2. Pseudocódigo 1 para la suma de n primeros números enteros

Entrada: variable n
Salida: variable *resultado*

```
1.  def suma_lenta(n)
2.      resultado = 0
3.      para i ← 1 hasta n hacer
4.          para j ← 1 hasta i+1 hacer
5.              resultado += 1
6.          fin para
7.      fin para
8.      retornar resultado
```

Algoritmo 3. Pseudocódigo 2 para la suma de n primeros números enteros

Entrada: variable n
Salida: variable *resultado*

```
1.  def suma_promedio(n)
2.      resultado = 0
3.      para i ← 1 hasta n hacer
4.          resultado += i
5.      fin para
6.      retornar resultado
```

Algoritmo 4. Pseudocódigo 3 para la suma de n primeros números enteros

Entrada: variable n
Salida: variable *resultado*

```
1.  def suma_rapida(n)
2.      resultado = n * (n + 1) // 2
3.      retornar resultado
```

1.4. METODOLOGÍA DE DESARROLLO DE SOFTWARE SCRUM

En vista de que la situación problema está enmarcada en el dominio complejo, aludiendo al marco de trabajo Cynefin presentado Snowden y Bone[60], y que el presente proyecto involucra el trabajo interdisciplinar y el desarrollo de un sistema de información, se contextualiza acerca de la metodología Ágil Scrum. En este contexto es recomendado el empleo de tácticas ágiles, ya que la implementación de prácticas emergentes y el cambio dinámico de los parámetros de desarrollo son necesarias para obtener resultados satisfactorios en tiempos fijos estipulados para la producción de software. Por esta razón se considera que las metodologías Ágiles, más específicamente Scrum, son convenientes para el proceso de desarrollo del sistema propuesto.

Scrum se consolida actualmente como el marco de trabajo que aplica los principios del manifiesto ágil más usado a nivel global. Los valores que estipula la *Agile Alliance* plasmados en el manifiesto ágil, son esencia del éxito de Scrum y de las metodologías ágiles en general[61]. Estos valores son los siguientes:

1. Individuos e interacciones por sobre procesos y herramientas.
2. Software funcionando por sobre documentación exhaustiva.
3. Colaboración del cliente por sobre la negociación de contratos.
4. Respuesta al cambio por sobre el seguimiento de un plan.

[60] SNOWDEN, David; BOONE, Mary. A Leader's Framework for Decision Making, Harvard Business Review. 2007.

[61] BECK, Kent; BEEDLE, Mike; BENNEKUM, Arie Van; COCKBURN, Alistair; CUNNINGHAM, Ward; FOWLER, Martin; GRENNING, James; HIGHSMITH, Jim; HUNT, Andrew; JEFFRIES, Ron; KERN, Jon; MARICK, Brian; MARTIN, Robert; MELLOR, Steve; SCHWABER, Ken; SUTHERLAND, Jeff; TGOMAS, Dave. Manifesto dor Agile Software Development. 2001.

Según estos valores, se puede apreciar que la cohesión entre el equipo asociado al proyecto es un factor determinante para garantizar un avance rápido con respecto a lo que se podría esperar mediante el enfoque tradicional de las metodologías secuenciales. Con una comunicación efectiva entre todos los integrantes del equipo, es viable planificar de forma dinámica la ejecución de las tareas propias del desarrollo del proyecto.

Por este motivo, la definición de los roles que establece Scrum, junto con sus tareas respectivas, es de suma importancia para la eficiencia en general del conjunto de métricas y recomendaciones que establece el marco de trabajo como tal. Los roles adoptables en Scrum son los siguientes:

- *Stakeholders*: son los clientes o usuarios finales del software.
- *Product Owner*: es le persona responsable del éxito del producto desde el punto de vista de los *stakeholders*.
- Equipo de desarrollo: está conformado por las personas responsables de la construcción del software.
- *ScrumMaster*: juega el papel de coach del equipo, procurando ayudarlo a alcanzar el mejor desempeño posible.

Si bien cada uno de estos roles tiene actividades distintas, la información del estado del proyecto debe ser visible en todo momento para cualquiera de ellos. Así mismo, el aporte intelectual y el *feedback* de cada uno de los integrantes es muy valorado, por lo que esta metodología incita al equipo a tener una estructura jerárquica transversal, creando un clima de confianza propicio para el aumento de la producción.

Entre los parámetros que sugiere Scrum para el desarrollo, además de la conformación del equipo con roles marcados, se encuentran los elementos útiles para establecer las normas concernientes al desarrollo del software. El *Product*

Backlog es la columna vertebral de la organización de tareas en Scrum. La filosofía de éste consiste en partir los segmentos necesarios para llevar a cabo el producto en la mayor cantidad de tareas independientes posibles. Estos segmentos son llamados PBI (*Product Backlog Items*), que pueden ser comparados con el concepto de actividades dentro de un cronograma tradicional[62].

Ya que el levantamiento de requerimientos se considera fundamental para cualquier proyecto de desarrollo de software, Scrum plantea un método para determinar lo que se necesita desarrollar, enfocándose en la perspectiva del usuario en cuanto a sus necesidades. A este formato particular de requerimiento se le denomina Historia de Usuario[63].

Cada Historia de Usuario está conformada por múltiples PBI, los cuales son organizados de manera jerárquica en el *Product Backlog*, en donde se priorizan las características más urgentes para lograr el Producto Mínimo Viable (MVP). Esta estructura apunta a la satisfacción de la principal métrica de avance en Scrum: el software funcionando.

Debido a ello, es comúnmente usado el concepto de *sprint*, el cual consta de un conjunto de PBI asociados a diferentes Historias de Usuario destinados a ser desarrollados en un plazo determinado de tiempo, usualmente menor a 4 semanas. Se pretende que cada *sprint* tenga características funcionales de alta prioridad y de gran valor para los *Stakeholders*, con el fin de obtener el MVP rápidamente.

---

[62] ALAIMO, Martin; SALIAS, Martin. Proyectos Ágiles con Scrum: Flexibilidad, aprendizaje, innovación y colaboración en contextos complejos. Ediciones Kleer, Buenos Aires, Argentina. ISBN 978-987-45158-1-0. 2013.

[63] COHN, Mike. User Stories Applied For Agile Software Development. Addison-Wesley and Pearson Education Incorporated. Boston, Massachussets. ISBN: 0-321-20568-5. 2004.

La variable fija que se tiene para la estimación de los PBI y los *sprints* es el tiempo de entrega. Ya que el tiempo de las entregas no se debe cambiar, el alcance de los *sprint* y el costo de ciertos PBI puede ser modificado. Esto genera una capacidad de desarrollo iterativa y evolutiva en cuanto a las características del producto.

Como vía de comunicación principal entre el equipo de desarrollo y los demás miembros del equipo, se plantean una serie de reuniones frecuentes. Los objetivos primordiales de estas reuniones consisten en identificar los PBI y construir el *Product Backlog*, planear y revisar los *sprints*, evaluar la calidad del trabajo en equipo a partir de la implementación de Scrum y reportar avances o impedimentos con respecto a PBI e Historias de Usuario específicas[64].

El dinamismo que aporta Scrum al desarrollo es soportado por la frecuencia de las reuniones entre los diferentes involucrados en el proyecto. Siempre es posible incluir, eliminar o modificar un PBI para conseguir un resultado específico acordado. La evaluación del resultado de los cambios hechos y la toma de decisiones a partir de los mismos se genera de manera rápida y efectiva producto de una comunicación abierta y constante del equipo, y a las características de las reuniones propuestas.

[64] ALAIMO, Martin; SALIAS, Martin. Proyectos Ágiles con Scrum: Flexibilidad, aprendizaje, innovación y colaboración en contextos complejos. Ediciones Kleer, Buenos Aires, Argentina. ISBN 978-987-45158-1-0. 2013.

2. ANTECEDENTES

El interés por el estudio de los sismos se remonta a cientos de siglos atrás, cuando los chinos ya describían, hace más de 3.000 años, el impacto de los movimientos telúricos. Griegos y Romanos documentaban a su vez el impacto de estos movimientos en la destrucción de construcciones arquitectónicas y ciudades enteras. Los Mayas y los Aztecas, se refieren a este fenómeno natural detallando los principales eventos que afectaron las regiones americanas. En un principio para estas civilizaciones se trataba de fenómenos de respuesta mítica, relacionado con creaturas fantásticas que vivían al interior de la Tierra y que al ejercer su movimiento provocaban este tipo de movimientos superficiales[65].

Uno de los primeros autores en publicar información sobre el estudio de los sismos fue Aristóteles, quien abandona las explicaciones mitológicas y postula que los movimientos terrestres se deben al efecto de la circulación de vientos muy fuertes al interior de la Tierra. A inicios del siglo XVII, Vicenzo Magnati[66], en 1688, fue uno de los primeros historiadores en recopilar información sobre estos movimientos telúricos, elaborando una lista de 91 sismos que causaron daños destructivos durante el periodo desde el 34 hasta 1687 d.C. Algunos artículos escritos entre 1700 y 1800 donde se documentaban estos hechos sísmicos fueron desprestigiados debido a la percepción poco objetiva de los autores, y a su explicación poco creíble de los hechos ocurridos.

[65] SERVICIO GEOLÓGICO MEXICANO. Causas, Características e Impactos de los Sismos. Secretaría de Economía de México. Disponible en: http://portalweb.sgm.gob.mx/museo/riesgos/sismos. 2013.

[66] CARCEDO AYALA, Fabián. Manual de Ingeniería Geológica. Ministerio de Industria y Energía, Instituto Tecnológico GeoMinero de España. 2005. 626 pp.

Con el avance de los desarrollos tecnológicos en los medios de transmisión de información, llegaría la invención del telégrafo en 1840, lo que posibilitaría comunicar los informes sismológicos de manera más eficiente, acelerada y con mayor facilidad. Más tarde, con la masificación de las líneas telefónicas dedicadas, se crearon redes de monitoreo sísmico alrededor de todo el mundo, que fueron mejorando su infraestructura a medida que la tecnología en materia de comunicaciones iba creciendo. Actualmente, este proceso de transmisión es realizado vía satélite hacia las estaciones encargadas, que demodulan las señales extrayendo los datos que incluyen el tiempo y la localización de cada evento.

Como consecuencia de estos numerosos estudios informales y muy variados sobre los fenómenos físicos sismológicos ocurrentes en la época, nació la ciencia de estudio de sismos: la Sismología. Se puede considerar como punto de partida de esta ciencia moderna el 1 de noviembre de 1755, día en el cual una sucesión de movimientos telúricos de gran fuerza sacudió a Lisboa (Portugal) en lo que se conoce como "el gran terremoto de Lisboa"[67], provocando graves daños estructurales en la cuidad y en el puerto.

Con el nacimiento de esta ciencia, explicaciones poco trascendentales e irregulares como las que Aristóteles, Plinio y otros antiguos historiadores y matemáticos plantearon, fueron desestimadas y nació la documentación de sismos en función de las fechas y la ocurrencia de los estos fenómenos. Fueron Cauchy, Poisson, Stokes y Rayleigh[68], durante el nacimiento del siglo XIX quienes propusieron una

[67] DÁVILA MADRID, Ramón. Notas Introductorias en Sismología. Posgrado en ciencias de la Tierra, Centro de Geociencias, Universidad Autónoma de México. 2011. 36 pp.

[68] SERVICIO GEOLÓGICO MEXICANO. Causas, Características e Impactos de los Sismos. Secretaría de Economía de México. Disponible en: http://portalweb.sgm.gob.mx/museo/riesgos/sismos. 2013.

descripción profunda de la propagación de las ondas elásticas en los materiales sólidos, detallando el comportamiento de las ondas superficiales y su clasificación.

Giuseppe Mercalli (1833)[69] elaboró una lista de más de 5.000 terremotos desde 1450 hasta 1881. Fue el primer autor en datar estos fenómenos naturales dando una explicación teórica y física, y proponiendo una escala de aceleración sísmica que es usada en la actualidad para clasificar los eventos sismológicos con base en esta componente. En 1935, Charles Francis Richter[70], sismólogo y físico americano del California Institute of Technology (CALTECH), junto a Beno Gutenberg, propusieron la escala logarítmica de Richter.

Actualmente se utilizan técnicas matemáticas y estadísticas más elaboradas que además de permitir el análisis de los eventos históricos que han acontecido, permiten el uso de esta información para la detección de futuros eventos sísmicos en un estado en el que el sismo está por arribar a la superficie. El uso de herramientas computacionales modernas como los modelos de Markov, la estadística Bayesiana, las máquinas de soporte vectorial y las redes neuronales artificiales, han posibilitado un estudio más profundo de estos eventos.

A continuación, se hace una descripción de investigaciones enmarcadas en el aprendizaje automático para la detección de sismos, agrupadas por las técnicas usadas, haciendo énfasis en aquellas que aplican las redes neuronales artificiales.

[69] SÁNCHEZ, Francisco. Los Terremotos y sus Causas. Instituto Andaluz de Geofísica y Prevención de Desastres Sísmicos. España. 2007. 24 pp.

[70] SERVICIO GEOLÓGICO MEXICANO. Causas, Características e Impactos de los Sismos. Secretaría de Economía de México. Disponible en: http://portalweb.sgm.gob.mx/museo/riesgos/sismos. 2013.

2.1. Modelos Ocultos de Markov (HMM)

En la investigación "Constructing a Hidden Markov Model based earthquake detector: application to induced seismicity" realizada por Beyreuther[71] et al. en Indonesia (2012) se usó la técnica de *Modelos Ocultos de Markov* (HMM) en conjunto con *Clustering de Estados* para mejorar el desempeño en tiempo de los HMM. El objetivo de dicha investigación fue reducir los niveles de fallos en la detección sísmica en redes sismológicas de poca densidad, apuntando a las estructuras existentes de monitoreo de volcanes y plantas geotérmicas de producción energética.

El sistema de detección propuesto fue implementado en una estación de monitoreo de una planta geotérmica ubicada en Indonesia, durante un periodo de 3.9 meses. Su desempeño en detección fue similar al mostrado por un conjunto de dos estaciones con el sistema tradicional de monitoreo. Además, se probó el método de detección con datos sísmicos del volcán Mt. Merapi, reafirmando las ventajas del sistema desarrollado en escenarios de monitoreo sísmico mediante una sola estación. Los resultados encontrados se presentan de forma cualitativa y no se presentan: la población escogida, la cantidad de observaciones y los datos cuantitativos que muestren el desempeño de los modelos.

[71] BEYREUTHER, Moritz; HAMMER, Conny; WASSERMANN, Joachim; OHRNBERGER, Matthias; MEGIES, Tobias. Constructing a Hidden Markov Model based earthquake detector: application to induced seismicity. Geophysical Journal International, Vol. 189, issue 1, 602–610 pp. ISSN 0956-540X. In: https://doi.org/10.1111/j.1365-246X.2012.05361.x. 2012.

2.2. Redes Bayesianas (DBN)

En el trabajo “A Machine Learning Approach for Improving the Detection Capabilities at 3C Seismic Stations”, C. Riggelsen y M. Ohrnberger[72] (2012), usan la técnica de *Redes Bayesianas Dinámicas* (DBN) y algoritmos de clasificación, para detectar la ocurrencia de un sismo, mediante la clasificación de los tramos de señal en las categorías: ruido o sismo. La fuente de datos usada para dicho procedimiento provino de las estaciones internacionales BOSA, ubicada en Boshof y Sudafrica, y LPAZ, ubicada en La Paz, Bolivia; ambas vinculadas al Servicio Geológico de los Estados Unidos (USGS). Con los registros conocidos de dichas estaciones, les fue posible obtener un desempeño de entre 85 y 97% en la clasificación binaria propuesta.

2.3. Máquinas de Soporte Vectorial (SVM)

Con la técnica de *Máquinas de Soporte Vectorial*[73] (SVM) de Ruano et al. (2014), consideraron las señales provenientes de la estación PVAQ de la red sismográfica del Instituto de Meteorología de Portugal (IM), con el que obtuvieron entre un 97.7 y 98.7% de sensibilidad y especificidad. Al exponer el sistema a los datos de prueba de otras estaciones diferentes del IM, el clasificador mostró entre un 88.4% y 99.4% de sensibilidad y especificidad, además de obtener un tiempo de procesamiento de 1.3 segundos en el mejor de los casos lo que, según los autores, lo hace aplicable a sistemas de alerta temprana. Vale la pena resaltar que en un enfoque inicial se

[72] RIGGELSEN, Carsten; Ohrnberger, Matthias. A Machine Learning Approach for Improving the Detection Capabilities at 3C Seismic Stations. Pure and Applied Geophysics, Vol. 171, issues 3-5, 395-411 pp. ISSN 0033-4553. In: https://doi.org/ 10.1007/s00024-012-0592-3. 2012.

[73] RUANO, A.E.; MADUREIRA, G.; BARROS, O.; KHOSRAVANI, H.R.; RUANO, M.G.; FERREIRA, P.M. Seismic detection using support vector machines. Elsevier Neurocomputing, Vol. 135, No. 5, 273-283 pp. ISSN 0925-2312. In: https://doi.org/10.1016/j.neucom.2013.12.020. 2014.

usó la técnica de *Perceptrones Multi-Capa* (MLP), pero fue descartada al presentar un menor desempeño con respecto al obtenido con SVM.

2.4. Redes Neuronales Artificiales (ANN)

Ibs-von Seht[74], en su trabajo "Detection and identification of seismic signals recorded at Krakatau volcano (Indonesia) using artificial neural networks" (2008), utiliza una *Red Neuronal Artificial de Feedforward* (ANN) para clasificar los eventos sísmicos del volcán Anak Krakatu de Indonesia. En dicha clasificación se estipularon 6 categorías: sismo de tipo volcán-tectónico, sismo de largo periodo, temblor, ruido de alta frecuencia y ruido de baja frecuencia. Se usaron los siguientes atributos para la clasificación: duración de la ventana umbral, la impulsividad del ruido, las frecuencias dominantes, el espectro de frecuencia total y el espectrograma del evento.

El monitoreo de la actividad fue ejecutado entre junio de 2005 y marzo de 2007, tomando más de 10.000 eventos que al final fueron representados en 924 observaciones de una estación, que se utilizaron para entrenar la red (462 observaciones) y validarla (462 observaciones). Las señales fueron filtradas digitalmente con un filtro pasa bajos de 1 Hz. La cantidad de identificaciones correctas de la red neuronal utilizada está entre un 80% y 97% del total de observaciones con una red neuronal con 120 neuronas en la primera capa oculta. Los autores concluyen que el aparente mal desempeño de la red pudo deberse a que las observaciones escogidas a partir de las muestras tomadas durante los dos años pudieron no haber sido representativas de todas las clases propuestas.

[74] IBS-VON SEHT, M. Journal of Volcanology and Geothermal Research. Detection and identification of seismic signals recorded at Krakatau volcano (Indonesia) using artificial neural networks. Hanover, Germany. 9 pp. 2008.

Hassan et al. presentan el uso de técnica MLP en su investigación denominada "Seismic Signal Classification using Multi-Layer Perceptron Neural Network"[75] (2013). La clasificación se hizo en cuatro clases: sismo local, sismo regional, sismo por explosión o sismo por maquinaria. Los atributos considerados para la clasificación de las señales sísmicas son: similitud en la envolvente, duración, centroide espectral, longitud del espectro y oblicuidad (*skewness*).

Los datos de entrada usados para este proyecto fueron proporcionados por 5 estaciones sísmicas monoaxiales instaladas alrededor de la ciudad de Agadir, Marruecos. El rendimiento de la red fue evaluado en términos de cuatro aspectos: sensibilidad, especificidad, precisión y error cuadrático medio, usando un conjunto de 343 observaciones para el entrenamiento del algoritmo. Se evaluaron 13 topologías en las que se variaba la cantidad de neuronas en la capa oculta, obteniendo el mejor resultado con 5 neuronas. A pesar de haber obtenido una precisión superior al 90% en la clasificación, los autores consideran que es necesario aumentar la confiabilidad de las características obtenidas para representar las señales, mediante la ampliación del conjunto de datos utilizados y un análisis más profundo de las señales pertenecientes a cada categoría. Se resalta que no se dio detalle de las características de los procesos de filtrado de las ondas.

Asimismo, Giudicepietro et al[76], en "Fast Discrimination of Local Earthquakes using a Neural Approach" (2017) busca la clasificación sismos, mediante una red neuronal MLP, en tres clases: sismos locales, regionales y telesismos. El estudio se plantea haciendo uso de los registros sísmicos de la estación sísmica SGG, operada por el

[75] AITLAASRI, El Hassan; AKHOUAYRI, Es-Saïd; AGLIZ, Driss; ATMANI, Abderrahman. Seismic Signal Classification using Multi-Layer Perceptron Neural Network. International Journal of Computer Applications, Vol. 79, No. 15. ISSN 0975-8887. 2013.

[76] GIUDICEPIETRO, Flora; ESPOSITO, Antonietta; RICCIOLINO, Patrizia. Fast Discrimination of Local Earthquakes Using a Neural Approach. Seismological Research Letters, Vol. 88, No. 4, 1089-1096 pp. In: https://doi.org/10.1785/0220160222. 2017.

Osservatorio Vesuviano (INGV), que se encuentra cerca de San Gregorio Matese, un pueblo situado en una zona sísmicamente activa en el sur de los Apeninos, Italia.

El algoritmo fue entrenado con 315 observaciones (5/8 del conjunto total de datos) y validado con 189 (3/8 del conjunto de datos) todos de una única estación, ejecutando una reducción de dimensionalidad por LPC (*Linear Predictive Coding*), en una ventana de detección de 1 segundo. La red neuronal posee 5 nodos en su capa oculta, funciones de activación de tangente hiperbólico para las unidades ocultas y sigmoide logística para los nodos de salidas. Mediante el uso de MLP se logró una correcta clasificación de entre el 97,7% y 98,5%, discriminando entre los resultados de las comparaciones de sismos locales contra sismos regionales y sismos locales contra telesismos respectivamente. Cuando la ventana fue ampliada a 4 segundos, se obtuvo una correcta clasificación del 99,49% comparando telesismos contra sismos locales. Los autores concluyen que el método propuesto es apto para sistemas de monitoreo y de alerta temprana. Cabe resaltar que no se registran los atributos tenidos en cuenta para la clasificación.

Vallejos y McKinnon[77] plasman en su investigación denominada “Logistic regression and neural network classification of seismic records” (2013) el uso tanto de *Regresión Logística*, como de *Redes Neuronales Artificiales*, para la clasificación de eventos micro sísmicos provenientes de dos minas ubicadas en Ontario, Canadá. Las categorías usadas fueron explosiones, eventos sísmicos y eventos reportados. De forma inicial se usaron las categorías explosiones y eventos sísmicos, obteniendo métricas de rendimiento similares para ambas técnicas, pero al introducir la tercera categoría (eventos reportados), la *Regresión Logística* presento un desempeño superior. En términos generales, para ambas técnicas de

[77] VALLEJOS, J.A.; MCKINNON, S.D. Logistic regression and neural network classification of seismic records. International Journal of Rock Mechanics and Mining Sciences, Vol. 62, 86-95 pp. In: https://doi.org/10.1016/j.ijrmms.2013.04.005. ISSN 1365-1609. 2013.

aprendizaje, se logró una precisión superior al 95% en cuanto a la clasificación de los eventos. No se registran la cantidad de muestras utilizadas para los procesos de entrenamiento y prueba, y la arquitectura de la red.

Un enfoque más específico lo exponen Kaur et al[78] en su investigación denominada "Detection and Identification of Seismic P-Waves using Artificial Neural Networks" (2013), el uso *de Redes Neuronales de Propagación hacia atrás* (BPNN) para la detección de la onda P de un conjunto de sismos locales y regionales cuyos datos fueron provistos por la Organización Central de Instrumentos Científicos (CSIO) en Chandigarh, India.

Los atributos usados y tomados a partir de sismogramas triaxiales fueron: grado de polarización (DOP), coeficiente de auto regresión (ARC), relación entre el tiempo corto promedio y el tiempo prolongado promedio (STA/LTA) y la proporción de potencia vertical sobre potencia total (RV2T). Se hizo uso de 60 sismogramas para el entrenamiento clasificados equitativamente en dos clases: onda P o ruido; y 100 sismogramas de validación de la red neuronal. Los sismogramas de onda P fueron anotados teniendo en cuenta 49 muestras hacia atrás del pico de onda y 51 hacia adelante. Mediante el uso de estos parámetros y la implementación de una ventana variable de 2 segundos con un filtro pasa banda de 1 a 8 Hz de frecuencia, los resultados mostraron una precisión entre el 90% y el 95% en la detección de la onda P. Sin embargo, los autores sugieren la inclusión de más parámetros para aumentar la confiabilidad del sistema de detección desarrollado.

[78] KAUR, Komalpreet; WADHAWA, Manish; PARK, E.K. Detection and Identification of Seismic P-Waves using Artificial Neural Networks. The 2013 International Joint Conference on Neural Networks, Dallas, Texas, United States of America. DOI: 10.1109/IJCNN.2013.6707117. 2013.

De manera similar, Reynen[79] en su investigación “Supervised machine learning on a network scale: application to seismic event classification and detection” (2017) logra un 99% de precisión en la clasificación de eventos sísmicos mediante el uso de *Regresión Logística*, considerando dos atributos para la regresión: grado de polarización (DOP) y espectrograma de frecuencia, teniendo como base la información de 13 estaciones triaxiales ubicadas en el sureste de California, Estados Unidos. Los datos fueron filtrados con un filtro pasa banda de 0,5 Hz a 49,9 Hz. El sistema desarrollado fue puesto a prueba durante una semana con datos continuos en Oklahoma, Estados Unidos, haciendo uso de los datos de 30 estaciones sísmicas. Con el método propuesto, fue posible detectar 25 veces más eventos que con la infraestructura de detección sísmica de Oklahoma. No se registra la exactitud del modelo ni su arquitectura.

El uso de Aprendizaje Automático en la sismología se refleja también en aplicaciones como la tomografía sísmica en la que se busca mapear el interior de la Tierra, buscando un modelamiento de su topografía. En la investigación denominada “Classification of Seismic Windows Using Artificial Neural Networks” (2011), Diersen et al[80] proponen el uso de Importance-Aided Neural Networks (IANN), un tipo de ANN en el que se busca mejorar la precisión de la red con la inclusión del conocimiento del experto, para clasificar ventanas sísmicas usadas para generar imágenes tomográficas del sur de California.

Los atributos considerados para la clasificación fueron: las frecuencias típicas, el ancho de banda, la energía máxima de la ventana y el coeficiente de correlación.

[79] REYNEN, Andrew. Supervised machine learning on a network scale: application to seismic event classification and detection. Department of Earth and Environmental Sciences, Faculty of Sciences, University of Ottawa. 65 pp. 2017.

[80] DIERSEN, Steve; LEE, En-Jui; SPEARS, Diana; CHEN, Po; WANG, Liqiang. Classification of Seismic Windows Using Artificial Neural Networks. International Conference on Computational Science. ISSN 1877-0509. 1572-1581 pp. 2011.

La red es entrenada con 1250 observaciones (71% del total de datos) y validada con 504 (29% del total de datos). Se probaron 53 topologías distintas, 13 con una capa oculta y entre 22 y 34 nodos en la capa; y 40 con dos capas ocultas, con 12 a 19 nodos en la primera capa y de 6 a 10 nodos en la segunda. Los criterios para escoger la mejor topología fueron: Error Cuadrático Medio, el promedio de casos iguales o con diferencia de ±5% de la salida esperada. Se tuvieron en cuenta tres topologías, siendo 16-19-7-1 la mejor configuración. La investigación concluye indicando que la IANN presentó una precisión de 99,60% en la clasificación de registros sísmicos, en comparación con el 99,21% mostrado por una Red Neuronal Artificial tradicional.

3. METODOLOGÍA Y PROCEDIMIENTO

Para el cumplimiento de los objetivos de forma colaborativa e integrada se siguió la metodología Ágil Scrum, en conjunto con la metodología de Prototipado, mediante la cual se generaron, de manera evolutiva, prototipos que se fueron actualizando mediante mejoras a las fallas identificadas. En el Anexo B se amplía la información sobre la metodología utilizada, teniendo en cuenta los requerimientos establecidos y detallados en el Anexo A.

El procedimiento para el desarrollo del sistema para la detección de movimientos sísmicos usando redes neuronales artificiales inicia con la descripción de la población y la muestra consideradas, los instrumentos utilizados para llegar a las fases de procesamiento, entrenamiento, validación y prueba del sistema, y la descripción de los módulos de prototipado desarrollados e integrados.

3.1. POBLACIÓN MUESTRAL

La Red Sismológica Nacional de Colombia (RSNC) empezó el almacenamiento de registros de eventos sísmicos el 1° de junio de 1993 hasta el 28 de febrero del 2018, cuando hubo un cambio en el software de registro y procesamiento de eventos sísmicos, pasando de SEISAN a SeisComp3. La cantidad de sismos registrados en Colombia en ese periodo es de 176.968, incluidos todos los eventos con epicentros y estaciones en los diversos departamentos del país. A este conjunto de datos se le considera como la población de eventos sísmicos.

La muestra analizada está compuesta por un conjunto de 60.785 eventos sísmicos locales con epicentro en el departamento de Santander, durante el periodo comprendido entre el primero de enero de 2010 y el 30 de septiembre de 2017, registrados por la RSNC y publicados en su plataforma web. A partir del primero de octubre de 2017, los archivos no contienen la información sísmica esperada, pues

la RSNC cambió el proceso de almacenamiento de datos y no se encuentran publicados en el servicio de descargas.

Cada uno de los eventos está compuesto por una serie de registros de las 85 estaciones sismológicas nacionales de la RSNC y 10 estaciones sismológicas internacionales ubicadas en Venezuela y Ecuador, entre otros países vecinos incluidos (ver Anexo C). Cada estación hace una medición de velocidad del suelo en diversas componentes espaciales, con una dimensión topológica que varía entre 1 y 16, siendo 1 y 2 casos poco comunes y más de 9 casos aislados. Normalmente la dimensión topológica es de 3 componentes: Z, N y E. Cada componente registra el evento sísmico ocurrente y una ventana anterior y posterior al evento registrado que varía en tamaño.

3.2. INSTRUMENTOS

Los instrumentos utilizados para el desarrollo del sistema involucran herramientas de software, con las que se dio seguimiento a la metodología planteada, diseño y desarrollo de los algoritmos y documentación de algunos apartes de este; y herramientas de hardware con las que se hizo el proceso de selección de datos, se procesaron los datos seleccionados y se ejecutó el algoritmo para la clasificación de eventos sísmicos.

3.2.1. Herramientas de Software

A continuación, se mencionan las herramientas de software que fueron usadas para la ejecución de las tareas citadas:

- Para el control de versiones y trabajo colaborativo durante todo el proceso de desarrollo de algoritmos y Prototipado, se hizo uso de la plataforma GitHub, que está escrita sobre el framework de desarrollo web Ruby on Rails y permite hospedar de forma pública o privada, códigos y archivos en general, brindando el servicio de versionamiento para estos.

- Para el seguimiento de la metodología Scrum planteada, se hizo uso del aplicativo web Zenhub que está asociado con GitHub y permite hacer un seguimiento a las tareas y actividades planteadas, definiendo unos tiempos de entrega de *sprints* y una distribución libre en la asignación de tareas a cada miembro del equipo.

- El lenguaje de desarrollo de cada uno de los algoritmos del sistema es Python en su versión 3.6.5 y se han hecho uso de las siguientes librerías libres para el procesamiento de los datos, entrenamiento, validación y prueba del clasificador:

    - *Librería math*: para ejecutar operaciones aritméticas, lógicas y relacionales básicas, generar semillas pseudo-aleatorias y manejo de decimales.
    - *Librería obspy*: para la lectura, procesamiento y almacenamiento de señales y series de tiempo sismológicas.
    - *Librería numpy*: para el procesamiento de arreglos multi-dimensionales de gran tamaño, operaciones aritméticas, lógicas y relacionales con arreglos, y operaciones matemáticas de alto nivel.
    - *Librería SciPy*: para encontrar raíces de polinomios, resolver ecuaciones lineales y no lineales, ajuste polinomial de datos, interpolación y transformadas en tiempo y frecuencia de los datos.
    - *Librería nolds*: para extraer atributos no lineales relacionados con la Teoría del Caos de conjuntos de datos sísmicos multi-dimensionales.
    - *Librería univariate*: para extraer atributos en otros dominios ajenos al tiempo y la frecuencia (los parámetros de *Hjorth* y la dimensión fractal por el algoritmo de *Petrosian*) mediante transformadas.
    - *Librería os*: para usar funcionalidades del sistema operativo como la creación, lectura y manipulación de archivos y carpetas, y el control de permisos.

- *Librería gc*: para administrar las operaciones del recolector de basura (*Garbage Collector*) al ejecutar ciertas operaciones de alta complejidad espacial.
- *Librería multiprocessing*: para la creación de sub-procesos, hilos y la distribución de tareas y recursos a los procesadores desde software.
- *Librería pickle*: para la serialización y deserialización de datos procesados y su posterior almacenamiento con el fin de reducir el peso de los mismos en disco.
- *Librería matplotlib*: para la representación visual de datos por medio de gráficas en dos dimensiones.
- *Librería time*: para la medición del tiempo que les toma a los algoritmos de procesamiento y clasificación, ejecutar las tareas pertinentes.
- *Librería pandas*: para la instanciación de estructuras de datos propicias para los procesos de aprendizaje automático.
- *Librería scikit-learn*: para el preprocesamiento de datos y el uso de algoritmos de Machine Learning.
- *Librería keras*: para la implementación de algoritmos y estrategias de Deep Learning.

- El ambiente de desarrollo del sistema fue Spyder, un software multiplataforma y de código abierto con licencia MIT para la programación científica en Python que tiene integradas las librerías detalladas. Está embebido en el software de código abierto Anaconda que permite la administración y uso de paquetes de Python y R y está enfocada a desarrollos relacionados con ciencias de datos y Machine Learning.

- Debido a que los algoritmos son ejecutados en diversas máquinas, se usó la aplicación de escritorio Teamviewer en su versión 13 con licencia académica para el control del sistema operativo, software y transferencia de la información.

- Para el diseño y diagramación de los prototipos, representación de los algoritmos en diagramas y corrección de imágenes usadas en la documentación, se hiso uso de la aplicación de escritorio de la suite de Adobe, Adobe Illustrator CC 2018 con licencia académica, facilitado por la UPB seccional Bucaramanga.

- Para el diseño de los algoritmos desarrollados se hizo uso de la aplicación de escritorio StarUML 2 de licencia GNU GPL, facilitado por la UPB seccional Bucaramanga.

- Para la representación gráfica de los resultados en tres dimensiones, se hizo de Matlab en su versión 2016a, proporcionado por la UPB seccional Bucaramanga.

- Para la búsqueda de información y descarga de los archivos sísmicos mediante la creación de snippets en Javascript se hizo uso del navegador Google Chrome en su versión 68 facilitado por la UPB seccional Bucaramanga.

- Para la documentación en prosa de los procedimientos y algoritmos seguidos y desarrollados se hizo uso de las herramientas ofimáticas proporcionadas por la UPB seccional Bucaramanga.

Las herramientas de desarrollo y codificación de algoritmos fueron utilizadas sobre el sistema operativo Ubuntu 17.10 de 64 bits. Las herramientas de documentación y diseño fueron utilizadas sobre sistema operativo Windows 10 de 64 bits.

3.2.2. Herramientas de Hardware

A continuación, se mencionan las herramientas de hardware que fueron usadas para la ejecución de las tareas citadas:

- Se usó un primer computador de escritorio para el preprocesamiento de la muestra con las siguientes características computacionales:

Tabla 6. Características computacionales del computador de escritorio 1 utilizado.

Característica	Descripción
Sistema operativo	Linux 17.10
Procesador	Procesador Intel Corei7 3.40GHz – 8 Cores
Memoria RAM	16 GB
Capacidad de almacenamiento	931 GB de disco duro interno HDD
Tarjetas de red	2
Dirección IPv4	10.152.164.31/24

- Se usó una máquina virtual en el servidor Cobol del Centro de Computación Avanzada (CCA) de la UPB seccional Bucaramanga, para la extracción de atributos, entrenamiento, validación y prueba del Proceso de Clasificación. La máquina virtual presenta las siguientes características computacionales:

Tabla 7. Características computacionales de la máquina virtual del CCA.

Característica	Descripción
Sistema operativo	Linux 17.10 de 64 bits
Cores CPU	32 Cores
Memoria RAM	64 GB
Capacidad de almacenamiento	600GB
Dirección IPv4	10.154.12.13/24

Se utilizaron los 31 núcleos de la máquina virtual, dejando un núcleo libre para tareas varias del sistema operativo y lo no relacionado al procesamiento y clasificación del Dataset. La distribución de tareas a los núcleos se encuentra descrita en la Sección 3.3.2.6 de extracción de atributos del Dataset de sismos.

- Se usó un segundo computador de escritorio para la conexión y control de la máquina virtual mencionada, pues este se encuentra al interior de la subred del CCA, lo que permite una trasferencia de archivos e instrucciones más rápida. El

computador de escritorio presenta las siguientes características computacionales:

Tabla 8. Características computacionales del computador de escritorio 2 utilizado.

Característica	Descripción
Sistema operativo	Dual-boot con una partición con Ubuntu 17.10 y una con Windows 10
Procesador	Procesador Intel Core i5 2.90GHz – 4 Cores
Memoria RAM	8 GB
Capacidad de almacenamiento	500 GB de disco duro interno HDD
Tarjetas de red	1
Dirección IPv4	10.154.12.15

- Se usó un computador personal para la prueba de los algoritmos de entrenamiento, validación y prueba del clasificador, previos a su implementación en la máquina virtual del CCA. Las características de este equipo son:

Tabla 9. Características computacionales del computador personal utilizado.

Característica	Descripción
Sistema operativo	MAC OS High Sierra 10.13.1
Procesador	Procesador Intel Core i5 – 2 Cores
Memoria RAM	8 GB
Capacidad de almacenamiento	120 GB de disco duro interno SSD
Tarjetas de red	1

3.3. PROCEDIMIENTO

El desarrollo del sistema para la detección de movimientos sísmicos usando redes neuronales artificiales se hizo mediante una metodología ágil de desarrollo Scrum y el versionamiento de los algoritmos integrados se hizo siguiendo el modelo de prototipos planteado por el desarrollo evolutivo de Prototipado. La medición del gasto computacional en tiempo de los algoritmos de procesamiento está expresada en términos de su función de complejidad temporal. A continuación, se hace una

descripción del procedimiento para el desarrollo del sistema por Prototipado y los módulos que lo componen, así como el cálculo de la complejidad temporal del mismo.

3.3.1. Prototipado

El clasificador para la detección de movimientos sísmicos fue desarrollado progresivamente de forma modular y a través de la metodología de prototipado. En la Figura 8 se presenta el diagrama de bloques general del sistema de detección que incluye el clasificador desarrollado a manera de prototipo.

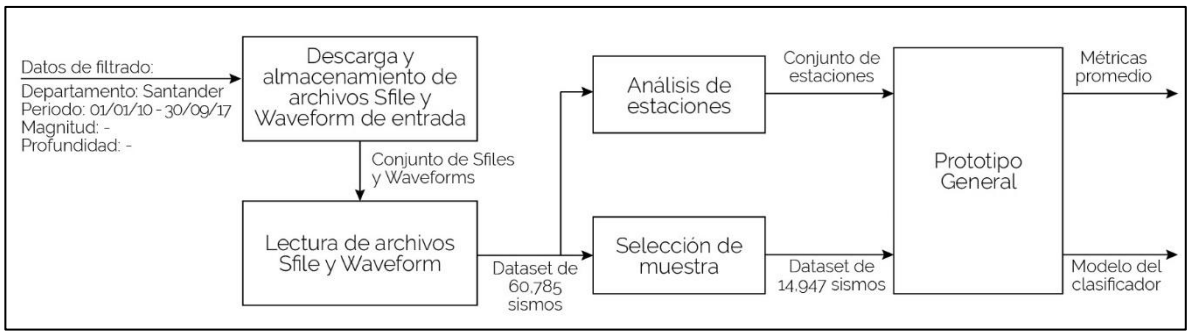


Figura 8. Diagrama general del sistema de detección.

Conforme se ha estipulado, el objetivo del sistema es la detección de eventos sísmicos y por tal motivo, es necesario obtener en primera instancia un conjunto de eventos sísmicos históricos para su análisis y posterior procesamiento. Así, tal como lo describe la figura, la entrada al sistema son aquellos datos de filtrado con los que se obtendrá el histórico sísmico requerido para el procesamiento, entrenamiento, validación y prueba del clasificador. Estos datos son:

- Departamento en el que se ha localizado el epicentro
- Periodo en el que se enmarca el histórico sísmico, disponible desde 1993 hasta el presente, tal como lo menciona la Sección 3.1 de población.

- Intervalo de magnitud de interés según la escala de Richter, con la que fueron registrados los eventos sísmicos.
- Intervalo de profundidad de interés desde los 500 m debajo de la superficie, hasta la profundidad más alta registrada.

Debido a que la detección de los eventos sísmicos está enmarcada en el departamento de Santander y que existe una alta diversidad de sismos en esta región, el clasificador debe tener una fase de entrenamiento con un histórico de datos que permita garantizar la heterogeneidad de la muestra. De esta forma, no se han estipulado datos de filtro de magnitud y profundidad, con el fin de mantener esta heterogeneidad.

3.3.1.1. Versionamiento Semántico

El versionamiento semántico de prototipos permite identificar los prototipos desarrollados de forma ordenada, otorgándoles un nombre que es combinación de números y letras, dependiendo de su avance en la consecución del objetivo y de los cambios que hayan sido ejecutados sobre este.

Para la asignación de etiquetas de versión a los prototipos de este proyecto, se hizo una adaptación del versionamiento de Tom Preston[81], inventor de *Gravatars* y cofundador de GitHub, y los RFC 2119[82] y 8174[83] de la IETF. La asignación de versiones se hace de la siguiente forma:

[81] PRESTON, Tom. Semantic Versioning 2.0.0. [Last Update: August 2018]. Available at: https://semver.org/.

[82] INTERNET ENGINEERING TASK FORCE IETF. RFC2119: Key words for use in RFCs to Indicate Requirement Levels. Available at: https://tools.ietf.org/html/rfc2119. 1997.

[83] INTERNET ENGINEERING TASK FORCE IETF. RFC8174: Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words. Available at: https://tools.ietf.org/html/rfc8174. 2017.

- El número inicial corresponde a la versión V0.0.1, según el formato (V$x.y.z$). El índice z corresponde a la versión de mejoramiento, el índice y a la versión secundaria y el índice x a la versión primaria.
- Un incremento de una unidad en el valor de la versión de mejoramiento Z (V$x.y.Z$ con $x > 0$) se hace cuando exista una mejora de alguna falla o error en la versión actual, compatible con versiones anteriores. Una corrección de errores se define como un cambio interno que corrige el comportamiento incorrecto.
- Un incremento de una unidad en la versión secundaria Y (V$x.Y.z$ con $x > 0$) se hace cuando se introduce una nueva funcionalidad en la versión actual, compatible con las versiones anteriores o si alguna de las funciones de versiones anteriores se marca como obsoleta.
- Un incremento de una unidad en la versión primaria X (V$X.y.z$ con $X > 0$) se hace cuando se introducen cambios en la versión actual que no son compatibles con las versiones anteriores y esto mejora sustancialmente el funcionamiento, y la versión está lista para ser publicada.
- Una adición de etiquetas al final de la versión (V$x.y.z - etiqueta$) se hace cuando se quiere hacer una diferenciación entre algunas funcionalidades particulares de la versión V$x.y.z$ y que, en cualquiera de los casos, siempre hay una precedencia de esta versión que no implica un cambio sustancial como para diferenciarlo de forma más general.

3.3.2. Descripción de módulos de los prototipos

El desarrollo de los prototipos ha sido de forma modular, de tal manera que la salida de cada uno de los módulos es la entrada del siguiente, hasta que el objetivo del prototipo sea cumplido, para una entrada dada. En la Figura 9 se muestra el diseño general de un prototipo.

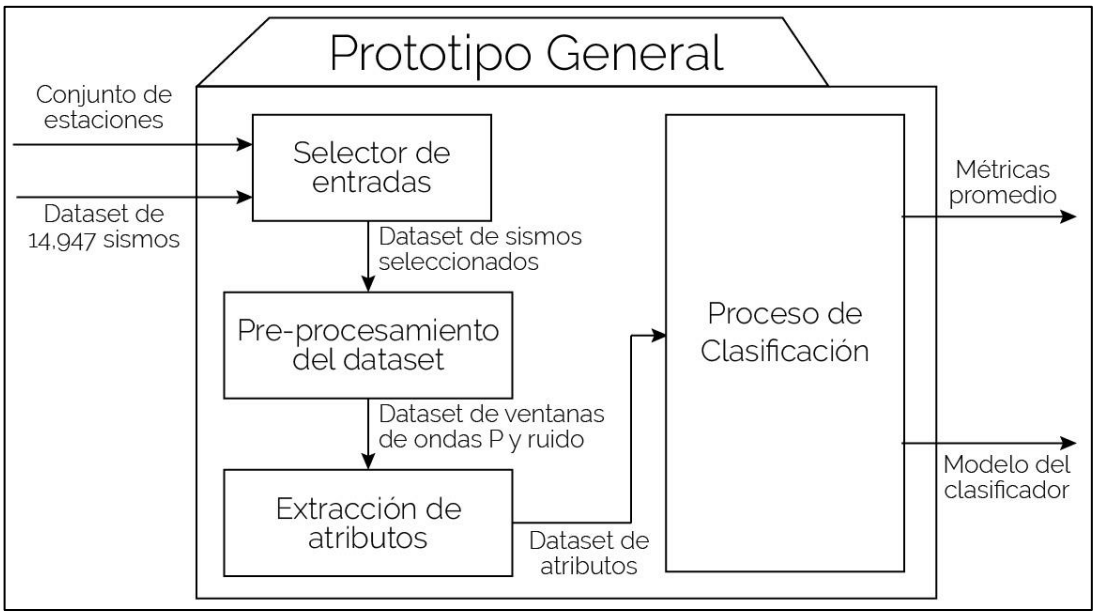


Figura 9. Diagrama de bloques del diseño de modular de un prototipo general.

Los módulos generales implementados en todos los prototipos desarrollados ejecutan las siguientes acciones:

- Descarga y almacenamiento de los archivos de entrada
- Lectura de los archivos de entrada
- Análisis de estaciones y selección de la muestra
- Preprocesamiento de los datos
- Selección y extracción de atributos
- Proceso de clasificación

3.3.2.1. Descarga y almacenamiento de los archivos de entrada

Para la obtención de los datos históricos sísmicos del departamento de Santander, se revisa la información que se encuentra en la base de datos de la RSNC que puede ser accedida mediante el aplicativo web con Localizador de Recurso URL *https://bdrsnc.sgc.gov.co/paginas1/catalogo/Consulta_Experta/consultaexperta_2.php*. La descarga de los archivos se hace mediante el *snippet* mostrado en la Figura

10 en lenguaje *javascript.* El proceso completo de descarga y almacenamiento se encuentra descrito con detalle en el Anexo D.

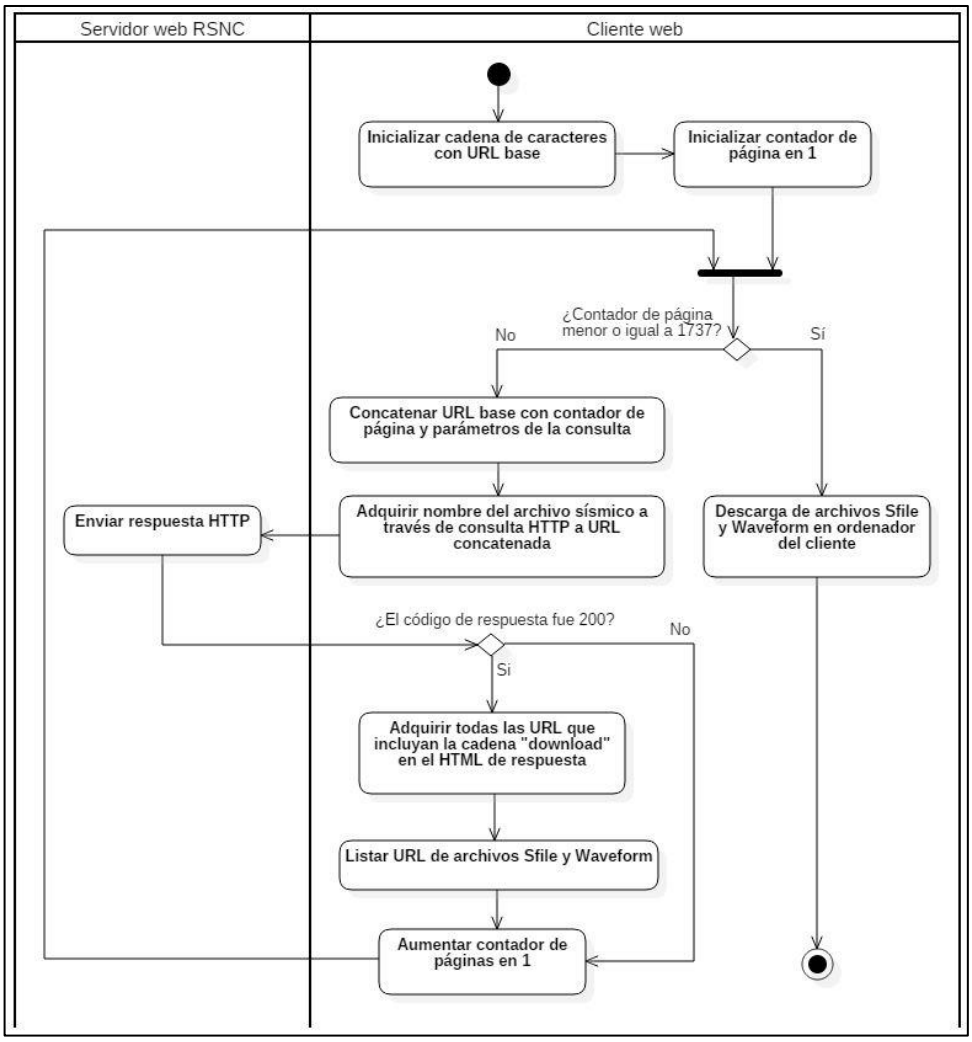


Figura 10. Diagrama de actividades del snippet desarrollado en Javascript para la descarga de archivos Sfile y Waveform.

El algoritmo ejecuta un ciclo en el que se accede a las 1737 páginas de recursos haciendo una petición HTTP y extrayendo los elementos HTML resultantes de las respuestas del servidor a las peticiones. Esto se hace teniendo en cuenta que el número de página es lo único que cambia en las URL. En los elementos encontrados, se filtran aquellos íconos que tengan una referencia *href* de descarga de archivos, 'download' para las trazas y 'REA' para los archivos binarios de

información sísmica. Cada una de las direcciones obtenidas de los íconos es concatenada en una cadena que al final imprime todas las referencias a los archivos en todas las páginas, tanto para los archivos de traza como para los binarios de información. Una vez obtenidas las direcciones URI donde se encuentran los archivos, se ejecuta un código en Bash que descarga estos recursos en una carpeta destinada para este fin.

3.3.2.2. Lectura de archivos de entrada

Una vez descargados los archivos SFiles y Waveforms, se procede a ejecutar una revisión detallada del formato encontrado y a desarrollar un algoritmo que permita la lectura de los datos en estos formatos, tal que no se pierda información.

Formato de los archivos de entrada

Regularmente, un archivo SFile de la RSNC tiene la siguiente estructura:

```
2015  310 0645 55.8 L   6.817 -73.156145.2  RSN  9 0.4 1.7LRSN                1
GAP=134        1.10       3.4     5.2  6.1 -0.1674E+01  0.7456E+01  0.5411E+00E
Epicentro: LOS SANTOS - SANTANDER                                             3
ACTION:UPD 15-04-30 16:24 OP:nd   STATUS:               ID:20150310064555 L   I
OLDACT:REG 15-03-10 16:45 OP:cam  STATUS:               ID:20150310063749     3
2015-03-10-0637-49M.COL___275                                                 6
STAT SP IPHASW D HRMM SECON CODA AMPLIT PERI AZIMU VELO AIN AR TRES W  DIS CAZ7
BAR2 EZ EP        646 14.81                             170   -0.6010 25.0 187
BAR2 EZ ES        646 30.59                             170   -0.1010 25.0 187
BAR2 EZ I         646 31.56         8.6 0.22            168           25.0 187
BRR  HZ EP        646 17.25                             153    0.4610 69.3 298
BRR  HN ES        646 33.79                             153    0.6310 69.3 298
BRR  HZ I         646 37.81         9.4 0.46            151           69.3 298
PAM  SZ IP     C  646 18.34                             151    0.4210 76.7  41
PAM  SN ES        646 35.40                             151    0.2310 76.7  41
PAM  SZ I         646 36.95         7.3 0.18            155           76.7  41
RUS  HZ EP        646 20.00                             143    0.3610  103 175
RUS  HE ES        646 37.43                             143   -0.8110  103 175
RUS  HZ I         646 39.52         5.9 0.32            144            103 175
PTB  HZ EP        646 22.20                             132   -0.46 9  147 258
PTB  HN ES        646 43.63                             132    0.02 9  147 258
PTB  HZ I         646 47.86         7.8 0.10            131            147 258
```

Figura 11. Archivo SFile que contiene la información sísmica del evento registrado el 10 de marzo de 2015. Adaptado de Red Sismológica Nacional de Colombia. 2018.

En la estructura de archivo mostrada pueden destacarse los siguientes campos:

- STAT: código de la estación que registra el evento. El listado de estaciones se muestra en el Anexo C.
- HRMM SECON: tiempo en horas, minutos y segundos de picado de la onda, dependiendo de la onda correspondiente, P o S.
- IPHASW: tipo de picado[84] (I), fase (PHAS) y peso (W). La fase identifica si se trata de la onda P o de la onda S que ha sido picada, adjuntando el tiempo de picado en el siguiente atributo.

Los archivos de forma de onda o *waveforms* definen una estructura binaria. Son los archivos en los que queda registrado el evento sísmico muestra a muestra, desde un tiempo antes de su inicio que es determinado por el sismólogo de turno, hasta que ha finalizado el evento. Este tipo de archivos se encuentran en formato *miniSEED* (*Data Only Standard for the Exchange of Earthquake Data*), un formato creado por IRIS (*Incorporated Research Institutions for Seismology*) con el que se pretende el registro de la actividad sísmica ocurrente y el intercambio de series de tiempo de los eventos ocurridos. Un archivo binario de forma de onda tiene a grandes rasgos, dos grandes secciones de contenido:

- Sección de metadatos donde se especifica información relacionada con cada una de las estaciones que pertenecen a la red de sismógrafos y acelerógrafos de la RSNC, que incluye las estaciones que ha identificado el evento sísmico.

[84] *def*. Cuando ocurre un evento sísmico, se desprenden distintos tipos de ondas entre las que se encuentran las ondas P y las ondas S. El tiempo aproximado de la llegada de la onda P, una vez ha pasado el evento sísmico, visto desde la perspectiva del sismólogo y contrastado con un algoritmo matemático de identificación de la onda P, se denomina tiempo de picado de la onda P. Tomado de: CHI DURÁN, Rodrigo Kimyen. Caracterización de trazas sísmicas en el campo cercano: Pisagua, Norte de Chile. Universidad de Chile, Facultad de Cienias Físicas y Matemáticas, Departamento de Ingeniería Eléctrica. 2015.

- Sección de registro de las series de tiempo de cada una de las estaciones. Si alguna estación no ha identificado el evento sísmico, la traza será ruido o en su defecto, una constante sobre el 0.

El formato de los archivos Sfile y Waveform se encuentra explicado con mayor detalle en el Anexo E.

Lectura de los archivos *Sfiles* y *Waveforms*

Para la lectura e interpretación de la información sísmica contenida en los archivos Sfile y Waveform se diseñaron y desarrollaron los procesos con base en el diagrama de clases mostrado en la Figura 12. Las clases de lectura de los archivos son *WaveformReader* y *SfileReader*, tienen dos métodos de lectura de los archivos: *get_traces*() y *get_params*(). Estos métodos recorren los archivos Waveform y Sfile, respectivamente, de tal forma que las trazas sean leídas como arreglos de tiempo con la magnitud y el contenido de los eventos sea convertido a un diccionario en donde se puedan filtrar los parámetros de interés que han sido mostrados en la sección anterior. El método de lectura de las trazas hace uso del método *read()* de la librería de Obspy para la lectura de archivos en formato SEED o sus derivados.

Una vez que son leídos los archivos con estos métodos, se instancia un objeto de la clase Evento mediante el método *get_event*() de la clase *EventReader*. Este objeto es una abstracción de cada uno de los eventos sísmicos y tiene la información contenida en los Sfile que describen el evento y un diccionario en el que se almacenan las componentes que registran las estaciones en los eventos. Este diccionario contiene objetos de tipo *TraceGroup* en el que se almacena el nombre de la estación, un arreglo con las componentes, la muestra en las que están registradas las ondas P y las ondas S y la distancia epicentral calculada entre la ubicación de la estación y la del epicentro registrado.

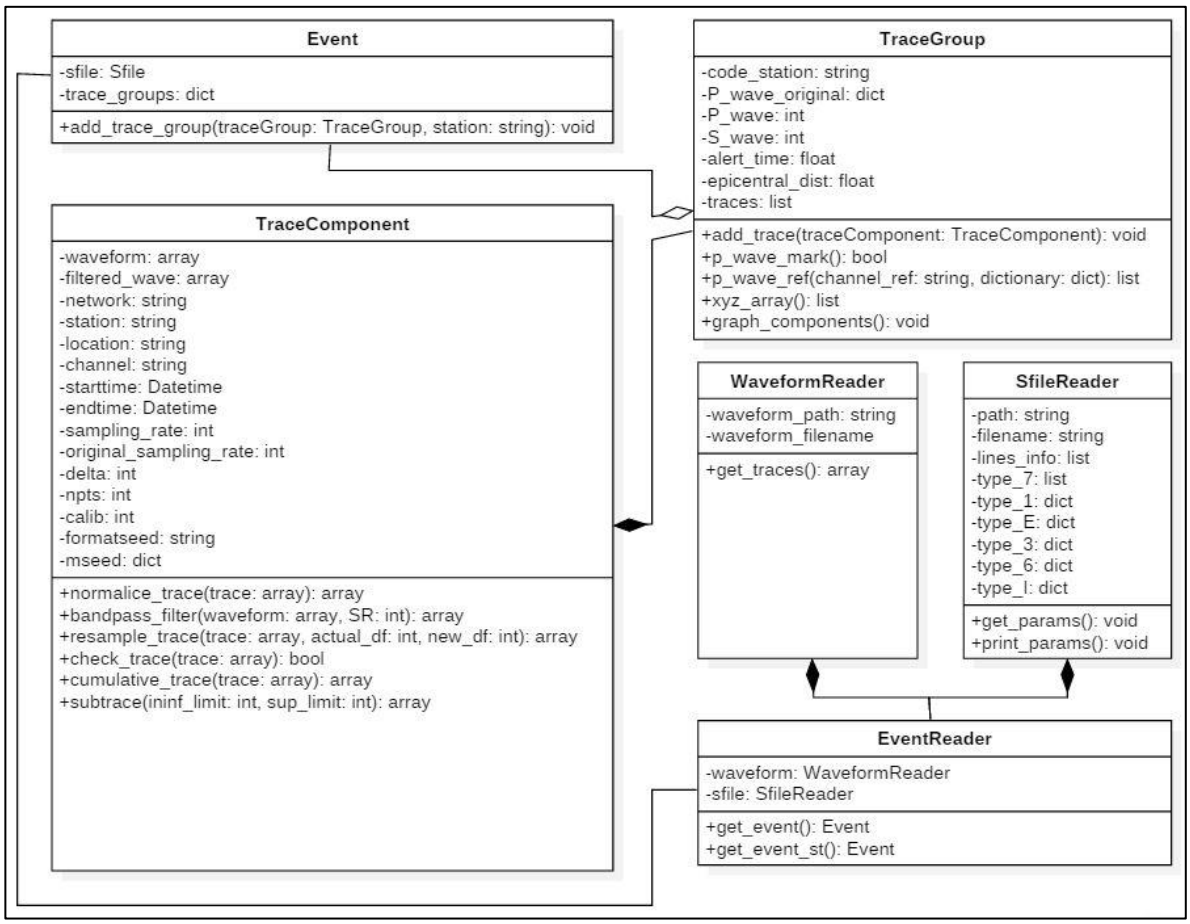


Figura 12. Diagrama de clases de los procesos de extracción y preprocesamiento de los archivos Sfile y Waveforms.

A su vez, las componentes son almacenadas en un arreglo de objetos *TraceComponent* que almacena toda la información de las trazas de cada componente de la estación en el evento, la hora de inicio, hora de fin, orientación del canal, número de muestras, arreglo de las muestras originales en la traza y de las muestras pre-procesadas en la traza, tal como se describe en las secciones siguientes.

3.3.2.3. Selección de la muestra

Una vez descargados y leídos los archivos de contenido Sfiles y Waveforms como se ha explicado en las secciones anteriores, se ejecuta un proceso de selección de

la muestra, de tal forma que los archivos resultantes puedan ser utilizados en todos los procesos siguientes. La selección de la muestra se hizo de la siguiente forma (ver Figura 13):

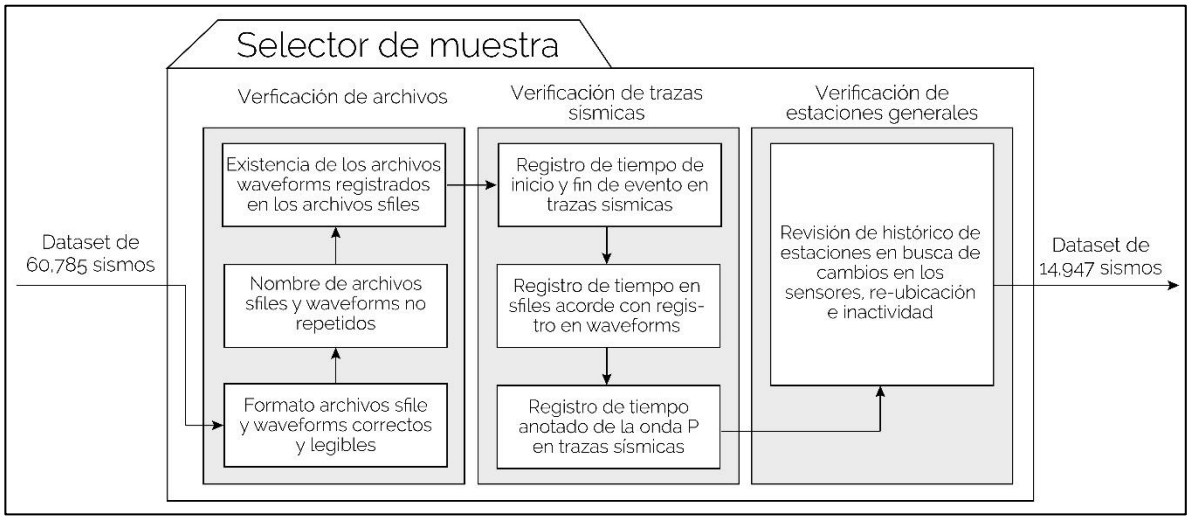


Figura 13. Proceso de selección de la muestra.

El proceso de selección inicia con una verificación de los archivos, en la que se hace una revisión de formato con el fin de descartar aquellos que presenten inconvenientes a la hora de ser procesados. Seguidamente, se eliminan archivos repetidos en el conjunto de datos. Esto sucede debido a que la RSNC presenta los archivos en forma de tablas web, en el que cada registro de la última tabla de la página anterior es el mismo registro de la primera fila de la tabla de la página siguiente, haciendo que se repita un archivo por página, dentro de las aproximadamente 1800 páginas de contenido.

Posteriormente y teniendo en cuenta que cada archivo Sfile tiene dentro de sus parámetros el nombre del Waveform con el que está asociada la información de las trazas sísmicas, se hace un proceso de revisión de la existencia de estos archivos. Aquellos archivos Sfile sin su correspondiente Waveform, son eliminados. De igual forma, aquellos archivos Waveform que no sean citados en ningún archivo Sfile también son eliminados.

Una vez que se han verificado los archivos, se procede a hacer una verificación general de su contenido. De esta forma, se verifica que el tiempo de inicio y fin de cada evento hayan sido registrados en los Sfiles y correspondan con los presentes en las trazas al interior de los Waveforms. De igual manera, es vital que en el contenido de los archivos Sfiles exista el registro de anotación de la llegada de la onda P en cada evento, pues el clasificador sísmico está basado en la detección de esta onda. Aquellos archivos que no cumplan estas condiciones de registro de tiempos son descartados.

Finalmente, se hace un análisis del histórico de estaciones sismológicas en búsqueda de la siguiente información:

- Posibles cambios en los sensores de medición de velocidad y aceleración que generen una variación en la dinámica de la señal registrada. Si esto sucede, debe escogerse el último periodo en el que no existan variaciones de este tipo, con el fin de mantener la dinámica de la sismicidad registrada por las estaciones.
- Posible reubicación de las estaciones que afecte la dinámica de sismicidad de la zona de evaluación. Si la estación es reubicada, la geografía y topología cambia, lo que hace que el modelo de velocidades varíe, haciendo que exista una diferencia de dinámica con respecto al lugar original.
- Posibles periodos de inactividad de las estaciones producto de inconvenientes técnicos, mantenimiento o cualquier factor que afecte el funcionamiento de las estaciones. Estos periodos causan mediciones erróneas o inexistentes, haciendo que se vea afectado el proceso de entrenamiento del clasificador.

Estas variables son analizadas con el fin de identificar los cambios, los posibles factores que afecten la continuidad de la dinámica de la sismicidad registrada por

las estaciones que pueda causar un mal aprendizaje del clasificador y afectar la detección.

3.3.2.4. Análisis de estaciones

Teniendo en cuenta la muestra detallada en la Sección 3.1, aunque esta contiene una heterogeneidad de datos apreciable, el costo computacional para analizarlos es considerablemente elevado. Por esta razón, se siguió un procedimiento de selección de la muestra que involucra un análisis de los siguientes criterios, relaciones y variables, conservando la heterogeneidad presente en la muestra original:

- Ubicación de las estaciones sismológicas en la geografía colombiana.
- Ubicación de las estaciones sismológicas y los eventos sísmicos registrados con epicentro en el departamento de Santander.
- Cantidad de sismos por estaciones y epicentro.
- Cantidad de sismos por magnitud identificada.
- Epicentro contra profundidad promedio.
- Latitud contra longitud y profundidad.
- Distancia epicentral promedio por estación.
- Tasa de muestreo de las estaciones a lo largo del periodo de evaluación.
- Cantidad de componentes de medición de las estaciones a lo largo del periodo de evaluación.
- Distribución de magnitudes y profundidades.

El objetivo de este procedimiento es la selección de las estaciones de mayor relevancia en la detección de eventos sísmicos, con el fin de reducir el costo computacional para el procesamiento de las señales históricas y del proceso de clasificación, manteniendo la heterogeneidad y diversidad de los datos en magnitud, profundidad y ubicación del epicentro al interior de la región del departamento de

Santander. El proceso de análisis de las estaciones es el mostrado en seguida (Figura 14).

Figura 14. Proceso de análisis de estaciones.

Debido a que se trata de un proceso de análisis que se ejecuta paralelamente al proceso de selección de la muestra, se debe tener en cuenta el conjunto de datos de la muestra original, derivada de la población especificada. Esos 60.785 datos de entrada pasan por tres procesos de forma simultánea:

- Un Mapeo de la ubicación geográfica de las estaciones y los sismos registrados, con el fin de identificar de forma visual la distribución de estos eventos y la posible formación de nidos de microsismicidad, tal que puedan ser detectados por estaciones locales y regionales aledañas.
- El conteo de la cantidad de sismos por estación y epicentro permite identificar nuevamente la diversidad de ubicación de eventos sísmicos registrados y determinar cuáles estaciones son las que han registrado mayor cantidad de sismos en el periodo de evaluación. Aquellas estaciones de mayor registro de eventos sísmicos pueden presentar mayor sensibilidad a la ocurrencia de estos eventos, están en una ubicación geográfica que favorece la detección o presentan mayor cercanía con los nodos sísmicos.

- El cálculo de la distancia epicentral promedio por estación para la identificación de las estaciones más cercanas que han detectado los eventos sísmicos registrados, lo que permite entrever la posibilidad futura de detección en línea y alerta sísmica usando las estaciones escogidas por medio de este parámetro.

Una vez ejecutados estos procesos, se procede a hacer un proceso de selección de las estaciones con la información recopilada, en el que se evalúan los criterios descritos y se obtiene un conjunto de estaciones base con las que se ejecutará los procesos de procesamiento, extracción de atributos y clasificación.

3.3.2.5. Preprocesamiento de datos

El preprocesamiento de los datos se ejecuta con el fin de hacer una serie de transformaciones y adecuaciones a los datos para prepararlos para los procesos posteriores: la extracción de atributos y el proceso de clasificación. Así, una vez que se ha seleccionado la muestra con las estaciones consideradas, se procede a ejecutar el proceso mostrado en la Figura 15.

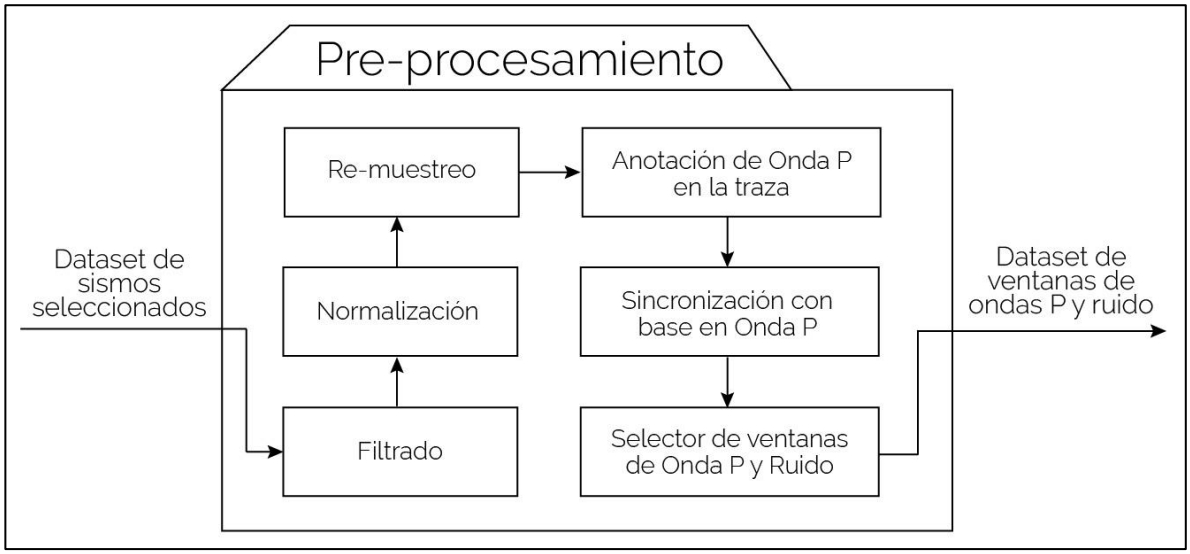


Figura 15. Proceso de preprocesamiento de datos sísmicos seleccionados.

Filtrado, normalización y re-muestreo de los datos

La primera etapa del preprocesamiento es el filtrado de los datos. Para este proceso debe tenerse en cuenta la existencia de dos grandes fuentes de ruido, el que se produce por las vibraciones terrestres y tiene causas naturales, y el que se produce en el instrumento de medición, o ruido instrumental, que es producto de los siguientes factores[85]:

- Tal como sucede en las profundidades del océano, en la superficie el ruido sísmico de los microsismos está en una banda dominante que va desde los 0.1 Hz hasta 1 Hz.
- Ruido del suelo de muy largo periodo (0.2 a 50 mHz).
- Ruido por fuentes naturales: viento que fricciona con el terreno, con los árboles y la vegetación o los objetos vibrantes; cascadas y afluentes que chocan con las piedras, entro otros (15 a 60 Hz).
- Ruido por fuentes humanas: plantas de poder, industrias, transformadores y equipos eléctricos, tráfico, entro otros; que generan ruido comúnmente en las bandas de 50 a 60 Hz, así como sub-armónicos en 30 Hz, 25 Hz y 12.5 Hz, dependiendo de la ubicación de la fuente.

De esta forma, puede identificarse una banda de frecuencia libre de algunos factores de ruido: la banda que va desde 1 Hz hasta 12 Hz. Tal como Küperkoch, et. al.[86] propone en la etapa de filtrado con la banda de 2 a 10 Hz y en contraste a

[85] BORMANN, Peter; WIELANDT, Erhard. Chapter 4: Seismic Signals and Noise. In: New Manual of Seismological Observatory Practice (NMSOP-2), IASPEI, GFZ German Research Centre for Geosciences, Potsdam. DOI: 10.2312/GFZ.NMSOP-2_CH4.

[86] KÜPERKOCH, L.; MEIER, Th.; DIEHL, T. Chapter 16: Automated event and phase detection. In: New Manual of Seismological Observatory Practice (NMSOP-2), IASPEI, GFZ German Research Centre for Geosciences, Potsdam. DOI: 10.2312/GFZ.NMSOP-2_CH4.

lo que Kaur, et. al.[87] ha identificado como una banda de filtrado de 1 a 8 Hz, se propuso un filtro pasa banda Butterworth de cuarto orden de 1 – 10 Hz como la unión de los dos subconjuntos de bandas de los autores citados.

Para el filtrado de las señales de forma computacional, se utiliza la función *bandpass* de la librería *obspy.signal.filter*, que tiene como entradas la señal a filtrar por componente, la frecuencia mínima y máxima del intervalo de frecuencias definido y la tasa de muestreo de la señal en Hz. La salida corresponde a la señal filtrada en la banda de frecuencias de entrada. La función *bandpass* emplea las funciones *iirfilter* y *sosfilt* de la librería *scipy.signal* para el uso de algoritmos de filtrado digital IIR (*Infinite Impulse Response*). Según National Instrument[88], los filtros IIR tienen menor complejidad espacial y una cantidad de operaciones menor para el filtrado, razón por la cual son recomendables para el filtrado digital de grandes cantidades de datos. El filtro Butterworth en complemento, según McGuire[89], et. al., tiene una respuesta en frecuencia suavizada y presenta una función de transferencia en la que los coeficientes de su polinomio pueden ser fácilmente calculados, disminuyendo la complejidad espacial y temporal a la hora de hacer un filtrado con grandes cantidades de datos, además de que presenta una buena respuesta en frecuencia de la banda de paso.

Una vez que las señales son filtradas, se pasan por un proceso de normalización por media aritmética enmarcado en un proceso de escalado de atributos (*Feature*

[87] KAUR, Komalpreet; WADHAWA, Manish; PARK, E.K. Detection and Identification of Seismic P-Waves using Artificial Neural Networks. The 2013 International Joint Conference on Neural Networks, Dallas, Texas, United States of America. DOI: 10.1109/IJCNN.2013.6707117. 2013.

[88] NATIONAL INSTRUMENTS. IIR Filters and FIR Filters. Part Number: 370858N-01. Available at: http://zone.ni.com/reference/en-XX/help/370858N-01/genmaths/genmaths/calc_filterfir_iir/. 2017.

[89] MCGUIRE, Michael. The advantages and disadvantages of the Butterworth filter, the Chebyshev filter, and the Bessel filter. Computer Engineering & Digital Signal Processing, University of Victoria. 2017.

Scaling). El proceso consiste en una centralización de la media que permite obtener un conjunto de datos sin la componente continua que poseen y un re-escalado por máximo y mínimo, con lo que se obtiene un conjunto de datos con amplitud en el intervalo [-1, 1]:

$$x' = \frac{x - media(x)}{max(x) - min(x)} \qquad \text{(Ecuación 21)}$$

Esto se hace con el fin de presentar un estándar en los datos para los procesos de extracción de atributos, entrenamiento, validación y prueba del clasificador, debido a que las estaciones sismológicas presentan una amplia diversidad de sensores y sensibilidades, lo que hace que las amplitudes difieran entre muestras de eventos y estaciones. El algoritmo seguido para la normalización por media de los datos es el siguiente:

Algoritmo 5. Pseudocódigo para la normalización por media aritmética de los datos

Entrada: arreglo con objetos de trazas sin normalizar
Salida: arreglo con objetos de trazas normalizadas

```
1.  def normalize(objetos_waveform)
2.      media ← mean(objetos_waveform)
3.      max_min ← max(objetos_waveform)- min(objetos_waveform)
4.      para muestra en objetos_waveform
5.          muestra ← (muestra – media)/max_min
6.      retornar objetos_waveform
```

El preprocesamiento de los datos en sus procesos de filtrado en la banda de frecuencias, normalización y re-muestreo es comprobado para algunos de los eventos sísmicos del conjunto de datos descritos en el proceso de selección de la muestra de la sección anterior y los resultados son presentados en la Sección 4.4.

Una vez filtrados y normalizados los datos, se ejecuta un proceso de re-muestreo de los datos debido a que éstos presentan una alta diversidad de tasas de muestreo,

producto de los diferente sensores y dispositivos de adquisición que están presentes en las estaciones sismológicas y en los algoritmos de digitalización, almacenamiento y procesamiento de la RSNC. El re-muestro se hace con el fin de estandarizar las tasas de muestreo, tal que todas las trazas presenten una tasa uniforme a lo largo del conjunto de datos. La tasa de muestreo base es 100 Hz, tasa identificada con mayor frecuencia en el conjunto de datos.

Para re-muestrear las trazas de forma computacional se hizo uso del método *resample* de la librería *scipy.signal* que hace un re-muestreo de un conjunto de datos a una tasa de muestreo de entrada dada, resultando en un conjunto de datos muestreados con esa tasa. Este proceso se hace por medio del método de re-muestreo usando la transformada de Fourier, tal como lo describe Hawkins[90] en "Fourier transform resampling: theory and application".

Anotación de Onda P, Sincronización y Selección de ventanas

Una vez que se ha filtrado, escalado y re-muestreado el conjunto de datos, las trazas se encuentran sin ruidos de alta frecuencia y de muy largo periodo, están en un intervalo definido de magnitudes y presentan la misma tasa de muestreo. El siguiente proceso ejecutado es el de anotación de la Onda P mediante la revisión de la información de los archivos Sfile, en el que se detalla la hora de picado de esta onda, tal como se describe en la Sección 3.3.2.2 en el atributo IPHASW.

Se hace una conversión simple por regla de tres de la muestra relativa al tiempo de picado en la traza y esta muestra queda registrada para cada una de las componentes registradas por estación de interés para el evento, teniendo en cuenta la relación entre la nueva tasa de muestreo y la original:

[90] HAWKINS, W. G. Fourier transform resampling: theory and application. IEEE Transactions on Nuclear Science Vol 44, No. 4. DOI: 10.1109/23.632725. 1543 – 1551 pp. 1997.

```
P_Wave = round((new_df/original_df)*P_Wave)
```

Donde `self.P_Wave` es la variable que representa la muestra en la que fue picada la Onda P y `(new_df/original_df)` es la relación entre las dos tasas de muestreo. El picado de la Onda S también fue ejecutado siguiendo el mismo proceso con el único fin de tener un registro del tamaño de la ventana en el proceso siguiente, tal que no se excediera la ventana de la Onda P hasta solapar la Onda S.

Una vez picadas las ondas, se procede a un proceso de sincronización de las trazas, pues los archivos Sfile y Waveform están sujetos a presentar distintos tiempos de inicio del evento en las componentes de las estaciones, aunque el tiempo de picado de la Onda P para todas las componentes es el mismo. De esta manera, la sincronización se hace con base en este tiempo de picado de la onda y teniendo en cuenta la componente sobre la que fue registrada. Por ejemplo, si la Onda P fue picada sobre la componente vertical Z, entonces las dos componentes restantes, si presentan un tiempo de inicio distinto, son ajustadas al tiempo de inicio de la componente Z sobre la que fue picada la onda. De igual forma, si existe un picado sobre otra componente, se aplica la misma lógica para la sincronización de las dos componentes restantes. Esta tarea se ejecuta por estación en todos los registros tenidos en cuenta en el selector de muestra y estaciones.

En el proceso de sincronización son eliminadas las primeras y últimas muestras de las trazas de las componentes con respecto a la traza de la componente de referencia en la que es picada la Onda P, con el fin de que exista una homogeneidad en la cantidad de muestras por componente, el mismo tiempo de inicio y el mismo tiempo de fin del evento, considerando que al inicio y fin de las trazas no existe información concluyente sobre el evento y son simplemente muestras ajenas a la ocurrencia del mismo. La sincronización se hace como se muestra en el siguiente algoritmo:

Algoritmo 6. Pseudocódigo para la sincronización de las trazas

Entrada: componentes de medición distribuidos en el canal de referencia y los otros dos canales
Salida: componentes sincronizados teniendo en cuenta la Onda P

```
1.  def sinc(canal_ref, canal_2, canal_3)
2.      dif_Refvs1 ← int((canal_ref.starttime - canal_1.starttime)* canal_ref.sampling_rate)
3.      dif_Refvs2 ← int((canal_ref.starttime - canal_2.starttime)* canal_ref.sampling_rate)
4.      si Refvs1 != 0
5.          canal_1.filter_wave ← canal_1[dif_Refvs1:]
6.      si Refvs2 != 0
7.          canal_2.filter_wave ← canal_2[dif_Refvs2:]
8.      npts_min ← min([len(canal_ref), len(canal_1), len(canal_2)])
9.      canal_ref ← canal_ref[0:npts_min]
10.     canal_1 ← canal_1[0:npts_min]
11.     canal_2 ← canal_2[0:npts_min]
12.     retornar canal_ref, canal_1, canal_2
```

La entrada al algoritmo son tres arreglos de canales que corresponden a las trazas de las tres componentes, distribuidas en: la componente de referencia sobre la que se ha anotado la Onda P y las 2 componentes restantes a sincronizar. La salida corresponde a las tres componentes sincronizadas con igual cantidad de muestras.

Una vez que las trazas están sincronizadas y con las ondas debidamente anotadas, se procede a extraer porciones de ventana de evento y ruido. Esto se hace con el fin de entrenar al clasificador en lo que será una señal relacionada con un evento sísmico y una relacionada con ruido, lo que representa la no ocurrencia de un evento. El tamaño de la ventana está definido acorde a lo encontrado por Kaur[91], et. al., en su investigación, proponiendo una ventana deslizante de 2 segundos, lo que es equivalente a una ventana estática de 200 muestras para el clasificador, teniendo en cuenta la frecuencia de muestreo de 100 Hz propuesta.

[91] KAUR, Komalpreet; WADHAWA, Manish; PARK, E.K. Detection and Identification of Seismic P-Waves using Artificial Neural Networks. The 2013 International Joint Conference on Neural Networks, Dallas, Texas, United States of America. DOI: 10.1109/IJCNN.2013.6707117. 2013.

La ventana de 200 muestras es extraída de las componentes registradas para el evento por las estaciones con la ventana alrededor de la muestra de picado de la Onda P, de tal forma que se han planteado las dos siguientes formas de posicionamiento:

- 100 de las 200 (50%) muestras de la ventana atrás de la onda y 99 muestras adelante.
- 180 de las 200 (90%) muestras de la ventana atrás de la onda y 19 muestras adelante.

Estas variaciones en las formas de posicionamiento de la onda se tuvieron en cuenta por los siguientes tres factores:

- El comportamiento de los atributos varía según la ventana escogida. Para verificar esto, se hallaron los atributos descritos en la siguiente sección utilizando una ventana deslizante a lo largo de las trazas, observando la dinámica presentada. Los resultados son descritos en la Sección 4.5.
- El desempeño final del clasificador con estas dos variaciones, cuyos resultados son descritos en la Sección 4.6.
- El tiempo en el cálculo de los atributos y en el proceso del clasificador comparado con el tiempo que toma recibir la siguiente ventana deslizante, lo que puede estar representado en etapas posteriores como el tiempo de generación de una alerta de tratarse de una detección en línea. Si existe un solapamiento de exactamente media ventana (100 muestras), la siguiente ventana arribaría un segundo de después, tiempo en el que debe ejecutarse la clasificación del sismo.

El algoritmo para la extracción de las ventanas es presentado a continuación:

Algoritmo 7. Pseudocódigo para la extracción de las ventanas de Onda P y Ruido

Entrada: traza de la componente registrada por estación, tamaño de la ventana, muestra anotada de la Onda P y porcentaje en la ubicación de la onda en la ventana de 0 a 1.
Salida: ventanas de Onda P y ruido

```
1.  def p_noise_extraction(traza, tamano_ventana, P_wave, porcent)
2.      inf_limit ← random.randint((2*tamano_ventana),(P_wave-(2*tamano_ventana)))
3.      init ← porcent* tamano_ventana
4.      Onda_P ← sub_trace(traza,P_wave-init,P_wave+(tamano_ventana -init))
5.      Ruido ← sub_trace(traza,inf_limit,inf_limit+ tamano_ventana)
6.      retornar Onda_P, Ruido
```

La entrada del algoritmo se compone de la traza de la componente registrada por la estación, el tamaño de la ventana (200 muestras), la muestra en la que se encuentra anotada la Onda P y porcentaje en la ubicación de la onda en la ventana de 0 a 1 (50% ó 90%). La salida corresponde a las ventanas de Onda P y ruido con la cantidad de muestras definidas en la entrada. El método *randint* permite generar números pseudoaleatorios enteros y el método *sub_trace* permite extraer una porción de traza mediante la definición del límite superior e inferior del arreglo.

3.3.2.6. Selección y extracción de atributos

La identificación de diferentes tipos de ondas sísmicas es llevada a cabo mediante el reconocimiento de diversas características de la señal. Para el conjunto de datos sísmicos de entrada de tres componentes de una fuente sísmica local, una variedad de atributos es detectada e identificada para su análisis. Estos son:

- Grado de Polarización de la Onda (DOP), calculado tridimensionalmente.
- Radio de Potencia Vertical contra Potencia Total (RV2T), calculado en las tres componentes.
- Entropía de Shannon, calculada en las tres componentes.
- Asimetría (*skewness*), calculada en las tres componentes.
- Kurtosis, calculada en las tres componentes.

- Dimensión de Correlación o dimensión fractal, calculada en tres componentes.

Otros atributos considerados, pero no incluidos en el entrenamiento, validación y prueba de la red neuronal son:

- Mediana, calculada sobre las tres componentes.
- Media geométrica, calculada sobre las tres componentes.
- Media armónica, calculada sobre las tres componentes.
- Desviación estándar, calculada sobre las tres componentes.
- Exponente de Lyapunov máximo, calculado sobre las tres componentes.
- Exponente de Hurst, calculado sobre las tres componentes.
- Parámetros de Hjorth: Actividad (*activity*), complejidad (*complexity*) y morbilidad (*morbidity*), calculados sobre las tres componentes.

Los primeros cuatro atributos fueron desestimados debido a que no describen el comportamiento dinámico en tiempo de las señales sísmicas como sí lo hacen los atributos estadísticos escogidos. Los últimos tres atributos fueron descartados por el costo computacional que representa encontrarlos, debido a que deben hacerse ajustes polinomiales continuamente hasta que exista una convergencia, lo que demanda gran uso de recursos de tiempo, memoria y procesamiento.

Los atributos fueron calculados haciendo uso de la ejecución de multi-procesos, distribuyendo tareas de forma proporcional a cada uno de los 31 núcleos disponibles en la máquina virtual del CCA. La distribución de las tareas se hace con base en el peso del objeto de trazas derivado del procesamiento de cada Waveform, de tal forma que a cada núcleo le corresponda una cantidad homogénea de datos a procesar.

Para esto, se considera la solución voraz adaptada del *problema de la mochila simple*[92], en la que se busca maximizar la asignación de tareas a cada núcleo, conservando la homogeneidad en la carga (que todos tengan la misma carga):

$$\sum_{i=1}^{n} w_i x_i \leq W \qquad \text{(Ecuación 22)}$$

Donde cada w_i corresponde al peso de objetos i; n es la cantidad de objetos a procesar; el vector de todos los x_i es el vector de solución al problema; y W es el peso máximo por núcleo, cuyo valor corresponde a la media aritmética de todos los pesos del conjunto de los n objetos, en caso de que el mayor peso no supere la media. En caso de que la supere, W corresponde al valor del peso del objeto de mayor peso. El algoritmo seguido para esta tarea es el siguiente:

Algoritmo 8. Pseudocódigo para distribuir las tareas a los núcleos en la extracción de atributos

Entrada: arreglo con objetos de trazas
Salida: arreglo con la distribución de objetos para cada núcleo

```
1.  def core_distr(objetos_waveform)
2.      pesos_objetos ← peso(objetos_waveform).sort()
3.      objetos_waveform ← ordenar_por_peso(objetos_waveform)
4.      cores_q ← os.cpu_count() - 1
5.      max ← max(pesos_objetos)
6.      media ← mean(pesos_objetos)
7.      W = (max<=mean)*mean + (max>mean)*max
8.      para i ←1 hasta cores_q
9.          para waveform en objetos_waveform
10.             si peso(waveform) <= W
11.                 cores[i].append(waveform)
12.                 W -= peso(waveform)
13.         W = (max<=mean)*mean + (max>mean)*max
14.     retornar cores
```

[92] *def*. El problema de la mochila es un problema de optimización en el que se busca incluir la mayor cantidad de ítems de características símiles y con un peso y relevancia determinados, con el fin de incluir estos elementos en una mochila que tiene una capacidad limitada. Adaptado de: HOROWITZ, Ellis; SAHNI, Sartaj. Computing partitions with applications to the knapsack problem. Journal of the Association for Computing Machinery, 21: 277–292, doi:10.1145/321812.321823, MR 0354006. 1974.

En el algoritmo, la función *peso(objetos_waveform).sort()* permite encontrar el peso del objeto en memoria y ordenarlo de forma descendente. El método *os.cpu_count()* permite calcular la cantidad de núcleos disponibles en la máquina y los métodos *max()* y *mean()* permiten encontrar el máximo y la media en un conjunto de datos.

Cálculo del DOP

Para calcular los valores propios de forma computacional, se utiliza la función *eigval* de la librería *obspy.signal.polarization*, que tiene como entradas las tres componentes espaciales de la señal sísmica, los cinco coeficientes polinomiales para el cálculo de la derivada de tiempo, que por omisión es uno, y el factor de normalización, que por omisión es uno. La salida está compuesta por: los tres valores propios de las señales sísmicas, el coeficiente de rectilinealidad, la planaridad de las ondas y los valores propios de la derivada de tiempo.

La función *egval* emplea el paquete LAPACK[93] (*Linear Algebra Package*) en su versión 3.8.0, propuesto por Jurkevics, 1988, en su investigación "Polarization analysis of three-component array data". Este paquete es libre, está escrito en Fortran 90 y provee subrutinas para la solución de ecuaciones lineales, sistemas lineales, valores propios y problemas singulares del álgebra lineal, a través de la implementación de algoritmos que comparten el vector de memoria y son ejecutados con multiprocesos (procesos con ejecución en paralelo).

Una vez que se hallan los valores propios, se aplica la Ecuación 3 para encontrar el Grado de Polarización de la onda tal como se muestra en el Algoritmo 9.

[93] Paquete desarrollado por la Universidad de Tennesee, la Universidad de California, Berkeley, la Universidad de Colorado en Denver y el Numerical Algorithms Group Ltd.

Algoritmo 9. Pseudocódigo para el cálculo del DOP de las ondas sísmicas

Entrada: conjunto de tres componentes espaciales Z, *V* y *E*
Salida: valor de salida correspondiente al DOP

```
15. def DOP(dataZ, dataV, dataE)
16.     eigenvalues ← eigval(datax = dataZ, datay = dataV, dataz = dataE, normf = 1.0)
17.     l1 ← (eigenvalues(0))(0)
18.     l2 ← (eigenvalues(1))(0)
19.     l3 ← (eigenvalues(2))(0)
20.     dop ← (((l1-l2)^2) + (l2-l3)^2 + (l3-l1)^2) / ((l1+l2+l3)^2)
21.     retornar dop
```

Cálculo del RV2T

El RV2T se calcula aplicando la Ecuación 6 tal como se muestra en el Algoritmo 10, teniendo en cuenta que las componentes de entrada han sido filtradas, normalizadas y re-muestreadas:

Algoritmo 10. Pseudocódigo para el cálculo del RV2T de las ondas sísmicas

Entrada: conjunto de tres componentes espaciales Z, *V* y *E*
Salida: valor de salida correspondiente al RV2T

```
1.  def RV2T(dataZ, dataV, dataE)
2.      num ← 0
3.      den ← 0
4.      tamano ← size(dataZ)
5.      para i ←0 hasta tamano
6.          num += (dataZ(i)^2)
7.          den += (dataZ(i)^2) + (dataV(i)^2) + (dataE(i)^2)
8.      fin para
9.      rv2t = num/dem
10.     retornar rv2t
```

Cálculo de la Entropía de Shannon

La Entropía de Shannon se calcula aplicando la Ecuación 7, tal como se muestra en el Algoritmo 11, teniendo en cuenta que la componente de entrada haya sido filtrada, normalizada y re-muestreada. La primera función, *bincount*, es de la librería

de Numpy y permite encontrar el número de ocurrencias de cada valor del arreglo de entrada, es decir, calcula el histograma del arreglo. Puesto que la entropía no contempla elementos nulos, con la función *nonzero*, de la librería Numpy, se especifican los índices del arreglo cuyo valor es distinto de cero. Posteriormente, se agrupan los valores de la probabilidad, con el arreglo original, de tal forma que se forma una matriz de dos columnas, mediante la función *vstack* de la librería Numpy. A esta matriz se le halla la transpuesta, con el fin de operar los valores como filas, mediante la función *T*, de la librería Numpy.ndarray.

Finalmente, se calcula el valor de la Entropía de Shannon, utilizado la función *entropy* de la librería scipy.stats, que tiene como entrada la distribución de probabilidad pk de la variable de estudio y como salida, la Entropía de Shannon, que es el cómputo de $pk * log(pk)$.

Algoritmo 11. Pseudocódigo para el cálculo de Entropía de Shannon de las ondas sísmicas

Entrada: traza de una componente espacial Z, V o E
Salida: valor de salida correspondiente a la Entropía de Shannon

```
1.  def Entropia_Shannon(data)
2.      data ← int(data)
3.      prob ← bincount(data)/size(data)
4.      non_zero ← (nonzero(prob))(0)
5.      salida ← vstack(non_zero, prob(non_zero)).T
6.      entropia ← entropy(salida)
7.      retornar entropia
```

Cálculo de la Asimetría y la Kurtosis

La Asimetría se calcula como se muestra en el Algoritmo 12, teniendo en cuenta que la componente de entrada haya sido filtrada, normalizada y re-muestreada. La

función *skew*[94] de la librería scipy.stats tiene como entrada el arreglo de datos correspondiente a una componente de la señal sísmica, el sesgo estadístico, que por omisión se computa y la política de propagación de los valores vacíos, que por omisión es afirmativa, lo que significa que para valores vacíos, la salida es inexistente o *nan*, de lo contrario, la salida es la asimetría calculada.

Internamente, la asimetría se calcula con la función *skew* de scipy.mstats_basic, que ejecuta la función *moment* de la librería de scipy.stats que permite calcular los momentos centrales estadísticos, teniendo como entrada la traza de datos y el momento a calcular. Para el cálculo del momento central, la librería usa exponenciación por cuadrados, que permite una solución más eficiente.

Algoritmo 12. Pseudocódigo para el cálculo de la asimetría de las ondas sísmicas

Entrada: traza de una componente espacial Z, V o E
Salida: valor de salida correspondiente a la asimetría

```
1. def Skewness(data)
2.     skewness ← skew(data)
3.     retornar skewness
```

Para el cálculo de la Asimetría por muestra, la librería tiene en cuenta el segundo y el tercer momento, aplicando la Ecuación 9 sobre los valores de los momentos calculados: $nval = sqrt((n-1.0)*n)/(n-2.0)*m3/m2**1.5$[95], donde *n* es el arreglo de entrada y m2, m3 son los momentos centrales calculados.

La Kurtosis se calcula como se muestra en el Algoritmo 13, teniendo en cuenta que la componente de entrada haya sido filtrada, normalizada y re-muestreada. La

[94] SCIPY TEAM. Skewness in: SciPy v1.1.0 Reference Guide. SciPy Documentation. Available at: https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.skew.html. 2018.

[95] ZWILLINGER, D; KOKOSKA, S. CRC Standard Probability and Statistics Tables and Formulae. Chapman & Hall: New York. 2000.

función *kurtosis*[96] de la librería scipy.stats tiene como entrada el arreglo de datos correspondiente a una componente de la señal sísmica, la bandera *Fisher* para identificar la preferencia del cálculo por Fisher o Pearson, el sesgo estadístico, que por omisión se computa y la política de propagación de los valores vacíos, que por omisión es afirmativa. En el caso de ser un cálculo de la kurtosis de Fisher, se sustraen tres unidades a la medida de kurtosis de la distribución dada.

Internamente, se ejecuta la función *moment* de la librería de scipy.stats que permite calcular los momentos centrales estadísticos, teniendo como entrada la traza de datos y el momento a calcular. Para el cálculo del momento central, la librería usa exponenciación por cuadrados.

Algoritmo 13. Pseudocódigo para el cálculo de la kurtosis de las ondas sísmicas

Entrada: traza de una componente espacial Z, V o E
Salida: valor de salida correspondiente a la asimetría

```
1. def Kurtosis(data)
2.     kurt ← kurtosis(data)
3.     retornar kurt
```

Para el cálculo de la kurtosis por muestra, la librería tiene en cuenta el segundo y el cuarto momento, aplicando la Ecuación 11 sobre los valores de los momentos calculados: $nval = 1.0/(n-2)/(n-3) * ((n ** 2 - 1.0) * m4/m2 ** 2.0 - 3 * (n-1) ** 2.0)$[97], donde *n* es el arreglo de entrada y m_2, m_4 son los momentos centrales calculados.

[96] SCIPY TEAM. Kurtosis in: SciPy v1.1.0 Reference Guide. SciPy Documentation. Available at: https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.kurtosis.html. 2018.

[97] ZWILLINGER, D; KOKOSKA, S. CRC Standard Probability and Statistics Tables and Formulae. Chapman & Hall: New York. 2000.

Cálculo de la Dimensión de Correlación (D2)

La Dimensión de Correlación se calcula usando el método *corr_dim* de la librería *Nolds* que computa funciones relacionadas con la Teoría del Caos. Los argumentos de entrada son: los datos de la serie de tiempo, la dimensión embebida del conjunto de datos y la forma de cálculo de la distancia entre los pares de puntos, que por defecto es Euclidiana. La dimensión de correlación se calcula usando la Ecuación 15 en la que se hace una aproximación logarítmica al valor buscado mediante la integral de correlación calculada por el método de ajuste polinomial *polyfit* propio de la librería *Numpy*. El algoritmo seguido para el cálculo de la dimensión de correlación es el siguiente:

Algoritmo 14. Pseudocódigo para el cálculo de la CD de las ondas sísmicas

Entrada: traza de una componente espacial Z, V o E, dimensión embebida de la serie de tiempo
Salida: valor de salida correspondiente a la Dimensión de Correlación

```
1. def Correlation_Dim(data, dim)
2.     CD ← nolds.corr_dim (data, dim)
3.     retornar CD
```

3.3.2.7. Proceso de Clasificación

En el proceso de clasificación, se implementan redes neuronales artificiales con diferentes configuraciones, cuyo objetivo es clasificar las observaciones entre onda P o ruido. Éste bloque recibe como entrada un dataset de atributos sísmicos y genera como salida un modelo del mejor clasificador en conjunto con sus métricas de desempeño. El dataset de atributos sísmicos se expresa en forma de matriz, en donde cada fila corresponde a una observación y cada columna a un atributo específico, como se muestra a modo de ejemplo en la Tabla 10. El archivo contenedor del dataset, es generado tras calcular los atributos sísmicos para cada evento y para cada una de las estaciones en consideración. Posteriormente, se exporta en formato *CSV* (*Comma Separated Values*).

Tabla 10. Ejemplo de dataset de atributos sísmicos.

DOP	RV2T	Entropy	Kurtosis	Skewness	CD	EsSismo
0.6564231	0.6924243	3.8994586	5.216548	4.156488	0.315648	1
0.0718984	0.2556487	1.6549723	0.154895	-0.324564	0.955534	0
0.7832564	0.5154658	4.0148748	6.681987	3.985165	0.156486	1
0.2546549	0.1566969	0.9941561	-0.078516	-0.301684	1.002416	0

Con el dataset de atributos sísmicos como entrada, se generan los bloques de entrenamiento a través de optimización de hiperparámetros con *Grid Search* y de validación usando *Monte Carlo Cross-Validation*, como se expresa en la Figura 16.

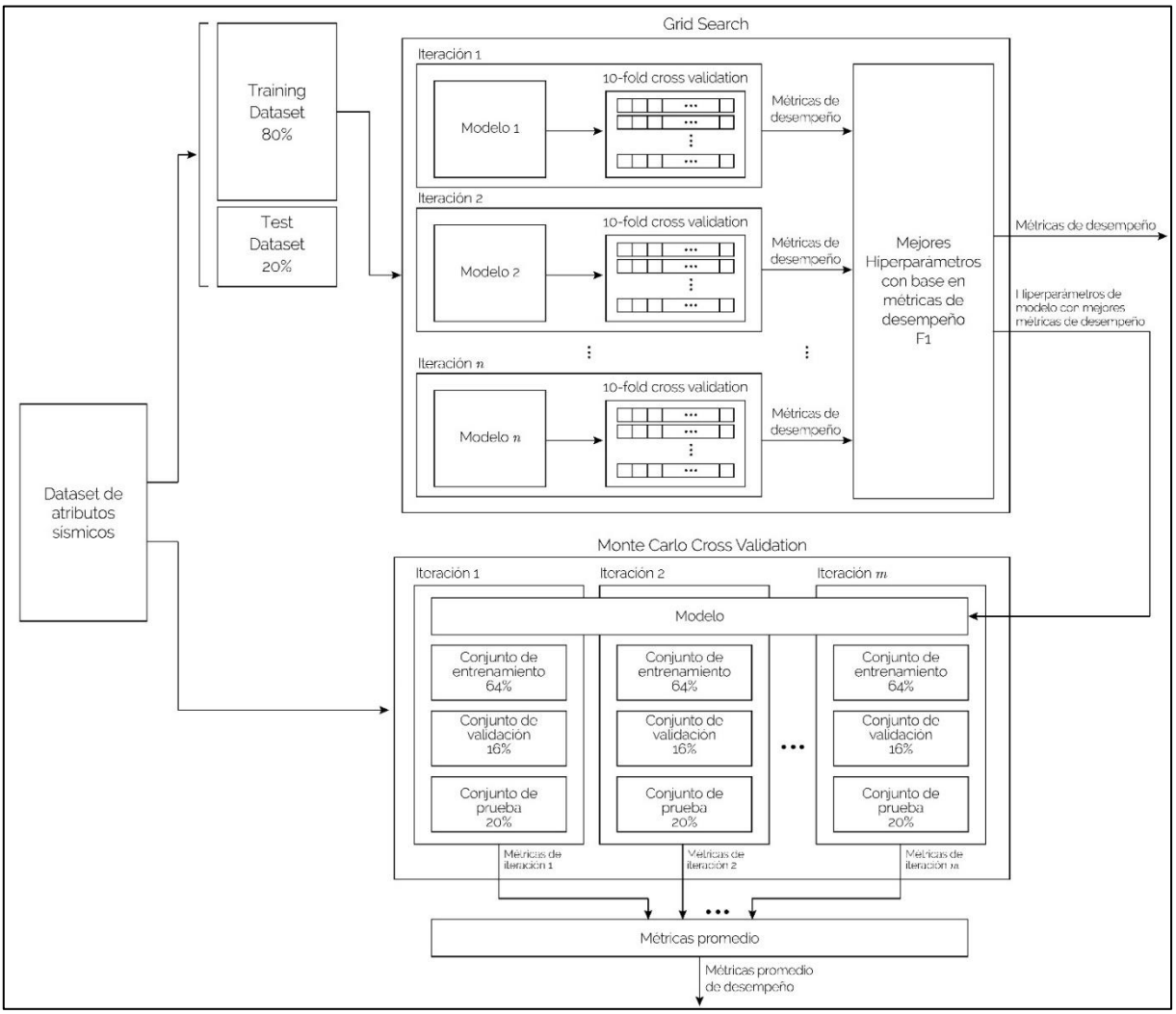


Figura 16. Proceso de clasificación completo.

El proceso general de clasificación es el siguiente:

1. Se adquiere el dataset de atributos sísmicos en forma de matriz.
2. Se estipulan semillas pseudoaletorias para asegurar la repetibilidad de los resultados.
3. Se divide el dataset de atributos sísmicos: 80% para entrenar (*training set*) y 20% para probar el desempeño (*test set*) y se proveen como entrada al bloque de *Grid Search*.
4. Se definen los hiperparámetros que se desean optimizar (*grid*), las métricas de desempeño, la cantidad de *folds* y de núcleos de procesamiento a usar.
5. Se generan *n* iteraciones correspondientes a la cantidad de combinaciones de hiperparámetros.
6. En cada iteración se usa *K-fold Cross-Validation* para determinar las métricas de cada modelo para cada combinación de hiperparámetros.
7. Se selecciona la combinación de hiperparámetros con mejores métricas y se exporta el modelo de red neuronal al bloque de *Monte Carlo Cross-Validation*.
8. Se generan *m* iteraciones en donde se entrena el modelo selecto en *Grid Search* con el dataset distribuido *m* veces de forma aleatoria entre *training set* (64%), *validation set* (16%) y *test set* (20%).
9. Se calculan las métricas de desempeño del modelo, promediando las métricas provistas por las *m* iteraciones del bloque *Monte Carlo Cross-Validation.*

Para realizar estos procedimientos a nivel algorítmico, se plantea la clase *TelluricoANN* (Figura 17), la cual posee métodos de entrenamiento y validación de redes neuronales enfocadas en la tarea de clasificación.

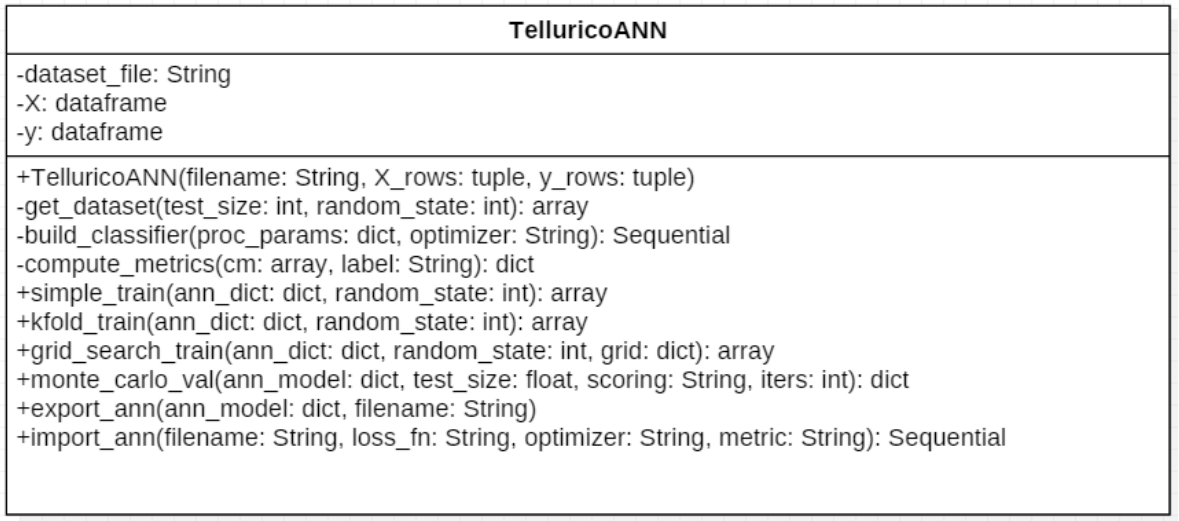


Figura 17. Diagrama de la clase TelluricoANN.

Como se puede apreciar en la Figura 17, existen múltiples métodos para realizar el entrenamiento de una red neuronal. Los métodos presentes en la clase *TelluricoANN* son los siguientes:

- *TelluricoANN*: es el constructor de la clase. Toma como parámetros de entrada el nombre del archivo del dataset (*filename)*, los índices de las columnas que corresponden a las variables independientes (*X_rows*) y los índices de las columnas asociadas a las variables dependientes (*y_rows*), mediante el uso de la librería *pandas*, con el método *read_csv* [98]. Se encarga de separar entradas de salidas e inicializar la instancia de la clase.

[98] PANDAS: DATA STRUCTURES FOR STATISTICAL COMPUTING IN PYTHON, McKinney. read_csv in: pandas v0.24.4 Reference Guide. Pandas Documentation. Available at: https://pandas.pydata.org/pandas-docs/stable/generated/pandas.read_csv.html. 2018.

- *get_dataset*: se encarga de dividir el dataset en *training set* y *test set* mediante el uso del método *train_test_split* de la librería *sklearn.model_selection*[99]. Toma como entrada el porcentaje del dataset asignado al *test_set* (*test_size*) y la semilla de generación de números pseudoaleatorios, para la selección de las observaciones de los grupos de *training* y *test set* (*random_state*), además de normalizar los datos a una misma escala, mediante la clase *StandardScaler*[100] de la librería *sklearn.preprocessing*. Provee como salida un arreglo de *dataframes* en donde se separan las variables independientes de *training set* y las de *test set* (*X_train, y_train, X_test, y_test*). Se usa de forma privada dentro de la clase, por los métodos *simple_train, kfold_train, grid_search_train* y *monte_carlo_val*, para generar los modelos de ANN.

- *build_classifier*: este método se usa para implementar la arquitectura del clasificador. Se usa de forma privada dentro de la clase, por los métodos *simple_train, kfold_train, grid_search_train* y *monte_carlo_val*, para generar los modelos de ANN. Toma como entrada un diccionario en donde se especifican las características estructurales de la red neuronal (cantidad de capas ocultas, funciones de activación, etc) y un *String* especificando el algoritmo de optimización a usar (Adam, Adadelta, Adagrad, etc). Utiliza las clases

[99] SCIKIT-LEARN: MACHINE LEARNING IN PYTHON, Pedregosa *et al*. train_test_split in: scikit-learn v0.19.2 Reference Guide. Scikit-learn Documentation. Available at: http://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html. 2018.

[100] SCIKIT-LEARN: MACHINE LEARNING IN PYTHON, Pedregosa *et al*. StandardScaler in: scikit-learn v0.19.2 Reference Guide. Scikit-learn Documentation. Available at: http://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html. 2018.

Sequential[101] de *keras.models* y *Dense*[102] de *keras.layers* para configurar las características generales de la red y las características de sus capas, respectivamente.

- *compute_metrics*: se encarga de realizar el cálculo de las métricas de desempeño más comunes asociadas a un modelo de clasificación (*FP rate, TP rate, Precision, Accuracy, Recall F1-Score*). Toma como entrada una matriz de confusión (*confusion_matrix*[103] de la librería *sklearn.metrics*) y un *label* para etiquetar la salida. Retorna un diccionario con el conjunto de métricas descrito.

- *simple_train*: este método realiza un entrenamiento singular de la red neuronal. Toma como entrada un diccionario que contenga todos los parámetros necesarios para entrenar la red, sin hacer optimización *de hiperparámetros* (*Grid Search*) ni usar técnicas de validación (*K-fold Cross Validation*) y el valor de la semilla de generación de números pseudoaleatorios. Implementa el método *get_dataset*, para hacer la separación entre *training, validation* y *test set*. Hace uso del método *build_classifier* para ensamblar la red neuronal y el método *fit* de la clase *Sequential* para entrenarla. Su salida es una instancia de la clase *Sequential* y un conjunto de métricas de desempeño para *validation set* y *test set* en forma de diccionario, a través del método *compute_metrics*.

[101] KERAS: THE PYTHON DEEP LEARNING LIBRARY, Chollet *et al*. Secuential in: keras v2.2.1 Reference Guide. Keras Documentation. Available at: https://keras.io/models/sequential/. 2018.

[102] KERAS: THE PYTHON DEEP LEARNING LIBRARY, Chollet *et al*. Dense in: keras v2.2.1 Reference Guide. Keras Documentation. Available at: https://keras.io/layers/core/#dense. 2018.

[103] SCIKIT-LEARN: MACHINE LEARNING IN PYTHON, Pedregosa *et al*. confusion_matrix in: scikit-learn v0.19.2 Reference Guide. Scikit-learn Documentation. Available at: http://scikit-learn.org/stable/modules/generated/sklearn.metrics.confusion_matrix.html. 2018.

- *kfold_train*: se encarga de realizar un entrenamiento singular de la red neuronal, implementando el método *simple_train,* pero generando una evaluación del modelo de clasificación a través de *K-Fold Cross Validation*. Toma como entrada un diccionario especificando las características estructurales de la red neuronal, además de incluir la cantidad de *folds,* la métrica y la cantidad de núcleos de procesamiento a usar para la evaluación y el valor de la semilla de generación de números pseudoaleatorios. Se utiliza el método *cross_val_score*[104] de la librería *sklearn.model_selection* para la validación y la clase *KerasClassifier*[105] de la librería *keras.wrappers.scikit_learn*, que sirve como puente para entrenar con los métodos de la clase *Sequential* y evaluar con *cross_val_score*. Se obtiene como salida una instancia de la clase *Sequential* y un conjunto de métricas de desempeño asociadas al clasificador en su proceso de entrenamiento y a su respectivo *test set*, a través del método *compute_metrics*.

- *grid_search_train*: este método se encarga de realizar el proceso de optimización de hiperparámetros mediante *Grid Search*. Toma como entrada un diccionario que contiene los parámetros de la red neuronal, la rejilla (*grid*) de hiperparámetros a optimizar, la cantidad de *folds,* la métrica para seleccionar el mejor conjunto de hiperparámetros y la cantidad de núcleos de procesamiento a usar para la búsqueda (*n_jobs*). Se usa la clase *GridSearchCV*[106] de la librería *sklearn.model_selection* para la optimización de hiperparámetros, que a su vez

[104] SCIKIT-LEARN: MACHINE LEARNING IN PYTHON, Pedregosa *et al*. cross_val_score in: scikit-learn v0.19.2 Reference Guide. Scikit-learn Documentation. Available at: http://scikit-learn.org/stable/modules/generated/sklearn.model_selection.cross_val_score.html. 2018.

[105] KERAS: THE PYTHON DEEP LEARNING LIBRARY, Chollet *et al*. KerasClassifier in: keras v2.2.1 Reference Guide. Keras Documentation. Available at: https://keras.io/scikit-learn-api/. 2018.

[106] SCIKIT-LEARN: MACHINE LEARNING IN PYTHON, Pedregosa *et al*. GridSearchCV in: scikit-learn v0.19.2 Reference Guide. Scikit-learn Documentation. Available at: http://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html. 2018.

usa *K-fold Cross Validation* para evaluar los modelos generados. Retorna como salida un modelo de clasificación de la clase *Sequential* y un conjunto de métricas de desempeño asociadas al clasificador en su proceso de entrenamiento y a su respectivo *test set*.

- *monte_carlo_val*: se encarga de evaluar un clasificador mediante el uso de *Monte Carlo Cross Validation*. Toma como entrada un modelo de clasificación de la clase *Sequential*, el tamaño del *test set* (*test_size*), la métrica para evaluar los modelos durante su entrenamiento (*scoring*), la cantidad de iteraciones (*iters*) y la cantidad de núcleos de procesamiento a usar para la búsqueda (*n_jobs*). Se usa el método *get_dataset* para generar un conjunto de *training, validation y test set* distinto por cada iteración. Se usa el método *simple_train* para entrenar el modelo ingresado y obtener un conjunto de métricas de *validation* y *test set* por cada iteración, a través del método *compute_metrics*. Se retorna el promedio de las métricas de todas las iteraciones.

- *export_ann*: se encarga de exportar el modelo de clasificación a archivos externos. Usa el método *classifier.model.to_json* de la clase *Sequential*. Genera un archivo *.json* para almacenar la información sobre la arquitectura de la red neuronal y un archivo *.h5* para almacenar los pesos, mediante el método *model.save_weights* de la clase *Sequential.*

- *import_ann*: se encarga de importar un modelo de clasificación a partir de archivos externos. Toma como entrada el nombre de los archivos *.json* y *.h5* para cargar la arquitectura y los pesos de la red, la función de error (*loss_fn*), el algoritmo de optimización (*optimizer*) y la métrica asociada al entrenamiento (*metric*). Usa los métodos *modelo_from_json* de la librería *keras.models*, *load_weights* y *compile* de la clase *Sequential* para cargar los archivos externos y generar el clasificador. Retorna una instancia de la clase *Sequential*.

Arquitectura del Clasificador

El tipo de red neuronal selecto para clasificar entre onda P y ruido, es *feedforward backpropagation*. La implementación de la arquitectura del clasificador se realiza mediante el método *build_classifier* de la clase *TelluricoANN* (Figura 17). Allí se deben especificar las características de la red neuronal con la que se va a realizar la tarea de clasificar el dataset. Teniendo en cuenta que la clase *Sequential*, de la librería *keras.models* representa la abstracción más general de la red neuronal de tipo *feedforward backpropagation* y que ésta implementa instancias de la clase *Dense* para cada capa de la ANN, se tiene que los parámetros configurables o hiperparámetros en consideración, son los siguientes:

- Cantidad de capas ocultas
- Cantidad de neuronas por capa
- Función de activación por capa
- Función de error
- Algoritmo de optimización
- Batch size
- Cantidad de epochs

Para inicializar cada una de las capas de la red neuronal con la clase *Dense*, es necesario especificar la cantidad de neuronas de entrada (para el caso de la capa de entrada), la cantidad de neuronas de salida y el tipo de función de activación. A su vez, para compilar la estructura de la instancia de la ANN, mediante la clase *Sequential*, es necesario especificar un algoritmo de optimización, una función de error y una o más métricas de desempeño asociadas al entrenamiento.

La elección de cada uno de estos hiperparámetros tiene un impacto significativo en el desempeño una red neuronal, en términos generales. Sin embargo, no es viable en cuanto a recursos computacionales, el optimizar todos los hiperparámetros

expuestos. Por ello, se seleccionaron los siguientes hiperparámetros para el proceso de optimización: algoritmo de optimización, *Batch size* y cantidad de *epochs*. Según esto, en la Figura 18 se puede apreciar un ejemplo de diccionario de entrada, que especifica las características necesarias para generar el modelo de red neuronal, requerido para iniciar el proceso de optimización de hiperparámetros.

```
ann = {   'init_info' : {
              'dataset_filename' : 'attr_matrix_prot04_1stats.csv',
              'X_rows'           : (0,14),
              'y_rows'           : (14,15)
          },
          'proc_params' : {
                      'test_size'       : 0.2,
                      'validation_size' : 0.2,
                      'metric'          : 'accuracy',
                      'arch'            : (14, 7, 4, 1),
                      'hidden_act_fn'    : 'relu',
                      'output_act_fn'    : 'sigmoid',
                      'loss_fn'         : 'binary_crossentropy',
                      'cv'              : 10,
          }

      }
```

Figura 18. Diccionario de inicialización de red neuronal.

Como se denota en la Figura 18, se tiene un conjunto de parámetros de inicialización que comprenden el nombre del archivo del dataset (*dataset_filename)* y los rangos de variables independientes (*X_rows*) y dependientes (*y_rows*). Se puede apreciar que los atributos consisten en 14 variables independientes y una dependiente. En *proc_params* se pueden apreciar parámetros como el tamaño del *test set* (*test_size*) y *validation set* (*validation_size*), así como la métrica de desempeño para el entrenamiento (*metric*).

Se puede observar que la arquitectura de la red, en cuanto a la cantidad de capas y neuronas por capa, se expresa en el parámetro *arch*. Para este ejemplo, se tienen 14 neuronas conformando de entrada, dos capas ocultas con 7 y 4 neuronas respectivamente y la capa de salida con una neurona asociada.

De la misma forma se especifica la función de activación de las capas ocultas (*hidden_act_fn*), que corresponde a la función rectificador (*relu*). Para la capa de salida, se especifica la función de activación sigmoidea (*sigmoid*), a través del parámetro *output_act_fn*. Por último, se tiene que la función de error es asignada como *log los* (*binary_crossentropy*), a través del parámetro *loss_fn* y que la cantidad de *folds* para la validación en el proceso de optimización de hiperparámetros toma un valor de 10 a través de la llave de diccionario *cv* (*cross-validation*).

Optimización de los hiperparámetros

Los posibles valores que pueden tomar los hiperparámetros seleccionados para el proceso de optimización mediante *Grid Search,* fueron escogidos mediante una revisión a los autores que proponen este tipo de parámetros.

Se consideraron los algoritmos de optimización, *Adadelta*, *RMSprop y Adam*, tal como lo propone Ruder (2017)[107] en un estudio en el que realiza una comparación entre el comportamiento de los algoritmos de optimización de redes neuronales, más usados en la actualidad. Allí concluye que dichos algoritmos de optimización presentan un desempeño similar en situaciones semejantes.

En cuanto al *batch size* y la cantidad de *epochs*, se consideraron los rangos de valores [8, 16, 32, 64, 128] y [10, 50, 100, 250, 500] respectivamente. Dichos rangos se plantean en concordancia con los resultados mostrados por Thoma (2017)[108] en donde se relaciona el desempeño de redes neuronales, con el *batch size* y la

[107] RUDER, Sebastian. An overview of gradient descent optimization algorithms. Insight Centre for Data Analytics, NUI Galway. arXiv:1609.04747v2 [cs.LG]. 2017.

[108] THOMA, Martin. Analysis and Optimization of Convolutional Neural Network Architectures. Department of Computer Science, Institute for Anthropomatics and FZI Research Center for Information Technology. arXiv:1707.09725v1 [cs.CV]. 2017.

cantidad de *epochs* en diferentes rangos, aplicados con *datasets* de prueba reconocidos.

Para encontrar los mejores hiperparámetros para los prototipos planteados, se usa el método *grid_search_train* de la clase *TelluricoANN* (Figura 18). El proceso general de búsqueda de los mejores hiperparámetros se puede apreciar en la Figura 19, teniendo en cuenta los hiperparámetros de optimización: algoritmo de optimización, *batch size* y cantidad de *epochs*.

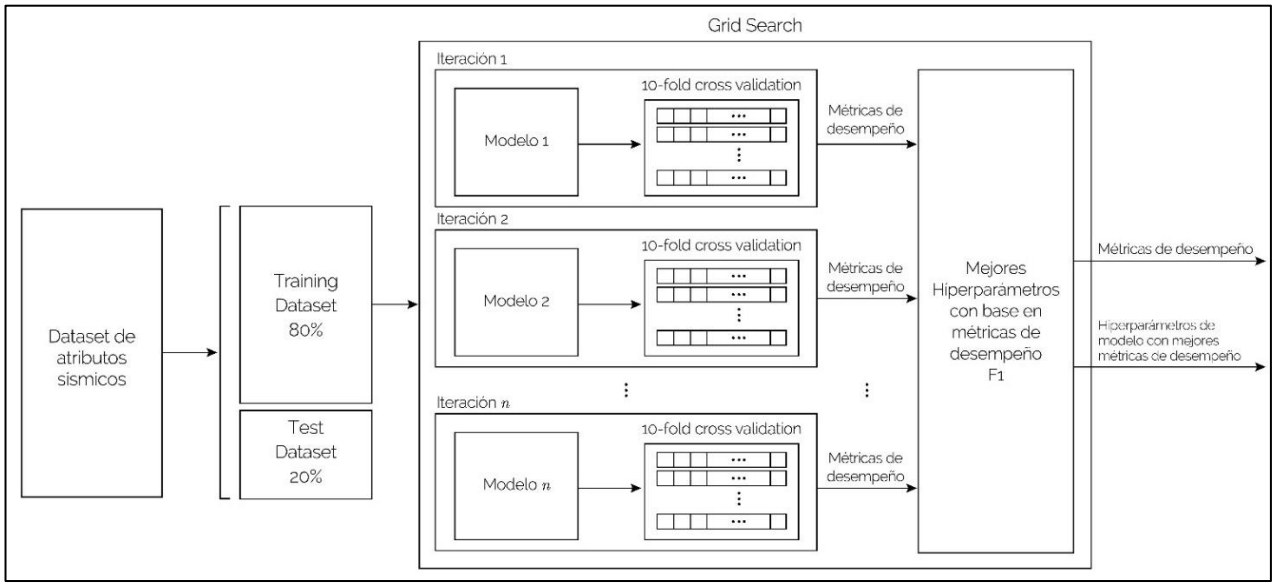


Figura 19. Proceso de optimización de hiperparámetros usando Grid Search.

La clase *GridSearchCV* de la librería *sklearn.model_selection* presente en el método *grid_search_train* de la clase *TelluricoANN* se usa para instanciar el objeto encargado de realizar el procedimiento de búsqueda de los mejores hiperparámetros. Las entradas requeridas por el constructor de esta clase son:

- *estimator*: consiste en el objeto que contiene la arquitectura del clasificador. Es una instancia de la clase *KerasClassifier* proveniente de la librería *keras.wrappers.scikit_learn*. La instancia de *KerasClassifier* requiere como

entrada una función de construcción del clasificador (*build_classifier*, cuya salida es un objeto de tipo *Sequential*).

- *param_grid*: matriz o rejilla de hiperparámetros a considerar en forma de diccionario de datos.
- *scoring*: métrica expresada en un *String* que se usa para comparar los modelos generados y sugerir la combinación de hiperparámetros más adecuada.
- c*v*: cantidad de *folds* expresado en un *int*, para realizar el procedimiento *K-fold Cross Validation* con cada uno de los modelos generados.
- *n_jobs*: cantidad de núcleos de procesamiento a usar para la búsqueda, expresado en un *int.*
- *refit*: booleano que indica si se provee como valor de retorno, el mejor modelo entrenado con los hiperparámetros selectos.

La métrica usada para seleccionar el mejor modelo es *F1-Score*, ya que con esta métrica es posible generalizar el comportamiento del modelo en cuanto a su habilidad para clasificar correctamente las observaciones, tanto negativas como positivas.

Una vez instanciado el objeto *GridSearchCV,* se obtiene el *training set*, separado en variables independientes y dependientes (*X_train, y_train*), mediante el método *get_dataset* y se usa el método *fit* de la misma clase, tomando como argumento *X_train* y *y_train* para iniciar el proceso de optimización de hiperparámetros.

Cada iteración *n* de *Grid Search (*Figura 20) consiste en una combinación específica de hiperparámetros en donde se genera un modelo de tipo *Sequential*, asociando la arquitectura expresada en el diccionario de entrada del método *grid_search_train* y los hiperparámetros que se consideran por cada iteración. Dicho modelo se valida mediante *K-fold Cross Validation*, teniendo *K = cv*, como número de *folds* y usando el *training set* ingresado en *GridSearchCV.fit*.

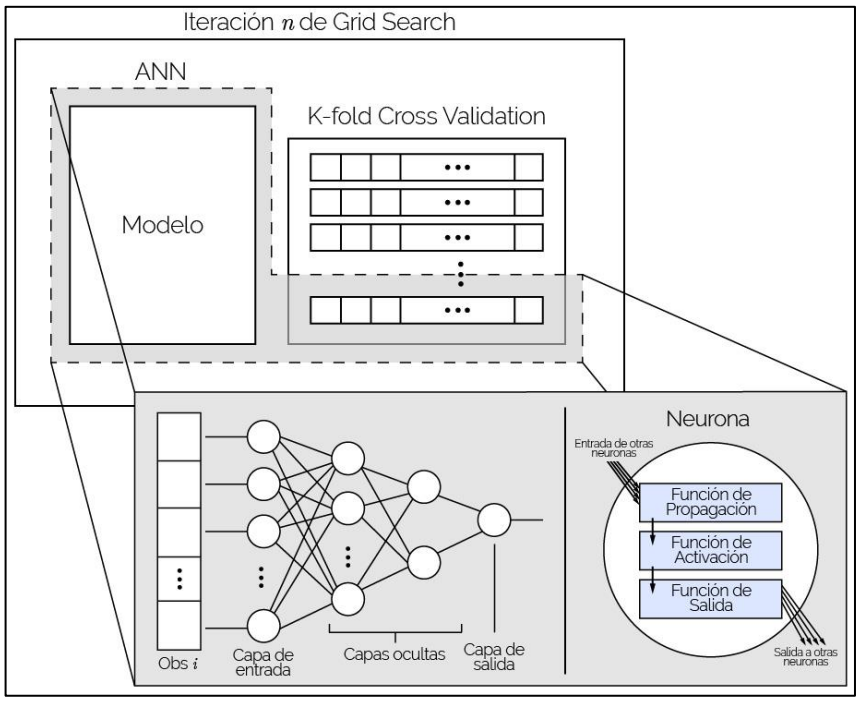


Figura 20. Diagrama de bloques de las iteraciones de Grid Search.

Como se puede apreciar en la Figura 20, por cada iteración de *Grid Search* se generan *K = cv* entrenamientos. Para este procedimiento, se seleccionó *K = 10*, como lo sugiere Davison (1997)[109]. Se tienen 3 hiperparámetros para optimizar, el primero de ellos puede tomar 3 valores (algoritmo de optimización), el segundo puede tomar 5 valores (*batch size*) y el tercero puede tomar 5 valores (cantidad de *epochs*). Según esto, se puede calcular que el total de entrenamientos o modelos generados durante el proceso de *Grid Search* es de $10 * 3 * 5^2 = 750$.

La cantidad de entrenamientos requerida para optimizar un conjunto de hiperparámetros requiere de prolongados tiempos de cómputo, especialmente cuando se usan *datasets* de gran volumen. Por ello en la clase *GridSearchCV,* se permite la opción de realizar combinaciones de hiperparámetros en paralelo

[109] DAVISON, A.C.; HINKLEY, D.V. Bootstrap Methods and their Applications. Cambridge University Press, Cambridge. 1997.

mediante el argumento *n_jobs.* Para realizar este procedimiento, se hace uso de la máquina virtual del CCA (Tabla 7), en donde es posible usar 31 núcleos de procesamiento de forma simultánea.

Tras finalizar el proceso de optimización de hiperparámetros, es posible acceder a los resultados de este mediante los siguientes atributos de la clase *GridSearchCV:*

- *best_score*: indica el valor de la métrica de desempeño, asociada al mejor clasificador considerado.
- *best_params*: retorna un diccionario con la selección de los mejores hiperparámetros.
- *best_estimator_*: provee el modelo de clasificación de tipo *Sequential*, entrenado con los mejores hiperparámetros.

Continuando con el flujo del método *grid_search_train*, al obtener los resultados de la búsqueda de los mejores hiperparámetros, se generan las métricas de desempeño para el mejor clasificador, usando el *test set*, mediante el método *compute_metrics* de la clase *TelluricoANN.*

Validación cruzada

Para aumentar la confiabilidad de las métricas asociadas al modelo de clasificación, se usa la técnica *Monte Carlo Cross Validation*. En éste método, a partir de un modelo de red neuronal de entrada, se realizan *m* entrenamientos y se calculan *m* conjuntos de métricas, distribuyendo las observaciones del *dataset* completo en grupos de *training set, validation set* y *test set*, como se muestra en la Figura 21.

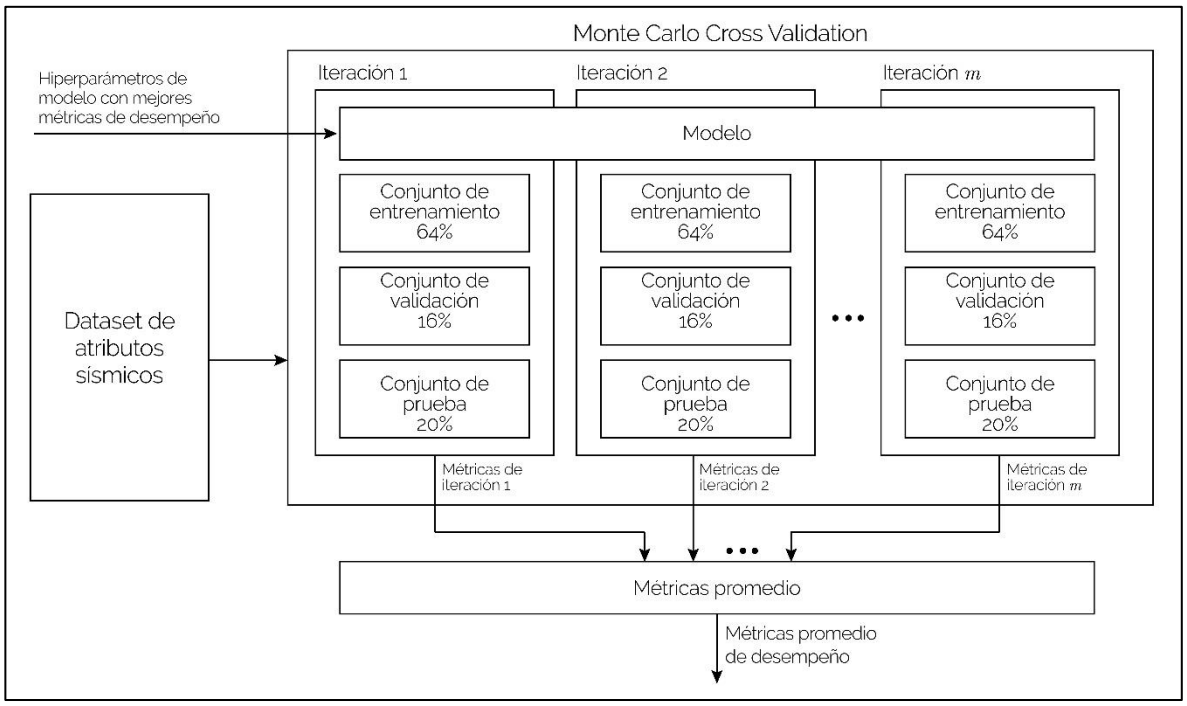


Figura 21. Proceso general de Monte Carlo Cross Validation.

El método *monte_carlo_val* de la clase *TelluricoANN* se encarga de realizar el proceso descrito en la Figura 21. Los argumentos de entrada de este método son los siguientes:

- *classifier*: instancia de la clase *Sequential*. Representa el modelo de red neuronal completo, con los mejores hiperparámetros, seleccionados a partir de *Grid Search.*
- *test_size*: número tipo *float.* Corresponde al porcentaje del *dataset* asignado como *test set*. El mismo porcentaje se usa para dividir el *training set* en *training set* y *validation set.*
- *scoring*: cadena de caracteres. Métrica usada para evaluar el desempeño del modelo durante el entrenamiento, usando el *validation set*.
- *iters*: número tipo *int.* Cantidad de procesos de entrenamiento y validación.

Para cada una de las iteraciones, se hace llamado al método *get_dataset*, para dividir el *dataset* en *training, validation* y *test set.* Del 100% del *dataset*, se destina

un 64% para el *training set* o conjunto de entrenamiento, 16% para el *validation set* o conjunto de validación y, 20% para el *test set* o conjunto de prueba. Por cada iteración se tiene un *random_state* distinto como argumento para *get_dataset*. De esta manera, las observaciones de todo el *dataset* tienen la posibilidad de aparecer en cualquiera de los tres grupos (*training, validation, test)*.

El número de iteraciones usadas es *iters = 10,* obteniendo así 10 distribuciones de observaciones distintas. Al igual que con el método *get_dataset*, en cada iteración se llama al método *simple_train*, con el objetivo de realizar un entrenamiento simple (entiéndase por simple a un entrenamiento sin técnicas de validación adicionales ni optimización de hiperparámetros). A diferencia de los entrenamientos de *Grid Search*, en el bloque de *Monte Carlo Cross Validation* solo se tiene en cuenta un modelo, lo que varía es la distribución de los datos.

En cada uno de los entrenamientos simples se entrena la red neuronal con el *training set*, se prueba su desempeño por cada *epoch* con el *validation set* y finalmente se prueba el desempeño general con el *test set*. Mediante el método *compute_metrics,* usando como argumento las matrices de confusión de validación y prueba, se obtienen las métricas asociadas al *validation set* y al *test set*, por cada entrenamiento simple, es decir, por cada una de las 10 iteraciones.

Tras obtener los 10 conjuntos de métricas, se calcula el promedio de cada tipo de métrica para el conjunto de validación y de prueba. Las métricas promedio obtenidas a partir del proceso descrito permiten determinar de forma clara el desempeño real del clasificador y, en conjunto con el modelo selecto en *Grid Search*, constituyen la salida general del proceso de clasificación.

3.3.3. Complejidad temporal

El desempeño en tiempo del clasificador permite identificar la rapidez con que es clasificado un evento sísmico, desde que una ventana ingresa, hasta que se clasifica como sismo o ruido, teniendo en cuenta los procesos de extracción de atributos y clasificación. Los procesos de selección de muestras, estaciones, preprocesamiento de datos, entrenamiento y validación del clasificador no son tenidos en cuenta para este estudio, debido a que son procesos de carácter previo a la clasificación del evento. La medición del tiempo de los procesos de extracción y clasificación se hacen separadamente de forma secuencial, conservando las mismas características del ambiente computacional. Se usó la máquina virtual del CCA con un solo núcleo para la ejecución de los procesos.

3.3.3.1. Complejidad temporal de la extracción de atributos

Para determinar el desempeño en tiempo del proceso de extracción de atributos debe tenerse en cuenta que las señales de entrada ya se encuentran filtradas, normalizadas y re-muestreadas, tal como se ha explicado en las secciones anteriores. Para medir el tiempo se hizo uso del método *time()* de la librería *time* escrita en C, que permite contar la cantidad de segundos decimales que han pasado desde el primero de enero de 1970 hasta el momento en el que es llamada la función, teniendo en cuenta el reloj ajustado del sistema operativo. Por ejemplo, si la función es llamada el primero de enero de 2018, la cantidad de segundos será aproximadamente de 1.513.728.000.

Las variables tenidas en cuenta para la medición del tiempo son:

- El tamaño de la ventana desde 5 muestras hasta 250 muestras con pasos de 5 muestras (49 ventanas evaluadas), con el fin de identificar la relación

existente entre un aumento y disminución de la cantidad de muestras, con respecto al tiempo de extracción de los atributos.

- La cantidad de estaciones desde 1 estación hasta 32 estaciones con incrementos de una estación (32 estaciones evaluadas), variable que está directamente relacionada con la cantidad de ventanas a procesar, lo que desemboca en un incremento en la cantidad de operaciones, factor que afecta el desempeño en tiempo del proceso de extracción de atributos.
- El desempeño en tiempo fue registrado para el evento sísmico ocurrido el 18 de junio de 2013 con epicentro en el departamento de Santander.

El registro del tiempo se hizo promediando el desempeño al hallar los atributos como una combinatoria entre las dos variables mencionadas (1.568 combinaciones), cada una ejecutada 50 iteraciones, para un total de 78.400 iteraciones. El Algoritmo 15 fue desarrollado para esta medición.

Al finalizar el proceso de cálculo del tiempo en el proceso de extracción de los atributos, se obtiene una matriz de dos dimensiones en la que las columnas representan cada una de las ventanas y las filas representan la cantidad de estaciones. El valor en cada celda (i, j) de la matriz es el tiempo que tarda el proceso en extraer los atributos para la cantidad de estaciones i y ventana j.

Algoritmo 15. Pseudocódigo para la medición del tiempo de la extracción de atributos

Entrada: trazas de las componentes del evento sísmico registrado en un arreglo de eventos y estaciones que registraron el evento (32).
Salida: matriz de tiempos.

```
1.  def time_complex(event, stats)
2.      iterations ← 50
3.      init_window ← 5
4.      incr_window ← 5
5.      max_window ← 250
6.      acumul ← 0.0
7.      stats_q ← len(stats)
8.      time ← float[stats, ((max_window – init_window)/incr_window)]
9.      para window ← init_window hasta max_window en pasos de incr_window
10.         para stat ← 1 hasta stats_q
11.             para iter ← 1 hasta iterations
12.                 start ← time.time()
13.                 para station en stats[:stats_q]
14.                     canalZ ← event[station].canalZ
15.                     canalN ← event[station].canalN
16.                     canalE ← event[station].canalE
17.                     DOP(canalZ, canalN, canalE)
18.                     RV2T(canalZ, canalN, canalE)
19.                     Entropia_Shannon(canalZ)
20.                     Skewness(canalZ)
21.                     Kurtosis(canalZ)
22.                     Correlation_Dim(canalZ)
23.                 end ← time.time()
24.                 acumul ← acumul + (end – start)
25.             times[stat][window/ incr_window-1] ← acumul/iterations
26.             acumul ← 0.0
27.     retornar times
```

3.3.3.2. Complejidad temporal del proceso de clasificación

Para determinar el desempeño en tiempo del proceso de clasificación debe tenerse en cuenta que las redes ya se encuentran entrenadas con el mejor modelo identificado en el proceso de optimización de hiperparámetros con *Grid Search*. Para medir el tiempo se hizo uso nuevamente del método *time()* de la librería *time.*

Las variables tenidas en cuenta para la medición del tiempo son:

- La cantidad de estaciones de medición de los eventos (4 estaciones). Para calcular el tiempo con estas variaciones se consideraron los 4 modelos de red neuronal desarrollados, pues un cambio en la cantidad de estaciones se traduce en una variación en las matrices de atributos y las ventanas de salida del proceso de selección de ventanas, lo que significa que la red debe ser re-entrenada con este nuevo cambio. El proceso de re-entrenamiento no fue tenido en cuenta para el registro del tiempo.
- La cantidad de observaciones del evento desde 1 hasta 1.000 eventos. El desempeño en tiempo fue registrado para el evento sísmico ocurrido el 18 de junio de 2013 con epicentro en el departamento de Santander.

El registro del tiempo se hizo promediando el desempeño al hallar los atributos como una combinatoria entre las dos variables mencionadas (4000 combinaciones), cada una ejecutada en 1000 iteraciones. El Algoritmo 16 fue desarrollado para esta medición.

Al finalizar el proceso de cálculo del tiempo en el proceso de extracción de los atributos, se obtiene una matriz de dos dimensiones en la que las columnas representan cada una de las estaciones y las filas representan la cantidad de observaciones. El valor en cada celda ($i,\ j$) de la matriz es el tiempo que tarda el proceso de clasificar las j observaciones para las i estaciones. No se verificó si la clasificación fuese correcta o incorrecta, solo se registró el tiempo en ejecutar este proceso, teniendo en cuenta únicamente casos clasificados previamente como positivos.

Algoritmo 16. Pseudocódigo para la medición del tiempo del proceso de clasificación

Entrada: matriz de atributos sísmicos extraídos (obs_attrs), vector de redes neuronales entrenadas (anns).
Salida: matriz de tiempos.

```
1.  def time_complex(obs_attrs, anns)
2.      iterations ← 1000
3.      cantidad_obs ← 1000
4.      cantidad_stats ← 4
5.      acumul ← 0.0
6.      time ← float[cantidad_obs, cantidad_stats]
7.      para i ← 1 hasta cantidad_stats
8.          para stats ← 1 hasta cantidad_stats
9.              para iter ← 1 hasta iterations
10.                 start ← time.time()
11.                 anns[stats].preduct(obs_attrs[:i])
12.                 end ← time.time()
13.                 acumul ← acumul + (end – start)
14.             times[stat][window/ incr_window-1] ← acumul/iterations
15.             acumul ← 0.0
16.     retornar times
```

4. RESULTADOS

En la siguiente sección se muestran los resultados obtenidos para los procesos de análisis de estaciones, selección de la muestra, preprocesamiento del conjunto de datos, extracción de atributos y la clasificación de los eventos sísmicos.

4.1. ANÁLISIS DE ESTACIONES Y SELECCIÓN DE LA MUESTRA

4.1.1. Análisis de estaciones

A continuación, se muestran los resultados del proceso de análisis de estaciones, teniendo en cuenta el orden de los procesos descritos en el diagrama de bloques presentado en la Figura 14.

4.1.1.1. Mapeo de ubicación geográfica

En primera instancia se hace un mapeo de las estaciones y eventos sísmicos en la muestra de 60.785 eventos descritos en la Sección 3.1. El conjunto de estaciones disponible, conformado por 85 estaciones de medición sismológica distribuidas por todo el territorio colombiano, se muestra en la Figura 22.

La ubicación de las estaciones sismológicas está asociada a la ubicación de las fallas geológicas. Los registros de actividad sísmica son frecuentemente localizados en las cercanías de las cordilleras que atraviesan el país, por este motivo, se puede evidenciar que en la región Andina hay una alta densidad de las estaciones de medición, en concordancia también con la ubicación de algunos de los centros urbanos más importantes, como las ciudades de Bogotá, Medellín y Bucaramanga. En el Anexo C, se puede observar la ubicación de cada estación expresada en longitud, latitud y altura con respecto al nivel del mar.

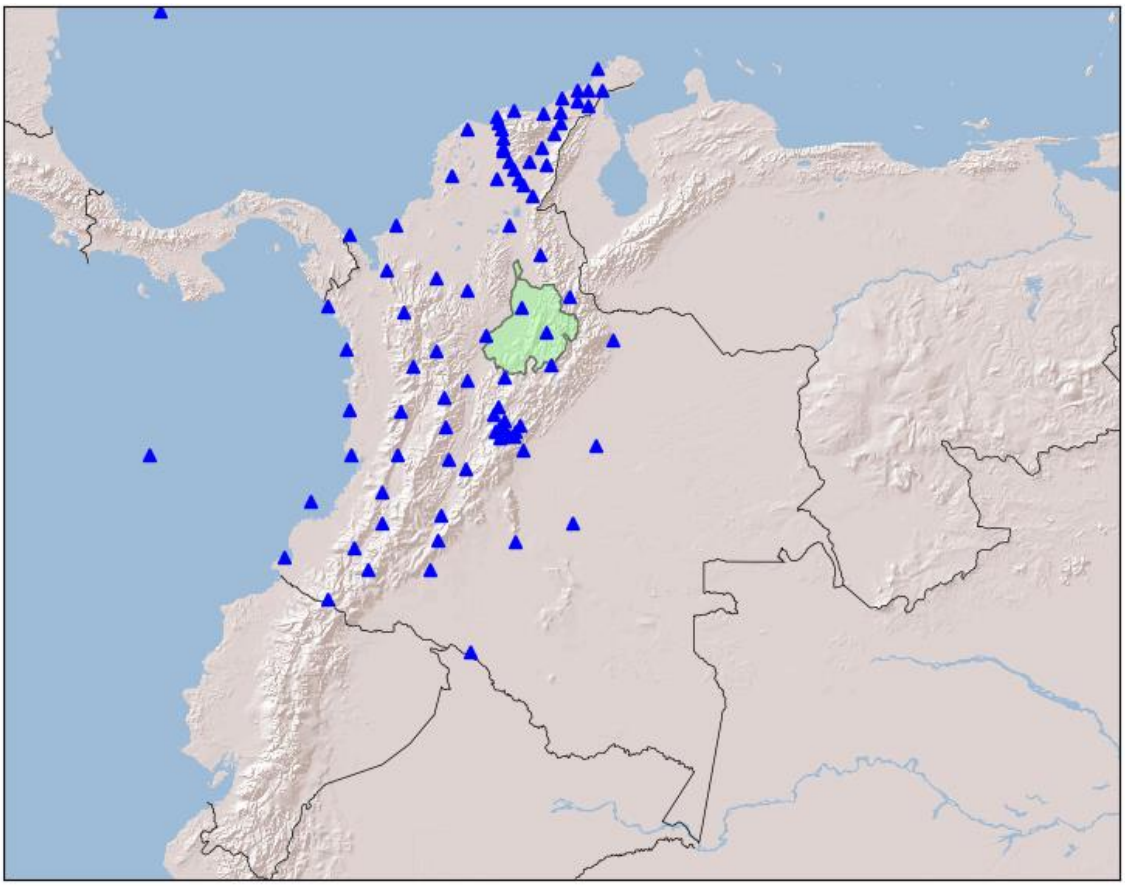


Figura 22. Estaciones sismológicas nacionales de la Red Sismológica Nacional de Colombia.

Debido a que la muestra seleccionada corresponde a los sismos cuyo epicentro ha sido localizado en el departamento de Santander, las estaciones aledañas al territorio santandereano tienen en primera instancia una mayor relevancia de ser consideradas para el proceso de clasificación. Esto se debe a que las ondas sísmicas, al desplazarse, alcanzan en menor tiempo las estaciones, pues los eventos sísmicos ocasionados por las interacciones entre las fallas presentes en el territorio recorren una menor distancia en la geografía del departamento para llegar a ser registrados por las estaciones cercanas.

En la Figura 23 se puede apreciar la relación entre la distribución geográfica de los epicentros sísmicos de la muestra en el departamento de Santander con su división

municipal, en conjunto con la representación de las estaciones sismológicas más cercanas al territorio.

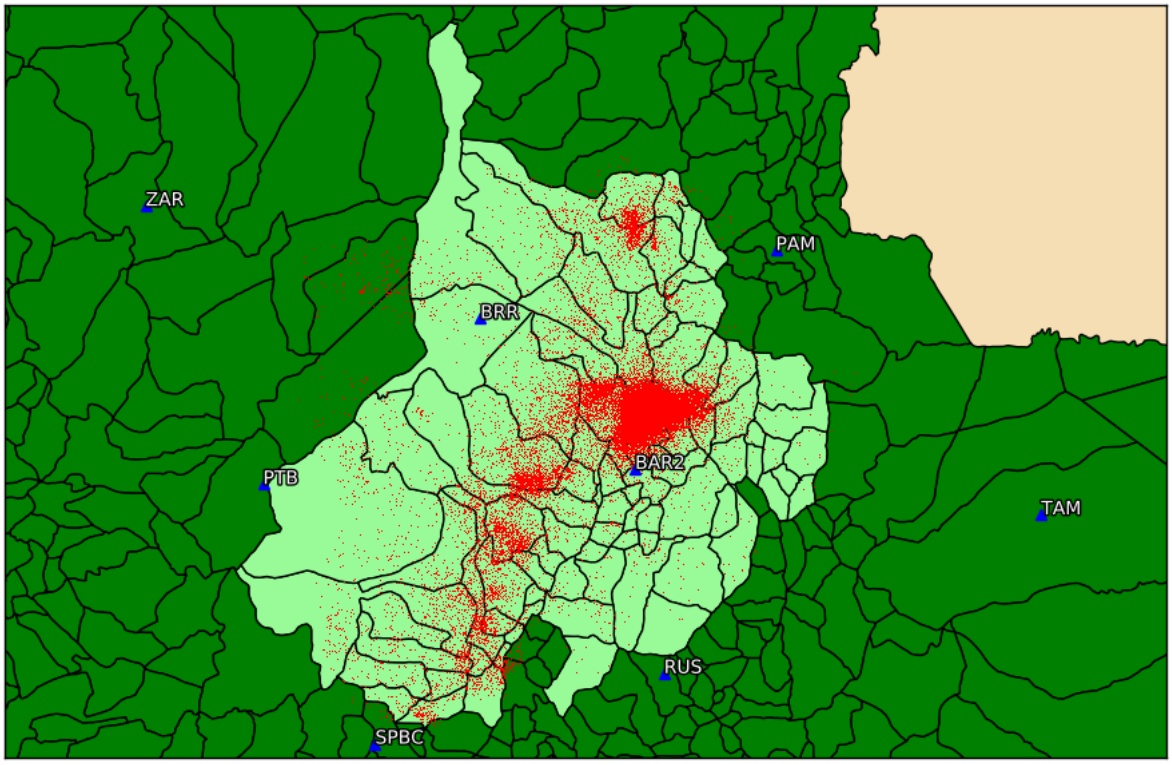


Figura 23. Distribución geográfica de los eventos sísmicos en Santander.

La distribución sísmica en el departamento posiblemente obedezca a la distribución tectónica como se ve en la Figura 24, en la que se aprecia que las aglomeraciones de eventos siguen la trayectoria de la cordillera oriental a través del departamento. Existe un agrupamiento notable de epicentros sísmicos localizado hacia el norte-oriente de la estación BAR2. Esta ubicación corresponde al municipio de Los Santos que, como se ha mencionado, es el segundo nido sísmico más activo del mundo y se encuentra cercano a múltiples fallas a las que debe su actividad.

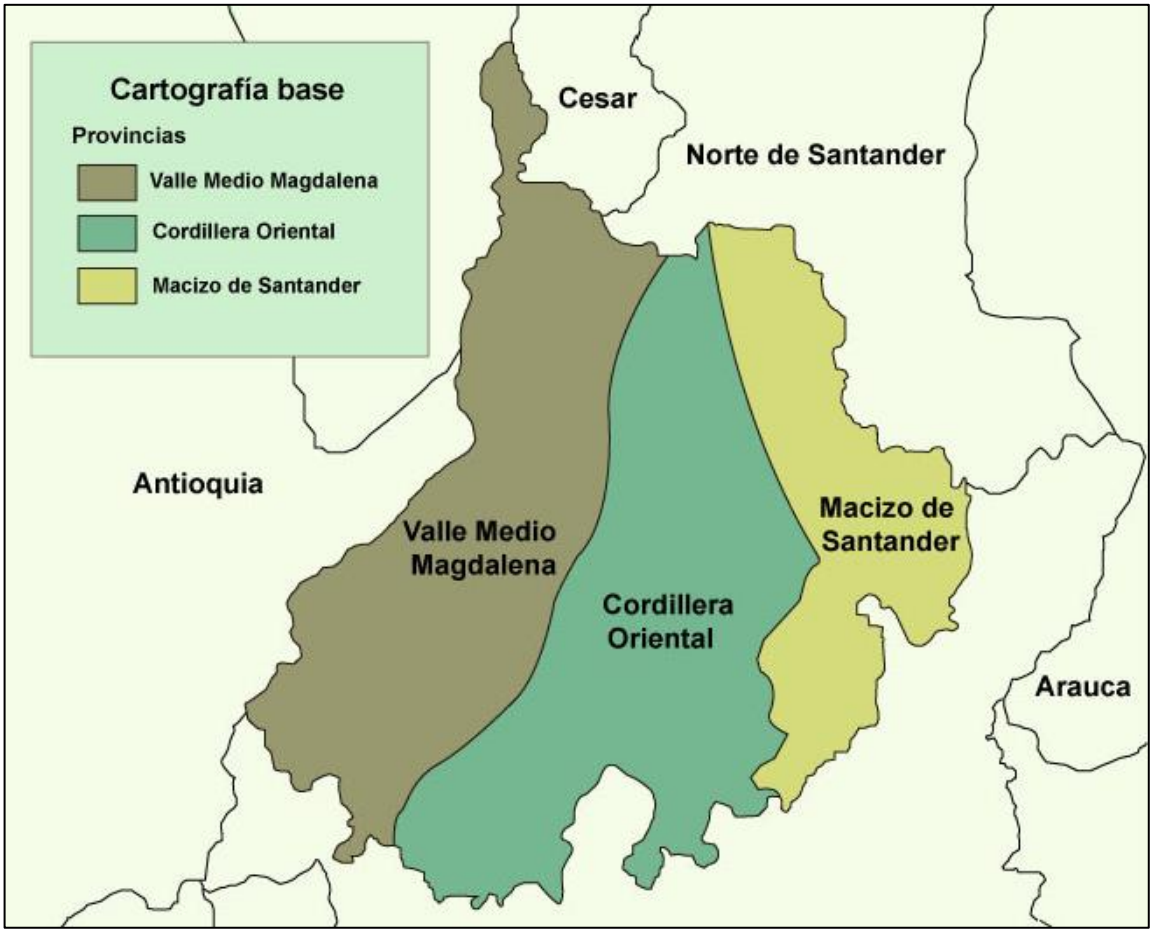


Figura 24. Esquema tectónico del departamento de Santander. Adaptado de: SERVICIO GEOLÓGICO COLOMBIANO. Evaluación y Monitoreo de Actividad Sísmica. Disponible en: https://www2.sgc.gov.co/ProgramasDeInvestigacion/geoamenazas/Paginas/actividad-ismica.aspx#. 2017.

4.1.1.2. Conteo de la cantidad de sismos por epicentro y estación

En la Figura 25, se pueden observar los 10 municipios que más sismos han registrado con epicentros en el departamento de Santander. Existen algunos epicentros registrados por fuera del departamento que fueron desestimados en la selección de la muestra debido a que, al estar por fuera de la región de interés, las señales pueden presentar una dinámica distinta, producto de un cambio en la geografía y topografía.

Mediante la gráfica detallada en la figura se puede comprobar que la sismicidad registrada en el municipio de Los Santos, supera en grandes proporciones a las registradas en otros municipios. El 59% (35.686 eventos) de la muestra de 60.785 sismos en Santander (2010 a 2017), equivalente al 20% de la población muestral (sismos de Colombia desde 1993 a 2017), corresponde a sismos cuyo epicentro fue localizado en Los Santos.

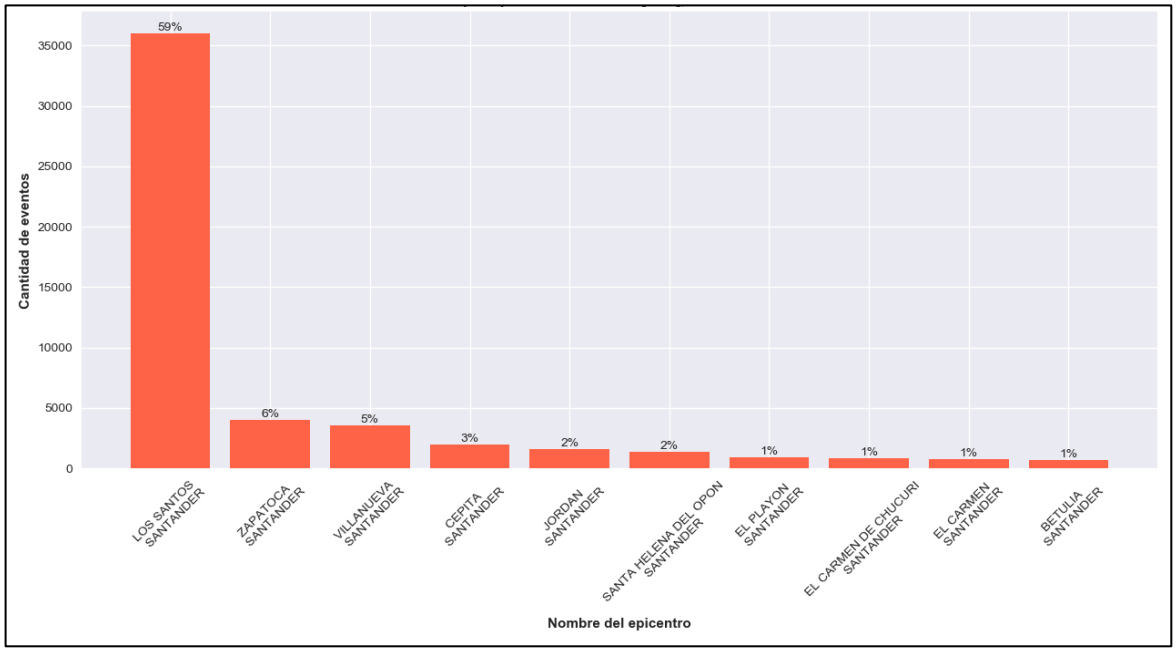


Figura 25. Gráfico de barras con la cantidad de eventos sísmico considerando los 10 epicentro que más sismos han registrado con epicentros en el departamento de Santander.

El 81% de todos los eventos sísmicos de la muestra tienen epicentro en 10 de los 87 municipios de Santander representados en la figura, lo que representa un factor importante para la selección de las estaciones, ya que el registrar los sismos provenientes estos 10 municipios se está cubriendo un gran porcentaje del territorio que es sísmicamente activo en el departamento.

En la Figura 22 pueden identificarse de forma visual las estaciones BAR2, BRR, RUS, PAM, PTB, SPBC, ZAR y TAM como las mejores opciones, entre el conjunto total de estaciones, para ser seleccionadas. Esto debido a su cercanía con los epicentros plasmados en el mapa y con los municipios mostrados. Como complemento y con el fin de corroborar esta información, en la Figura 26 se muestran las 10 estaciones que han registrado mayor cantidad de eventos sísmicos. En primer lugar, se encuentra la estación BAR2, que registró el movimiento sísmico del 89% de los eventos de la muestra (53.831 eventos).

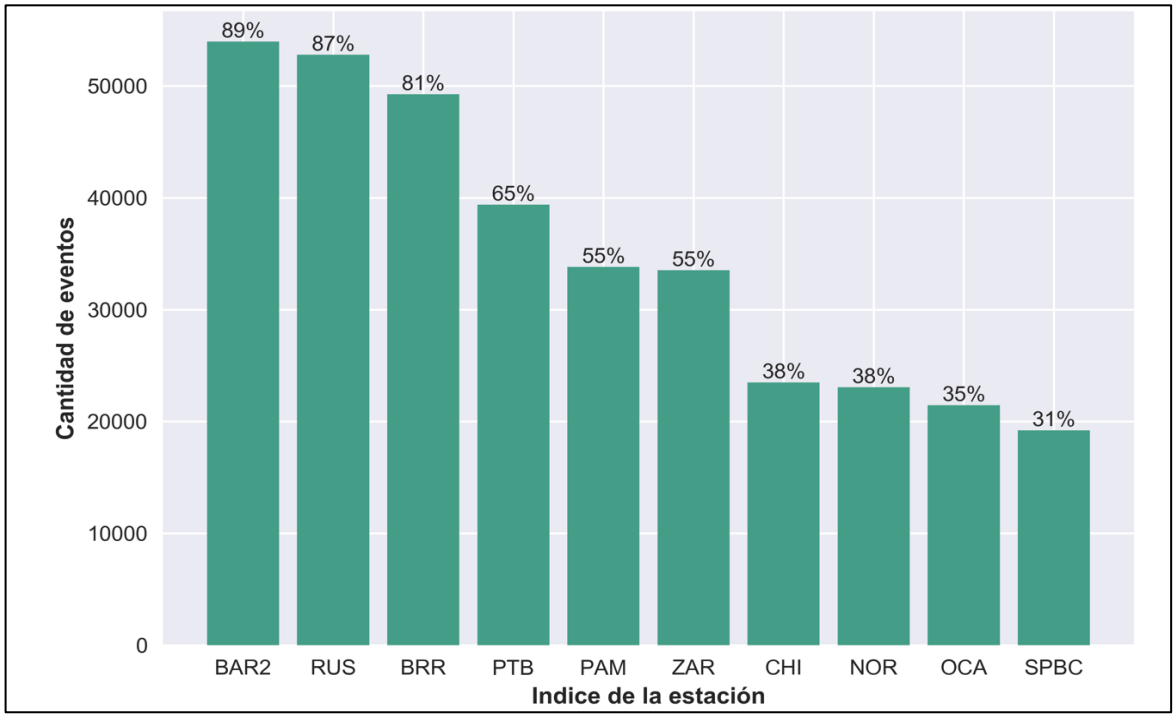


Figura 26. Gráfico de barras con la cantidad de eventos sísmicos detectados por las estaciones sismológicas con epicentros en el departamento de Santander. Registro de las 10 primeras estaciones.

La relevancia de la ubicación geográfica de las estaciones con respecto a las aglomeraciones de los epicentros sísmicos mostradas en la Figura 22 se pueden comprobar con los porcentajes mostrados en la Figura 26. En términos generales, mientras más cerca está la estación del epicentro, más posibilidades tiene de detectar el sismo y registrar su traza. Cabe aclarar que esto depende en gran

medida de la capacidad técnica de la estación y de la geología que recorren las ondas sísmicas entre el hipocentro y la estación. Sin embargo, se considera a la cantidad de sismos detectados, como uno de los criterios de más peso para seleccionar las estaciones. Al contrastar estas estaciones con las escogidas a nivel geográfico, puede notarse que las estaciones SPBC y TAM detectan pocos eventos, a comparación de las estaciones más próximas., debido a esto, estas estaciones son descartadas, manteniendo las seis primeras estaciones mostradas en la Figura 26, en su respectivo orden, como las más importantes para el estudio: BAR2, RUS, BRR, PTB, PAM y ZAR.

La relación de la cantidad de sismos detectados por cada estación, asociados con el epicentro correspondiente a cada evento se detalla en la Figura 27, en la que la cantidad de sismos registrados por cada una de las 10 estaciones de mayor cantidad de registros es asociada con la ubicación del epicentro.

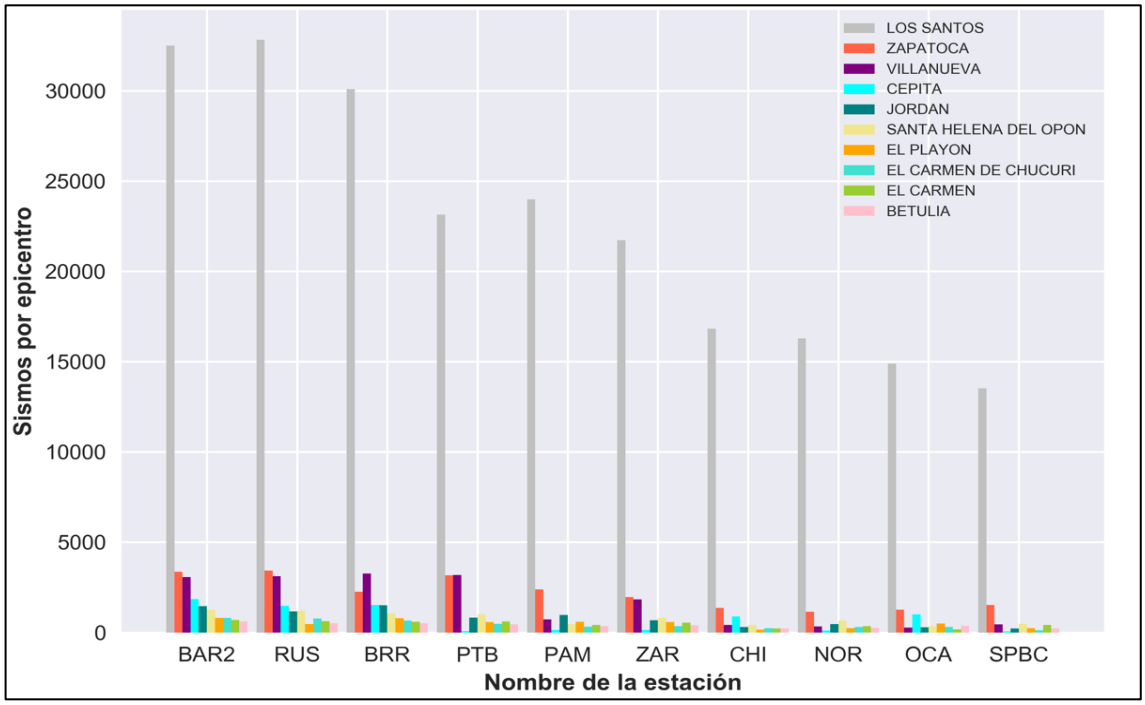


Figura 27. Gráfico de barras con la cantidad de eventos sísmicos detectados por las estaciones sismológicas en contraste con los 10 epicentros más identificados en el departamento de Santander. Registro de las 10 primeras estaciones.

4.1.1.3. Distancia epicentral promedio por estación

Como se ha mostrado en los procesos de mapeo geográfico y conteo de los eventos sísmicos por epicentro y estación, existen seis estaciones predominantes según los criterios de distancia visual y cantidad de sismos registrados: BAR2, RUS, BRR, PTB, PAM y ZAR. Con el fin de precisar en la cercanía de las estaciones a los epicentros que son visualizados en los mapas mostrados, se calculó una medida cuantitativa de la distancia de la estación a los epicentros o distancia epicentral. Los resultados son mostrados en la Figura 28 y 29, contrastando los valores de distancia con la cantidad de sismos registrados por estación.

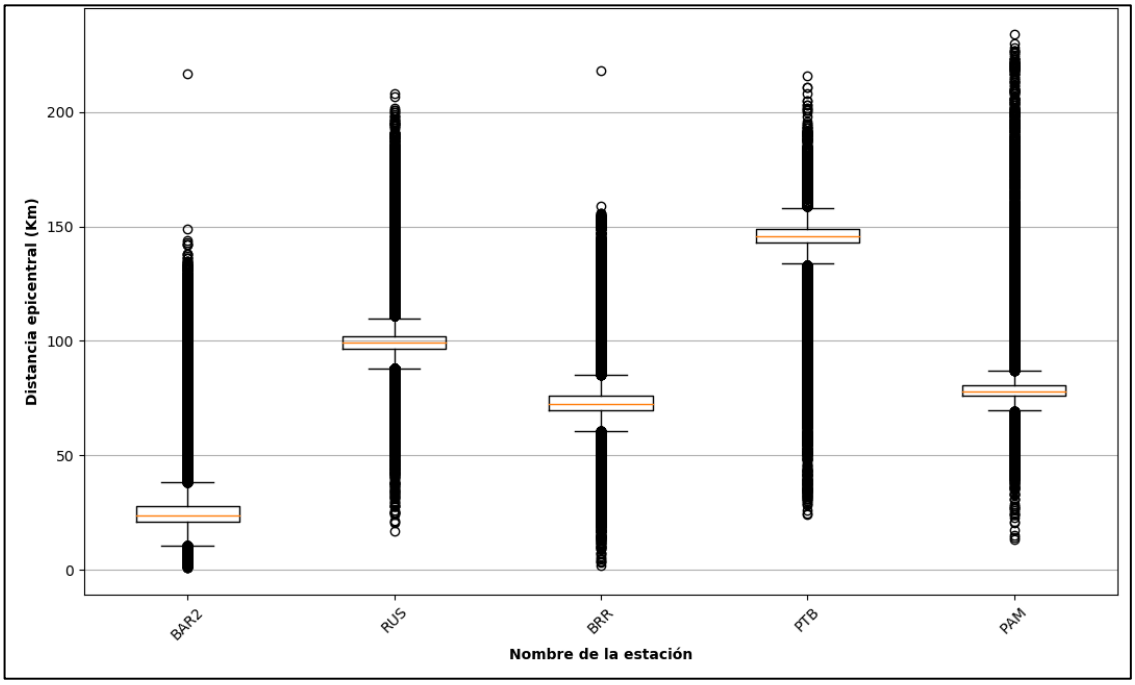


Figura 28. Diagrama de caja de la distancia epicentral promedio registrada por las estaciones.

En la Figura 28 se aprecia un diagrama de caja en el que se presenta la distribución de distancia epicentral obtenida por cada una de las 10 estaciones que más eventos sísmicos registraron. Puede notarse en el gráfico que la estación más cercana a los epicentros sísmicos es BAR2, seguida de BRR, PAM, RUS y PTB. La estación ZAR se encuentra considerablemente más alejada (mayor distancia epicentral) a las

estaciones mencionadas, razón por la cual fue descartada. Puede notarse que existe una gran cantidad de valores atípicos en la medición de la distancia epicentral posiblemente debido a variabilidad en las mediciones por parte de las estaciones sismológicas o a errores en el cálculo por parte de la RSNC.

En la Figura 29 se relaciona la cantidad de sismos registrada por estación con la distancia epicentral promedio de los registros analizados. Puede notarse la misma distribución de distancias epicentrales mostradas en la figura anterior y las estaciones pueden ser ordenadas por distancia epicentral como: BAR2, BRR, PAM, RUS y PTB. Sin embargo, existe una variación en el orden si las estaciones se ordenan por cantidad de sismos registrados: BAR2, RUS, BRR, PTB y PAM. Según esto, pese a que la estación RUS está considerablemente más alejada de Los Santos que la estación BAR2, detectó mayor cantidad de sismos con epicentro en Los Santos, RUS detectó 3.630 (6%) sismos más que BAR2.

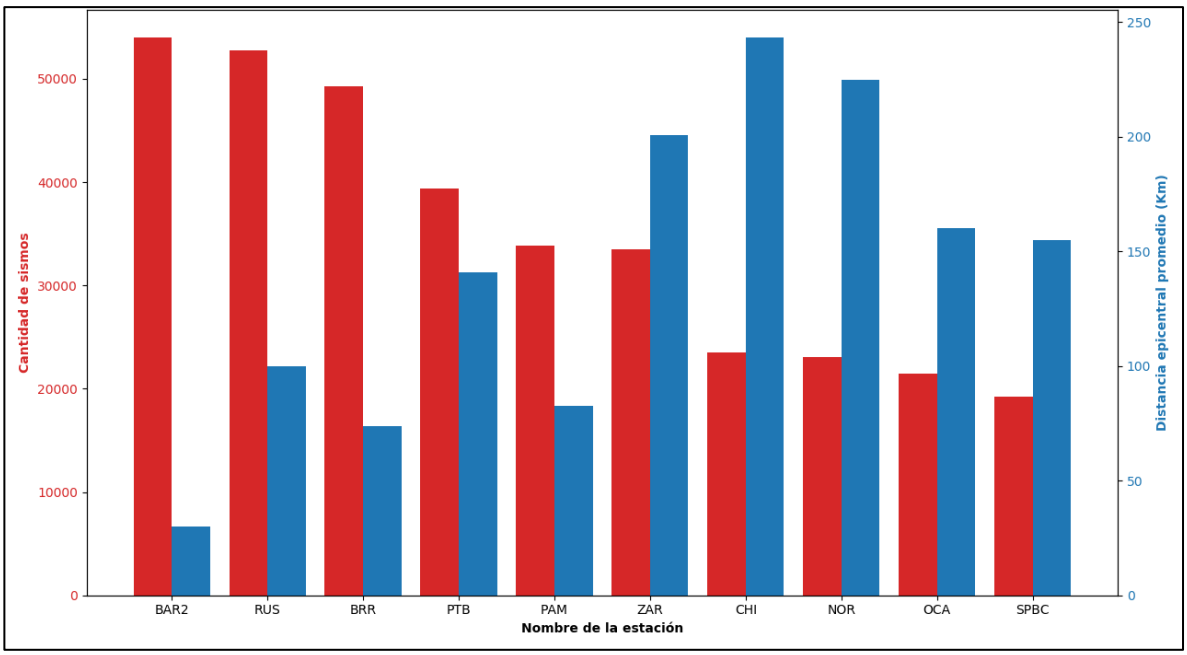


Figura 29. Gráfico de barras con la cantidad de eventos sísmicos detectados por las estaciones sismológicas en contraste con la distancia epicentral. Registro de las 10 primeras estaciones en orden de cantidad de sismos registrados.

Al analizar los criterios de forma integrada (cantidad de sismos detectados y distancia epicentral media), el procesamiento de los datos sísmicos se enfocó en las trazas correspondientes a las estaciones BAR2, RUS, BRR, PTB y PAM. Debe tenerse en cuenta que, si se analizan los criterios individualmente, pueden presentarse los siguientes dos casos en la selección de las estaciones:

- Al comparar la distancia epicentral entre BRR y RUS, se puede observar que BRR tiene un valor menor. Si se toma únicamente este criterio para seleccionar entre éstas dos estaciones, estaría presentándose un resultado sesgado ya que RUS detecta mayor cantidad de sismos en comparación con BRR, lo que podría perjudicar procesos futuros de localización y alerta de eventos sísmicos.

- Los sismos detectados por las estaciones PAM y ZAR tienen valores similares. Si se plantea el escenario en que la cantidad de sismos detectados por ZAR fuese ligeramente mayor a PAM y esto se usa como argumento para seleccionar a la estación ZAR sobre PAM, se estaría sesgando ya que la distancia epicentral de PAM es mucho menor a la de ZAR, lo que podría perjudicar procesos futuros de localización y alerta de eventos sísmicos.

4.1.2. Selección de la muestra

El resultado de los procesos de selección de muestra es el siguiente:

- De la población de 176.968 registros sísmicos con epicentros en la geografía colombiana desde 1993 hasta 2017, se seleccionan 60.785 registros (34% de la población original) desde el 2010 hasta el 2017 que tienen epicentro en el departamento de Santander.
- De los 60.785 registros sísmicos del departamento de Santander se descartaron 4.629 archivos ilegibles o con el formato incorrecto, reduciendo la muestra a 56.156 (32% de la población original).

- De los 56.156 registros sísmicos se descartaron 1.850 archivos repetidos, reduciendo la muestra a 54.306 (31% de la población original).
- De los 54.306 registros sísmicos se descartaron 2.306 archivos *Sfile* y *Waveform* que no estuvieron registrados de forma integrada, reduciendo la muestra a 52.000 (29% de la población original).
- De los 52.000 registros sísmicos se descartaron 3.698 archivos por no presentar registros de tiempo de inicio, fin y Onda P, reduciendo la muestra a 48.032 (27% de la población original).
- De los 48.032 registros sísmicos se descartaron 33.085 archivos del 2010 al 2017 debido a que las estaciones presentaron cambios de sensores, estuvieron inhabilitadas durante este periodo y/o tuvieron cambios en la tasa de muestreo, reduciendo la muestra a 14.947 (8% de la población original).

La selección de la muestra y de las estaciones es la mostrada en la siguiente figura:

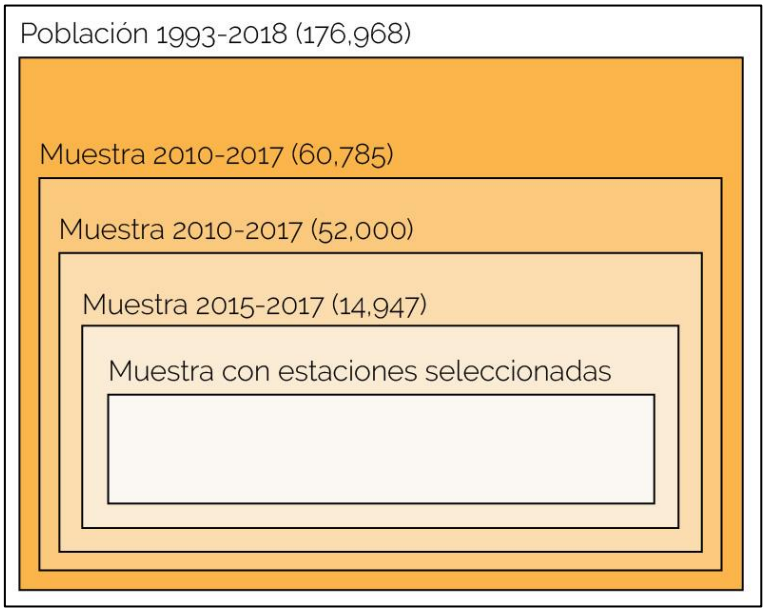


Figura 30. Reducción en la población a medida que se aplican los procesos de selección de muestras y estaciones.

La cantidad de registros utilizados para los procesos de preprocesamiento, entrenamiento, validación y prueba del clasificador dependen de la cantidad de

estaciones escogidas en el proceso de análisis. Tal como se describió en la sección anterior, las estaciones escogidas son: BAR2, BRR, RUS, PAM y PTB. Sin embargo, la estación BAR2, aunque presenta la menor distancia epicentral y la mayor cantidad de eventos sísmicos registrados, tiene un sensor de velocidad que registra únicamente la componente vertical, razón por la cual no pueden ser tomados adecuadamente los atributos en el proceso de extracción. En consecuencia, esta estación fue descartada y las estaciones consideradas fueron: BRR, RUS, PAM y PTB.

Para comprobar la heterogeneidad de los datos frente a la magnitud y profundidad de los eventos sísmicos, se graficaron las distribuciones de estas dos variables en la muestra de 14.947 sismos. Los resultados se muestran en las Figuras 31 y 32. Las magnitudes fueron graficadas en intervalos de 0.1 en la escala de Richter y se encuentran en el entre 0.8 y 8.0 con una mediana en 1.6. Las profundidades fueron graficadas en intervalos de 1 Km y se encuentran entre 50 y 190 Km con una mediana sobre 147 Km.

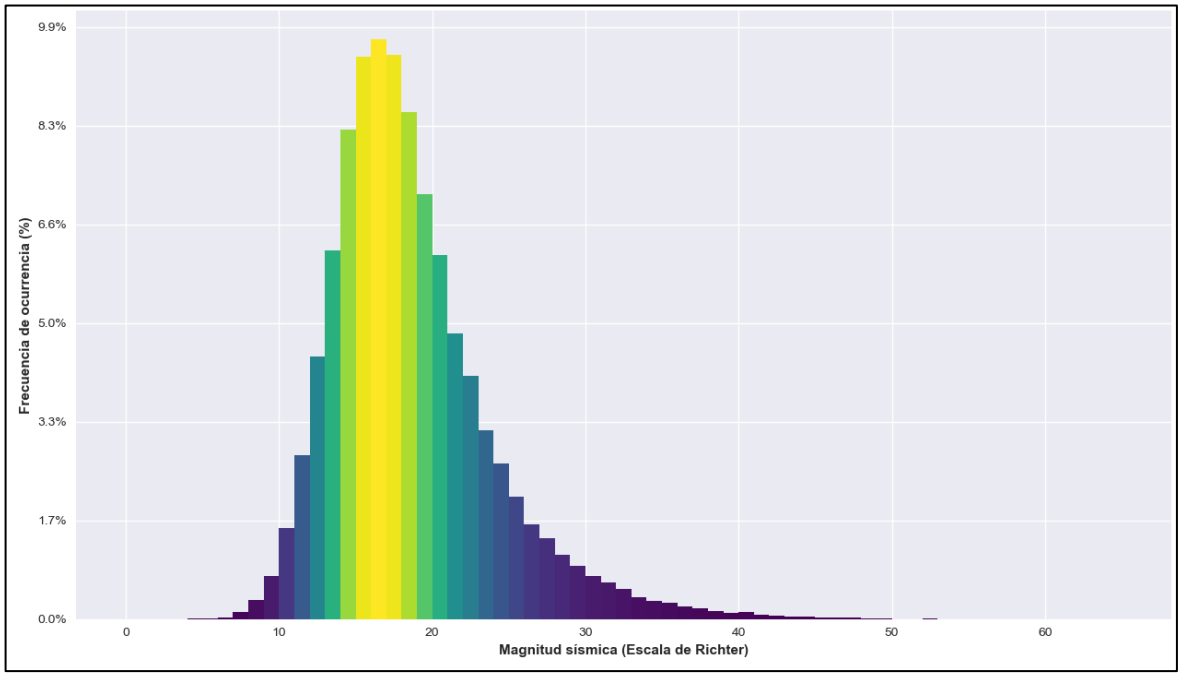


Figura 31. Distribución de magnitudes en la muestra de 14.947 eventos sísmicos.

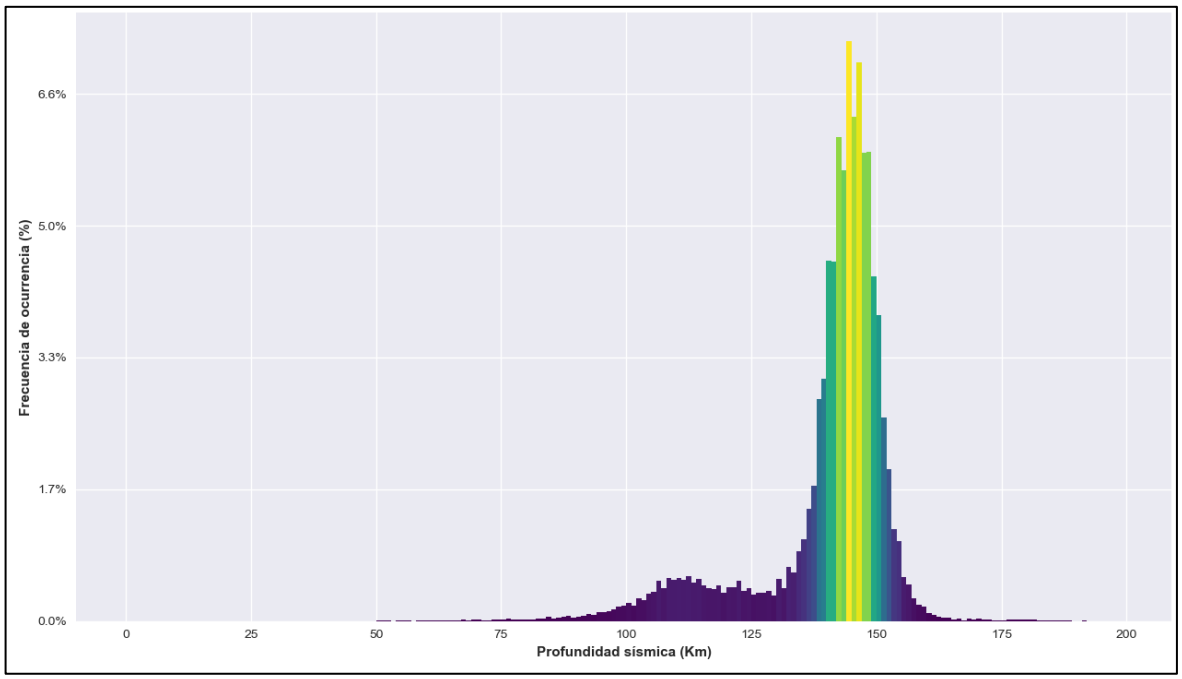


Figura 32. Distribución de profundidades en la muestra de 14.947 eventos sísmicos.

4.2. PROTOTIPADO

Los prototipos desarrollados y etiquetados mediante el versionamiento semántico, en orden cronológico, son:

- Prototipo V0.0.1
- Prototipo V0.0.2
- Prototipo V0.1.0
- Prototipo V0.1.1
  - Prototipo V0.1.1-3stat
  - Prototipo V0.1.1-4stat
- Prototipo V0.2.0

En las secciones siguientes se presentarán los resultados obtenidos con el último prototipo desarrollado, el V0.2.0, debido a que este integra los módulos especificados, obteniendo las mejores métricas de desempeño. Los prototipos predecesores se explican en detalle en el Anexo F.

El prototipo V0.2.0 está compuesto de la siguiente forma:

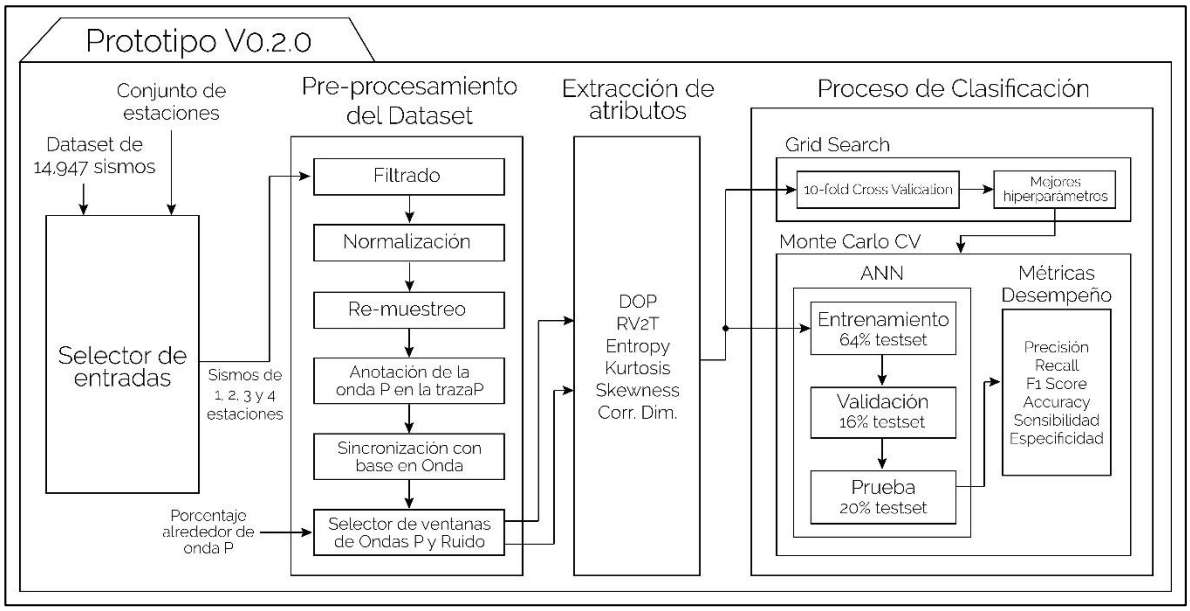


Figura 33. Diagrama de bloques del prototipo V0.2.0.

Una vez ejecutados los procesos de selección de muestras y estaciones, se obtiene un conjunto de 14.947 eventos sísmicos y unas estaciones de interés que ingresan a un proceso de selección de la entrada. Una vez seleccionada, pasa por el preprocesamiento de los datos, los atributos son extraídos y el proceso de clasificación es ejecutado.

Los resultados de estos procesos que han sido explicados en la Sección 3.1, son mostrados en las secciones siguientes. Con el fin de evidenciar el desempeño del clasificador al variar la cantidad de estaciones y como se ha explicado en la sección de Análisis de Estaciones, se escogieron 4 variaciones: ejecutar los procedimientos con 1 estación (BRR), con 2 estaciones (BRR y RUS), con 3 estaciones (BRR, RUS y PAM) y con 4 estaciones (BRR, RUS, PAM y PTB), siguiendo el orden que ha sido especificado en la sección mencionada.

4.3. SELECCIÓN DE LA ENTRADA

Una vez seleccionada la muestra, se selecciona la entrada que pasará por los procesos de preprocesamiento, extracción de atributos y proceso de clasificación. En el proceso de selección de entrada se evalúan los criterios mostrados en el siguiente diagrama de bloques:

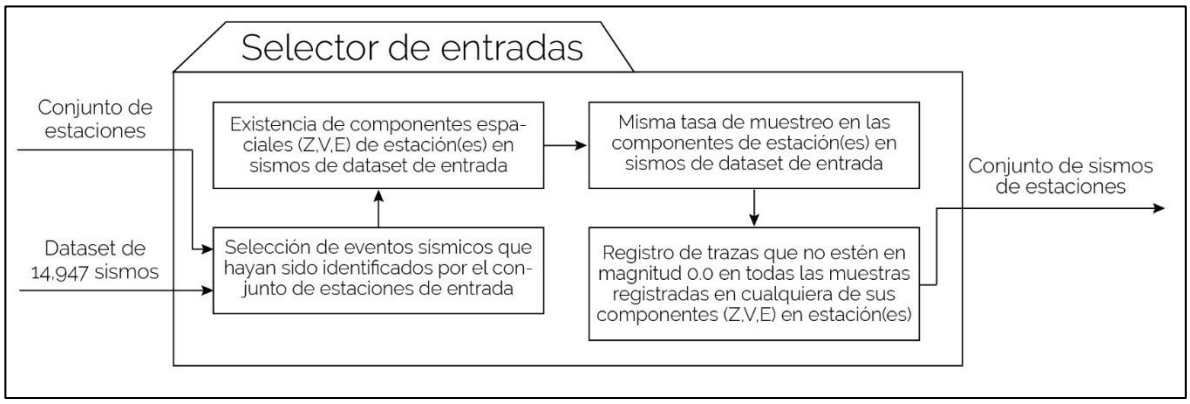


Figura 34. Diagrama de bloques del selector de entradas.

En primera instancia, se identifican aquellos eventos que hayan sido registrados por el conjunto de estaciones de forma que todas queden incluidas. Si existen eventos que no fueron registrados por una o más estaciones de las estipuladas en el conjunto de estaciones, el evento es descartado. El segundo criterio está relacionado con la existencia de las tres componentes espaciales en los eventos seleccionados. Si para el evento existen estaciones de interés que no hayan registrado en las tres componentes, el evento es descartado. Las componentes evaluadas son: Z, V y E, en cualquiera de los tipos de banda que los sismómetros de las estaciones puedan presentar (H, S, B, entre otros).

Una vez seleccionados los eventos con la misma cantidad de componentes, se verifica que la tasa de muestreo de dichas componentes sea la misma. Si se identifican eventos cuyas componentes tienen distintas tasas de muestreo, estos eventos son descartados. Cabe anotar que este es un criterio que presenta casos atípicos y es probablemente atribuible a errores en la toma de datos, en el

almacenamiento y/o en el algoritmo de procesamiento, pues se trata de un mismo sensor el que registra las componentes y se presupone que debe tener la misma tasa de adquisición de datos para sus componentes.

El último criterio evaluado corresponde a la identificación de componentes de las estaciones de interés que registren trazas en las que no exista una variación en la magnitud y permanezcan en un valor nulo (magnitud en 0.0) durante todo el tiempo de muestreo. Aquellos eventos en los que esto suceda son descartados.

Al finalizar el proceso de selección de la entrada, de los 14.947 se seleccionaron:

- 13.905 eventos con onda P y ruido (7.9% de la población y 23% de la muestra inicial) en los que está presente la estación BRR.
- 9.715 eventos con onda P y ruido (5.5% de la población y 16% de la muestra inicial) en los que están presentes las estaciones BRR y RUS.
- 6.418 eventos con onda P y ruido (3.6% de la población y 11% de la muestra inicial) en los que están presentes las estaciones BRR, RUS y PTB.
- 5.144 eventos con onda P y ruido (2.9% de la población y 8.5% de la muestra inicial) en los que están presentes las estaciones BRR, RUS, PTB y PAM.

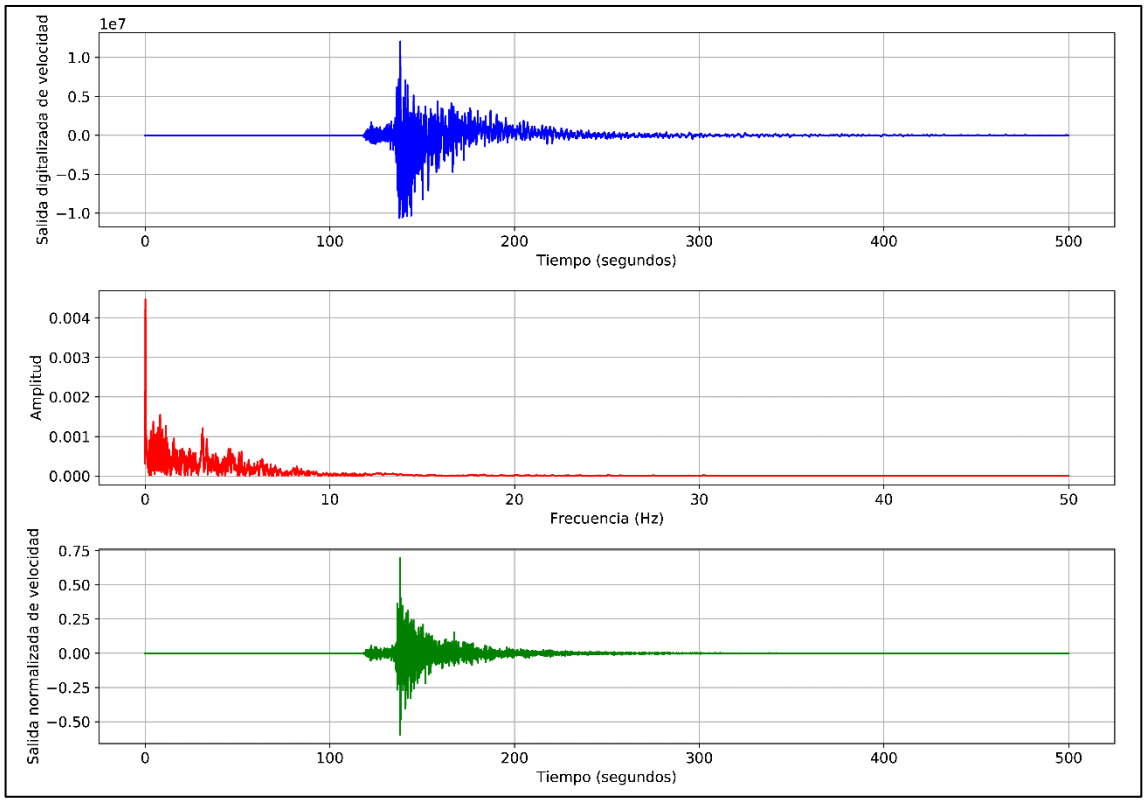
Cada uno de los eventos contiene las trazas correspondientes a las estaciones, donde pueden extraerse ventanas de onda P y ruido de señal. Con el fin de hacer comparables los procesos de clasificación probados con 1, 2, 3 y 4 estaciones, se tomó la muestra de 5.144 eventos sísmicos en estos casos. Los resultados presentados en las siguientes secciones tienen en cuenta estas consideraciones.

4.4. PRE-PROCESAMIENTO DE LOS DATOS

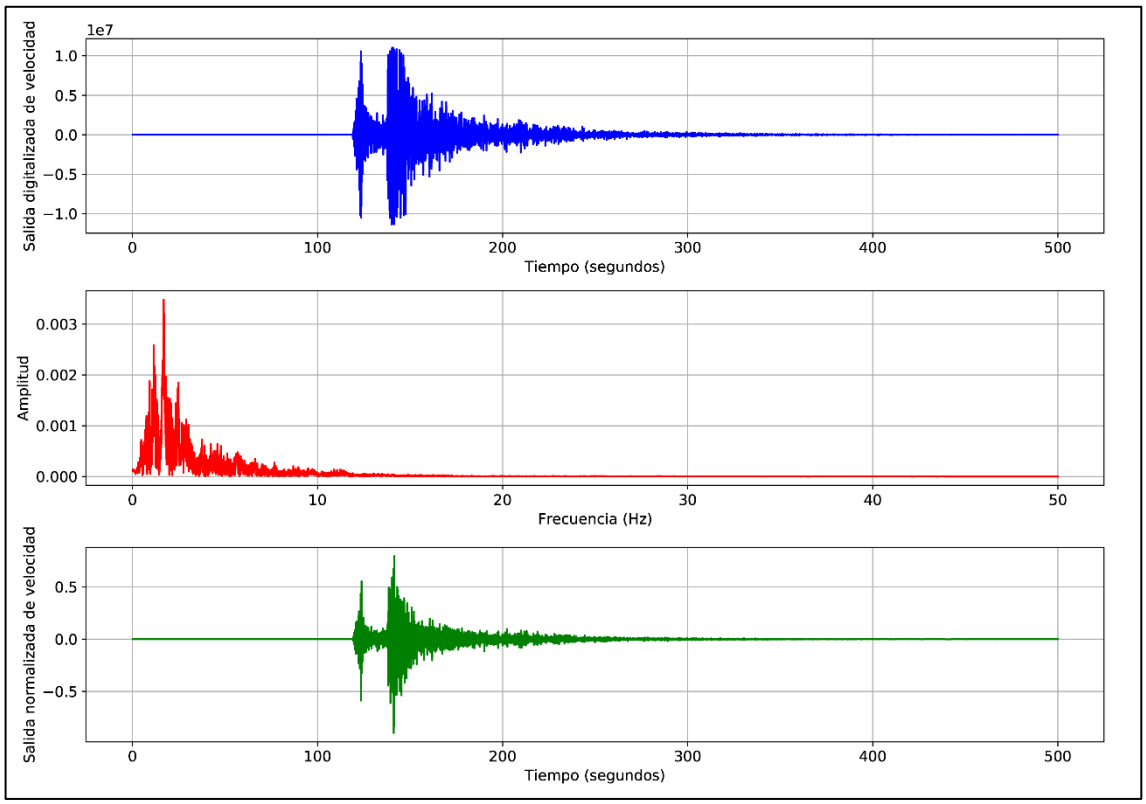
Una vez ejecutado el procedimiento descrito en la Sección 3.3.2.5 se obtienen los resultados presentados a continuación.

4.4.1. Filtrado, normalización y re-muestreo de los datos

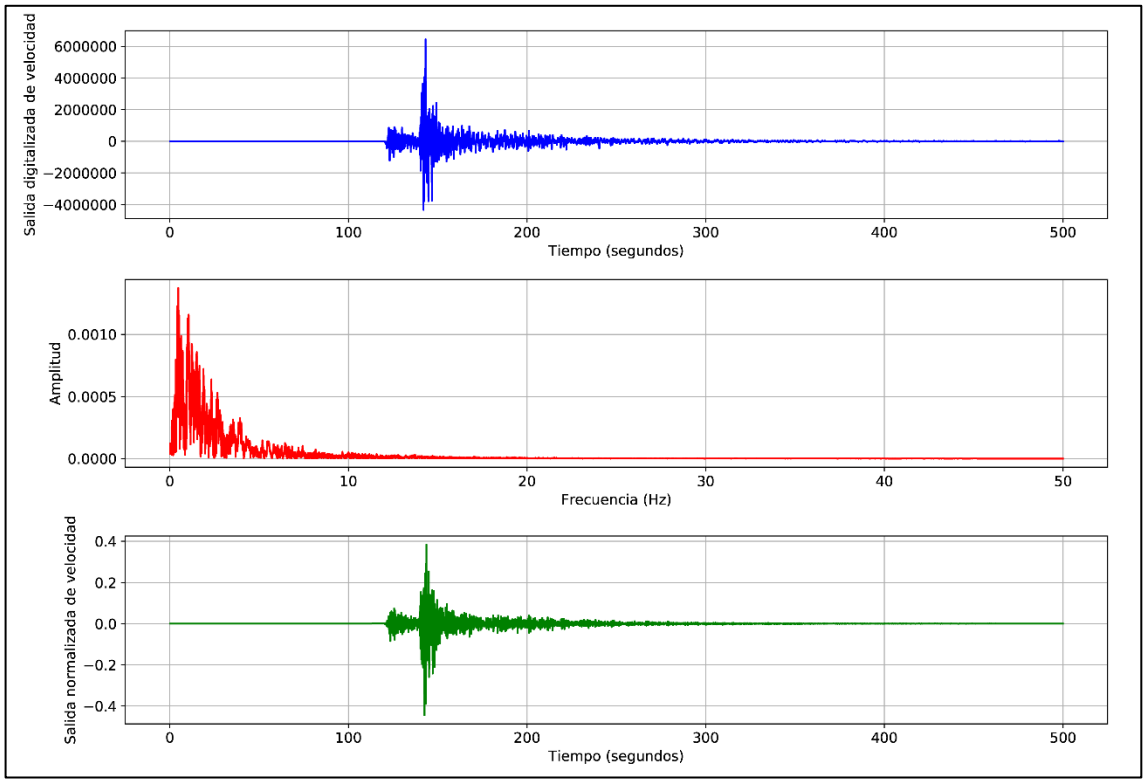
En la Figura 35 se muestra la traza original de la componente vertical de las estaciones BRR, PAM, RUS y PTB del evento sísmico ocurrido el 10 de marzo de 2015 con epicentro en el departamento de Santander.



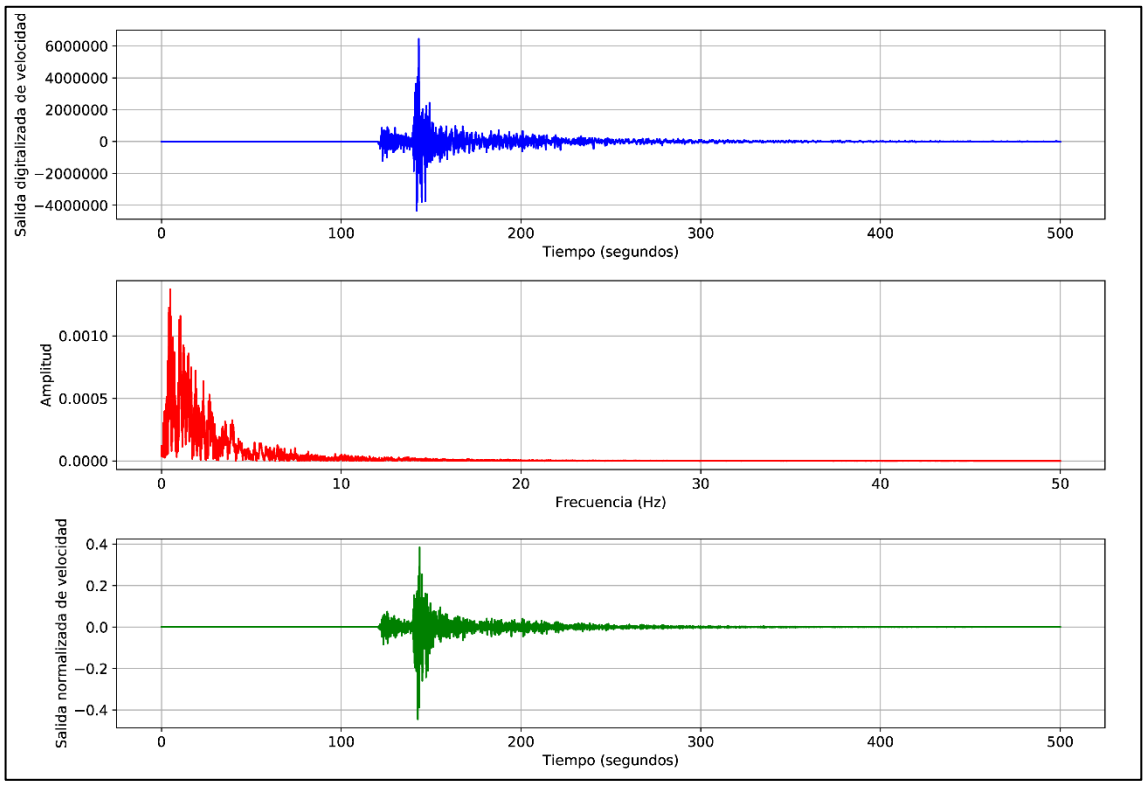
(a)


1e7
Salida digitalizada de velocidad
1.0
0.5
0.0
−0.5
−1.0
0
100
200
300
400
500
Tiempo (segundos)
Amplitud
0.003
0.002
0.001
0.000
0
10
20
30
40
50
Frecuencia (Hz)
Salida normalizada de velocidad
0.5
0.0
−0.5
0
100
200
300
400
500
Tiempo (segundos)

(b)


Salida digitalizada de velocidad
6000000
4000000
2000000
0
−2000000
−4000000
0
100
200
300
400
500
Tiempo (segundos)
Amplitud
0.0010
0.0005
0.0000
0
10
20
30
40
50
Frecuencia (Hz)
Salida normalizada de velocidad
0.4
0.2
0.0
−0.2
−0.4
0
100
200
300
400
500
Tiempo (segundos)

(c)



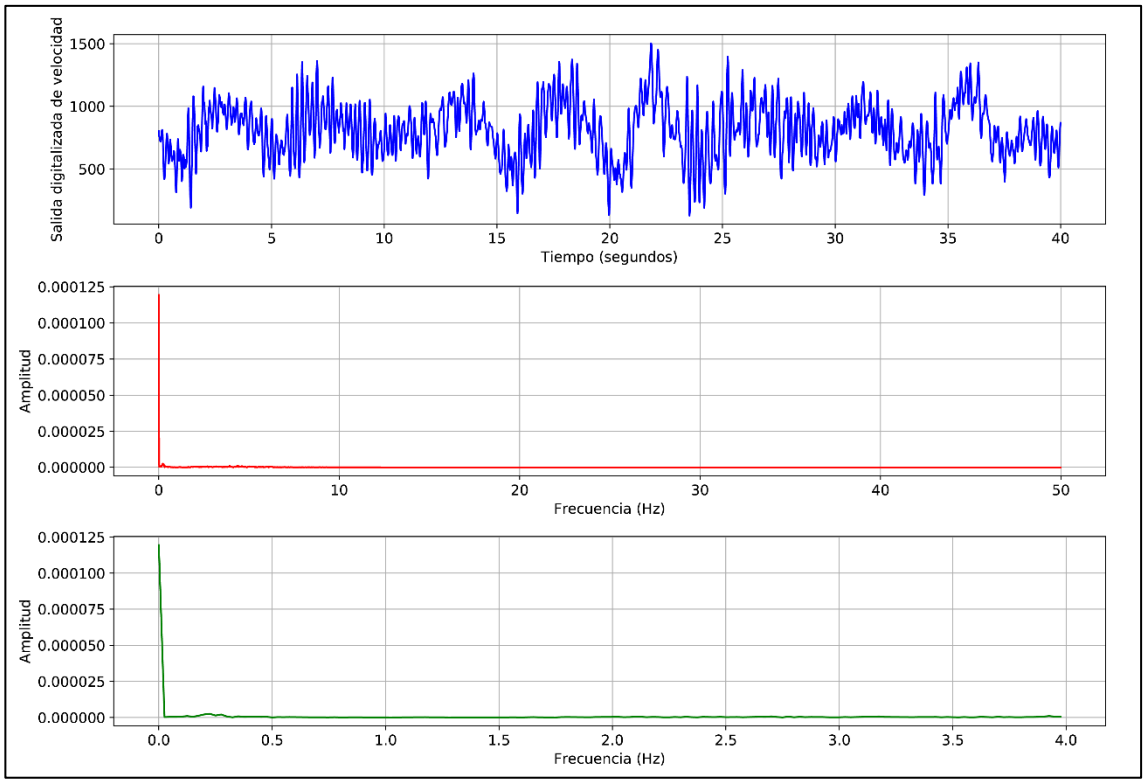
(d)

Figura 35. Evento sísmico del 10 de marzo del 2015 con epicentro en el departamento de Santander, registrado por las estaciones de interés: (a) BRR, (b) PAM, (c) RUS y (d) PTB.

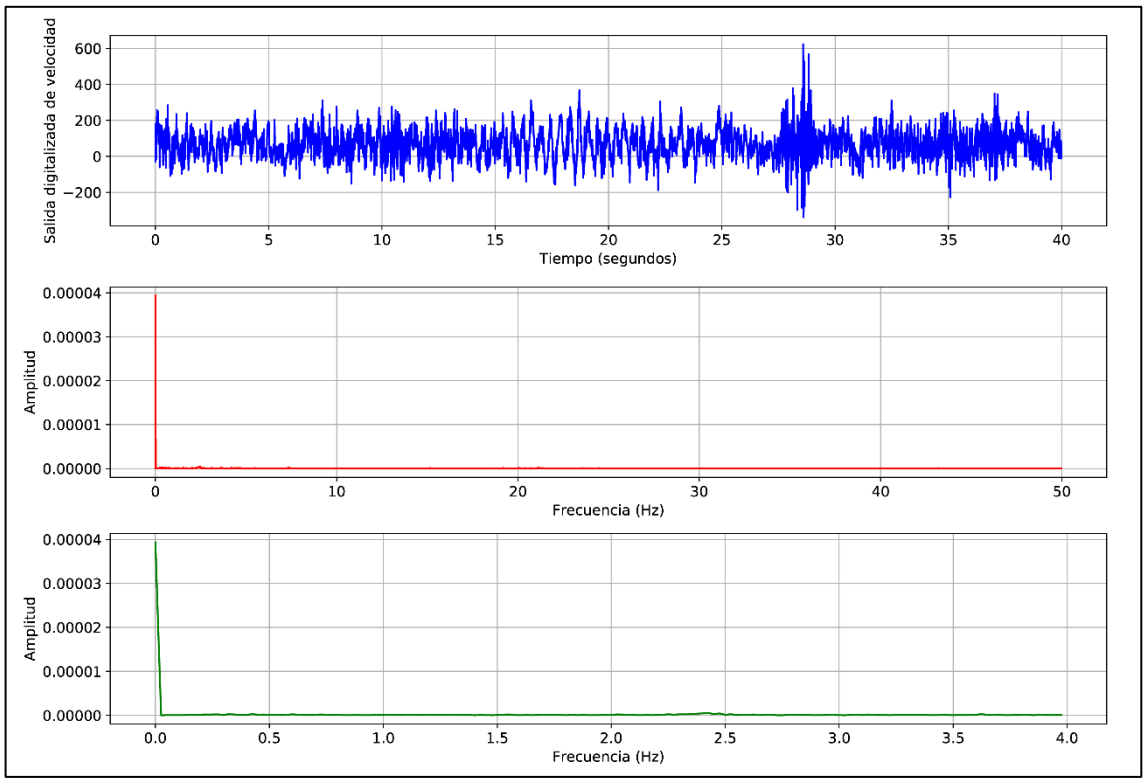
En la figura mostrada, la sección superior (en azul) de cada gráfica corresponde a la señal sísmica registrada por la componente de la estación mostrada. Cada una de las señales mostradas presenta una media de velocidad distinta de cero: la señal de la estación BRR presenta una media de 8,193, la señal de la estación PAM tiene una media de 9,789, la señal RUS tiene una media de 7,623 y la señal de la estación PTB tiene una media de 10,813. Sin embargo, por las características de los gráficos y las magnitudes que manejan las señales, son imperceptibles estos valores. En la sección intermedia (en rojo) de las gráficas se muestra la distribución de frecuencias de la transformada de Fourier de la señal sísmica completa. Puede notarse que la mayoría del contenido frecuencial se encuentra entre 0 y 10 Hz.

El ruido presenta un componente frecuencia en el intervalo de 0 a 1 Hz, tal como lo muestra la Figura 36, en la que se evidencia una ventana de ruido de 4.000 muestras en azul, su respectiva distribución de frecuencias en rojo y un acercamiento de 0 a 4 Hz a la distribución de frecuencias, en verde. Una vez filtradas las señales con el filtro pasa banda de 1 a 10 Hz Butterworth de 4to grado, normalizadas por media aritmética y re-muestreadas, se obtienen las gráficas de la sección inferior (en verde) de la Figura 35. En el caso de las señales del evento presentado, la tasa de muestreo original es de 100 Hz, razón por la cual no fue ejecutada ninguna operación de re-muestreo.

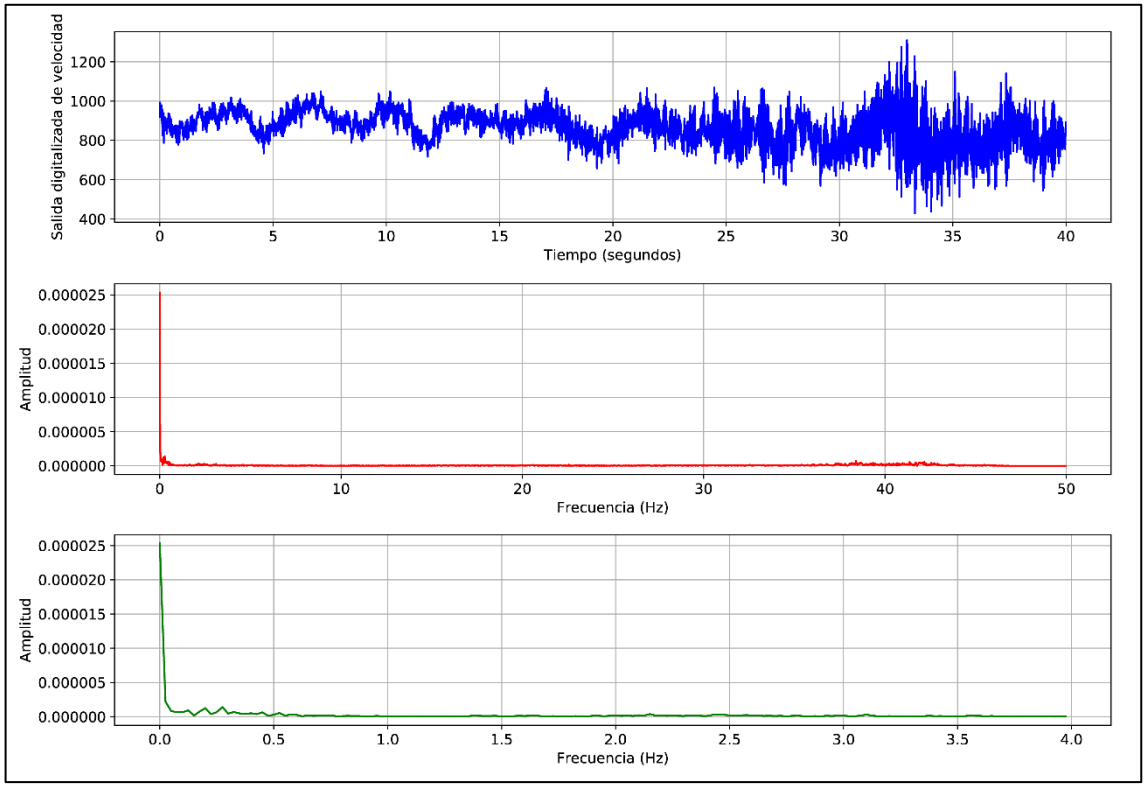
Puede observarse en las ventanas de ruido en verde mostradas en la Figura 36 que las componentes de ruido de mayor preponderancia se encuentran entre 0 y 1 Hz, tal como se describió en la sección anterior. El ruido de alta frecuencia que se aprecia puede percibirse en la distribución de frecuencias de la curva en rojo.



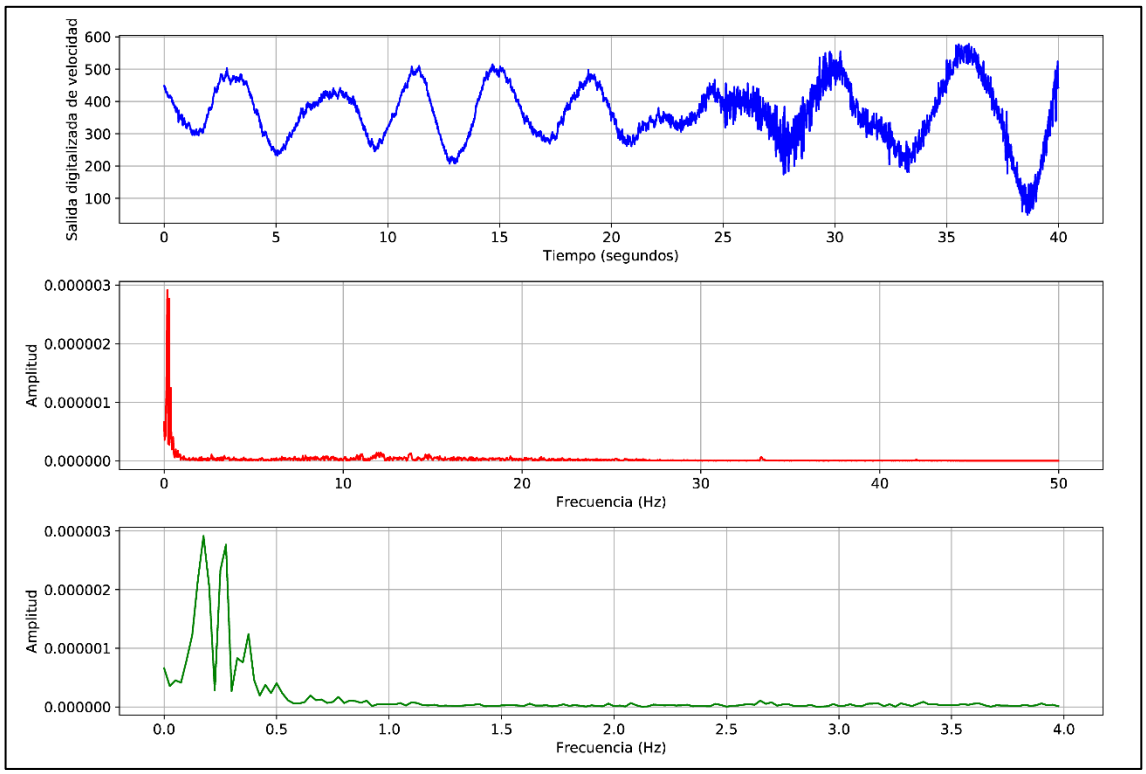
(a)


Salida digitalizada de velocidad
Tiempo (segundos)
Amplitud
Frecuencia (Hz)
Amplitud
Frecuencia (Hz)

(b)


Salida digitalizada de velocidad
Tiempo (segundos)
Amplitud
Frecuencia (Hz)
Amplitud
Frecuencia (Hz)

(c)



(d)

Figura 36. Ruido sísmico del evento del 10 de marzo del 2015 con epicentro en el departamento de Santander, registrado por las estaciones de interés: (a) BRR, (b) PAM, (c) RUS y (d) PTB.

4.4.2. Anotación de Onda P, Sincronización y Selección de ventanas

En la Figura 37 se muestra el registro filtrado, normalizado y re-muestreado de la componente vertical de la estación BRR del evento sísmico ocurrido el 18 de junio de 2013. La ventana de 200 muestras se desliza alrededor de la Onda P, tal como se ha detallado. Una ampliación de la ventana de la Figura 37 se encuentra en la Figura 38, en donde se aprecian las 200 muestras de ventana en el instante en el que la Onda P se encuentra al 50% de la ventana.

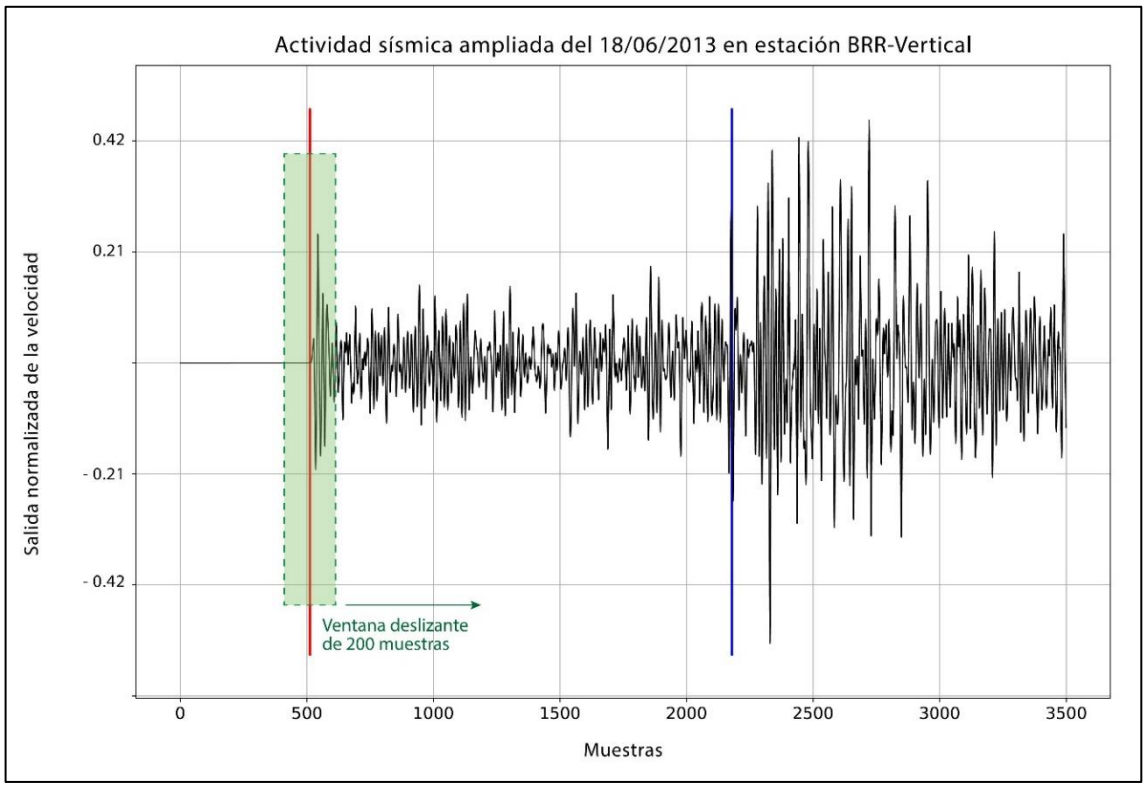
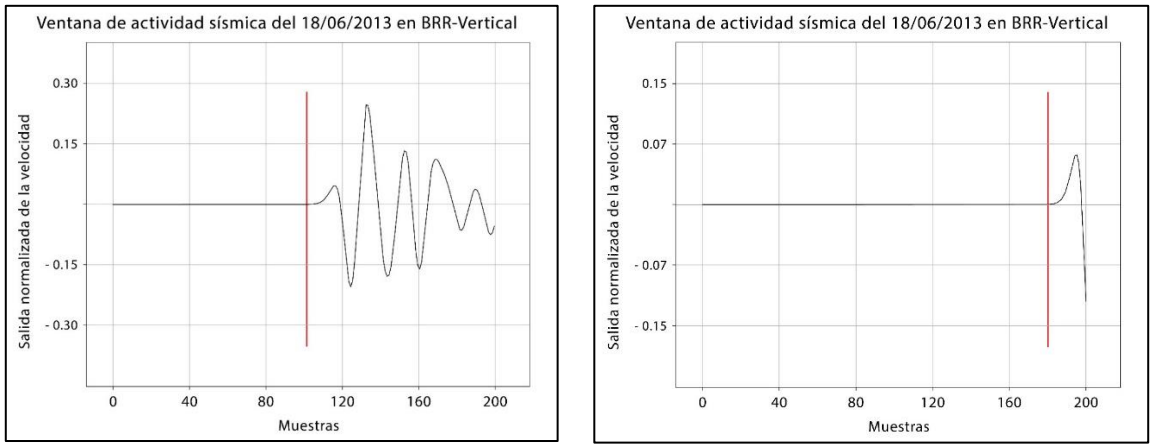


Figura 37. Movimiento de la ventana deslizante en la señal registrada por la estación BRR en su componente vertical del evento ocurrido el 18 de junio de 2013 con epicentro en Santander.



(a) (b)

Figura 38. Ventana del evento sísmico mostrado en la Figura 37 con Onda P en el: (a) 50% y (b) 90%.

Las ventanas de ruido tienen la misma longitud que las ventanas de ondas P y fueron extraídas en posiciones aleatorias de las trazas, teniendo en cuenta una sola

ventana por evento sísmico y un intervalo de muestras comprendido entre la muestra 200 hasta 200 muestras antes del registro de la Onda P. Un ejemplo de una porción de ruido en el evento visualizado en la Figura 37 es el que se muestra en la Figura 39, en el que fueron registradas las primeras 3.500 muestras después de la muestra 200, es decir, de la muestra 200 a la muestra 3.700. La ventana aleatoria extraída de la posición 595 de la traza original (muestra 395 en la traza de la figura) se muestra en la Figura 40.

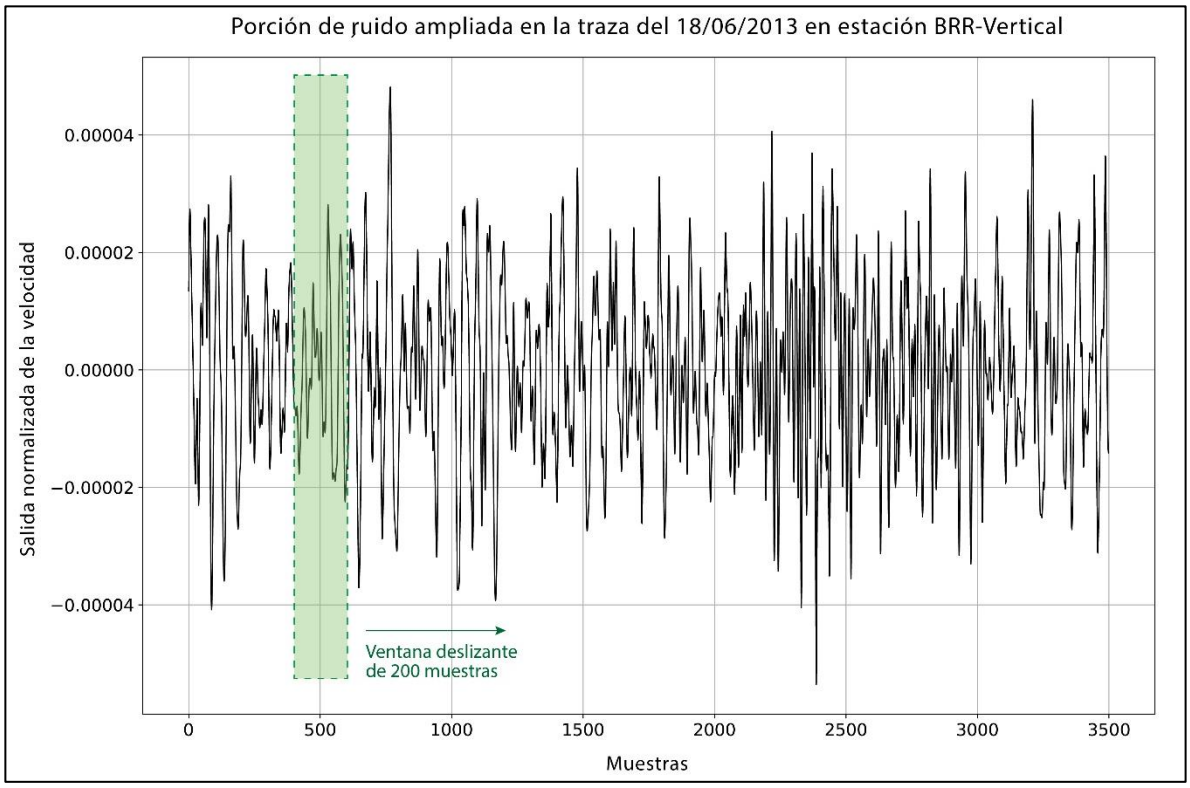


Figura 39. Movimiento de la ventana deslizante en el ruido sísmico de la señal registrada y mostrada en la Figura 37.

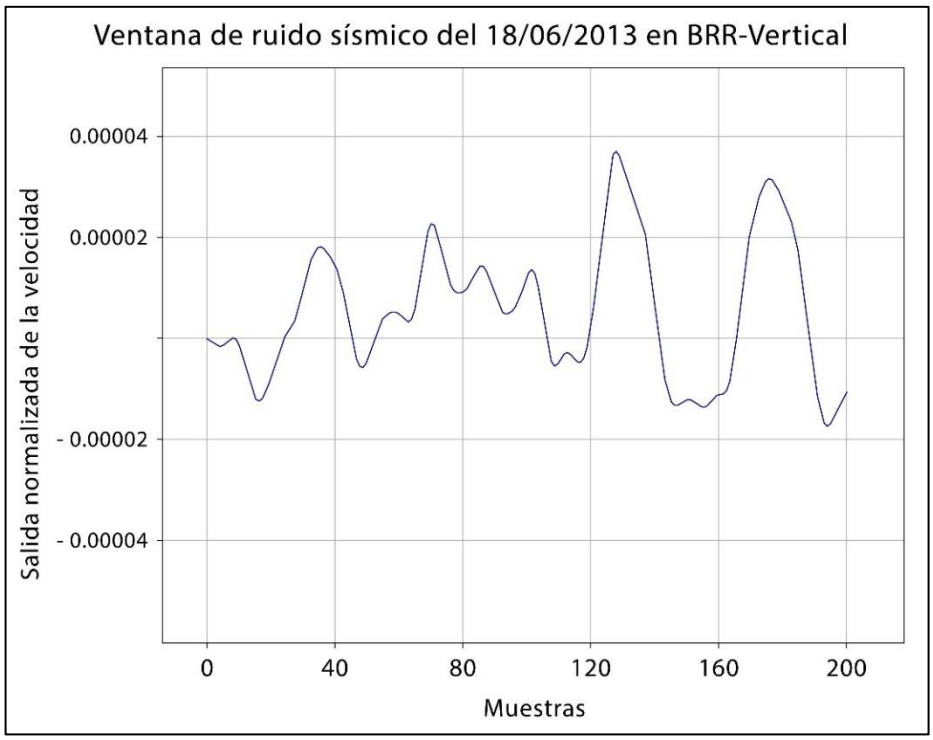
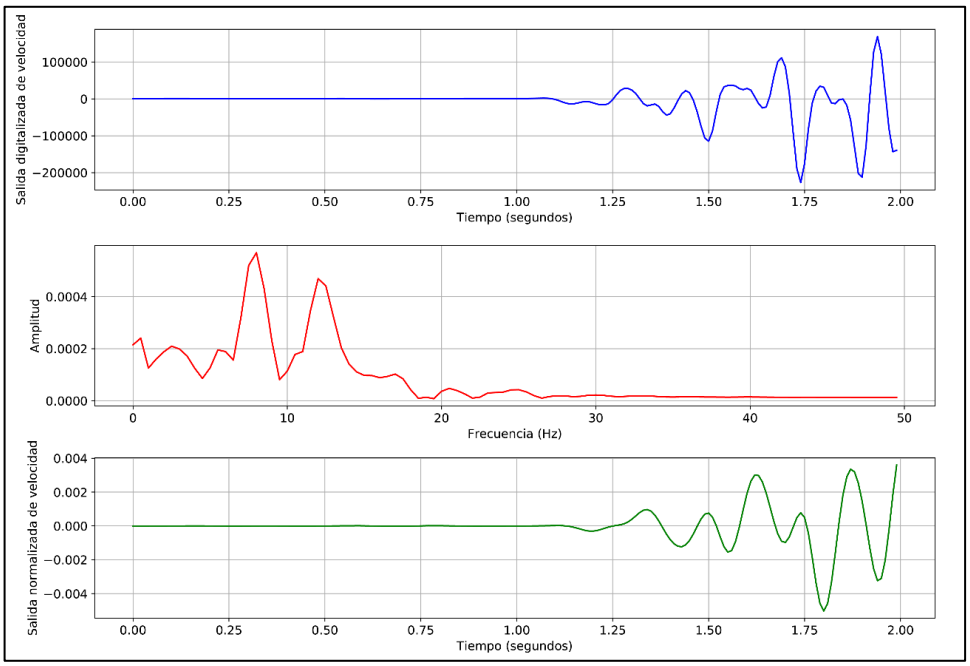
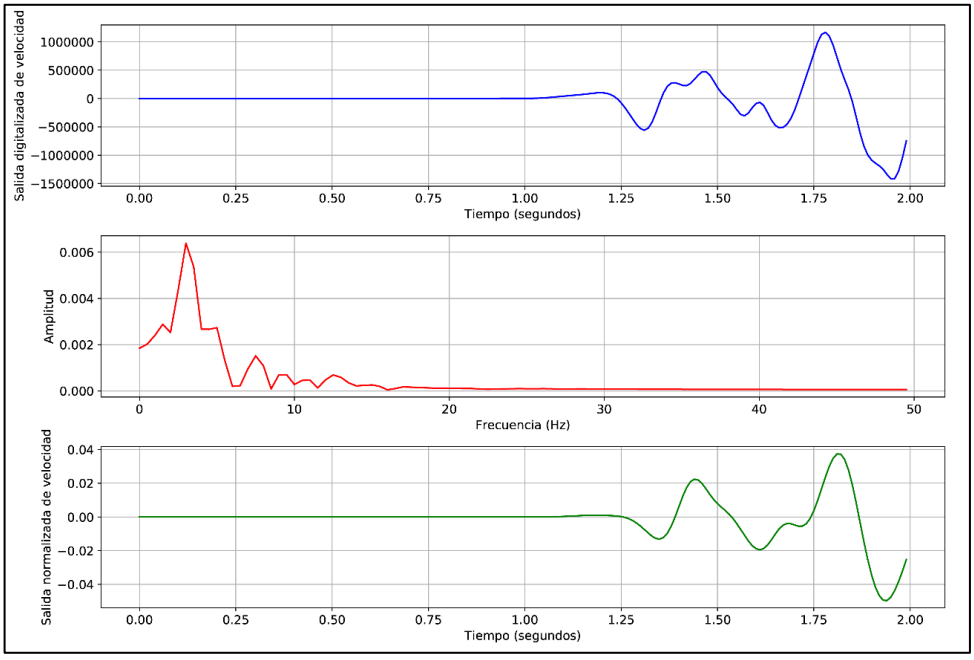


Figura 40. Ventana del evento sísmico mostrado en la Figura 39 con Onda P en el 50%.

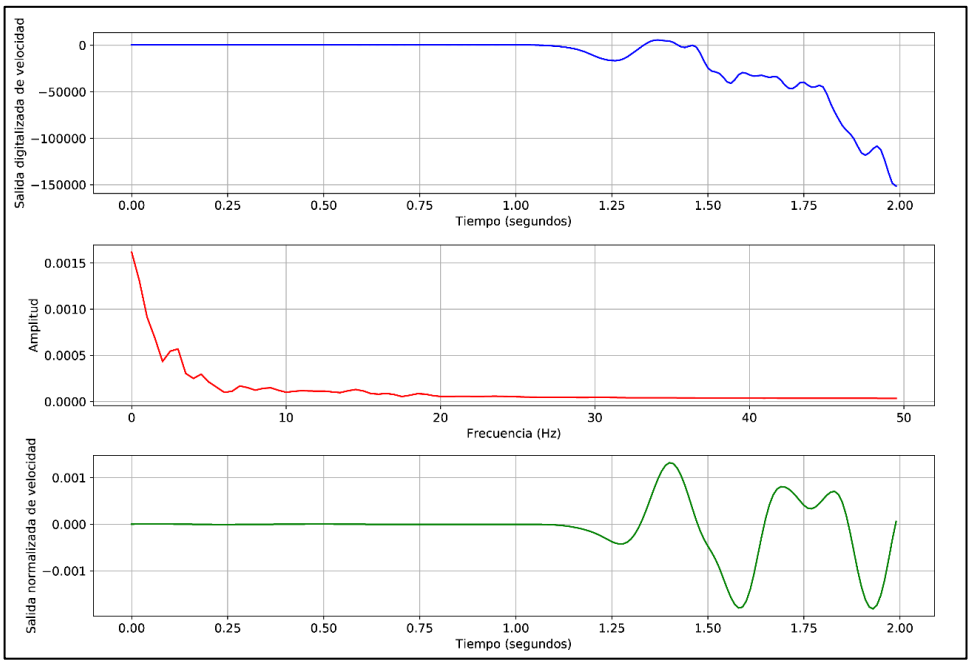
En la Figura 41 se presenta una vista general del proceso de filtrado aplicado a las ventanas extraídas de las señales de los eventos sísmicos, en donde se observan las porciones de señal en las ventanas normalizadas por media y re-muestreadas sin afectar sus componentes frecuenciales (sección en azul de la gráfica), la distribución de frecuencia mediante la Transformada de Fourier (sección en rojo de la gráfica) y el resultado del filtrado de la ventana (sección en verde de la gráfica), de las estaciones BRR, PAM, RUS y PTB del evento sísmico ocurrido el 10 de marzo de 2015 con epicentro en el departamento de Santander.


Salida digitalizada de velocidad
100000
0
−100000
−200000
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00
Tiempo (segundos)
Amplitud
0.0004
0.0002
0.0000
0
10
20
30
40
50
Frecuencia (Hz)
Salida normalizada de velocidad
0.004
0.002
0.000
−0.002
−0.004
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00
Tiempo (segundos)

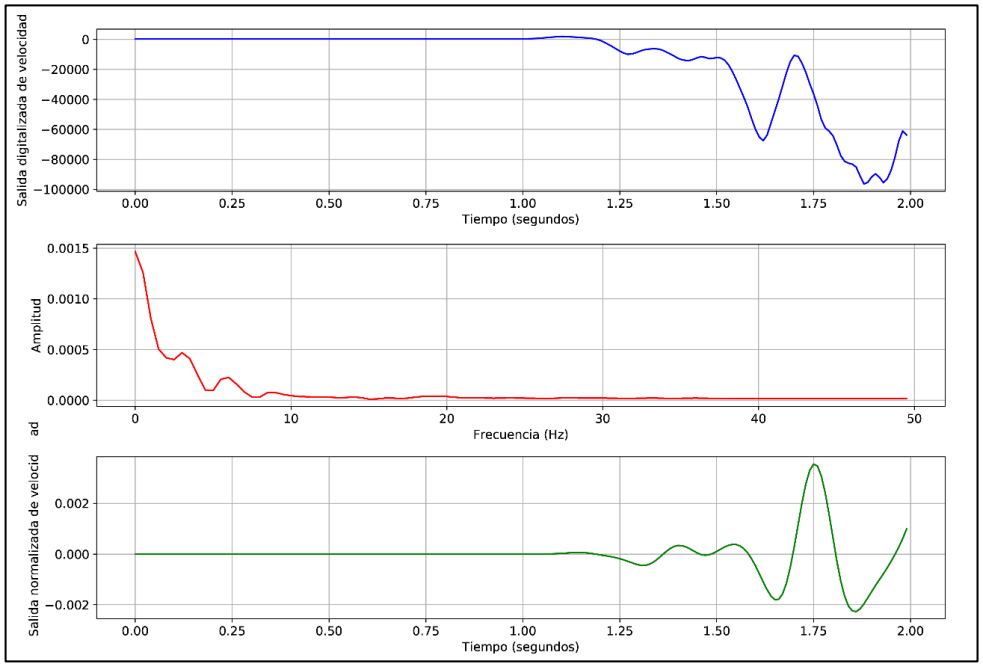
(a)


Salida digitalizada de velocidad
1000000
500000
0
−500000
−1000000
−1500000
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00
Tiempo (segundos)
Amplitud
0.006
0.004
0.002
0.000
0
10
20
30
40
50
Frecuencia (Hz)
Salida normalizada de velocidad
0.04
0.02
0.00
−0.02
−0.04
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00
Tiempo (segundos)

(b)



(c)

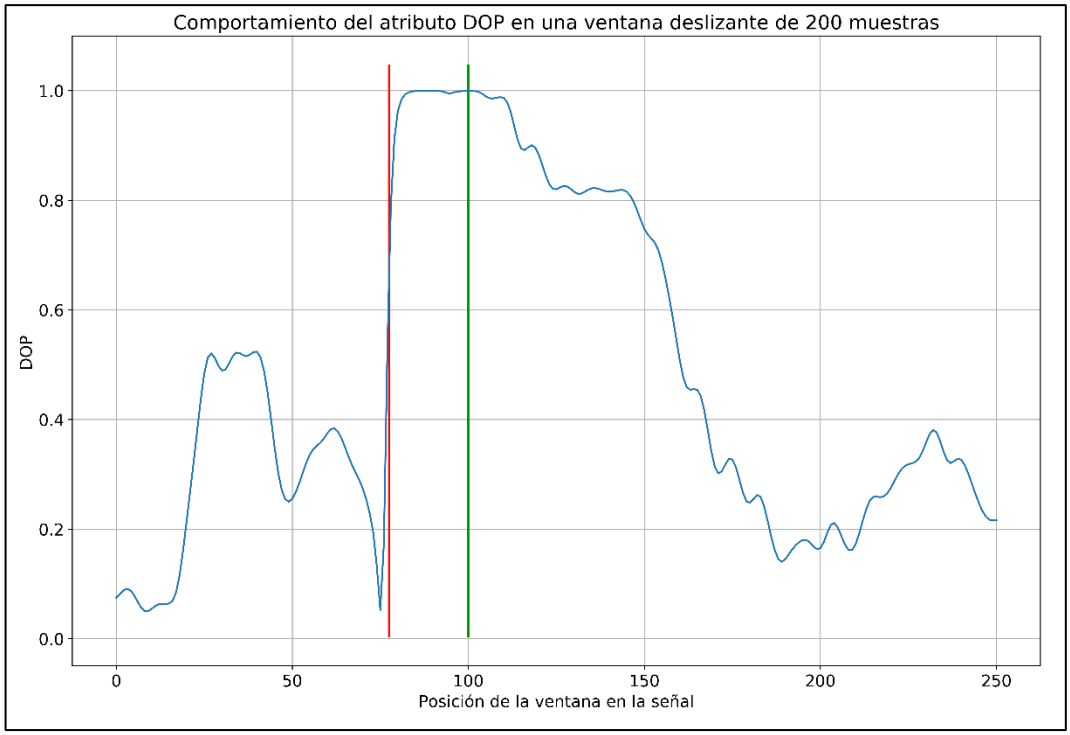


(d)

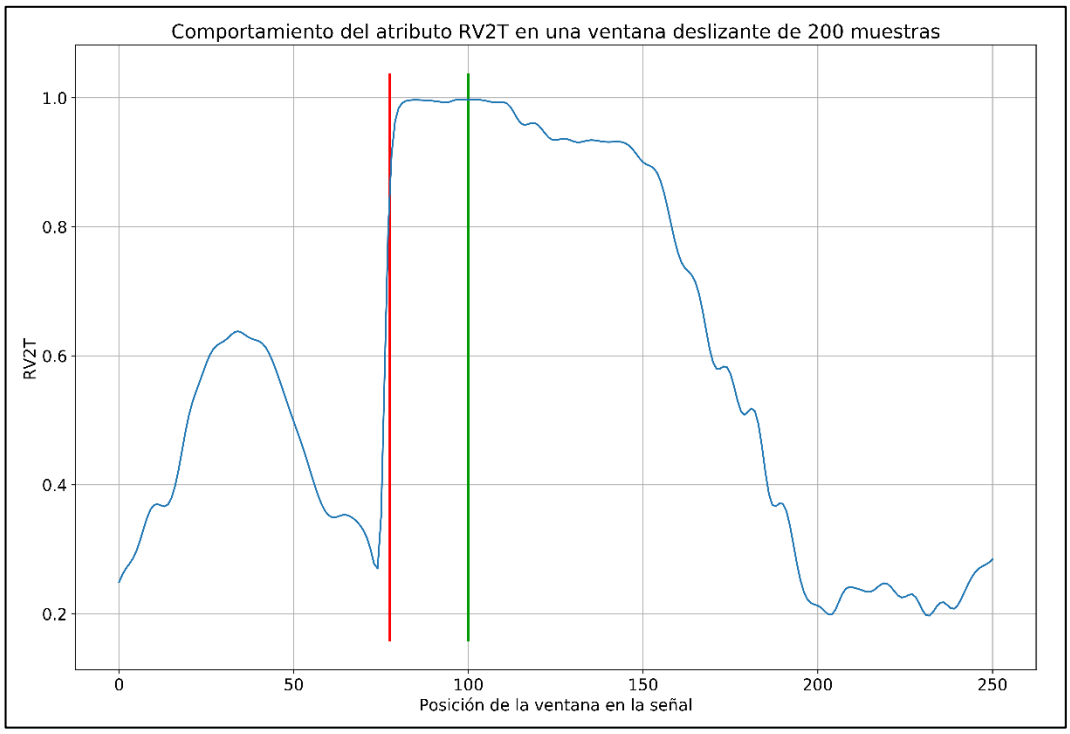
Figura 41. Filtrado, normalización y re-muestreo en ventanas de Onda P del evento del 10 de marzo del 2015 con epicentro en el departamento de Santander, registrado por las estaciones de interés: (a) BRR, (b) PAM, (c) RUS y (d) PTB.

4.5. SELECCIÓN Y EXTRACCIÓN DE ATRIBUTOS

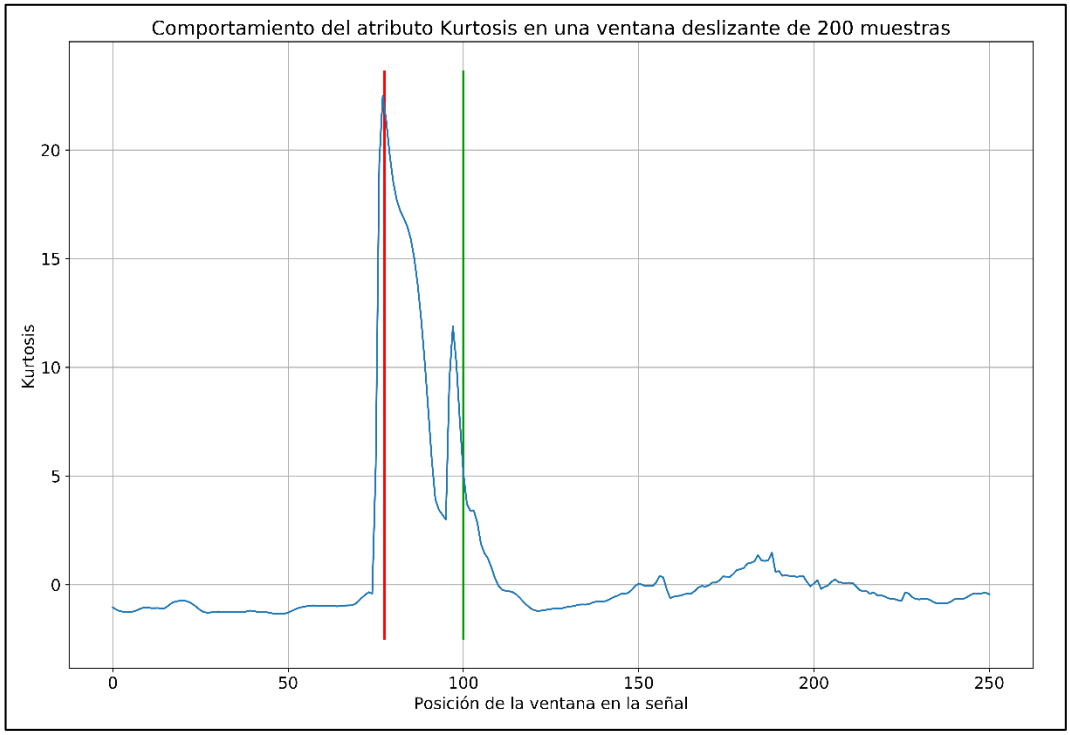
Al calcular los atributos por componente en las estaciones registradas para los eventos sísmicos históricos entrenados, validados y probados, se obtiene lo mostrado en la Figura 42. La línea vertical en color rojo identifica la ventana en la cual la Onda P se encuentra en la posición del 90% y la línea vertical verde indica la ventana en la cual la Onda P se encuentra en el 50% de la ventana. El comportamiento de los atributos ha sido graficado teniendo en cuenta un corrimiento de una ventana de 200 muestras cada 100 sobre las componentes de las estaciones de interés de cada uno de los eventos analizados. En la figura se aprecia el comportamiento de los atributos para la componente vertical de la estación BRR del sismo ocurrido el 6 de junio de 2013. El eje de la variable independiente representa la posición de la ventana que se va moviendo cada 100 muestras y la variable dependiente muestra la magnitud del atributo calculado sobre la ventana en desplazamiento.



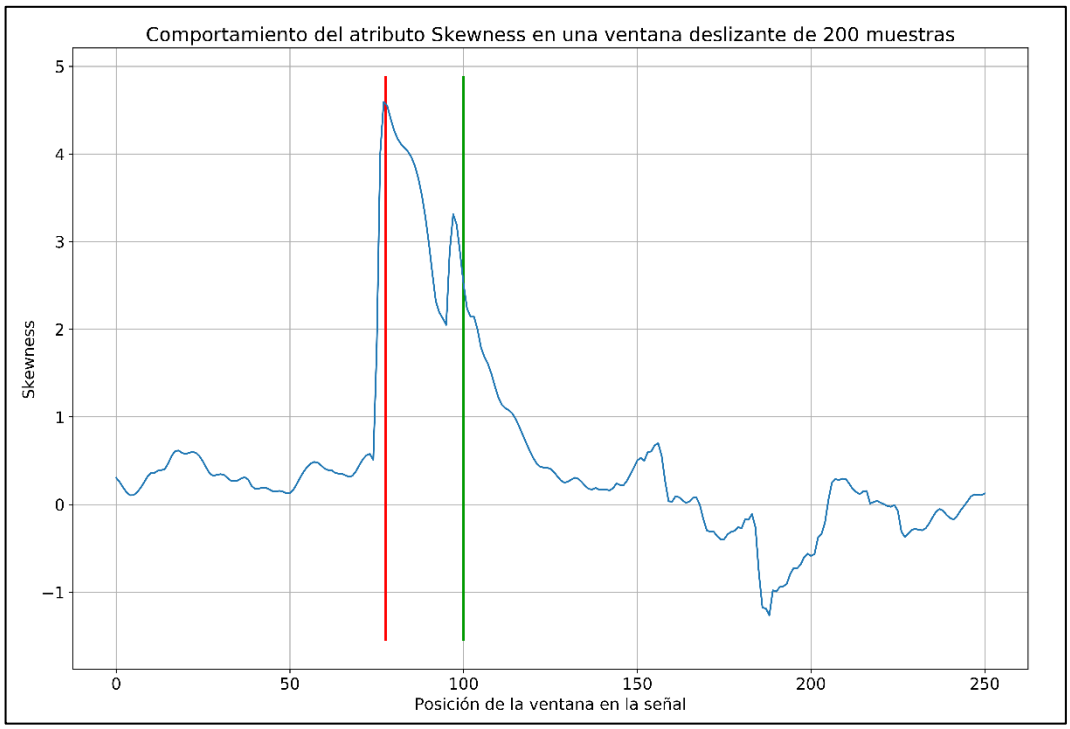
(a)


Comportamiento del atributo RV2T en una ventana deslizante de 200 muestras
RV2T
1.0
0.8
0.6
0.4
0.2
0
50
100
150
200
250
Posición de la ventana en la señal

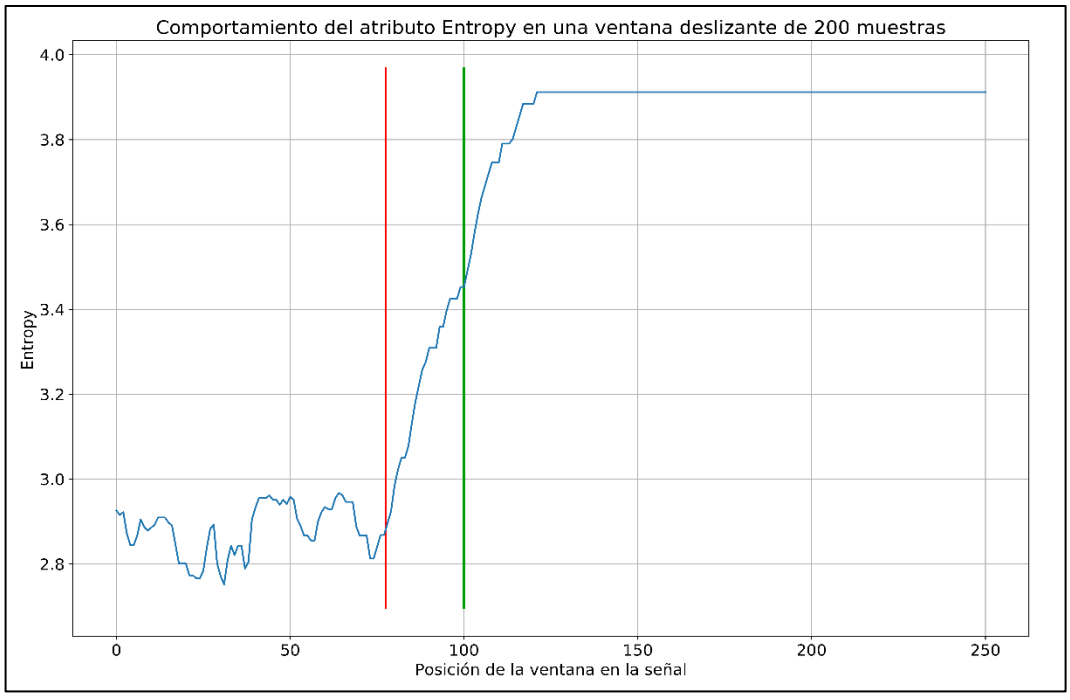
(b)


Comportamiento del atributo Kurtosis en una ventana deslizante de 200 muestras
Kurtosis
20
15
10
5
0
0
50
100
150
200
250
Posición de la ventana en la señal

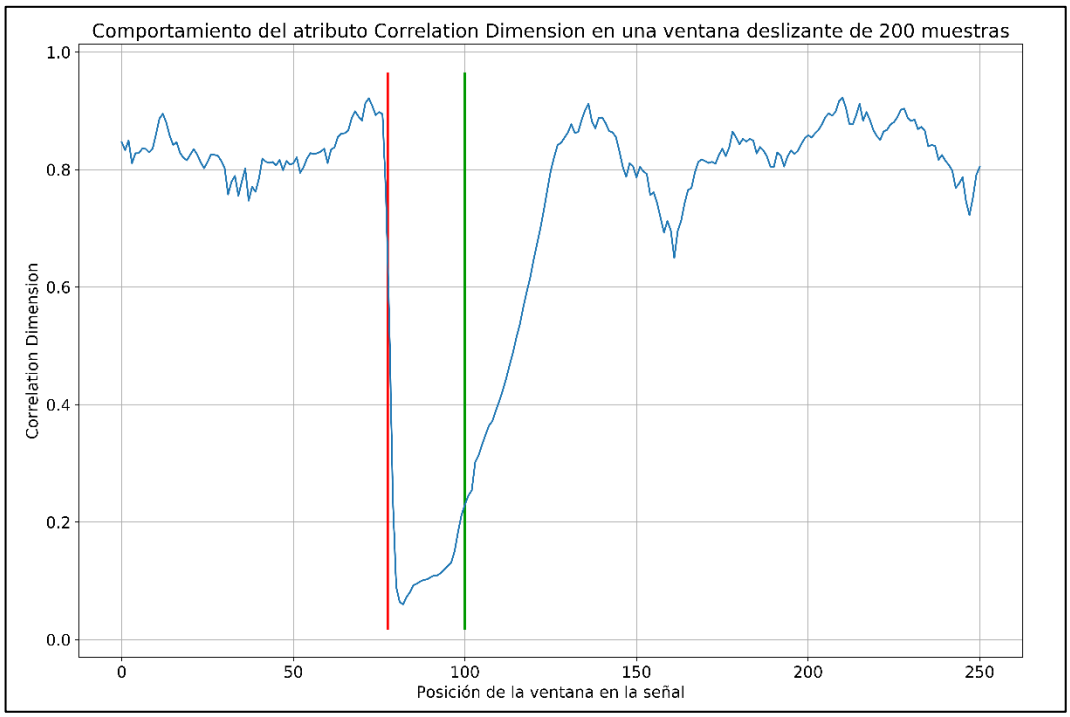
(c)


Comportamiento del atributo Skewness en una ventana deslizante de 200 muestras
Skewness
Posición de la ventana en la señal
5
4
3
2
1
0
−1
0
50
100
150
200
250

(d)


Comportamiento del atributo Entropy en una ventana deslizante de 200 muestras
Entropy
Posición de la ventana en la señal
4.0
3.8
3.6
3.4
3.2
3.0
2.8
0
50
100
150
200
250

(e)



(f)

Figura 42. Comportamiento de los atributos a medida que la ventana se desliza sobre la traza de la componente: (a) DOP, (b) RV2T, (c) Kurtosis, (d) Asimetría, (e) Entropía, (f) Dimensión de Correlación.

Puede notarse que la dinámica de los atributos cambia cuando la Onda P aparece en las ventanas. Mientras la onda está fuera de ellas, el atributo tiene una dinámica regular que identifica al ruido. Cuando aparece la onda, se presenta un cambio en la dinámica que se va atenuando a medida que la traza es recorrida hasta el final, donde se presenta la Onda S y nuevamente ruido sísmico.

4.6. PROCESO DE CLASIFICACIÓN

El proceso de clasificación para el prototipo V0.2.0 contempla un *dataset* de atributos sísmicos que incluye los datos provenientes de 4 estaciones sismológicas (BRR, RUS, PAM, PTB) con marca de onda P, ubicada en el 50% de la ventana. Este *dataset* contiene 56 columnas, correspondientes a los 14 atributos sísmicos

calculados para cada estación, con respecto a cada una de las observaciones y al indicador de la clase a la que se asocia cada observación (Onda P o Ruido). Existen 10.288 observaciones, expresadas en forma de filas, en donde 50% pertenecen a la clase Onda P (expresado como 1) y 50% a la clase Ruido (expresado como 0).

Como se pudo observar en el proceso de optimización de hiperparámetros (Figura 19), el *dataset* es dividido en 80% para *training set* y 20% para *test set* al ingresar al bloque de *Grid Search*. Esto quiere decir que 8.572 observaciones son usadas para el proceso optimización de hiperparámetros y 2.143 observaciones se usan para probar el desempeño del modelo sugerido por el proceso de *Grid Search.* Por otro lado, el *dataset* completo (las 10.288 observaciones) entra al bloque de *Monte Carlo Cross Validation* y es dividido en un 64% para *training set* (6.584 observaciones), 16% para *validation set* (1.646 observaciones) y 20% para *test set* (2.058 observaciones).

Arquitectura del Clasificador

Las características de la arquitectura propuesta para el clasificador asociado al prototipo V0.2.0, de acuerdo con los hiperparámetros que se incluyeron en el proceso de optimización mediante *Grid Search*, se seleccionaron considerando los criterios propuestos por algunos autores. A continuación, se presenta la arquitectura propuesta para la red neuronal artificial tipo *feedforward backpropagation* usada para el proceso de clasificación:

- Cantidad de capas ocultas: 2. Según Huang (2003)[110], el uso de 2 capas ocultas puede reducir significativamente la cantidad de neuronas ocultas requeridas para representar conjuntos grandes de datos de entrada.

[110] HUANG, Guang-Bin. Learning capability and storage capacity of two-hidden-layer feedforward networks. IEEE Transactions on Neural Networks, 14(2), 274–281. DOI:10.1109/tnn.2003.809401. 2003.

- Cantidad de neuronas por capa: 55–28–14-1. Se considera que el uso de arquitecturas piramidales tiene un buen desempeño en términos generales Larochelle *et al* (2009)[111].
- Función de activación por capa: para las neuronas ocultas se usó ReLU (*Rectifier Linear Unit*), debido al buen desempeño generalizado que muestran en las redes multicapa unidireccionales (*feedforward*)[112,113]. Para la neurona de salida, se seleccionó Sigmoid**,** ya que esta función realiza una regresión logística de la salida, permitiendo la obtención de la probabilidad de que una observación pertenezca a cierta clase[114].
- Función de error: se seleccionó *Log Loss* (*Binary Cross Entropy*), debido a su desempeño superior en comparación a otras funciones de error, en la tarea de clasificación binaria[115].

Optimización de Hiperparámetros

Tras obtener un modelo de clasificación con la arquitectura definida anteriormente y siguiendo el procedimiento descrito en la Sección 3.3.2.7, se obtuvieron los siguientes hiperparámetros para el mejor clasificador:

---

[111] LAROCHELLE, Hugo, *et al*. Exploring Strategies for Training Deep Neural Networks. The Journal of Machine Learning Research. Volume 10, 12/1/2009.

[112] AGARAP, Abien. Deep Learning using Rectified Linear Units (ReLU). Department of Computer Science Adamson University. arXiv:1803.08375 [cs.NE]. 2018.

[113] HANSSON, Magnus; OLSSON, Christoffer. Feedforward neural networks with ReLU activation functions are linear splines. Lund University, Sweden. ISSN: 1654-6229. 2017.

[114] GOODFELLOW, Ian; BENGIO, Yoshua; COURVILLE, Aaron. Chapter 6: Deep Feedforward Networks. In: Deep Learning. MIT Press. ISBN: 978-0-26-203561-3. 168-224 pp. 2016.

[115] JANOCHA, Katarzyna; CZARNECKI, Wojciech. On Loss Functions for Deep Neural Networks in Classification. Theoretical Foundations of Machine Learning 2017 (TFML 2017). arXiv:1702.05659v1 [cs.LG]. 2017.

- Algoritmo de optimización: *Adadelta*
- Batch size: 64
- Cantidad de epochs: 10

Con los hiperparámetros hallados, es posible generar las curvas de aprendizaje asociadas al proceso de entrenamiento del clasificador, usando diferente cantidad de *epochs*, como se muestra en la Figura 43.

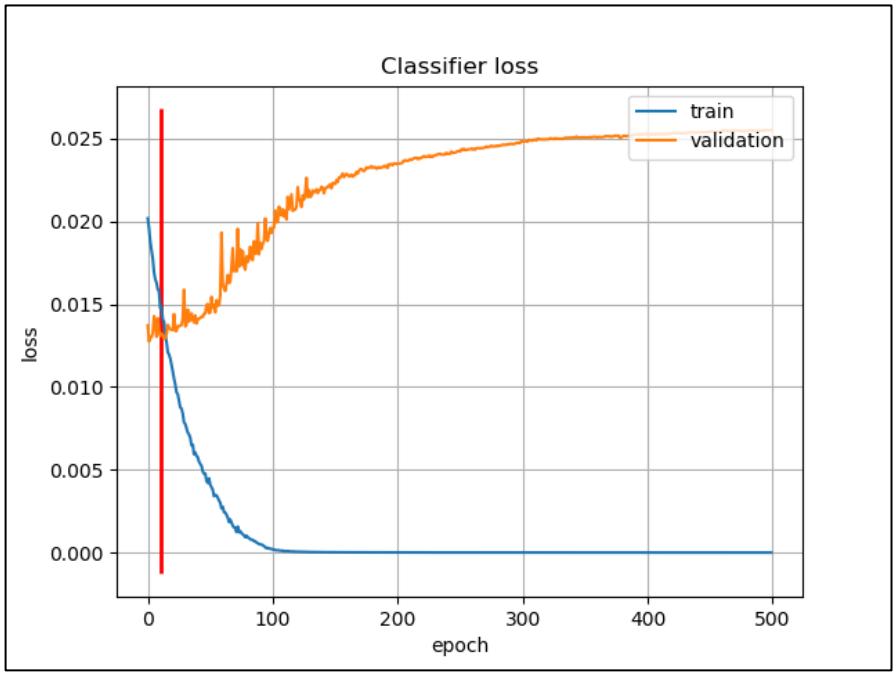


Figura 43. Curva de aprendizaje (error) del clasificador.

Se puede apreciar el resultado de la función de error presentado durante un entrenamiento simple para un modelo de red neuronal con la arquitectura descrita en la Tabla 11, en donde se expone el error de entrenamiento y de validación para un total de 500 *epochs*. La línea azul, que indica el error de entrenamiento, presenta un comportamiento decreciente que consigue llegar cerca de cero tras superar los 100 *epochs*. La separación agresiva de la línea azul con respecto a la naranja (error de validación) indican un comportamiento de *overfitting*. Por ello, a medida que el error de entrenamiento tiende a cero, el de validación (línea naranja) tiende a

aumentar, debido a que la red neuronal pierde la capacidad de clasificar nuevas observaciones.

La línea roja ubicada verticalmente corresponde al valor de 10 *epochs*, valor sugerido a través de *Grid Search*. Teniendo en cuenta que las opciones de cantidad de *epochs* son [10, 50, 100, 250, 500], se puede observar que gracias al proceso de optimización de hiperparámetros, se seleccionó una cantidad de *epochs* apropiada para evitar el *overfitting*. Cabe resaltar que la respuesta de la función de error depende de otros hiperparámetros como el algoritmo de optimización, el *batch size* y la arquitectura general de la red.

Como salida del bloque de *Grid Search* y como entrada al bloque de *Monte Carlo Cross Validation,* se obtiene el modelo de red neuronal descrito en la Tabla 11. En la Tabla 12 se muestran las métricas calculadas a partir del proceso de *K-fold Cross Validation* usado dentro del *Grid Search*.

Tabla 11. Características del clasificador del prototipo V0.2.0 entrenado con 4 estaciones y la onda P al 50% de la ventana.

Característica	Valor
Número de capas ocultas	2
Número de neuronas por capa oculta	29 y 14
Número de neuronas de entrada	55
Número de neuronas de salida	1
Función de activación de capa de entrada	Rectificador
Función de activación de capas ocultas	Rectificador
Función de activación de capa de salida	Sigmoide
Función de entrenamiento	Adadelta
Topología	55-28-14-2
Batch Size	8
Número de epochs	10
Cantidad de folds	10
Tipo de aprendizaje	Feedforward
Función de propagación	Backpropagation

Tabla 12. Métricas de desempeño de Grid Search.

Métrica	Valor
Sensibilidad	0,9880
Especificidad	0,9943
Precisión	0,9944
Recall	0,9880
F1	0,9912
Accuracy	0,9911

Validación cruzada

Con la salida del bloque *Grid Search*, obteniendo los mejores hiperparámetros (algoritmo de optimización: *Adadelta*, *Batch size*: 64, cantidad de *epochs*: 10), se usa el modelo generado como entrada para el bloque *Monte Carlo Cross Validation,* adicionando como entrada el *dataset* de atributos sísmicos (10.288 observaciones). Siguiendo el procedimiento descrito en la Sección 3.3.2.7 para validación cruzada, se muestran las métricas resultantes para el conjunto de prueba con las variaciones en estaciones y posiciones de la onda P en las Tabla 13 y en la Figura 44. En el Anexo G se muestran las métricas detalladas.

Tabla 13. Métricas de salida para el *Test set* del bloque *Monte Carlo Cross Validation* para el prototipo V0.2.0 con 4 estaciones y la onda P al 50% de la ventana.

Iteración	Sensibilidad	Especificidad	Precisión	*Recall*	F1	*Accuracy*
Iteración 1	0.9890	0.9915	0.9917	0.9890	0.9903	0.9902
Iteración 2	0.9934	0.9963	0.9962	0.9934	0.9948	0.9949
Iteración 3	0.9926	0.9934	0.9935	0.9926	0.9931	0.9930
Iteración 4	0.9943	0.9945	0.9943	0.9943	0.9943	0.9944
Iteración 5	0.9899	0.9905	0.9909	0.9899	0.9904	0.9902
Iteración 6	0.9886	0.9945	0.9943	0.9886	0.9915	0.9916
Iteración 7	0.9887	0.9954	0.9953	0.9887	0.9920	0.9921
Iteración 8	0.9900	0.9952	0.9954	0.9900	0.9927	0.9925
Iteración 9	0.9886	0.9918	0.9914	0.9886	0.9900	0.9902
Iteración 10	0.9882	0.9952	0.9954	0.9882	0.9918	0.9916
Promedio	0.9903	0.9938	0.9938	0.9903	0.9921	0.9921

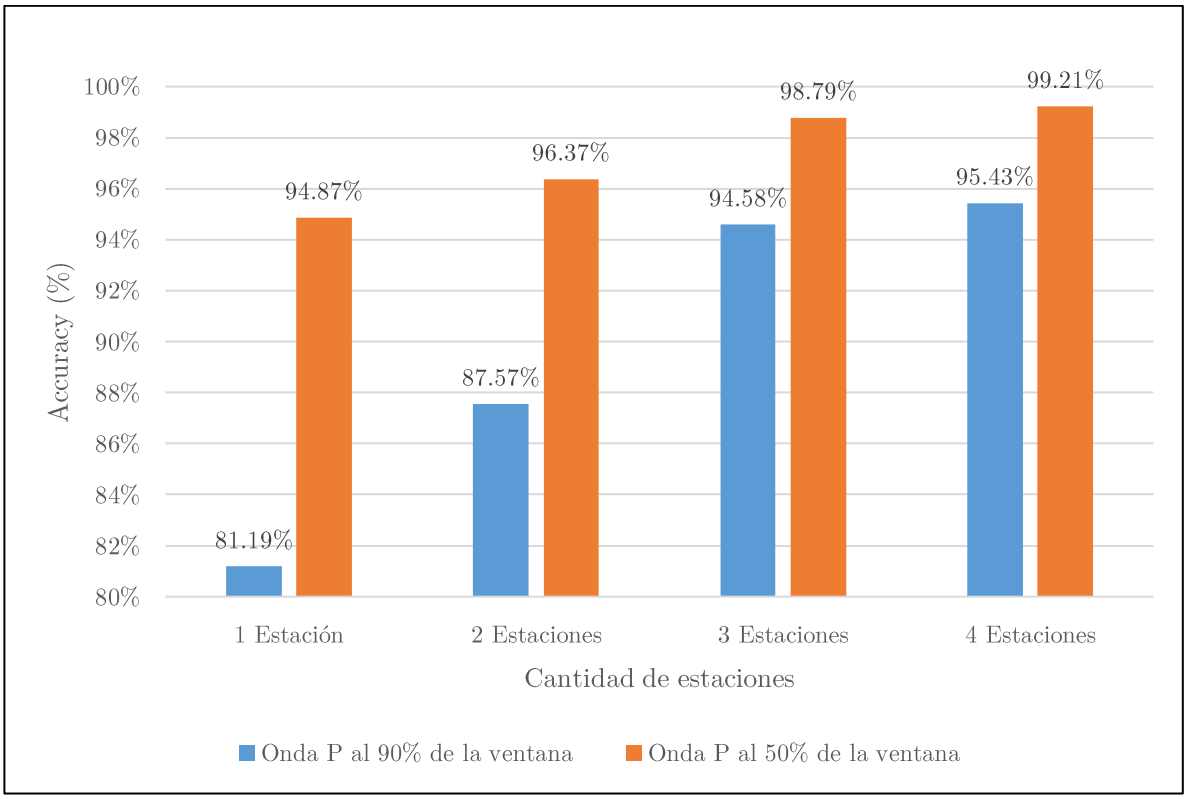


Figura 44. Comparativo del desempeño del clasificador concerniente al prototipo V0.2.0 con variaciones en la cantidad de estaciones y en la posición de la onda P en la ventana de observación.

La Tabla 13 describe las métricas asociadas para el *test set* del prototipo V0.2.0. Allí se pueden evidenciar los valores para las 10 iteraciones propuestas para el bloque de *Monte Carlo Cross Validation* y el promedio de dichas métricas. En la Figura 44 se aprecian las diferencias en el desempeño del clasificador del prototipo V0.2.0 cuando existen variaciones en la cantidad de estaciones y en la posición de la onda P en la ventana. Puede notarse que existe un incremento en las métricas de desempeño cuando la cantidad de estaciones es aumentada, considerando la misma cantidad de datos de entrada (10.288). De igual forma, al variar la posición de la onda P del 90% al 50% de la ventana, existe un incremento en las métricas de desempeño del clasificador, obteniendo el mejor desempeño con 4 estaciones y la onda P al 50% de la ventana.

Por otro lado, como se puede apreciar en la Tabla 13, el rango de variación de los valores de las métricas es reducido (menor al 1%). En la Tabla 14 se exponen los rangos de variación para la Sensibilidad, la Especificidad y el *F1-Score* entre la peor y la mejor iteración encontradas.

Tabla 14. Rangos de variación de sensibilidad, especificidad y F1-Score.

Métrica	Menor valor	Mayor valor	Rango de variación
Sensibilidad	98,82%	99,43%	0,61%
Especificidad	99,05%	99,63%	0,58%
F1	99,0%	99,48%	0,48%

Esta corta brecha entre las iteraciones de Monte Carlo permite afirmar que el modelo es generalizable para los datos de entrada.

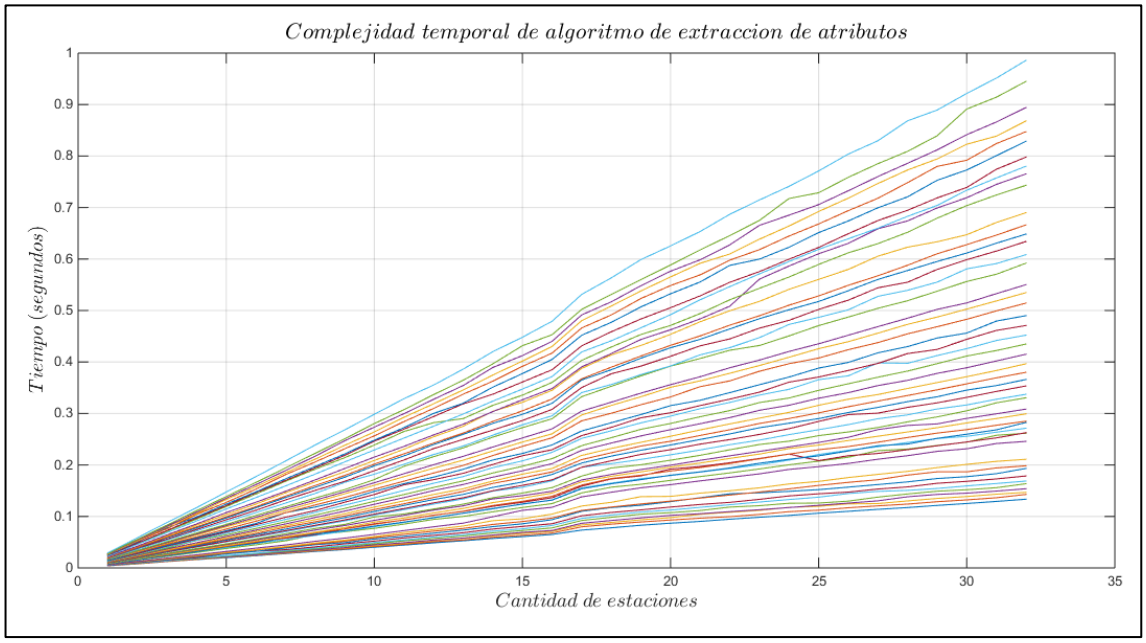
4.7. COMPLEJIDAD TEMPORAL

A continuación, se presentan los resultados del desempeño en tiempo relacionado con los procesos de extracción de atributos y clasificación, teniendo en cuenta el procedimiento detallado en la Sección 3.3.3.

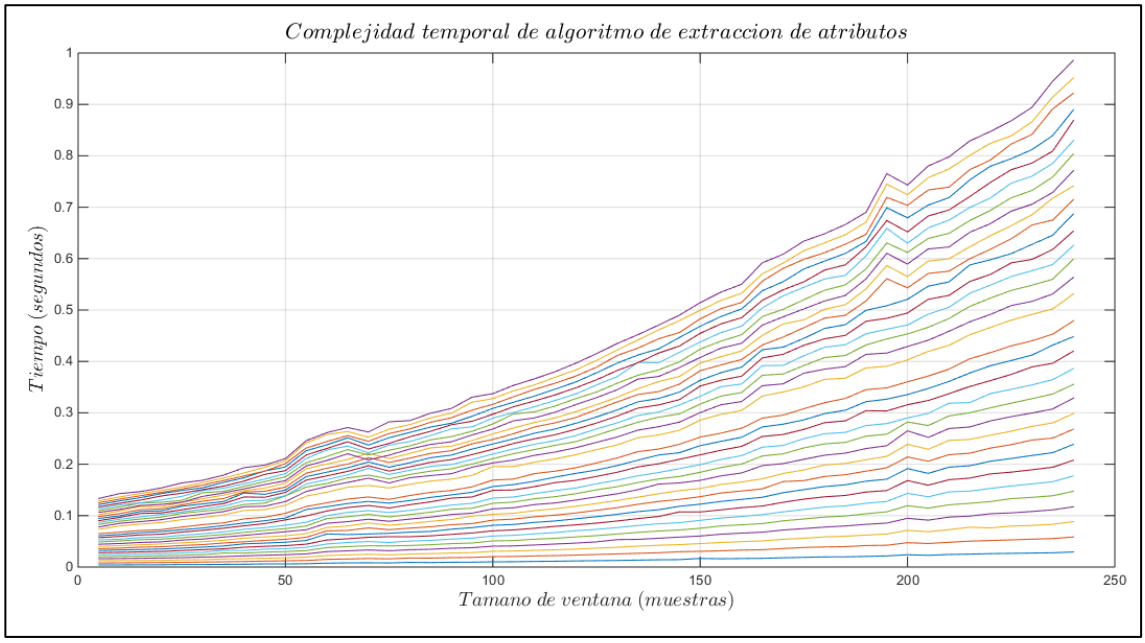
4.7.1.1. Complejidad temporal de la extracción de atributos

Al ejecutar el procedimiento en el que se consideran las variables de cantidad de estaciones, tamaño de la ventana y tiempo, los resultados del desempeño en tiempo del proceso de extracción de atributos son los que se muestran en la Figura 45. En Figura 45(a) se muestra el comportamiento en tiempo cuando la cantidad de estaciones es incrementada a medida que se ejecuta la extracción de atributos. Las diversas curvas mostradas corresponden a las variaciones en el tamaño de la ventana de extracción, desde una ventana de 5 muestras, hasta una ventana de 245 muestras, conservando la variación en la cantidad de estaciones. Puede

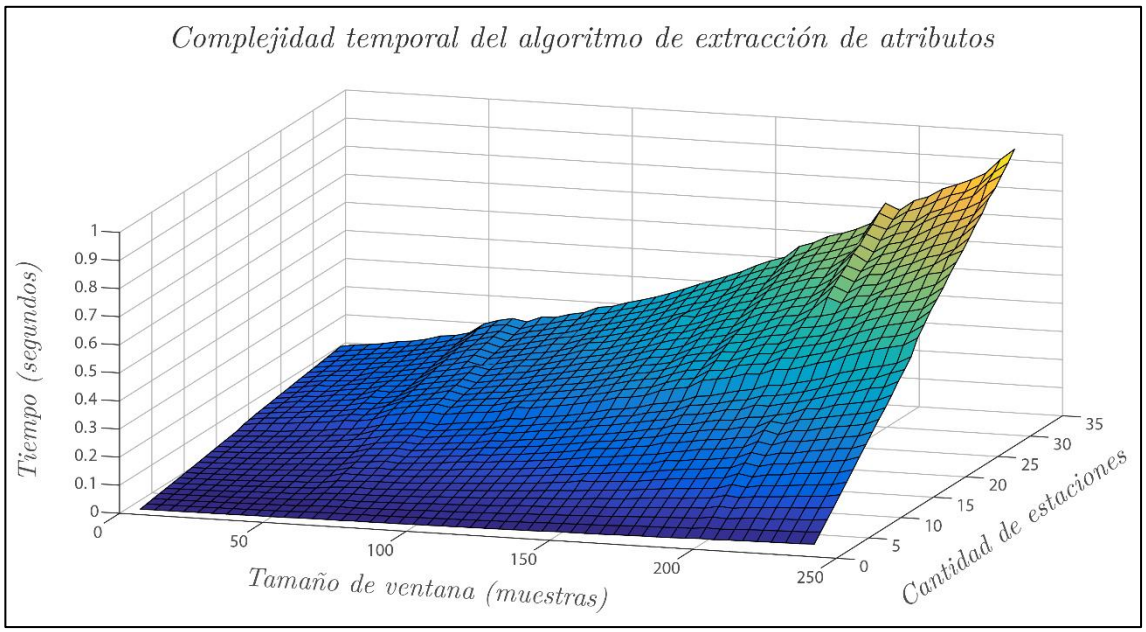
observarse que se trata de un comportamiento lineal que puede generalizarse como O(n).



(a)



(b)



(c)

Figura 45. Comportamiento en tiempo del proceso de extracción de atributos teniendo en cuenta: (a) una vista bidimensional que relaciona la cantidad de estaciones contra el tiempo de extracción, (b) una vista bidimensional que relaciona la variación en el tamaño de la ventana contra el tiempo de extracción y, (c) una vista tridimensional que relaciona las tres variables nombradas.

En la Figura 45(b) se muestra el comportamiento en tiempo cuando la ventana sobre la que se ejecuta la extracción de atributos es variada. Las diversas curvas mostradas corresponden a las variaciones en la cantidad de estaciones, desde la extracción sobre una estación, a la extracción sobre 32 estaciones, conservando la variación en la ventana. Puede observarse que se trata de un comportamiento polinomial que se resume en un comportamiento cuadrático de la forma $O(n^2)$.

En la superficie de la Figura 45(c) se muestra el comportamiento en tiempo general cuando el tamaño de las ventanas de extracción de atributos es variado, al igual que la cantidad de estaciones. La ecuación general para expresar la complejidad temporal del extractor de atributos, teniendo en cuenta las variaciones en la longitud de la ventana y en la cantidad de estaciones, es la siguiente:

$$O(m \cdot n^2) \qquad \text{(Ecuación 23)}$$

Donde *m* es la cantidad de estaciones y *n* es el tamaño de la ventana. Puede notarse en las tres gráficas que para una ventana de 200 muestras como la usada y una cantidad máxima de 4 estaciones, el tiempo para la extracción de atributos de esa ventana es alrededor de 86 milisegundos. Para una ventana deslizante con un solapamiento de 100 muestras (50% de la ventana), es decir un corrimiento de 1 segundo, el cálculo de los atributos demora alrededor de un 8,5% del tiempo de llegada de la siguiente ventana, lo que deja un 91,5% del tiempo libre para el proceso de clasificación, antes de que la siguiente ventana arribe.

4.7.1.2. Complejidad temporal del proceso de clasificación

Al ejecutar el procedimiento en el que se consideran las variables de cantidad de estaciones, cantidad de observaciones y tiempo, los resultados del desempeño en tiempo del proceso de clasificación son los que se muestran en la Figura 46. La Figura 46(a)) muestra el comportamiento en tiempo cuando la cantidad de estaciones es incrementada a medida que se ejecuta el proceso de clasificación.

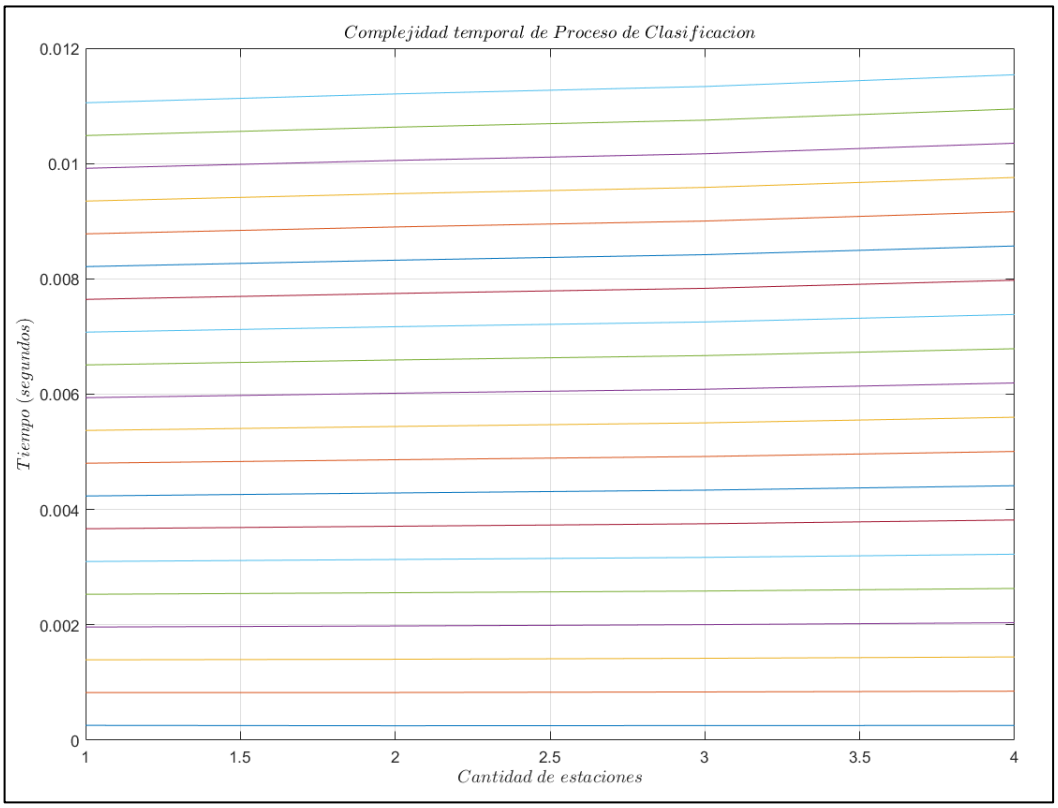
Las diversas curvas mostradas corresponden a las variaciones en la cantidad de observaciones, desde 1 observación hasta 1.000 observaciones, conservando la variación en la cantidad de estaciones. Con el fin de que la gráfica fuese apreciable, se muestran los incrementos en pasos de 50 observaciones. Puede notarse que se trata de un comportamiento lineal que va creciendo en pendiente que puede generalizarse como O(n).

En la Figura 46(b) se muestra el comportamiento en tiempo cuando la cantidad de observaciones entrantes es variada. Las cuatro curvas mostradas corresponden a las variaciones en la cantidad de estaciones, desde la extracción sobre una estación a la extracción sobre 4 estaciones, conservando la variación en las observaciones. Puede notarse que se trata de un comportamiento nuevamente lineal que se puede expresar como O(n).

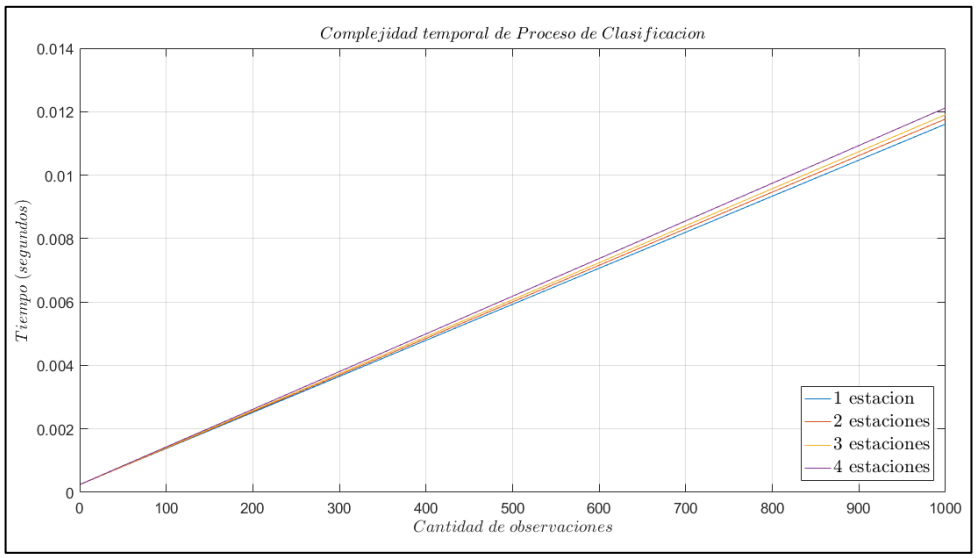
En la superficie de la Figura 46(c) se muestra el comportamiento en tiempo general cuando la cantidad de observaciones es variada junto con la cantidad de estaciones. La ecuación general para expresar la complejidad temporal del proceso de clasificación, teniendo en cuenta estas variaciones, es la siguiente:

$$O(m \cdot q) \qquad \text{(Ecuación 24)}$$

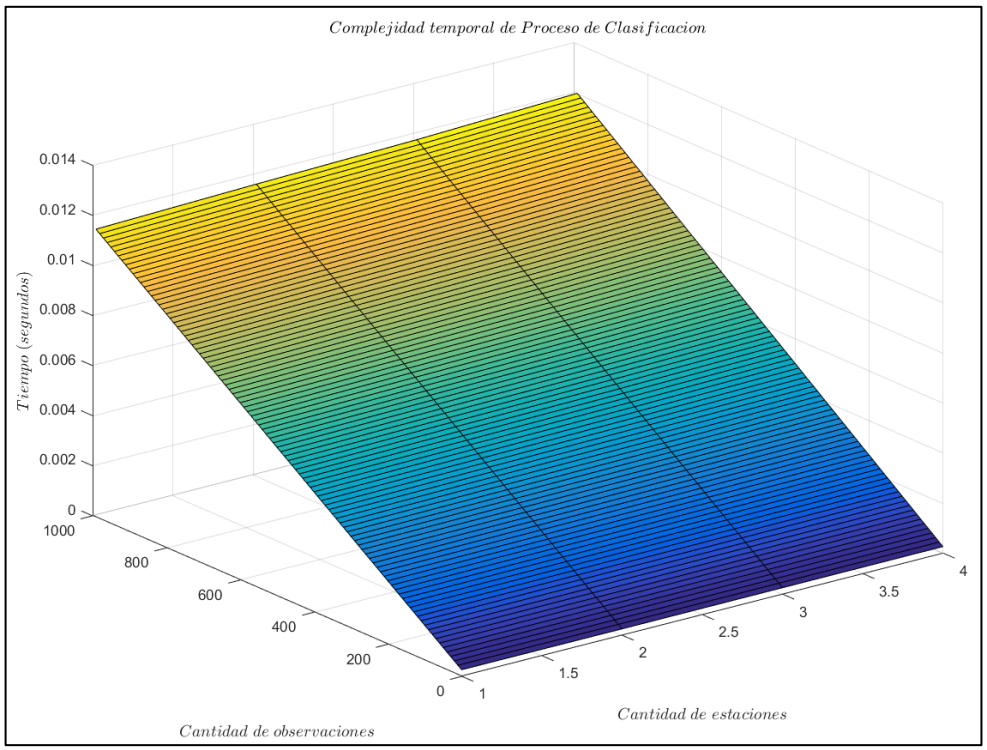
Donde *m* es la cantidad de estaciones y *q* es la cantidad de observaciones. En la gráfica mostrada en la Figura 47 se hace una ampliación y proyección al comportamiento en tiempo del clasificador con una observación y una variación de 1 a 32 estaciones, con el fin de hacer comparables los desempeños en tiempo de extracción de atributos con el proceso de clasificación.



(a)



(b)



(c)

Figura 46. Comportamiento en tiempo del proceso de clasificación teniendo en cuenta: (a) una vista bidimensional que relaciona la variación en la cantidad de estaciones contra el tiempo de clasificación, (b) una vista bidimensional que relaciona la cantidad de observaciones contra el tiempo de clasificación y (c) una vista tridimensional que relaciona las tres variables nombradas.

El comportamiento del clasificador puede especificarse en términos de variaciones en el tamaño de la ventana como constante, pues el resultado del proceso de extracción de atributos es un valor cuantitativo que representa toda la ventana, no importa su tamaño. Si el comportamiento en tiempo contempla esta variable, teniendo en cuenta la tendencia en la variación en la cantidad de estaciones mostrada en la Figura 47 sobre una observación, se obtiene lo mostrado en la Figura 48.

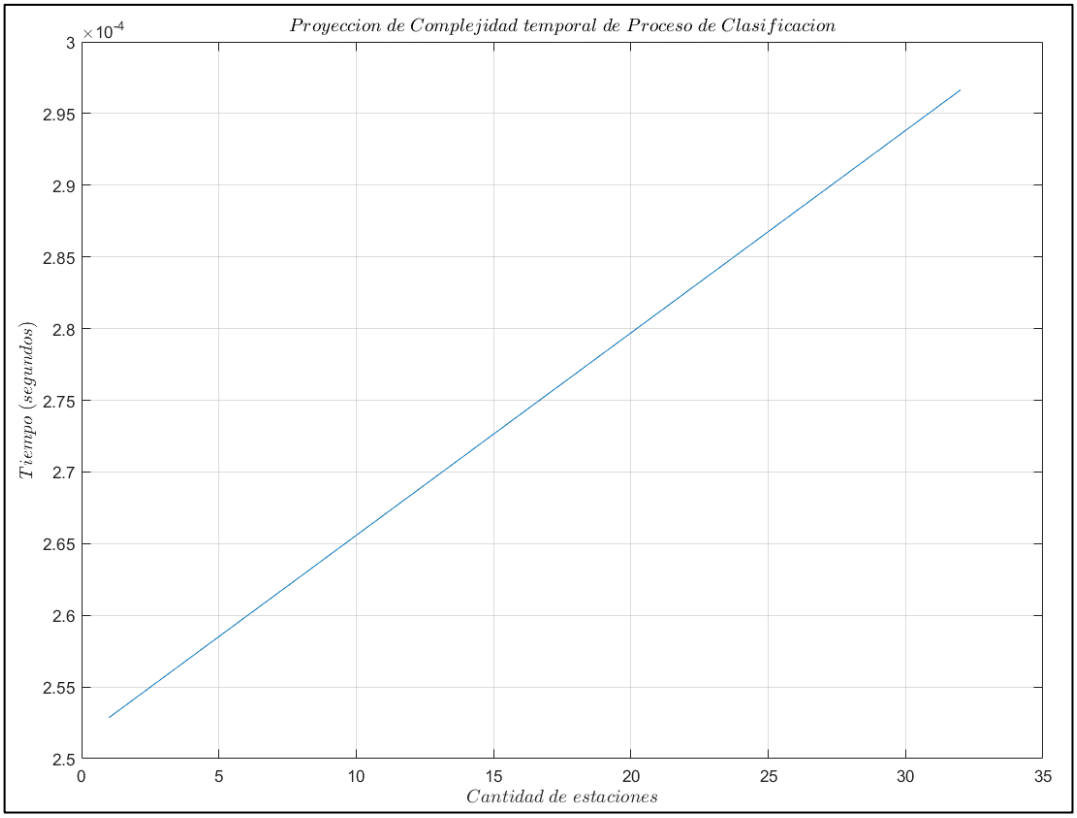


Figura 47. Tendencia del comportamiento en tiempo del proceso del proceso de clasificación para una variación en la cantidad de estaciones sobre una única observación de entrada.

Puede notarse en la gráfica que, para una observación y una cantidad máxima de 4 estaciones, el tiempo para la clasificación de esa observación es alrededor de 0,25 milisegundos. Para una ventana deslizante con un solapamiento de 100 muestras (50% de la ventana), es decir un corrimiento de 1 segundo, el proceso de clasificación demora alrededor de un 0,025% del tiempo de llegada de la siguiente

ventana lo que, en suma, junto con el proceso de extracción de atributos, deja un 91,48% (913,8 milisegundos) del tiempo libre antes de que la siguiente ventana arribe.

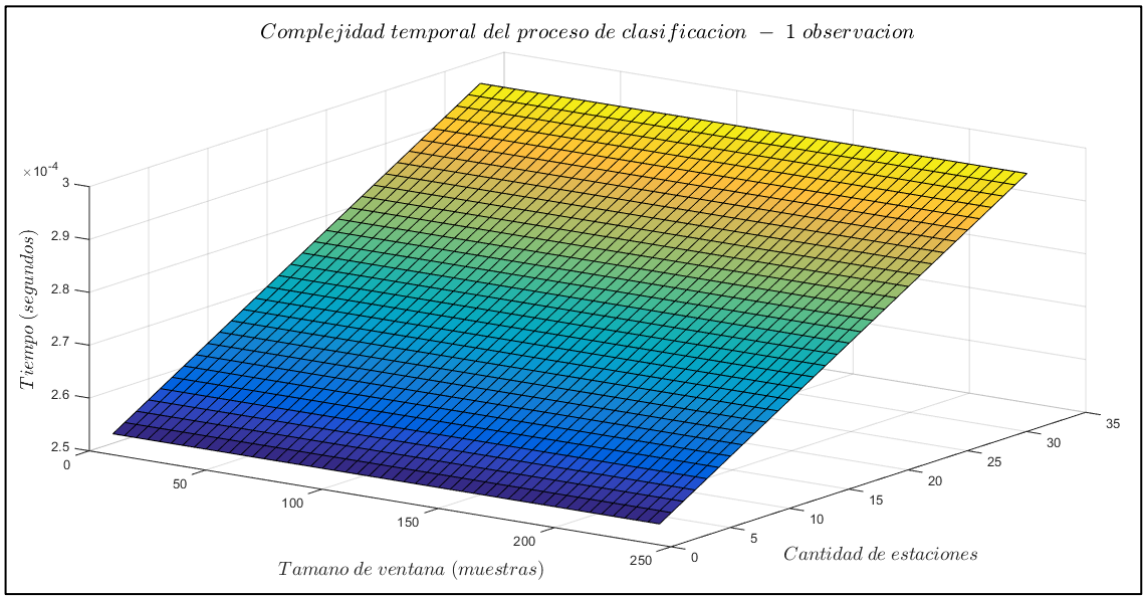


Figura 48. Tendencia del comportamiento en tiempo del proceso del proceso de clasificación para una variación en la cantidad de estaciones sobre una variación de la ventana de extracción de atributos y una única observación de entrada.

En la Figura 49 se muestra un comparativo entre el comportamiento en tiempo del proceso de extracción de atributos (curva superior) y el proceso de clasificación (curva inferior).

Finalmente, la complejidad temporal del clasificador, desde que la ventana de onda P o ruido entran al proceso de extracción de atributos, hasta que es clasificada, puede generalizarse como $O(m \cdot n^2) + O(m)$, lo que resulta en una complejidad temporal $O(m \cdot n^2)$.

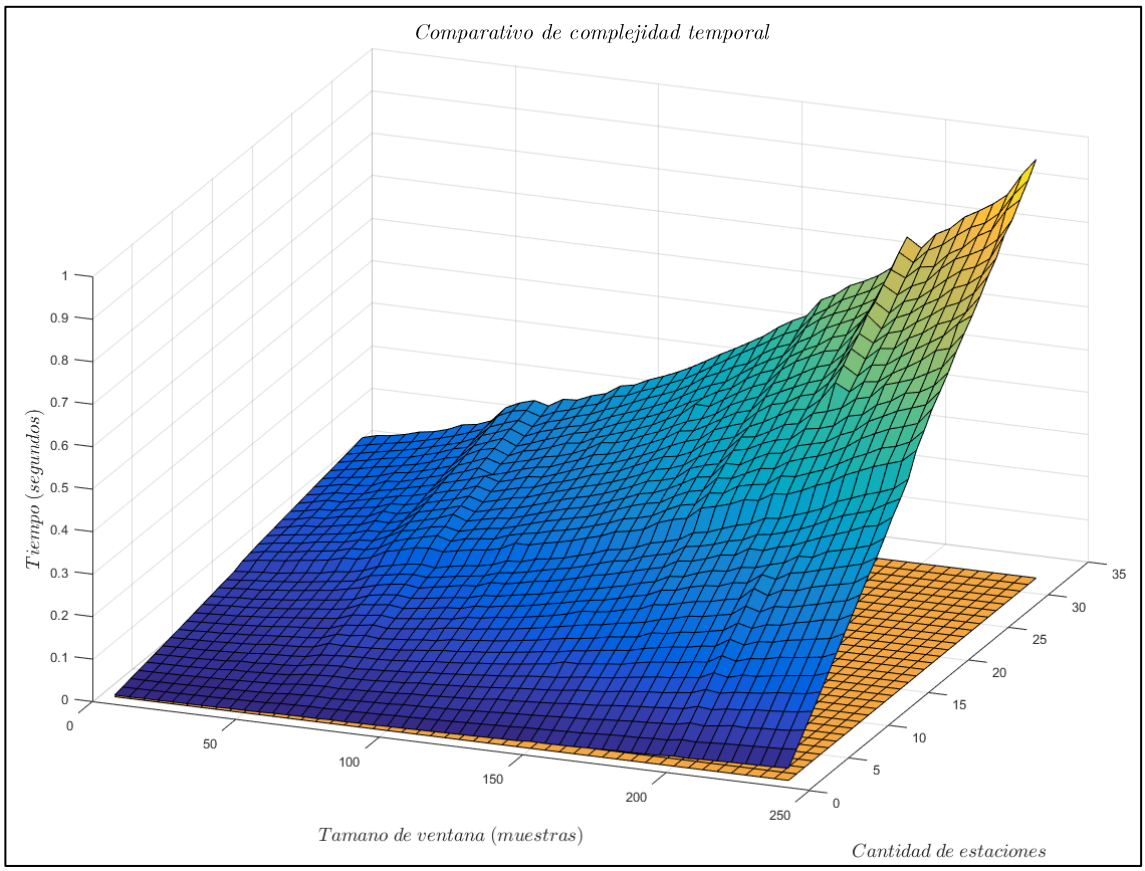


Figura 49. Comparativo entre el comportamiento en tiempo del proceso de extracción de atributos (superficie en azul) y el proceso de clasificación (superficie en naranja).

5. CONCLUSIONES

El sistema para la detección de movimientos sísmicos desarrollado de forma modular permitió la clasificación binaria fuera de línea de los eventos sísmicos, mediante el preprocesamiento, extracción de atributos, entrenamiento, validación y prueba de un modelo de red neuronal artificial. El estrecho margen de diferencia entre las métricas de desempeño por iteración en la validación de Monte Carlo permitió garantizar la generalización del modelo de clasificación.

De acuerdo con el análisis de las estaciones sismológicas, al considerar aspectos de distancia epicentral y conteo de eventos sísmicos, teniendo en cuenta una muestra de eventos locales heterogénea en magnitud y profundidad con epicentro registrado en el departamento de Santander, las estaciones de mayor viabilidad para la clasificación de los eventos fueron BRR, RUS, PAM y PTB. La variación en la cantidad de estaciones y el posicionamiento de la onda P en la ventana de detección influyeron directamente en el tiempo y desempeño del clasificador. El incremento en la cantidad de estaciones mejoró notablemente el desempeño del clasificador, en contraste con el aumento mínimo en el costo temporal. El posicionamiento de la onda P al 50% de la ventana aumentó el desempeño del clasificador.

Gracias a los procesos de filtrado, normalización y re-muestreo ejecutados en el preprocesamiento de las señales, se resaltaron las características de interés y se redujeron las componentes indeseables. Se consideraron seis atributos para esta tarea: DOP, RV2T, Kurtosis, Asimetría, Entropía y Dimensión de Correlación (CD). Estos últimos cuatro atributos permitieron una mejora en el desempeño del clasificador, resaltando que la Entropía y la CD proporcionaron una descripción de la dinámica no lineal de las señales sísmicas.

Por otro lado, la flexibilidad de la metodología Scrum orientada al desarrollo modular y evolutivo de los prototipos propició el trabajo colaborativo y permitió la identificación temprana de las falencias presentadas, lo que condujo a una corrección oportuna de errores y un aumento progresivo en el desempeño del clasificador.

Los resultados obtenidos mostraron un 99,21% de exactitud en la clasificación binaria de eventos sísmicos en Santander (segundo nido sísmico más activo del mundo) reportados del 2015 al 2017. Estos resultados están en concordancia con lo reportado por otros autores: Ibs-von Seht, et. al., quienes obtuvieron un 97% de exactitud en la clasificación con datos entre 2005 y 2007 de Indonesia, y Hasan, et. al., quienes obtuvieron un 90% de precisión en la clasificación con datos entre 2013 y 2014 de Marruecos.

6. RECOMENDACIONES Y TRABAJOS FUTUROS

Con el propósito de continuar con el trabajo realizado y como propuesta para proyectos futuros, se plantean las siguientes recomendaciones referentes al sistema de detección de eventos sísmicos:

- Aumentar las variaciones en el número de estaciones, haciendo un estudio más profundo y detallado del impacto que esto conlleve en aspectos sismológicos y en los procesos de clasificación de eventos sísmicos.

- Aumentar la cantidad de variaciones en la posición de la onda P en la ventana, midiendo las métricas de desempeño promedio, con el fin de analizar cuál es la posición de la onda P en la ventana con la que pueda detectarse con mejor desempeño un evento sísmico.

- Evaluar el impacto en el desempeño del clasificador de la extracción de atributos adicionales, de transformaciones entre los atributos presentados y la expresión de las señales en otros dominios.

- Evaluar la posibilidad de utilización de la optimización de hiperparámetros por búsqueda aleatoria, Bayesiana y evolutiva, con el fin de encontrar valores satisfactorios a un menor costo computacional.

- Ampliar el enfoque de clasificación binaria fuera de línea a clasificación multiclase para la estimación de la magnitud del evento sísmico registrado.

- Estudiar a profundidad las implicaciones de la complejidad temporal en la clasificación y detección de eventos sísmicos, tal que pueda extrapolarse la información analizada a futuros procesos de alerta.

- Considerar y validar otras técnicas de *Machine Learning* más robustas que permitan obtener métricas de desempeño más altas y mayor repetibilidad en el proceso.

- Ampliar el enfoque de clasificación fuera de línea con redes neuronales y *Machine Learning* a la detección en línea de eventos sísmicos.

- Investigar sobre el uso de redes neuronales y clasificación en la localización de eventos sísmicos, tal que pueda estimarse por medio de patrones y dinámicas, la ubicación del epicentro sísmico, con el fin de coadyuvar en las acciones de alerta futuras.

- Investigar sobre la implementación de modelos adaptativos y evolutivos que permitan considerar las condiciones geográficas y topográficas cambiantes y adaptarse a las mismas.

BIBLIOGRAFÍA

AGARAP, Abien. Deep Learning using Rectified Linear Units (ReLU). Department of Computer Science Adamson University. arXiv:1803.08375 [cs.NE]. 2018.

AITLAASRI, El Hassan; AKHOUAYRI, Es-Saïd; AGLIZ, Driss; ATMANI, Abderrahman. Seismic Signal Classification using Multi-Layer Perceptron Neural Network. International Journal of Computer Applications, Vol. 79, No. 15. ISSN 0975-8887. 2013.

ALAIMO, Martin; SALIAS, Martin. Proyectos Ágiles con Scrum: Flexibilidad, aprendizaje, innovación y colaboración en contextos complejos. Ediciones Kleer, Buenos Aires, Argentina. ISBN 978-987-45158-1-0. 2013.

AMAZON WEB SERVICES. Evaluating ML Models. Amazon Machine Learning Developer Guide. Disponible en: https://docs.aws.amazon.com/machine-learning/latest/dg/evaluating_models.html. 2018.

ASOCIACIÓN COLOMBIANA DE INGENIERÍA SÍSMICA. Reglamento Colombiano de Construcción Sismo Resistente (NSR-10). 2010

BECK, Kent; BEEDLE, Mike; BENNEKUM, Arie Van; COCKBURN, Alistair; CUNNINGHAM, Ward; FOWLER, Martin; GRENNING, James; HIGHSMITH, Jim; HUNT, Andrew; JEFFRIES, Ron; KERN, Jon; MARICK, Brian; MARTIN, Robert; MELLOR, Steve; SCHWABER, Ken; SUTHERLAND, Jeff; TGOMAS, Dave. Manifesto dor Agile Software Development. 2001.

BENGIO, Yoshua. Practical Recommendations for Gradient-Based Training of Deep Architectures. Cornell University Library. 33 pp. 2012.

BERGSTRA, James; BENGIO, Yoshua. Random Search for Hyper-Parameter Optimization. Journal of Machine Learning Research, Vol. 13. ISSN: 1532-4435. 281-305 pp. 2012.

BEYREUTHER, Moritz; HAMMER, Conny; WASSERMANN, Joachim; OHRNBERGER, Matthias; MEGIES, Tobias. Constructing a Hidden Markov Model based earthquake detector: application to induced seismicity. Geophysical Journal International, Vol. 189, issue 1, 602–610 pp. ISSN 0956-540X. In: https://doi.org/10.1111/j.1365-246X.2012.05361.x. 2012.

BOOCH, Grady y RUMBAUGH, James, JACOBSON, Ivar. El Lenguaje unificado de modelado, Madrid: Addison Wesley Longman, 1999, 432 pág.

BOON, Mei Ying; HENRY, Bruce; SUTTLE, Catherine; DAIN, Stephen. The correlation dimension: A useful objective measure of the transient visual evoked potential? Journal of Vision January, Vol. 8, No. 6. DOI: 10.1167/8.1.6. 2008.

BORMANN, Peter; WIELANDT, Erhard. Chapter 4: Seismic Signals and Noise. In: New Manual of Seismological Observatory Practice (NMSOP-2), IASPEI, GFZ German Research Centre for Geosciences, Potsdam. DOI: 10.2312/GFZ.NMSOP-2_CH4.

CARCEDO AYALA, Fabián. Manual de Ingeniería Geológica. Ministerio de Industria y Energía, Instituto Tecnológico GeoMinero de España. 2005. 626 pp.

CASTELLANO, Pablo. K-fold Cross Validation [Digital image]. Retrieved from Wikimedia Commons website: https://commons.wikimedia.org/wiki/ File:Esquema_castellà.jpg. (2014).

COHN, Mike. User Stories Applied For Agile Software Development. Addison-Wesley and Pearson Education Incorporated. Boston, Massachussets. ISBN: 0-321-20568-5. 2004.

COORNAERT, Michel. Topological Dimension and Dynamical Systems. University of Strasbourg, Strasbourg, France. Springer Editorial. 3-66 pp. ISBN 978-3-319-19793-7. 2015.

CORREAL, Juan Francisco. ¿Cuán vulnerable es Colombia ante un sismo? El Tiempo, Colombia. Disponible en: http://www.eltiempo.com/archivo/documento/CMS-16571309. 2016.

CORMEN, Thomas; LEISERSON, Charles; RIVEST, Ronald; STEIN, Clifford. Introduction to Algorithms. 2nd Edition. Massachusetts Institute of Technology; Boston, Massachusetts, United States of America. 2001. ISBN 0-262-03293-7.

CROSS, Michael. Chapter 9: Dimensions, in Physics 161: Introduction to Chaos. California Institute of Technology. 9 pp. 2000.

DALRYMPLE, Brent. The Age of the Earth. California: Stanford University Press. ISBN 0-8047-1569-6. 492 pp. 1991.

DÁVILA MADRID, Ramón. Notas Introductorias en Sismología. Posgrado en ciencias de la Tierra, Centro de Geociencias, Universidad Autónoma de México. 2011. 36 pp.

DAVISON, A.C.; HINKLEY, D.V. Bootstrap Methods and their Applications. Cambridge University Press, Cambridge. 1997.

DIERSEN, Steve; LEE, En-Jui; SPEARS, Diana; CHEN, Po; WANG, Liqiang. Classification of Seismic Windows Using Artificial Neural Networks. International Conference on Computational Science. ISSN 1877-0509. 1572-1581 pp. 2011.

DIETTERICH, Thomas. Approximate Statistical Tests for Comparing Supervised Classification Learning Algorithms. The MIT Press, Vol. 10, No. 7. ISSN: 0899-7667.1895-1923 pp. 1998.

DUCHI, John; HAZAN, Elad; SINGER, Yoram. Adaptive Subgradient Methods for Online Learning and Stochastic Optimization. Journal of Machine Learning Research Vol. 12. 2121-2159 pp. 2011.

EFRON, Bradley; TIBSHIRANI, Robert. "Improvements on cross-validation: The .632 + Bootstrap Method". Journal of the American Statistical Association. Vol. 92 (438). 548–560 pp. DOI: 10.2307/2965703. JSTOR 2965703. MR 1467848. 1997.

ESTRADA, Luis. Apuntes de Sismología. Universidad Nacional de Tucumán UNT, México. Facultad de Ciencias exactas y tecnología, Departamento de Geodesia y Topografía. 2012. 31 pp.

FERNANDEZ, Benito; PARLOS, A.G.; TSAI, W. K. Nonlinear dynamic system identification using artificial neural networks (ANNs). IJCNN International Joint Conference on Neural Networks. DOI: 10.1109/IJCNN.1990.137706. 1990.

GALPERIN, E. I. The Polarization of Seismic Waves and its Potential for Studying the Rocks Surrounding the Borehole. ISBN: 978-94-009-5195-2. DOI: https://doi.org/10.1007/978-94-009-5195-2_12.

GIUDICEPIETRO, Flora; ESPOSITO, Antonietta; RICCIOLINO, Patrizia. Fast Discrimination of Local Earthquakes Using a Neural Approach. Seismological Research Letters, Vol. 88, No. 4, 1089-1096 pp. In: https://doi.org/10.1785/0220160222. 2017.

GOODFELLOW, Ian; BENGIO, Yoshua; COURVILLE, Aaron. Chapter 6: Deep Feedforward Networks. In: Deep Learning. MIT Press. ISBN: 978-0-26-203561-3. 168-224 pp. 2016.

GRELLIER, Oliver. Parameter tuning : 5 x 2-fold CV statistical test. [Last query: 25 August, 2018]. Available at: https://www.kaggle.com/ogrellier/parameter-tuning-5-x-2-fold-cv-statistical-test. 2018.

HANSSON, Magnus; OLSSON, Christoffer. Feedforward neural networks with ReLU activation functions are linear splines. Lund University, Sweden. ISSN: 1654-6229. 2017.

HARRINGTON, Peter, Machine Learning in Action. Manning publications Co, United States of America. ISBN: 978-16-172-9018-3. 382 pp. 2012.

HAWKINS, W. G. Fourier transform resampling: theory and application. IEEE Transactions on Nuclear Science Vol 44, No. 4. DOI: 10.1109/23.632725. 1543 – 1551 pp. 1997.

HAYKIN, Simon. Neural Networks and Learning Machines. Pearson, Prentice Hall. Hamilton, Ontario, Canada. 3rd ed. ISBN: 978-0-13-147139-9. 938 pp. 2009.

HINTON, Geoffrey; SRIVASTAVA, Nitish; SWERSKY, Kevin. Overview of mini-batch gradient descent. In: Neural Networks for Machine Learning. Computer Science, University of Toronto. 31 pp. 2018.

HOBIGER, Manuel. Polarization of surface waves: characterization, inversion and application to seismic hazard assessment. Earth Sciences. Université de Grenoble. NNT: 2011GRENU005. 309 pp.

HOROWITZ, Ellis; SAHNI, Sartaj. Computing partitions with applications to the knapsack problem. Journal of the Association for Computing Machinery, 21: 277–292, doi:10.1145/321812.321823, MR 0354006. 1974.

HUANG, Guang-Bin. Learning capability and storage capacity of two-hidden-layer feedforward networks. IEEE Transactions on Neural Networks, 14(2), 274–281. doi:10.1109/tnn.2003.809401. 2003.

HUM, Sean Victor. Wave Polarization. ECE422: radio and Microwave Wireless Systems, The Edward S. Roger Sr. Department of Electrical & Computer Engineering, University of Toronto, Canada. 4 pp.

IBS-VON SEHT, M. Journal of Volcanology and Geothermal Research. Detection and identification of seismic signals recorded at Krakatau volcano (Indonesia) using artificial neural networks. Hanover, Germany. 9 pp. 2008.

INTERNET ENGINEERING TASK FORCE IETF. RFC2119: Key words for use in RFCs to Indicate Requirement Levels. Available at: https://tools.ietf.org/html/rfc2119. 1997.

INTERNET ENGINEERING TASK FORCE IETF. RFC8174: Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words. Available at: https://tools.ietf.org/html/rfc8174. 2017.

JADON, Shruti. Introduction to Different Activation Functions for Deep Learning. Available at: https://medium.com/@shrutijadon10104776/survey-on-activation-functions-for-deep-learning-9689331ba092. 2018.

JANOCHA, Katarzyna; CZARNECKI, Wojciech. On Loss Functions for Deep Neural Networks in Classification. Theoretical Foundations of Machine Learning 2017 (TFML 2017). arXiv:1702.05659v1 [cs.LG]. 2017.

KAUR, Komalpreet; WADHAWA, Manish; PARK, E.K. Detection and Identification of Seismic P-Waves using Artificial Neural Networks. The 2013 International Joint Conference on Neural Networks, Dallas, Texas, United States of America. DOI: 10.1109/IJCNN.2013.6707117. 2013.

KINGMA, Diederik; LEI BA, Jimmy. Adam: a method for stochastic optimization. International Conference on Learning Representations ICLR. Cornell University Library. 15 pp. 2015.

KRAMER L. Steven. Geotechnical Earthquake Engineering. Prentice Hall, USA, 1996.

KRIESEL, David. A Brief Introduction to Neural Networks. In: https://doi.org/10.1016/0893-6080(94)90051-5. 244 pp. 2005.

KÜPERKOCH, L.; MEIER, Th.; DIEHL, T. Chapter 16: Automated event and phase detection. In: New Manual of Seismological Observatory Practice (NMSOP-2), IASPEI, GFZ German Research Centre for Geosciences, Potsdam. DOI: 10.2312/GFZ.NMSOP-2_CH4.

KUROSE, James F. Ross, Keith W. Computer Networking: A top-down approach featuring the Internet. 4th edition. Addison Wesley, 2008.

LAROCHELLE, Hugo, et al. Exploring Strategies for Training Deep Neural Networks. The Journal of Machine Learning Research. Volume 10, 12/1/2009.

LEUNG, Henry. Chaotic Signal Processing. University of Calgary, Calgary, Alberta, Canada. 152 p. ISBN 978-1-61197-325-9. 2014.

LIANG, Zhiqiang; WEI, Jianming; ZHAO, Junyu; LIU, Haitao; LI, Baoqing; SHEN, Jie; ZHENG, Chunlei. The Statistical Meaning of Kurtosis and Its New Application to Identification of Persons Based on Seismic Signals. 8(8): 5106–5119. DOI: 10.3390/s8085106. 2008.

MCGUIRE, Michael. The advantages and disadvantages of the Butterworth filter, the Chebyshev filter, and the Bessel filter. Computer Engineering & Digital Signal Processing, University of Victoria. 2017.

MANDELBROT, B. How long is the coast of Britain Statistical self-similarity and fractional dimension. Science Review, No. 156, 636–638 pp. 1967.

MATICH, Damián. Redes Neuronales: Conceptos Básicos y Aplicaciones. Universidad Tecnológica Nacional, Rosario, Argentina. 55 pp. 2001.

NATIONAL INSTRUMENTS. IIR Filters and FIR Filters. Part Number: 370858N-01. Available at: http://zone.ni.com/reference/en-XX/help/370858N-01/genmaths/genmaths/calc_filterfir_iir/. 2017.

NILSSON; N. Introduction to Machine Learning; Department of Computer Science, Standford University; Standford, CA 94305; 1st Edition; 188 pp. 2005.

PANDAS: DATA STRUCTURES FOR STATISTICAL COMPUTING IN PYTHON, McKinney. read_csv in: pandas v0.24.4 Reference Guide. Pandas Documentation. Available at: https://pandas.pydata.org/pandas-docs/stable/generated/pandas.read_csv.html. 2018.

PEARSON, K. Das Fehlergesetz und seine Verallgemeinerungen durch Fechner und Pearson. A Rejoinder. Biometrika. 1905;4:169–212.

PERELBERG, Azik; HORNBOSTEL, Scott. Applications of seismic polarization analysis. Geophysics Journal, Vol. 59, No. 1. ISBN: 0926-9851. pp. 119-130. 1994.

PRESTON, Tom. Semantic Versioning 2.0.0. [Last Update: August 2018]. Available at: https://semver.org/.

REYNEN, Andrew. Supervised machine learning on a network scale: application to seismic event classification and detection. Department of Earth and Environmental Sciences, Faculty of Sciences, University of Ottawa. 65 pp. 2017.

RICHARD, A; GROENEVELD, G. M. Measuring skewness and kurtosis. The Statistician. 1984;33:391–399.

RIGGELSEN, Carsten; Ohrnberger, Matthias. A Machine Learning Approach for Improving the Detection Capabilities at 3C Seismic Stations. Pure and Applied Geophysics, Vol. 171, issues 3-5, 395-411 pp. ISSN 0033-4553. In: https://doi.org/10.1007/s00024-012-0592-3. 2012.

ROBLES, Gregorio; AMOR, Juan José; GONZÁLEZ-BARAHONA, Jesús; HERRAIZ, Israel. From Pigs to Stripes: A Travel trhough Debian. Universidad Rey Juan Carlo, Móstoles, Madrid, España. 2005.

RUANO, A.E.; MADUREIRA, G.; BARROS, O.; KHOSRAVANI, H.R.; RUANO, M.G.; FERREIRA, P.M. Seismic detection using support vector machines. Elsevier Neurocomputing, Vol. 135, No. 5, 273-283 pp. ISSN 0925-2312. In: https://doi.org/10.1016/j.neucom.2013.12.020. 2014.

RUDER, Sebastian. An overview of gradient descent optimization algorithms. Cornell University Library. 14 pp. 2017.

RUIZ, Carlos Alberto; BASUALDO, Marta Susana. Redes Neuronales: conceptos básicos y aplicaciones. Universidad Tecnológica Nacional, Facultad Regional Rosario, Argentina. 55 pp. 2001.

SÁNCHEZ, Francisco. Los Terremotos y sus Causas. Instituto Andaluz de Geofísica y Prevención de Desastres Sísmicos. España. 2007. 24 pp.

SARRIA MOLINA, Alberto. Ingeniería sísmica. Ediciones Uniandes, 2a. Ed, Bogotá, D.C., 1995.

SATO, Michikazu. Some remarks on the mean, median, mode and skewness. Australian Journal of Statistics 39(2), 219-224. DIO: https://doi.org/10.1111/j.1467-842X.1997.tb00537.x. 1997.

SCHNEIDER, Thomas. Information theory primer with an appendix on logarithms, National Cancer Institute. DOI: http://dx.doi.org/10.13140/2.1.2607.2000. 2007.

SCIKIT-LEARN PROJECT .Underfitting vs. Overfitting. Scikit-learn 0.19.2 online documentation Available at: http://scikit-learn.org/stable/auto_examples/model_selection/plot_underfitting_overfitting.html. 2017.

SCIPY TEAM. Skewness in: SciPy v1.1.0 Reference Guide. SciPy Documentation. Available at: https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.skew.html. 2018.

SERVICIO GEOLÓGICO COLOMBIANO. Evaluación y Monitoreo de Actividad Sísmica. Disponible en: https://www2.sgc.gov.co/ProgramasDeInvestigacion/geoamenazas/Paginas/actividad-sismica.aspx#. 2017.

SERVICIO GEOLÓGICO COLOMBIANO. Redes de estaciones, Instrumentación. Disponible en: seisan.sgc.gov.co/RSNC/index.php/red-de-estaciones/instrumentación.

SERVICIO GEOLÓGICO MEXICANO. Causas, Características e Impactos de los Sismos. Secretaría de Economía de México. Disponible en: http://portalweb.sgm.gob.mx/museo/riesgos/sismos. 2013.

SHEARER; Peter. Introduction to Seismology. Cambridge University Press. ISBN 978-0-521-88210-1. 2009.

SISTEMA DE INFORMACIÓN EN GESTIÓN DEL RIESGO DE DESASTRES. Red Sismológica Nacional de Colombia. [Última consulta: 25 de agosto de 2018]. Disponible en: http://www.redriesgos.gov.co/red-sismologica-nacional/. 2018.

SNOWDEN, David; BOONE, Mary. A Leader's Framework for Decision Making, Harvard Business Review. 2007.

TARBUCK, Edward; LUTGENS, Frederick. Ciencias de la Tierra, una introducción a la Geología Física. Universidad Autónoma de Madrid. Pearson, Prentice Hall, 8va Ed. ISBN: 978-84-832-2690-2. 736 pp. 2005.

THOMA, Martin. Analysis and Optimization of Convolutional Neural Network Architectures. Department of Computer Science, Institute for Anthropomatics and

FZI Research Center for Information Technology. arXiv:1707.09725v1 [cs.CV]. 2017.

VALLEJOS, J.A.; MCKINNON, S.D. Logistic regression and neural network classification of seismic records. International Journal of Rock Mechanics and Mining Sciences, Vol. 62, 86-95 pp. In: https://doi.org/10.1016/j.ijrmms.2013.04.005. ISSN 1365-1609. 2013.

WALBER, Matt. Precision and recall [Digital image]. Retrieved from Wikimedia Commons website: https://commons.wikimedia.org/wiki/File:Precisionrecall.svg. (2014).

WATKINS, Joseph. Moments and Generating Function. Department of Mathematics, University of Arizona. 2009.

ZARIFI, Zoya; HAVSKOV, Jens: HANYGA, Andrezej. An insight into the Bucaramanga nest. Tectonophysics. 1-13 pp. 2007.

ZEILER, Matthew. ADADELTA: AN ADAPTIVE LEARNING RATE METHOD. Cornell University Library. 6 pp. 2012.

ZHANG, Yuli; WU, Huaiyu; CHENG, Lei. Some new deformation formulas about variance and covariance. Proceedings of 4th International Conference on Modelling, Identification and Control (June 2012 - ICMIC2012). pp. 987–992.

ZHENG, Alice. Evaluating Machine Learning Models; A Begginer's Guide to Key Concepts and Pitfalls, O'Reilly Media, Inc. United States of America. 1st ed. ISBN: 978-1-491-93246-9. 48 pp. 2015.

ZWILLINGER, D.; KOKOSKA, S. CRC Standard Probability and Statistics Tables and Formulae. Chapman & Hall: New York. 2000.

ANEXO A – ACTA DE REQUERIMIENTOS

1. PREFACIO

Este documento describe los requerimientos del SISTEMA PARA LA DETECCIÓN DE MOVIMIENTOS SÍSMICOS USANDO REDES NEURONALES ARTIFICIALES a implementar, cuyo objetivo principal es la detección de movimientos sísmicos con los datos registrados por los sismogramas de la Red Sismológica Nacional de Colombia, mediante el uso de redes neuronales artificiales y clasificación de segmentos de señales sísmicas.

2. ALCANCE

Este documento de requerimientos de sistema es la base del desarrollo del sistema para la detección de movimientos sísmicos usando redes neuronales artificiales.

3. ENTORNO

El proceso de traslación de las capas continentales que la Tierra sufría en la era primitiva se mantiene actualmente, produciendo en la corteza terrestre movimientos verticales y horizontales que en promedio representan un desplazamiento de 100 micrómetros anuales. Esto se debe a que dicha corteza se encuentra compuesta por placas (trozos de litosfera) que conforman los fondos marinos y las superficies continentales. El movimiento de estas placas es el producto de las presiones internas de las corrientes del manto terrestre y las diferencias de densidad y temperatura, lo que causa roces y choques con placas contiguas, produciendo roturas, elevaciones, plegamientos montañosos o hundimientos de una placa bajo la otra (subducción). Estos desplazamientos internos se perciben como movimientos superficiales en las capas de la litosfera que son imperceptibles en la

mayoría de los casos, pero que, en ocasiones, producto de la energía acumulada, resultan en vibraciones perceptibles que se transforman en sismos de baja y de gran magnitud.

La tasa de más de 300.000 sismos anuales registrados a lo largo del territorio del planeta, en otras palabras, la ocurrencia de un sismo cada 2 minutos, indica que las capas superficiales están sometidas a movimientos sísmicos permanentemente y son vulnerables ante la ocurrencia de estos eventos. El 0,025% de esos sismos, es decir, aproximadamente 75 de ellos, presentan una magnitud de energía sísmica liberada elevada, causando graves daños en la infraestructura, alteraciones geológicas, ecológicas y, sobre todo, pérdidas humanas.

Dadas las características sismológicas de Colombia y a la necesidad de contar con sistemas de monitoreo y alerta temprana, la UPB seccional Bucaramanga, en conjunto con la Universidad Francisco de Paula Santander de Cúcuta, han definido el proyecto de investigación: “Desarrollo de un sistema de monitoreo y alerta de movimientos sísmicos – Tellurico”, el cual ha sido planteado de manera modular, distribuido en el desarrollo de 4 componentes: detección del evento sísmico, localización del evento sísmico, cálculo y visualización del mapa de intensidades y alerta al usuario. Para que exista una alerta temprana es necesaria la detección en línea del evento sísmico cuando aún no se ha presenciado el impacto considerable del movimiento en la superficie. Ante esta situación, se plantea el desarrollo de un prototipo que consiste en un sistema de detección de movimientos sísmicos, haciendo uso de planteamientos estadísticos, análisis de señales en línea y redes neuronales artificiales.

4. REQUERIMIENTOS SOFTWARE

A continuación, se citan los requerimientos para el desarrollo del sistema:

4.1. REQUERIMIENTOS FUNCIONALES

FSR1 El sistema permitirá:

A. La descarga y almacenamiento del histórico sísmico en el periodo comprendido desde el año 2010 hasta el año 2017.

B. La identificación de las características propias de las señales sísmicas a partir de la información del histórico sísmico: tiempo de inicio, tiempo de fin, duración, magnitud, profundidad, tiempo de llegada de onda P, tiempo de llegada de onda S, periodo, frecuencias típicas[116] y componentes espaciales, entre otros.

C. El preprocesamiento de los datos sísmicos, ejecutando procesos de: filtrado de señal, normalización por media aritmética, re-muestreo y sincronización de las componentes.

D. La selección de las señales de entrada para el proceso de clasificación por redes neuronales y la extracción de ventanas de onda P y ruido.

E. El cálculo y extracción de los atributos (*features*) de entrada para le Red Neuronal Artificial (ANN), de las señales de entrada seleccionadas, para las estaciones definidas[117].

[116] Las frecuencias típicas son identificadas por medio de una revisión del estado del arte y/o un análisis del comportamiento de las señales sísmicas en frecuencia.

[117] Las estaciones son definidas según un análisis de la ocurrencia de eventos sísmicos de la región que contempla la magnitud, profundidad y componentes de cada observación en el histórico. De igual forma, el periodo de medición puede variar, ser acotado o ampliado según el proceso lo requiera.

F. Clasificar por medio de la ANN las observaciones sísmicas en dos categorías: datos correspondientes a sismos y datos correspondientes a ruido, midiendo el desempeño de esta tarea por medio de métricas de evaluación.

G. La verificación del desempeño de la clasificación mediante el uso de métricas y procedimientos especializados para esta tarea[118].

4.2. REQUERIMIENTOS NO FUNCIONALES

NFSR1 Deberá existir un histórico de datos sísmicos en el periodo comprendido desde el año 2010 hasta el año 2017, como resultado de un procesamiento y almacenamiento de la Red Sismológica Nacional de Colombia.

NFSR2 La Red Neuronal Artificial (ANN) estará compuesta por:

- Una cantidad de nodos de entrada, intermedios es variable según el desempeño que marquen las métricas escogidas. La topología será cambiante y estará definida por la cantidad de atributos extraídos de las observaciones y en función de una ecuación simple de decremento de las capas.
- Un conjunto de observaciones que serán divididas en: conjunto de datos de entrenamiento, conjunto de datos de validación y conjunto de datos de prueba. Con los tres conjuntos será evaluado el desempeño de la ANN y la toma de decisiones se hará sobre los resultados aplicados al conjunto de pruebas.

[118] Las métricas de medición del desempeño son definidas en las fases posteriores. Se contemplan: Accuracy, Precisión, Recall, Especificidad, Sensibilidad y F1.

NFSR3 El sistema permitirá la integración entre los módulos de almacenamiento, selección y lectura de la entrada, preprocesamiento, extracción atributos y clasificación

4.3. REQUERIMIENTOS DE TESTING

ST1 Verificar la descarga y almacenamiento exitoso de los datos sísmicos en el periodo estipulado mediante la revisión de los archivos.

ST2 Contrastar las características sísmicas almacenadas en los archivos de la RSNC con las obtenidas por los procesos de identificación del sistema: tiempo de inicio, tiempo de fin, duración, magnitud, profundidad, tiempo de llegada de onda P, tiempo de llegada de onda S, periodo, frecuencias típicas y componentes espaciales, entre otros.

ST3 Verificar que en los registros de señal exista:

- Una media aritmética de 0.0.
- La misma cantidad de muestras para todas las componentes de una misma observación.
- Un contenido frecuencial filtrado a las frecuencias típicas determinadas.
- El tiempo de inicio y fin de las señales sea equivalente entre sus distintas componentes espaciales.

ST4 Verificar la aleatoriedad de la clasificación manual de las señales correspondientes a ruido y el tamaño fijo de las ventanas para todas las observaciones y sus componentes espaciales.

ST5 Verificar la complejidad temporal del proceso de cálculo de los atributos.

ST6 Verificar la clasificación mediante métricas de desempeño, utilizando el conjunto de datos de entrenamiento, validación y prueba.

ST7 Verificar la clasificación y escogencia de la mejor topología mediante pruebas de Monte Carlo.

4.4. MATRIZ REQ. FUNCIONALES VS. REQ. DE TESTING

En la siguiente tabla se muestra la relación existente entre los requerimientos funcionales del sistema y los requerimientos de prueba (testing) definidos:

Tabla 1. Relación existente entre los requerimientos funcionales del sistema y los requerimientos de prueba definidos.

	Requerimiento de Testing						
Requerimiento funcional	ST1	ST2	ST3	ST4	ST5	ST6	ST7
FSR1A	X						
FSR1B		X					
FSR1C			X				
FSR1D				X			
FSR1E					X		
FSR1F						X	
FSR1G						X	X

5. RESTRICCIONES

5.1. RESTRICCIONES SOFTWARE

Las restricciones de *Software* son:

1. El clasificador hará su proceso de clasificación fuera de línea o en *batch*, con los datos almacenados en el histórico, sin ningún solapamiento de ventanas ni procedimientos adicionales que impliquen la clasificación en línea del evento sísmico.

2. La clasificación se hará binaria, identificando si se trata de un evento sísmico o de ruido, obviando parámetros multiclase o cualquier otra característica que implique la clasificación en más de dos clases.
3. El sistema estará operando desde el Centro de Cómputo Avanzado (CCA) de la UPB seccional Bucaramanga, con una dirección IP privada y sin acceso al público.

5.2. RESTRICCIONES DE HARDWARE

Las restricciones de hardware son:

1. Se hará uso de los servidores ubicados en el Centro de Cómputo Avanzado (CCA) de la UPB seccional Bucaramanga, con las prestaciones computacionales respectivas, tanto para el procesamiento como para el almacenamiento de la información.
2. El almacenamiento inicial del histórico sísmico se hará en discos duros externos o internos brindados por la UPB y accedidos mediante protocolos SATA y USB en equipos de cómputo convencionales.

ANEXO B – PROFUNDIZACIÓN A LA METODOLOGÍA DE DESARROLLO

Para el cumplimiento de los objetivos estipulados y la generación de los prototipos descritos en la sección de metodología, se planteó la asociación expuesta en la Figura B1, entre objetivos, requerimientos e historias de usuario.

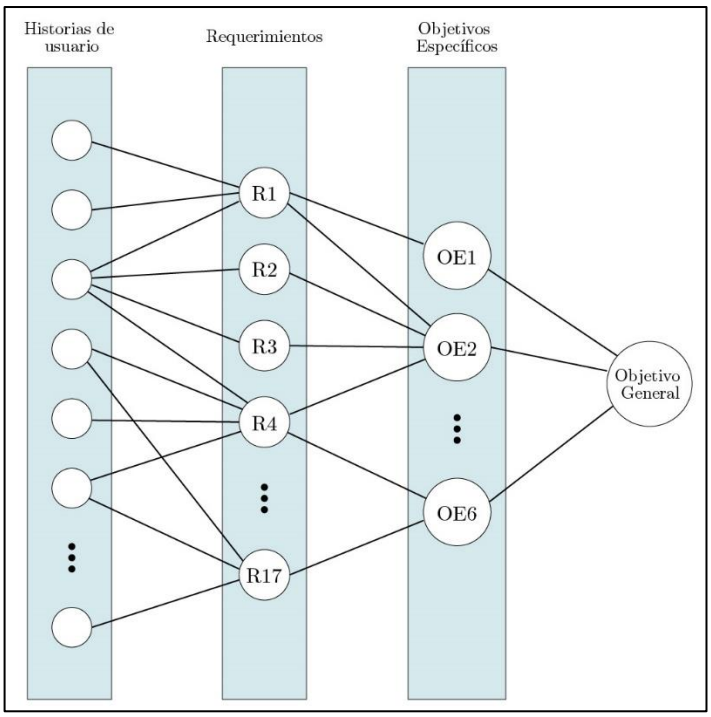


Figura B1. Diagrama de relación entre objetivos, requerimientos e historias de usuario.

En la figura puede apreciarse que los requerimientos ampliaron los objetivos añadiendo un análisis técnico de las necesidades para el cumplimiento de los mismos. Las historias de usuario transformaron los requerimientos en tareas más concretas que fueron seccionadas en PBIs (*Product Backlog Items*). Mediante el cumplimiento de las historias de usuario, se pudieron cumplir los requerimientos, los objetivos específicos, cumpliendo así con el objetivo general.

Las historias de usuario comunicaron a los *sprints* y los prototipos con los requerimientos y objetivos. En los *sprints* se consideraron un conjunto de PBIs para realizar dentro de un plazo de 4 semanas (1 mes), tratando de maximizar la prioridad de las tareas realizadas. Los prototipos fueron el resultado de los procesos llevados a cabo en uno o más *sprints*, marcando la evolución de los prototipos con el fin de cumplir con lo estipulado en las historias de usuario y requerimientos, apuntando al cumplimiento de los objetivos.

Los prototipos se conformaron a partir de cuatro tipos de historias de usuario (Figura B3):

- Historias de usuario de inicio: marcaron el punto de partida del proceso investigativo inicial que sirvió de soporte para la contextualización del estado del arte y la terminología de desarrollo.
- Historias de usuario evolutivas: fueron aquellas que mantuvieron su finalidad a lo largo del proceso iterativo de prototipado, mejorando los resultados asociados entre cada prototipo.
- Historias de usuario invariantes: fueron agrupadas en tareas que se realizaron una vez por prototipo y/o sprint.
- Historias de usuario de fin: permitieron culminar con el proceso de prototipado, dando paso a la documentación y presentación de resultados.

Esta concepción del uso de las historias de usuario fue el puente de entre la metodología ágil Scrum y la metodología de Prototipado.

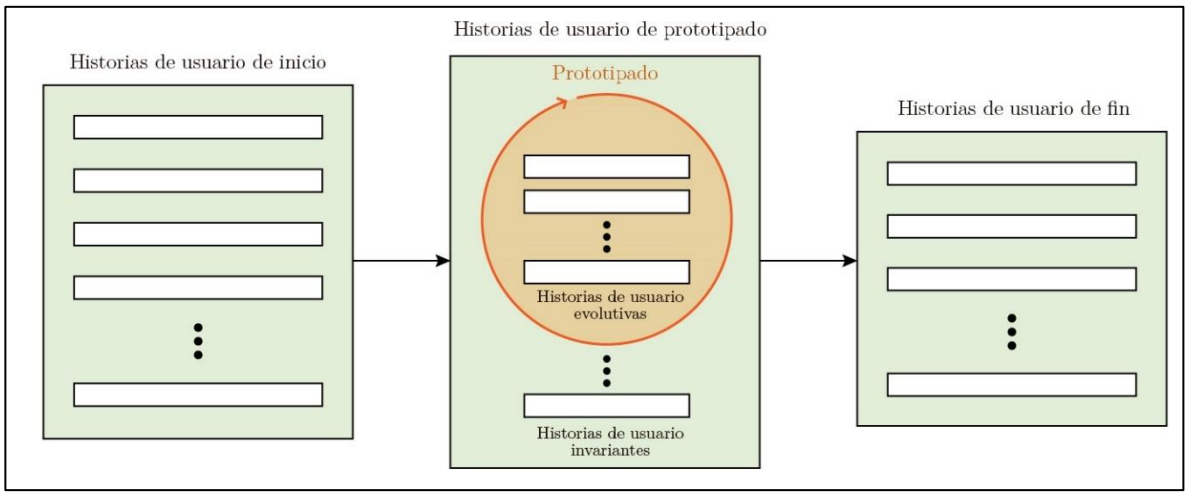


Figura B2. Relación de las historias de usuario en la metodología de Prototipado y el desarrollo del sistema.

De acuerdo con la relación entre las historias de usuario y los prototipos planteada, la conformación de *sprints* usada para el desarrollo del proyecto, se puede detallar en la Figura B3.

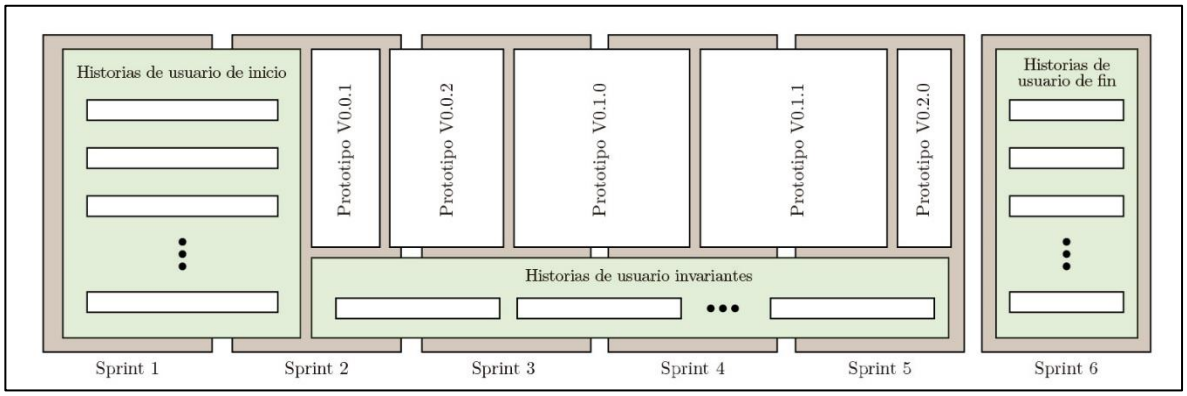


Figura B3. Desarrollo de prototipos dentro de *sprints*.

El sistema fue desarrollado en 6 *sprints*. Se puede apreciar en la Figura B3, que cada *sprint* tiene características distintas: el *sprint* 1 y el 6 no tienen asociado el desarrollo de ningún prototipo, debido a que fueron ejecutadas las historias de usuario de inicio que dan paso al Prototipado. Los *sprints* 2, 3, 4 y 5 contienen el proceso de Prototipado expuesto en el que fueron desarrollados solapadamente los

prototipos. Sin embargo, estos *sprints* contienen historias de usuario que no están asociadas al Prototipado, las historias de usuario invariantes. Por otra parte, se puede observar que algunos prototipos se desarrollaron en un solo *sprint*, mientras que otros inician en uno y terminan en otro.

En la Tabla B1 se puede apreciar la relación entre los objetivos, requerimientos e historias de usuario.

Tabla B1. Relación entre objetivos, requerimientos e historias de usuario.

Objetivo específico	Requerimientos	Historias de usuario
1	FSR1A, FSR1B, ST1, ST2	US1, US2, US3, US5, US6, US7, US9
2	FSR1B,FSR1D, NFSR1, ST2, ST4	US10, US11, US12, US13, US14
3	FSR1B, FSR1C, FSR1E, ST2, ST3, ST5	US16, US17, US18, US19, US20, US21, US22, US24, US25, US26, US27, US29
4	NFSR2, NFSR4	US8, US15, US23, US30,
5	FSR1F, FSR1G, ST6, ST7	US4, US28 US31, US32, US33, US34, US35, US36, US38
6	FSR1, NFSR4, ST1, ST2, ST3, ST4, ST5, ST6, ST7	US37, US39

Como se puede apreciar, los objetivos específicos del proyecto fueron vinculados con ciertos requerimientos funcionales, no funcionales y de *testing*. A su vez, fue posible asociar las historias de usuario a los requerimientos y, por ende, a los objetivos específicos. Las historias de usuario se vincularon a uno o más *sprints*. Su cumplimiento varía en el orden, de acuerdo con la relevancia que se determina para cada una de ellas dentro de cada *sprint*.

En las Tablas B2 y B3 se pueden observar las historias de usuario que fueron consideradas en cada *sprint* y la estimación del tiempo requerido para realizarlas (en horas). Además, se detallan las historias de usuario necesarias para producir los prototipos planteados.

Tabla B2. Relación entre historias de usuario, *sprints* y prototipos.

Código	Descripción	Estimación (h)
US1	Indagar sobre las técnicas de aprendizaje automático que podrían ser usadas en el contexto sismológico.	32
US2	Indagar sobre técnicas de preprocesamiento de señales sísmicas y librerías en python para tal utilidad.	28
US3	Identificar librerías de *machine learning* y *deep learning* en python.	8
US4	Estipular criterios para la selección de los hiperparámetros y la arquitectura de la red neuronal.	20
US5	Analizar el formato propio de los archivos *Sfile* y *Waveform*.	56
US6	Desarrollar un *snippet* para la descarga automática del histórico sísmico entre 2010 y 2017.	20
US7	Identificar las características y atributos de las señales, presentes en el histórico sísmico almacenado.	24
US8	Diseñar la arquitectura general del sistema	24
US9	Desarrollar clases de lectura de *Sfiles* y *Waveforms*.	48
US10	Remover archivos y señales que contengan errores de formato.	12
US11	Generar gráficas que permitan la caracterización de los eventos de entrada registrados en los *Sfiles*.	52
US12	Seleccionar las estaciones sismológicas para la disminución del costo computacional, mediante el análisis de las gráficas generadas.	12
US13	Remover archivos en donde no estén presentes las estaciones seleccionadas.	8
US14	Documentar el análisis de las estaciones.	56
US15	Diseñar la arquitectura del módulo de preprocesamiento.	28
US16	Filtrar las señales seleccionadas para reducir las componentes frecuenciales indeseables.	32
US17	Normalizar las señales seleccionadas para homogeneizar valores que se encuentran en diferentes escalas.	16
US18	Re-muestrear las señales seleccionadas para homogeneizar las tasas de muestreo.	16
US19	Anotar la onda P en cada traza, según lo estipulado en los *Sfiles*.	16
US20	Sincronizar las señales con base en la anotación de la onda P.	20
US21	Seleccionar las ventanas de detección de onda P y ruido.	20
US22	Documentar del módulo de preprocesamiento.	56
US23	Diseñar la arquitectura del módulo de extracción de atributos.	28
US24	Extraer atributos de carácter estadístico en las señales seleccionadas.	32
US25	Extraer atributos de carácter frecuencial en las señales seleccionadas.	32
US26	Extraer atributos de carácter no lineal en las señales seleccionadas.	56
US27	Integrar los módulos de preprocesamiento y extracción de atributos.	56

Código	Descripción	Estimación (h)
US28	Generar matrices de atributos sísmicos que sirvan como un *dataset* manipulable por los métodos de la ANN.	56
US29	Documentar del módulo de extracción de atributos.	56
US30	Diseñar la arquitectura del módulo de clasificación.	28
US31	Desarrollar la clase que implemente las opciones de entrenamiento y validación de la ANN.	56
US32	Ejecutar entrenamientos mediante validación cruzada por *k-fold* y optimización de hiperparámetros.	56
US35	Generar *datasets* para diversos números de estaciones y ventanas de onda P con base en la matriz de atributos sísmicos.	28
US36	Validar el desempeño del clasificador a través de validación cruzada por Monte Carlo.	32
US37	Estimar la complejidad temporal del proceso de extracción de atributos y de clasificación.	28
US38	Documentar del proceso de clasificación.	56
US39	Elaboración del informe final.	140

Tabla B3. Relación entre *sprints* y prototipos.

	Estimación por *Sprint* (h)	Prototipos asociados
Sprint 1	212	–
Sprint 2	124	Prototipo V0.0.1 Prototipo V0.0.2
Sprint 3	84	Prototipo V0.0.2 Prototipo V0.1.0
Sprint 4	84	Prototipo V0.1.0 Prototipo V0.1.1
Sprint 5	140	Prototipo V0.1.1 Prototipo V0.2.0
Sprint 6	224	–
Prototipado	476	–
Total	1.344	5

Cada una de las historias de usuario planteadas en la Tabla B3 se divide en múltiples tareas concretas (PBIs) con el objetivo de desarrollar aspectos específicos necesarios para el cumplimiento de la historia de usuario. Por ejemplo, la historia

de usuario US1 implicó las siguientes sub-tareas: revisar la estructura y contenido HTML de los recursos web de la RSNC para la descarga de archivos sísmicos, codificar el *snippet* usando *JavaScript* en la aplicación cliente y realizar la descarga de los archivos por medio de *Bash Scripting*. Cada una de estas sub-tareas fue estimada y representada en forma de *Issue* en ZenHub. Así, la estimación asociada a cada historia de usuario en la Tabla B2 es el total de la suma de las estimaciones de sus sub-tareas más un margen dispuesto para imprevistos.

Al relacionar los prototipos propuestos con las historias de usuario y su estimación, se puede decir, de forma aproximada, cuánto tiempo tomó realizar cada prototipo. De la misma forma, con el desarrollo de cada prototipo y sus historias de usuario mediante las relaciones estipuladas en la Tabla B1, se pudo apreciar el cumplimiento de los requerimientos y los objetivos específicos.

Seguimiento del cronograma y las actividades

Usando las funcionalidades de ZenHub y Github es posible relacionar los conceptos de Scrum de forma que:

- Los *sprints* se convirtieron en *Milestones* de ZenHub: los *sprints* se reflejan en ZenHub/GitHub con *Milestones*. Se establece una fecha de inicio y finalización (normalmente entre una a cuatro semanas) y se agregan una descripción y un título. El proceso de creación de un *Milestone* se describe en la Figura B4.
- Las historias de usuario se convirtieron en *Issues* de GitHub: éstos representaron actividades por hacer, completar o analizar. Según esto, las historias de usuario características de *Scrum* asociaron a *Issues* específicos, configurando a su vez las plantillas de emisión de GitHub para añadir detalles o criterios de aceptación. Las tareas específicas de cada *issue* fueron detalladas en modo de *checklist* y el progreso se midió a medida que estos ítems fueron completados. En el

ejemplo de la Figura B5, los *Issues* se configuraron para pertenecer *Milestones* (*Sprint*), vincular etiquetas, estimar su duración, entre otras opciones.

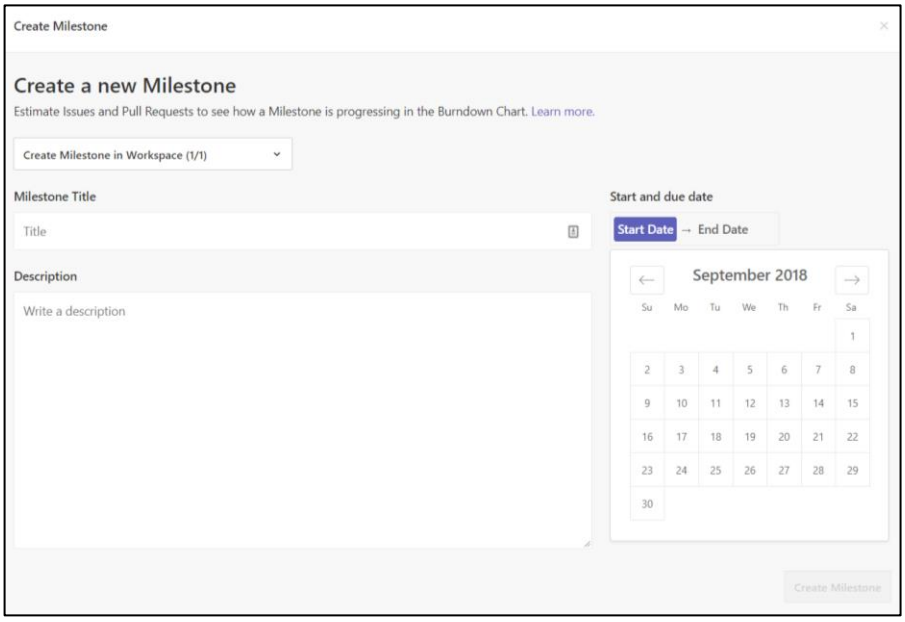


Figura B4. Creación de *Milestones* en ZenHub.

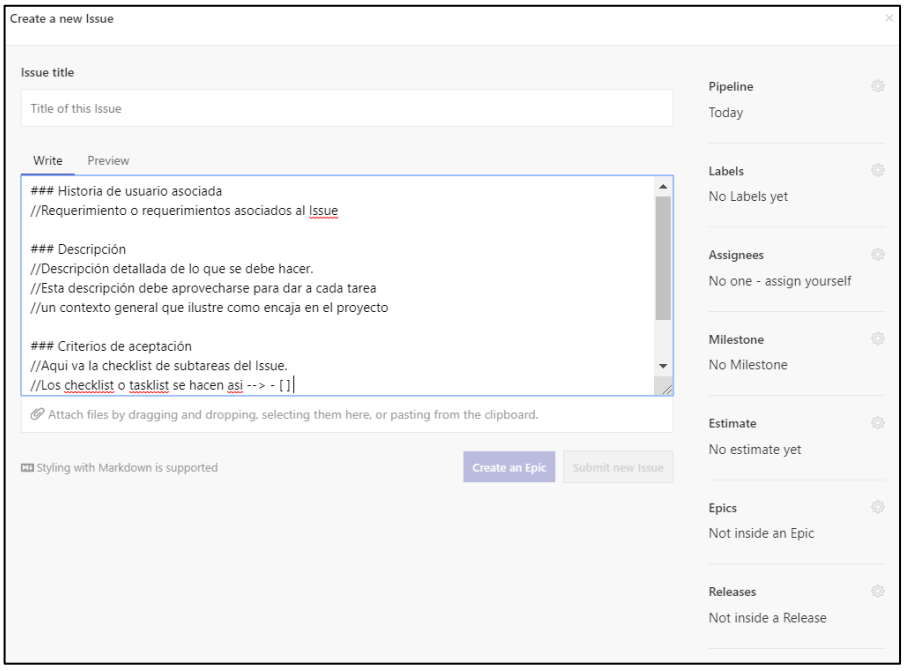
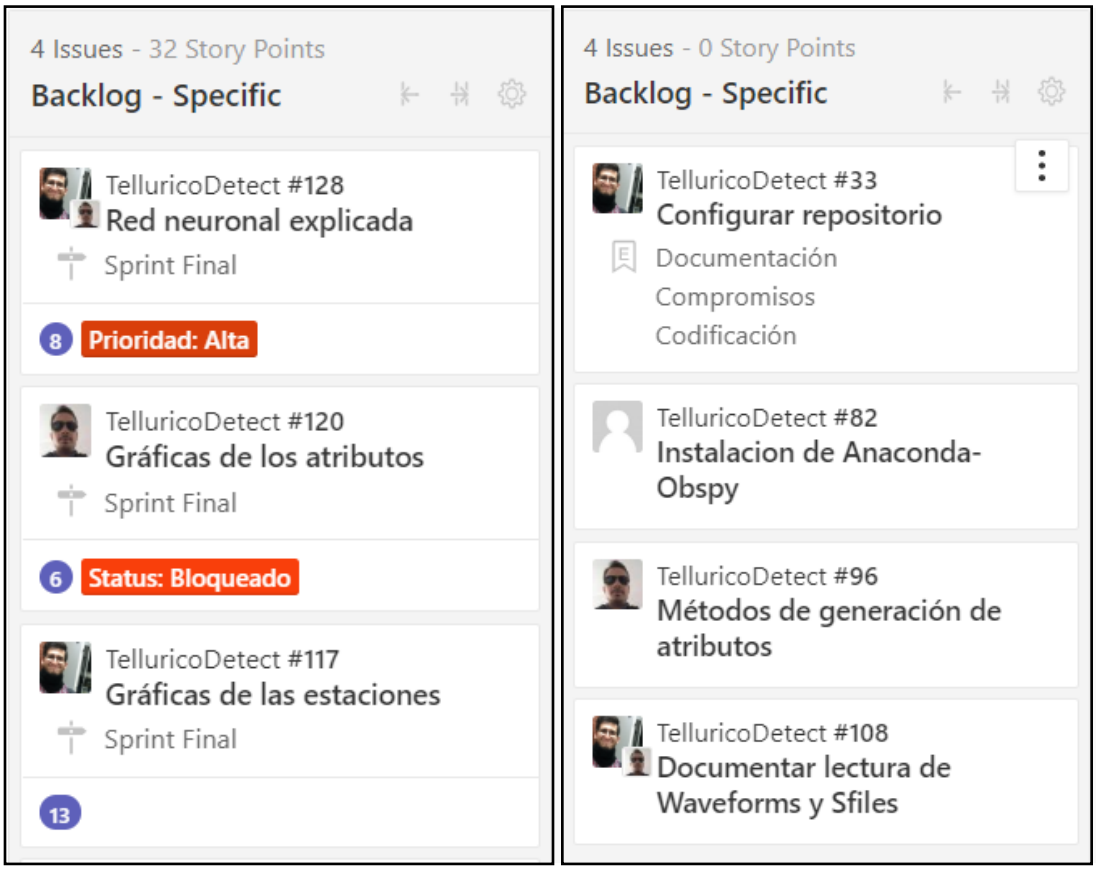


Figura B5. Ejemplo de creación de *Issues* en ZenHub.

- Los *Product Backlogs* se transformaron en paneles de Zenhub: el *backlog* incluye todos los ítems o *issues* que se realizaron para finalizar el producto. En ZenHub es posible tener una estructura en forma de pila, mediante un panel, en donde se pueden organizar los *issues* en orden de prioridad. En la Figura B6(b) se puede evidenciar un ejemplo de *product backlog*, en donde los *issues* están ordenados por prioridad, pero no están asociados a *milestones* ni tienen estimaciones.



(a) (b)

Figura B6. Paneles de: (a) *Sprint Backlog* y (b) *Product Backlog*.

Existen múltiples filtros útiles para mostrar la información relacionada con *issues* etiquetados, *milestones*, personas asignadas, entre otras. De esta manera se pueden observar las tareas específicas acordadas para cada *sprint*, detallando su estimación y las personas asignadas.

Para aportar al proceso general de organización, fue posible crear múltiples paneles como los visualizados en la Figura B6(a). En dichos paneles se pudo establecer un ciclo de vida para los *Issues*, en donde, por ejemplo, fue posible usar un panel de *New Issues* para ingresar tareas sin clasificaciones generales, *Icebox* para tareas cuya solucón es incierta, *Review* Q/A para tareas en revisión y *Closed* para tareas finalizadas. Asimismo, se pude disponer de paneles para registrar las conclusiones de las reuniones propias de *Scrum*, para recopilar información relacionada al proceso de desarrollo y para documentar los productos del trabajo realizado.

ANEXO C – LISTADO DE ESTACIONES SISMOLÓGICAS DE LA RSNC

En las Tablas C1 y C 2 se relacionan las 85 estaciones sismológicas nacionales de la Red Sismológica Nacional de Colombia (RSNC), ordenadas de forma ascendente por su identificador.

Tabla C1. Estaciones sismológicas nacionales de la RSNC por nombre y departamento.

	Identificador	Nombre	Departamento	Municipio
1	APAC	Apartadó	Antioquia	Apartadó
2	ARGC	Ariguaní	Magdalena	Ariguaní
3	BAR2	Barichara	Santander	Barichara
4	BBAC	Balboa	Cauca	Balboa
5	BET	Betania	Huila	Yaguara
6	BRR	Barranca	Santander	Barrancabermeja
7	CAP2	Capurganá	Choco	Acandí
8	CAQC	Cáqueza	Cundinamarca	Cáqueza
9	CBOC	Ciudad Bolívar	Antioquia	Ciudad Bolívar
10	CHI	Chingaza	Cundinamarca	La calera
11	CRJC	Cerrejón	La Guajira	Barrancas
12	CRU	Cruz	Nariño	La cruz
13	CS01	Arenas blancas	Cesar	Bosconia
14	CS02	El desastre	Cesar	Bosconia
15	CS03	San Sebastián	Cesar	Bosconia
16	CS04	Loma linda	Cesar	Bosconia
17	CS05	Bosconia	Cesar	Bosconia
18	CS06	El copey	Cesar	El copey
19	CS07	Mariangola Villa Rita	Cesar	Bosconia
20	CS08	Seminario juan pablo ii	Cesar	Bosconia
21	CUM	Cumbal	Nariño	Cumbal
22	CVER	Cruz verde	Cundinamarca	Bogotá D.C.
23	DBB	Dabeiba	Antioquia	Dabeiba
24	FLO2	Florencia	Caquetá	Florencia
25	FOM	Fómeque	Cundinamarca	Fómeque
26	GARC	Garzón	Huila	Garzón
27	GJ01	Cabo de la vela	La Guajira	Uribia
28	GJ02	Uribia internado camino verde	La Guajira	Uribia
29	GJ03	Manaure Escuela Comunidad Musichi	La Guajira	Manaure
30	GJ04	Ceura	La Guajira	Maicao
31	GJ05	Riohacha comunidad Epinayu	La Guajira	Riohacha
32	GJ06	Maicao	La Guajira	Maicao
33	GJ08	Barbacoas	La Guajira	Riohacha
34	GJ09	Rio ancho escuela de la naturaleza	La Guajira	Dibulla

	Identificador	Nombre	Departamento	Municipio
35	GJ10	San juan del cesar	La Guajira	San Juan del Cesar
36	GR1C	Isla Gorgona 2	Cauca	Guapi
37	GUA	Guaviare	Guaviare	San José del Guaviare
38	GUY2C	Guyana2	Caldas	Villamaría
39	HEL	Santa helena	Antioquia	Medellín
40	JAMC	Jamundí	Valle del cauca	Jamundí
41	LCBC	Los Córdobas	Córdoba	Los Córdobas
42	MACC	La macarena	Meta	La Macarena
43	MAL	Málaga	Valle del cauca	Buenaventura
44	MAP	Isla Malpelo	Valle del cauca	Buenaventura
45	MD01	Santa rosa	Magdalena	Fundación
46	MD02	Fundación	Magdalena	Fundación
47	MD03	Aracataca	Magdalena	Aracataca
48	MD04	Rio frio	Magdalena	Ciénaga
49	MD05	Ciénaga	Magdalena	Ciénaga
50	MD06	Guachaca	Magdalena	Santa marta
51	MD07	Barranquilla	Atlántico	Barranquilla
52	NOR	Norcasia	Caldas	Norcasia
53	OCA	Ocaña	Santander	Ocaña
54	ORTC	Ortega	Tolima	Ortega
55	PAL	San José del palmar	Choco	San José del palmar
56	PAM	Pamplona	Norte de Santander	Pamplona
57	PIZC	Pizarro	Choco	Bajo Baudó
58	POP2	Popayán	Cauca	Popayán
59	PRA	Prado	Tolima	Prado
60	PRV	Providencia	Archipiélago de San Andrés. Prov. Y Santa Catalina	Providencia
61	PTA	Punta ardita	Choco	Jurado
62	PTB	Puerto Berrío	Antioquia	Puerto Berrío
63	PTGC	Puerto Gaitán	Meta	Puerto Gaitán
64	PTLC	Puerto Leguízamo	Putumayo	Puerto Leguízamo
65	QUET	Quetame	Cundinamarca	Quetame
66	RGSC	La regadera - Sumapaz	Cundinamarca	Bogotá D.C
67	ROSC	Rosal	Cundinamarca	El rosal
68	RUS	Rusia	Boyacá	Duitama
69	SBTC	Sibaté	Cundinamarca	Sibaté
70	SJC	San Jacinto	Bolívar	San Jacinto
71	SMAR	Santa marta	Magdalena	Santa marta
72	SML1C	San Martín de Loba	Bolívar	San Martín de loba
73	SMORC	Sierra morena	Bogotá	Cundinamarca
74	SOL	Solano	Choco	Bahía solano
75	SPBC	San pablo de Borbur	Boyacá	San pablo de Borbur
76	TABC	Cerro el tablazo	Cundinamarca	Madrid
77	TAM	Tame	Arauca	Tame
78	TOL	Tolima	Tolima	Ibagué
79	TUM	Tumaco	Nariño	Tumaco
80	TVCAC	Tv cable	Suba	Cundinamarca
81	URE	San José de Ure	Córdoba	San José de Uré

	Identificador	Nombre	Departamento	Municipio
82	URI	Uribia	La Guajira	Uribia
83	VIL	Villavicencio	Meta	Villavicencio
84	YOT	Yotoco	Valle del cauca	Yotoco
85	ZAR	Zaragoza	Antioquia	Zaragoza

Fuente: DATOS ABIERTOS, GOBIERNO DIGITAL DE COLOMBIA. Estaciones Red Sismológica Nacional de Colombia, Servicio Geológico Colombiano. Última actualización: 16 de agosto de 2017. [revisado el 31 de julio de 2018] Disponible en: https://www.datos.gov.co/Minas-y-Energ-a/Estaciones-Red-Sismol-gica-Nacional-de-Colombia-Se/sefu-3xqc.

Tabla C2. Detalle de las estaciones sismológicas nacionales de la RSNC por posición geo-referenciada y altitud.

	Identificador	Latitud	Longitud	Altitud
1	APAC	7.9	-76.58	210
2	ARGC	9,858	-74,246	187
3	BAR2	6,592	-73,182	1,864
4	BBAC	2,022	-77,247	1,713
5	BET	2,723	-75,418	557
6	BRR	7,107	-73,712	137
7	CAP2	8,646	-77,359	229
8	CAQC	4,402	-73,986	2,041
9	CBOC	5,864	-76,012	1,401
10	CHI	4.63	-73,732	3,14
11	CRJC	11.02	-72,882	827
12	CRU	1,568	-76,951	2,761
13	CS01	9,477	-73,473	59
14	CS02	10.15	-73,198	170
15	CS03	9,709	-73,665	47
16	CS04	9,849	-73,778	51
17	CS05	10,034	-73,893	90
18	CS06	10,213	-73,965	167
19	CS07	10.21	-73,556	100
20	CS08	10.51	-73,284	260
21	CUM	0.941	-77,825	3,42
22	CVER	4,521	-74,074	3,608
23	DBB	7,018	-76.21	756
24	FLO2	1,583	-75,653	365
25	FOM	4,475	-73,859	2,381
26	GARC	2,187	-75,493	1,999
27	GJ01	12,182	-72,117	20
28	GJ02	11.72	-72,309	21
29	GJ03	11,727	-72,546	5
30	GJ04	11,471	-72,533	46

	Identificador	Latitud	Longitud	Altitud
31	GJ05	11,541	-72,848	7
32	GJ06	2,022	-77,247	1,713
33	GJ08	11,252	-72,877	100
34	GJ09	11,218	-73,248	75
35	GJ10	10,792	-73,016	252
36	GR1C	3,003	-78,167	39
37	GUA	2,545	-72,627	217
38	GUY2C	5,224	-75,365	3,605
39	HEL	6,191	-75,529	2,815
40	JAMC	3,215	-76,673	1,4
41	LCBC	8,857	-76,368	75
42	MACC	2,145	-73,848	283
43	MAL	4,013	-77,335	75
44	MAP	4,004	-81,606	137
45	MD01	10,424	-74,094	96
46	MD02	10,513	-74,111	66
47	MD03	10,704	-74,111	75
48	MD04	10,895	-74,151	38
49	MD05	11,022	-74,212	27
50	MD06	11,281	-73,877	19
51	MD07	10,909	-74,856	39
52	NOR	5,564	-74,869	536
53	OCA	8.239	-73.319	1436
54	ORTC	3,909	-75,246	446
55	PAL	7.34	-72.7	3,676
56	PAM	4,965	-77.36	38
57	PIZC	2.54	-76,676	1,869
58	POP2	7.34	-72.7	3,676
59	PRA	3,714	-74,886	457
60	PRV	13,376	-81,364	63
61	PTA	7,147	-77,808	78
62	PTB	6.54	-74,456	260
63	PTGC	4,199	-72,134	170
64	PTLC	-0.171	-74,797	240
65	QUET	4,381	-73,883	2,196
66	RGSC	4,368	-74,186	3,266
67	ROSC	4.84	-74.32	2,987
68	RUS	5,893	-73,083	3,697
69	SBTC	4,477	-74,287	2,71
70	SJC	9,897	-75.18	596
71	SMAR	11,164	-74,225	122
72	SML1C	8,862	-73,992	36
73	SMORC	4,575	-74.17	2,822
74	SOL	6,226	-77,409	38
75	SPBC	5,652	-74,072	799
76	TABC	5,011	-74,204	3,5

	Identificador	Latitud	Longitud	Altitud
77	TAM	6,436	-71,791	457
78	TOL	4,585	-75.32	2,577
79	TUM	1,824	-78,727	50
80	TVCAC	4,718	-74,085	2,685
81	URE	7,752	-75,533	251
82	URI	11,702	-71,993	68
83	VIL	4,112	-73,694	1,109
84	YOT	3,983	-76,345	1,04
85	ZAR	7,492	-74,858	205

Fuente: DATOS ABIERTOS, GOBIERNO DIGITAL DE COLOMBIA. Estaciones Red Sismológica Nacional de Colombia, Servicio Geológico Colombiano. Última actualización: 16 de agosto de 2017. [revisado el 31 de julio de 2018] Disponible en: https://www.datos.gov.co/Minas-y-Energ-a/Estaciones-Red-Sismol-gica-Nacional-de-Colombia-Se/sefu-3xqc.

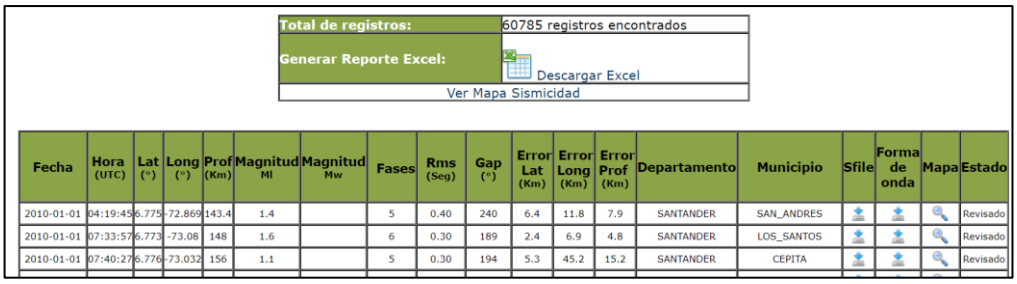
ANEXO D – DESCARGA DE ARCHIVOS SFILE Y WAVEFORM

Para la obtención de los datos históricos sísmicos del departamento de Santander, se revisa la información que se encuentra en la base de datos de la RSNC que puede ser accedida mediante el aplicativo web con Localizador de Recurso URL *https://bdrsnc.sgc.gov.co/paginas1/catalogo/Consulta_Experta/consultaexperta_2.php*. En la vista web que se obtiene al consultar el recurso (ver Figura D1(a)), pueden hacerse búsquedas avanzadas desde el 1 de junio de 1993 hasta el 28 de febrero de 2018.

La búsqueda o consulta sísmica experta ejecutada para la visualización de los registros sísmicos del departamento de Santander, en un tiempo comprendido entre el primero de enero del año 2010 al 30 de septiembre de 2017, se hace siguiendo el proceso mostrado en la Figura D1. La búsqueda arroja 60.785 registros sísmicos distribuidos en 1.737 secciones con tablas de 36 filas cada una. Esto significa que cada día se registran, en promedio, 21 sismos en el departamento o 600 sismos mensuales.

(a)

(b)



(c)

Figura D1. Proceso seguido para la búsqueda avanzada de registros sísmicos de la RSNC*:* (a) formulario inicial de búsqueda avanzada, (b) formulario diligenciado con los parámetros de búsqueda y (c) resultado tabulado de los registros sísmicos divididos en secciones de 36 filas cada uno.

El URL de cada una de las secciones de la tabla de 36 filas que contiene los registros sísmicos tiene la siguiente estructura:

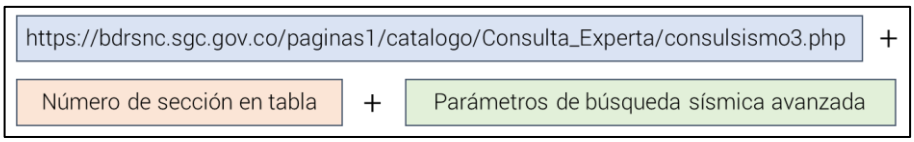


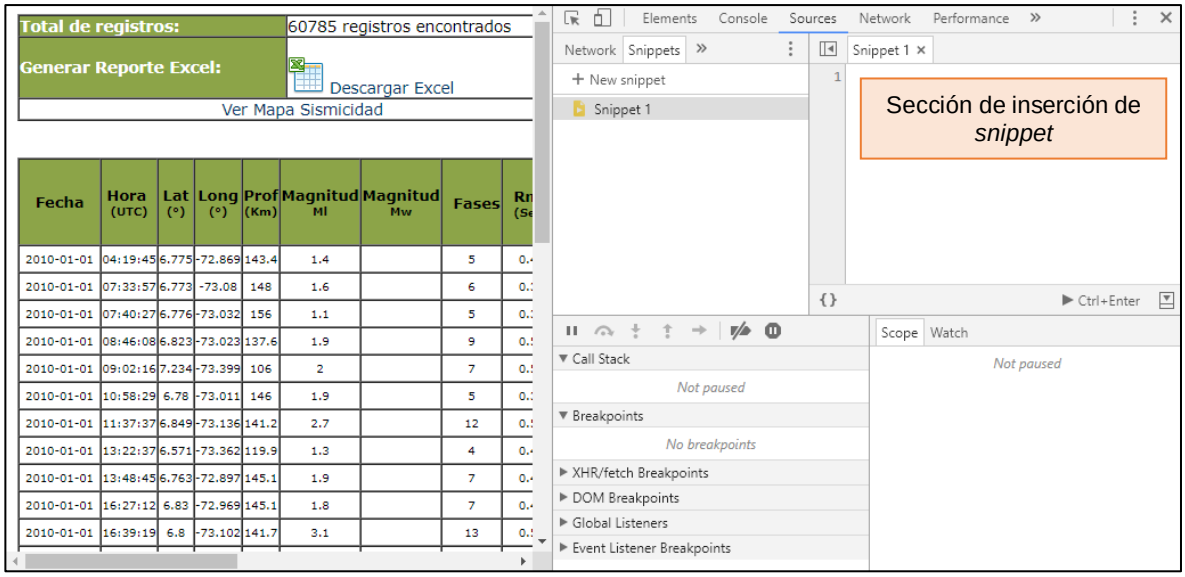
Figura D2. Estructura de la URL para la búsqueda avanzada de registros sísmicos de la RSNC.

Un ejemplo de URL es el siguiente:

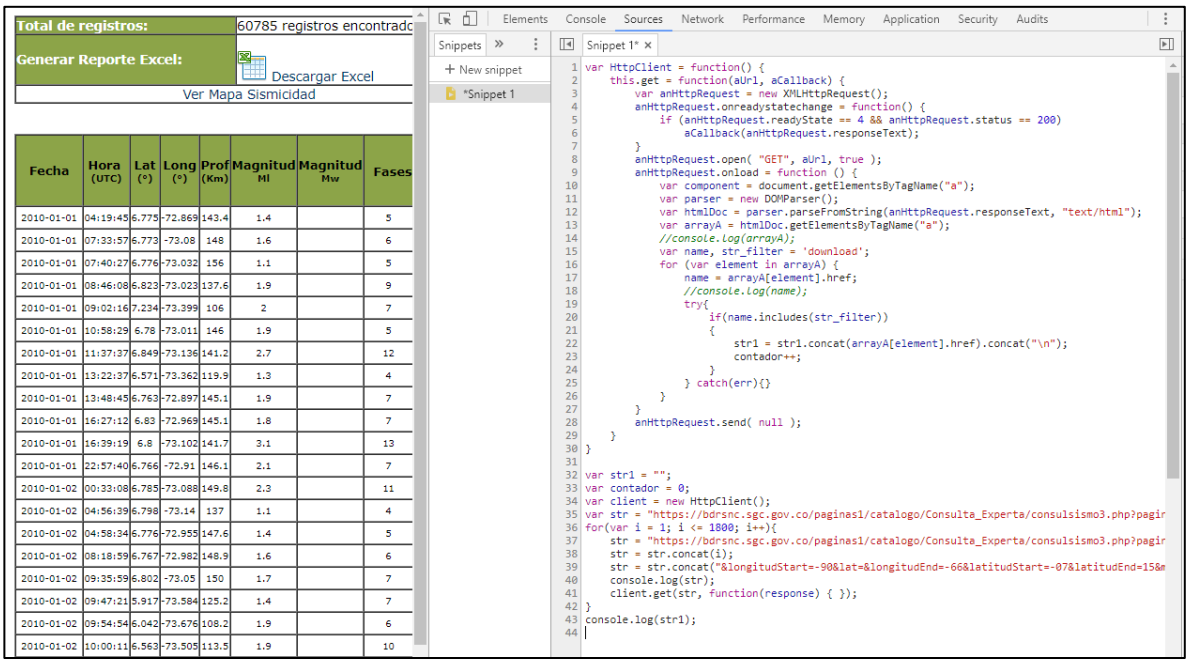
https://bdrsnc.sgc.gov.co/paginas1/catalogo/Consulta_Experta/consulsismo3.php?pagina=1&longitudStart=-90&lat=&longitudEnd=-66&latitudStart=-07&latitudEnd=15&magnitudStart=0&magnitudEnd=9&magnitudmwStart=0&magnitudmwEnd=9&depthStart=0&depthEnd=700&rmsStart=0&rmsEnd=10&inicial=01/01/2010&final=10/10/2017&contipo=cuadrante&longcentral=&radio=&registro1=35&departamento=SANTANDER&municipio=&gapinicio=0&gapfinal=360&eprofmin=0&elongmin=0&elatmin=0&eprofmax=999&elongmax=999&elatmax=999

Como se observa, la primera sub-cadena de la URL (*https://bdrsnc.sgc.gov.co/pag*

inas1/catalogo/Consulta_Experta/consulsismo3.php) es invariante para cualquier sección de la tabla y el recurso hacia el cual se dirigirá la misma. La segunda sub-cadena (*pagina=1*) denota la página en la que se encuentra la sección, en este caso, la primera página. La sub-cadena final expresa el valor de cada uno de los atributos en la búsqueda avanzada ejecutada. Puesto que la búsqueda se hizo sobre el departamento de Santander, puede notarse en el atributo de '*departamento'* que está referenciado, al igual que la fecha de inicio (atributo de nombre '*inicial*') y la fecha de fin (atributo de nombre '*final*'). La descarga de los archivos se hace mediante un *snippet*, una porción de codificación *javascript* que permite ser ejecutada de forma integrada con otros módulos web ya existentes. El *snippet* se introduce de la siguiente forma:



(a)



(b)

Figura D3. Proceso seguido para la inserción del *snippet*: (a) vista de inserción de recursos de red y *snippets*, (b) inserción del *snippet* en la ventana de inspección.

Para la creación del *snippet* se tiene en cuenta que:

1. Los recursos HTML resultantes contienen tablas divididas en secciones, cada una con la misma cantidad de filas. Las filas contienen información correspondiente a los parámetros de búsqueda mencionados en la Figura D2, que están asociados a los siguientes atributos de un evento sísmico: fecha, hora, latitud, longitud, profundidad, magnitud, fases, GAP, errores, departamento, municipio y los archivos binarios y de forma de onda.

2. El URL, como ya se ha mencionado, presenta la misma estructura y la única variación entre páginas está en la modificación del número de página. Los parámetros de búsqueda y la dirección del recurso inicial permanecen invariantes.

3. Los archivos binarios y de forma de onda están representados por un mismo ícono de descarga que corresponde a una imagen cuadrada de 16 pixeles de nombre “descargar1.png”.

El *snippet* construido a partir de estas consideraciones es el siguiente:

```
var HttpClient = function() {
    this.get = function(aUrl, aCallback) {
        var anHttpRequest = new XMLHttpRequest();
        anHttpRequest.onreadystatechange = function() {
            if (anHttpRequest.readyState == 4 && anHttpRequest.status == 200)
                aCallback(anHttpRequest.responseText);
        }
        anHttpRequest.open( "GET", aUrl, true );
        anHttpRequest.onload = function () {
            var component = document.getElementsByTagName("a");
            var parser = new DOMParser();
            var htmlDoc = parser.parseFromString(anHttpRequest.responseText, "text/html");
            var arrayA = htmlDoc.getElementsByTagName("a");
            //console.log(arrayA);
            var name, str_filter = 'download';
            for (var element in arrayA) {
                name = arrayA[element].href;
                try{
                    if(name.includes(str_filter))
                    {
                        str1 = str1.concat(arrayA[element].href).concat("\n");
                        contador++;
                    }
                } catch(err){}
            }
        }
        anHttpRequest.send( null );
    }
}

var str1 = "";
var contador = 0;
var client = new HttpClient();
var str = "https://bdrsnc.sgc.gov.co/paginas1/catalogo/Consulta_Experta/consulsismo3.php?pagina=";
for(var i = 1; i <= 1737; i++){
    str = "https://bdrsnc.sgc.gov.co/paginas1/catalogo/Consulta_Experta/consulsismo3.php?pagina=";
    str = str.concat(i);
    str = str.concat("&longitudStart=-90&lat=&longitudEnd=-66&latitudStart=-
07&latitudEnd=15&magnitudStart=0&magnitudEnd=9&magnitudmwStart=0&magnitudmwEnd=9&depth
Start=0&depthEnd=700&rmsStart=0&rmsEnd=10&inicial=01/01/2010&final=31/12/2017&contipo=cu
adrante&longcentral=&radio=&registro1=35&departamento=SANTANDER&municipio=MUNICIPIO&gapi
nicio=0&gapfinal=360&eprofmin=0&elongmin=0&elatmin=0&eprofmax=999&elongmax=999&elatmax=
999");
    console.log(str);
    client.get(str, function(response) { });
}
console.log(str1);
```

El algoritmo ejecuta un ciclo en el que se accede a las 1.737 páginas de recursos haciendo una petición HTTP y extrayendo los elementos HTML resultantes de las respuestas del servidor a las peticiones. Esto se hace teniendo en cuenta que, según la consideración número 2 mencionada, el número de página es lo único que cambia en la URL de cada una de las páginas y este número corresponde al iterador del ciclo. En los elementos encontrados se filtran aquellos íconos que tengan una referencia *href* de descarga de archivos, 'download' para las trazas y 'REA' para los archivos binarios de información sísmica, teniendo en cuenta la consideración número 3 mencionada. Cada una de las direcciones obtenidas de los íconos es concatenada en una cadena que al final imprime todas las referencias a los archivos en todas las páginas, tanto para los archivos de traza como para los binarios de información.

La cadena de las URL donde se encuentran cada uno de los archivos binarios de información tiene la siguiente forma, tal y como se muestra en la Figura 2:

https://bdrsnc.sgc.gov.co/REA_APOLO/2010/01/05-0720-04L.S201001
https://bdrsnc.sgc.gov.co/REA_APOLO/2010/01/05-0846-40L.S201001
https://bdrsnc.sgc.gov.co/REA_APOLO/2010/01/05-1241-35L.S201001
⋮
https://bdrsnc.sgc.gov.co/REA_APOLO/2017/12/30-2131-17L.S201712
https://bdrsnc.sgc.gov.co/REA_APOLO/2017/12/30-2302-52L.S201712
https://bdrsnc.sgc.gov.co/REA_APOLO/2017/12/31-0026-53L.S201712

La cadena de las URL donde se encuentran cada uno de los archivos de trazas de formas de onda tiene la siguiente forma (la dirección de los recursos reemplazados por la cadena de tres puntos es '*paginas1/catalogo/Consulta_Experta/*'), tal y como se muestra en la Figura D4:

https://bdrsnc.sgc.gov.co/.../download.php?file=2010/01/2010-01-13-2020-00S.COL___095
https://bdrsnc.sgc.gov.co/.../download.php?file=2010/01/2010-01-13-2048-41S.COL___095
https://bdrsnc.sgc.gov.co/.../download.php?file=2010/01/2010-01-14-0324-07S.COL___095
⋮
https://bdrsnc.sgc.gov.co/.../download.php?file=2017/12/2017-12-31-1508-00M.COL___462
https://bdrsnc.sgc.gov.co/.../download.php?file=2017/12/2017-12-31-1518-00M.COL___453
https://bdrsnc.sgc.gov.co/.../download.php?file=2017/12/2017-12-31-1938-00M.COL___475

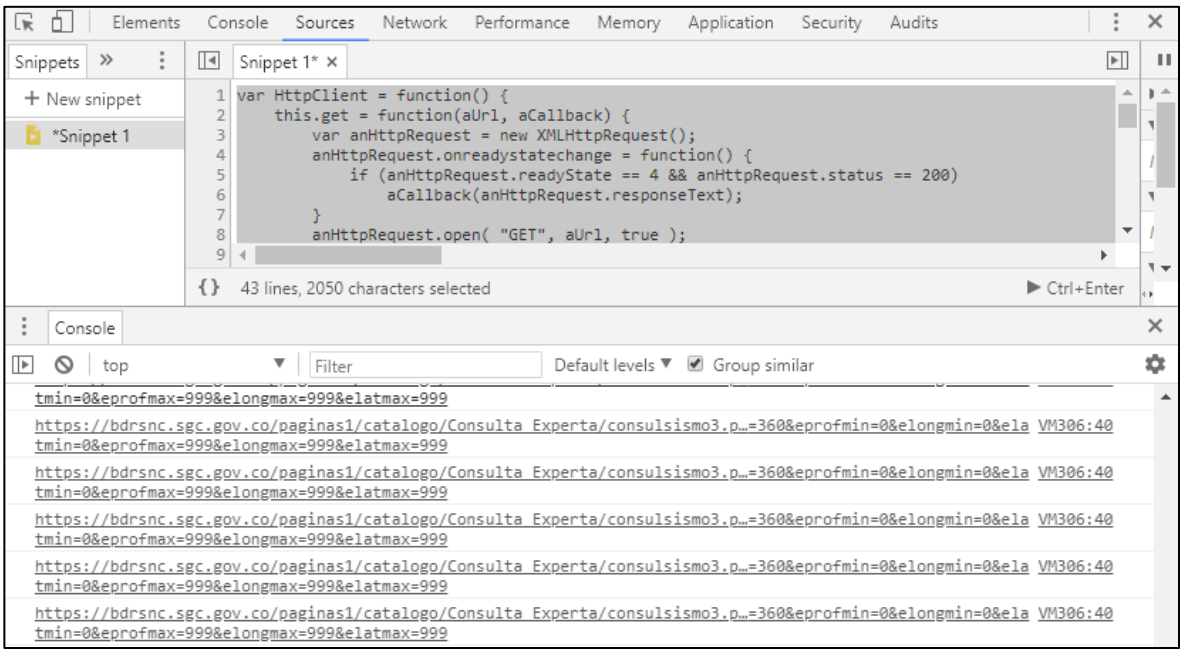


Figura D4. Salida obtenida en consola al ejecutar el *snippet* sobre el recurso HTML del formulario de búsqueda avanzada resultante.

Para la descarga de los archivos se usa un código en *Bash* que permita la lectura de los archivos con las direcciones URL y descargue los recursos en una carpeta destinada para este fin. El código usado fue el siguiente:

```
PathDownload=`awk -F ":" '/PathDownload/ { print $2 ;} ' RSNC_Parameter.ascii`
SeisanFile=`awk -F ":" '/SeisanFile/ { print $2 ;} ' RSNC_Parameter.ascii`
MseedFile=`awk -F ":" '/MseedFile/ { print $2 ;} ' RSNC_Parameter.ascii`
Sfiles=`awk -F ":" '/Sfiles/ { print $2 ;} ' RSNC_Parameter.ascii`

cd $HOME/Descargas
pwd
for archivo in `ls Waveforms*.txt`; do
  mv $archivo $PathDownload
  sed -i '1d' $PathDownload/$archivo
  wget --no-check-certificate -i $PathDownload/$archivo
  rm $PathDownload/$archivo
done

for archivo in `ls Sfiles*.txt`; do
  echo $archivo
  mv $archivo $PathDownload
  sed -i '1d' $PathDownload/$archivo
  wget --no-check-certificate -i $PathDownload/$archivo
  rm $PathDownload/$archivo
done
```

Las primeras líneas del código definen la ubicación de los archivos binarios de información y de formas de onda. Posteriormente se descargan, se almacenan en esta ubicación y se eliminan de la ubicación actual para mantener el orden de recursos. La descarga se hace con el comando *wget --no-check-certificate –i* con el que se especifica que no es necesaria la verificación del certificado de la fuente para hacer la descarga.

ANEXO E – FORMATO DE LOS ARCHIVOS SFILE Y WAVEFORM DE LA RSNC

Regularmente, un archivo *SFile* de la RSNC tiene la siguiente estructura:

```
2015  310 0645 55.8 L   6.817 -73.156145.2  RSN  9 0.4 1.7LRSN               1
GAP=134        1.10       3.4     5.2  6.1 -0.1674E+01  0.7456E+01  0.5411E+00E
Epicentro: LOS SANTOS - SANTANDER                                            3
ACTION:UPD 15-04-30 16:24 OP:nd   STATUS:               ID:20150310064555 L   I
OLDACT:REG 15-03-10 16:45 OP:cam  STATUS:               ID:20150310063749     3
2015-03-10-0637-49M.COL___275                                                6
STAT SP IPHASW D HRMM SECON CODA AMPLIT PERI AZIMU VELO AIN AR TRES W  DIS CAZ7
BAR2 EZ EP        646 14.81                             170   -0.6010 25.0 187
BAR2 EZ ES        646 30.59                             170   -0.1010 25.0 187
BAR2 EZ I         646 31.56         8.6 0.22            168           25.0 187
BRR  HZ EP        646 17.25                             153    0.4610 69.3 298
BRR  HN ES        646 33.79                             153    0.6310 69.3 298
BRR  HZ I         646 37.81         9.4 0.46            151           69.3 298
PAM  SZ IP     C  646 18.34                             151    0.4210 76.7  41
PAM  SN ES        646 35.40                             151    0.2310 76.7  41
PAM  SZ I         646 36.95         7.3 0.18            155           76.7  41
RUS  HZ EP        646 20.00                             143    0.3610  103 175
RUS  HE ES        646 37.43                             143   -0.8110  103 175
RUS  HZ I         646 39.52         5.9 0.32            144            103 175
PTB  HZ EP        646 22.20                             132   -0.46 9  147 258
PTB  HN ES        646 43.63                             132    0.02 9  147 258
PTB  HZ I         646 47.86         7.8 0.10            131            147 258
```

Figura E1. Archivo *SFile* que contiene la información sísmica del evento registrado el 10 de marzo de 2015. Adaptado de Red Sismológica Nacional de Colombia. 2018.

En ocasiones, los parámetros encontrados en el encabezado pueden variar y las características mostradas dependen de la tipología del evento sísmico, de la entidad que lo estructure y del algoritmo utilizado para el almacenamiento de los datos. Con el fin de simplificar el entendimiento del formato de los datos, se ha considerado el formato convencional encontrado en el 100% de los datos registrados y descargados del servicio de la RSNC. Este formato es el siguiente:

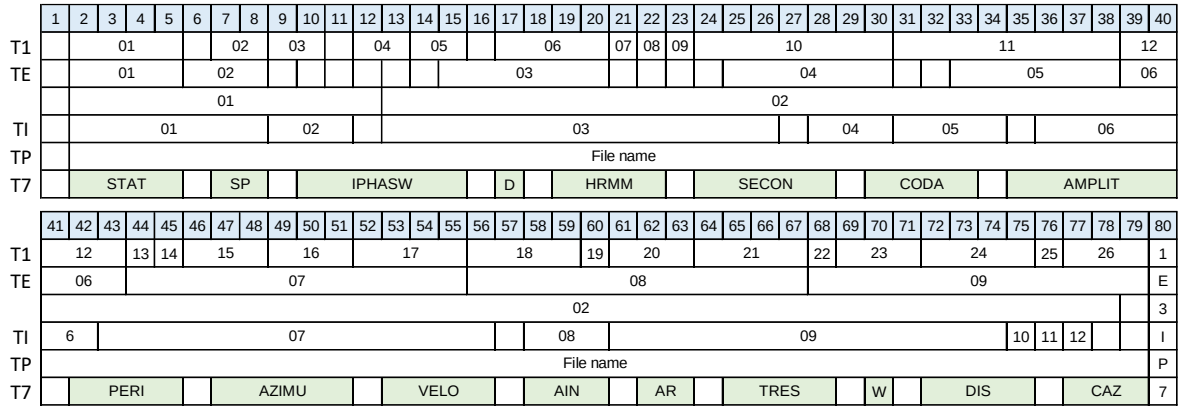


Figura E2. Formato del encabezado de los archivos *SFiles*.

Esta estructura obedece a un formato internacional denominado *Formato Nórdico.* El formato recibe este nombre, pues fue una estructura inicialmente planteada por cinco países nórdicos para la expresión y el detallado de eventos sísmicos. Al consignar los parámetros de forma ordenada y ser el único formato datado a la época, se convirtió en un formato referente a nivel internacional y posteriormente en un estándar de representación de los eventos sísmicos y la consigna de la información. Los archivos escritos en este formato se componen de dos grandes secciones: el encabezado, generalmente las cinco primeras líneas que identifican la información resumida registrada del evento, y el cuerpo, las líneas restantes que identifican la información sísmica registrada por las estaciones.

Cada una de las líneas posee una longitud de 80 caracteres y está estructurada en un tipo de sub-formato. El indicador del formato se encuentra en el último carácter de la línea. Las líneas del encabezado pueden ser superiores a 5, pues pueden incluir comentarios y aspectos adicionales de cada entidad y particularidades del evento sísmico. Los comentarios obedecen al formato Tipo 3 y son líneas continuas sin seccionamiento por caracteres. Usualmente la información del lugar de epicentro (municipio – departamento) se describe con esta tipología de línea en el archivo y es una línea que la RSNC suele adicionar al formato.

La primera línea del encabezado del archivo está estructurada según una descripción paramétrica Tipo 1 en la que se consigna información de fecha y magnitud, incluidos los siguientes atributos:

01. Año de ocurrencia del evento
02. Mes de ocurrencia del evento
03. Día de ocurrencia del evento
04. – 06. Hora de ocurrencia del evento
07. Modelo usado para la localización del evento
08. Indicador de distancia:
    a. Local (L)
    b. Regional (R)
09. ID del evento:
    a. Explosión (E)
    b. Explosión probable (P)
    c. Volcánica (V)
10. Latitud
11. Longitud
12. Profundidad
13. Indicador de profundidad
14. Indicador de localización
15. Entidad que reporta el hipocentro del evento
16. Número de estaciones usadas para la medición
17. RSM del tiempo residual
18. Magnitud No. 1
19. Tipo de magnitud
20. Entidad que reporta la magnitud 1
21. Magnitud No. 2
22. Tipo de magnitud
23. Entidad que reporta la magnitud 2
24. Magnitud No. 3
25. Tipo de magnitud
26. Entidad que reporta la magnitud 3

La razón de inclusión de diversas magnitudes para un mismo evento se debe a que múltiples agencias oficiales de sismología pueden publicar el mismo sismo con diversas magnitudes. En este trabajo se ha considerado la medición de las magnitudes publicadas por la RSNC como guía en los posteriores procesos de procesamiento y análisis de la información.

La segunda línea está estructurada según una descripción paramétrica Tipo E en la que se consigna información concerniente con los errores del cálculo en la localización del hipocentro y la covarianza entre los tres ejes de medición. La tercera línea no tiene una estructurada fija y la información contenida puede variar según lo

estipule la entidad, en este caso, la RSNC. Sin embargo, en los casos estudiados, esta línea brinda información relacionada con el lugar geográfico donde fue localizado el epicentro del evento sísmico.

La cuarta línea está estructurada según una descripción paramétrica Tipo I, en la que se consigna una descripción administrativa de las acciones ejecutadas cuando fue identificado el evento sísmico. La quinta línea está estructurada según una descripción paramétrica Tipo P en la que se consigna el nombre del archivo de forma de onda (*waveform*) relacionado con el evento y que contiene toda la información detallada de las estaciones, sus componentes y las trazas de medición.

La primera línea del cuerpo del archivo está estructurada según una descripción paramétrica Tipo 7 en la que se consignan los siguientes atributos:

01. STAT: código de la estación registrada. Según datos de la RSNC, existen en la actualidad 85 estaciones sismológicas nacionales y 10 internacionales en Venezuela y Ecuador cuyos datos son analizados para la identificación de eventos sísmicos. El listado de estaciones se muestra en el Anexo C.
02. SP: tipo de medidor en estación. Está dividido en dos componentes: el código de banda y el código del instrumento. En las Tablas E1 y E2 se muestran las tipologías para cada caso. Usualmente, suelen encontrarse las siguientes combinaciones, teniendo en cuenta dos tipos de instrumentos, el sismómetro y el acelerómetro:
    a. HH: sismómetro banda ancha a 100 Hz
    b. HL: sismómetro largo periodo a 100 Hz
    c. HN: acelerómetro
    d. BH: sismómetro banda ancha a 40 Hz
    e. EH: sismómetro corto periodo a 100 Hz

03. IPHASW: tipo de picado[119] (I), fase (PHAS) y peso (W). La fase identifica si se trata de la onda P o de la onda S que ha sido picada, adjuntando el tiempo de picado en el siguiente atributo.
04. D: polaridad
05. HRMM SECON: tiempo en horas, minutos y segundos de picado de la onda, dependiendo de la onda correspondiente, P o S.
06. CODA: duración del evento en segundos
07. AMPLIT: amplitud
08. PERI: periodo de la onda donde se lee la amplitud
09. ASIMU: azimut en la estación
10. VELO: velocidad aparente de la fase
11. SNR: razón de señal a ruido
12. AR: azimut residual de la localización
13. TRES: tiempo residual
14. W: peso que le da el sistema a la fase picada
15. DIS: distancia epicentral[120] en Km
16. CAZ: azimut del evento a la estación

[119] *def*. Cuando ocurre un evento sísmico, se desprenden distintos tipos de ondas entre las que se encuentran las ondas P y las ondas S. El tiempo aproximado de la llegada de la onda P, una vez ha pasado el evento símico, visto desde la perspectiva del sismólogo y contrastado con un algoritmo matemático de identificación de la onda P, se denomina tiempo de picado de la onda P. Tomado de: CHI DURÁN, Rodrigo Kimyen. Caracterización de trazas sísmicas en el campo cercano: Pisagua, Norte de Chile. Universidad de Chile, Facultad de Cienias Físicas y Matemáticas, Departamento de Ingeniería Eléctrica. 2015.

[120] *def*. Distancia epicentral: distancia que existe entre el epicentro localizado y cualquier punto de interés. Tomado de: RICHTER, Charles Francis. Elementary Seismology, W. H. Freeman and Company Inc, USA (Indian Reprint in 1969 by Eurasia Publishing House Private Limited, New Delhi). 1958.

Tabla E1. Tipo de banda de sismómetros y acelerómetros de medición[121].

Código de banda	Tipo de banda	Frecuencia de muestreo (Hz)
F	-	≥ 1000 a < 5000
G	-	≥ 1000 a < 5000
D	-	≥ 250 a < 1000
C	-	≥ 250 a < 1000
E	Extremadamente corto periodo	≥ 80 a < 250
S	Corto periodo	≥ 10 a < 80
H	Alto ancho de banda	≥ 80 a < 250
B	Banda ancha	≥ 10 a < 80
M	Periodo medio	> 1 a < 10
L	Largo periodo	≈ 1
V	Muy largo periodo	≈ 0.1
U	Ultra largo periodo	≈ 0.01
R	Extremadamente largo periodo	≥ 0.0001 a < 0.001
P	En el orden de 0.1 a 1 día	≥ 0.00001 a < 0.0001
T	En el orden de 1 a 10 días	≥ 0.000001 a < 0.00001
Q	Mayor a 10 días	< 0.000001
A	Canal administrativo	variable
O	Canal opaco	variable

Tabla E2. Tipo de instrumento de medición sísmica.

Código de instrumento	Tipo de instrumento
H	Sismómetro de alta ganancia
L	Sismómetro de baja ganancia
G	Gravímetro
M	Sismómetro de posicionamiento de masa
N*	Acelerómetro

Los archivos de forma de onda o archivos *waveforms* son archivos con una estructura binaria que lucen como la mostrada en la Figura E3. Son los archivos en

[121] INCORPORATED RESEARCH INSTITUTIONS FOR SEISMOLOGY (IRIS). SEED Reference Manual, SEED format version 2.4, Standard for the Exchange of Earthquake Data. August 2012.

*Esta notación depende de la red sismológica y del cambio de notación en el estándar en el 2000. Para años previos, puede apreciarse una notación con la letra G o la letra L.

los que queda registrado el evento sísmico muestra a muestra, desde un tiempo antes de su inicio que es determinado por el sismólogo de turno, hasta que ha finalizado el evento. Este tipo de archivos se encuentran en formato *miniSEED (Data Only Standard for the Exchange of Earthquake Data*), un formato creado por IRIS (*Incorporated Research Institutions for Seismology*) con el que se pretende el registro de la actividad sísmica ocurrente y el intercambio de series de tiempo de los eventos ocurridos.

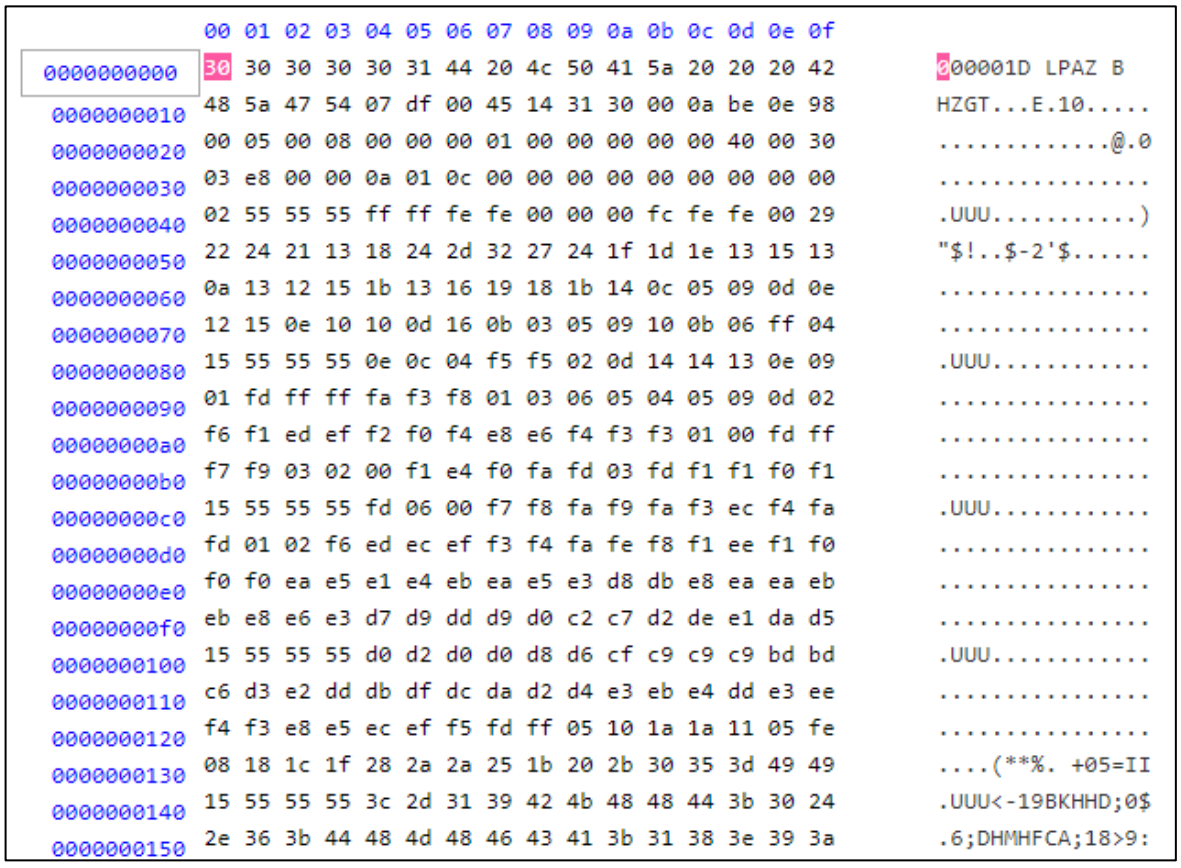


Figura E3. Archivo binario *waveform* que contiene la información específica de trazas del evento registrado el 10 de marzo de 2015.

El formato SEED y sus formatos derivados permiten un intercambio de datos a través de protocolos TCP/IP teniendo en cuenta las características de frecuencia y tiempo de la señal sísmica: bajos contenidos frecuenciales, alto rango dinámico y alta precisión en el tiempo. Esto se debe a que la mayoría de los formatos usados

para la compresión de la información son moderadamente tolerantes a fallos y pérdida de datos. Sin embargo, en el ámbito sísmico, la pérdida de una porción de señal sísmica o la malinterpretación de los datos por el algoritmo significaría una omisión de un posible comportamiento que derive en un evento sísmico no detectado.

Mientras SEED es un estándar de formato complejo con paquetes de datos de 512 bytes denominados *Blockettes* usados para la distribución de la información, miniSEED es un formato abreviado limitado en información que satisface los requerimientos necesarios para la comunicación por medio de los protocolos TCP/IP, sin que se presenten pérdidas o fallos por omisión. En SEED, los *Blockettes* se encuentran numerados del 1 al 2000, incluyendo el 1000 y el 1001 que suelen usarse para el formato abreviado miniSEED[122]. Este último presenta los siguientes campos en su estructura:

1. Encabezado fijo de 48 bytes (ver Tablas E3 y E4)
2. Uno o dos *Blockettes* usualmente de tipo 1000 y opcionalmente de tipo 1001
3. Datos

Tabla E3. Estructura del encabezado fijo de formatos SEED.

	Nombre del campo	Tipo de dato	Posición del byte	Longitud (bytes)
1	Número de secuencia	ASCII	1-6	6
2	Indicador de calidad	ASCII	7	1
3	Byte reservado	ASCII	8	1
4	Código de estación	ASCII	9-13	5
5	Identificador de ubicación	ASCII	14-15	2
6	Identificador del canal	ASCII	16-18	3
7	Código de red	ASCII	19-20	2
8	Registro de la hora de inicio	BTIME	21-30	10
9	Número de muestras	UWORD	31-32	2
10	Factor de tasa de muestreo	WORD	33-34	2

[122] INCORPORATED RESEARCH INSTITUTIONS FOR SEISMOLOGY (IRIS). SEED Reference Manual, SEED format version 2.4, Standard for the Exchange of Earthquake Data. August 2012.

	Nombre del campo	Tipo de dato	Posición del byte	Longitud (bytes)
11	Factor de multiplicación de la tasa	WORD	35-36	2
12	Banderas de actividad	UBYTE	37	1
13	Banderas de I/O	UBYTE	38	1
14	Banderas de calidad de datos	UBYTE	39	1
15	Número de *Blockettes* que siguen	UBYTE	40	1
16	Corrección de tiempo	LONG	41-44	4
17	Corrimiento en inicio de datos	UWORD	45-46	2
18	Corrimiento en inicio de 1er *Blockette*	UWORD	47-48	2

Los tipos de datos descritos en la Tabla E3 son los siguientes:

Tabla E4. Tipos de dato del encabezado fijo y todos los *Blockettes* siguientes de los formatos SEED.

Tipo de campo	Número de bits	Descripción del campo
UBYTE	8	Cantidad sin signo
IBYTE	8	Complemento de la cantidad sin signo
UWORD	16	Cantidad sin signo
WORD	16	Complemento de la cantidad sin signo
ULONG	32	Cantidad sin signo
LONG	32	Complemento de la cantidad sin signo
CHAR	8	Caracter
FLOAT	32	Flotante IEEE

El tipo de dato BTIME con el que se registra el tiempo de inicio del evento sísmico o cualquier información sísmica contenida en el archivo con formato SEED o formatos derivados, tiene la estructura mostrada en la Tabla E5.

Tabla E5. Tipos de dato BTIME del encabezado fijo de los formatos SEED.

	Nombre del campo	Tipo de dato	Posición del bit	Posición del byte	Longitud (bits)
1	Año de ocurrencia de evento	UWORD	1-16	1-2	16
2	Día del año (enero 1 es el día 1)	UWORD	17-32	3-4	16
3	Hora del día (0 a 23)	UBYTE	33-40	5	8
4	Minutos del día (0 a 59)	UBYTE	41-48	6	8
5	Segundos del día (0 a 59)	UBYTE	49-56	7	8
6	Campo sin uso	UBYTE	57-64	8	8
7	Segundos por debajo de 0.0001 (0 a 9999)	UWORD	65-80	9-10	16

Puede notarse que el archivo binario de forma de onda tiene a grandes rasgos, dos grandes secciones de contenido:

- Sección de metadatos donde se especifica información relacionada con cada una de las estaciones que pertenecen a la red de sismógrafos y acelerógrafos de la RSNC, que incluye las estaciones que ha identificado el evento sísmico.
- Sección de registro de las series de tiempo de cada una de las estaciones. Si alguna estación no ha identificado el evento sísmico, la traza será ruido o en su defecto, una constante sobre el 0.

ANEXO F – PROTOTIPOS DESARROLLADOS

Los prototipos desarrollados y etiquetados mediante el versionamiento semántico, en orden cronológico, son:

- Prototipo V0.0.1
- Prototipo V0.0.2
- Prototipo V0.1.0
- Prototipo V0.1.1
  - Prototipo V0.1.1-3stat
  - Prototipo V0.1.1-4stat
- Prototipo V0.2.0

En esta sección se detalla el desarrollo de los prototipos predecesores al prototipo V0.2.0.

1. PROTOTIPO V0.0.1

El Prototipo V0.0.1 mostrado en la Figura F1 es la primera versión de los procesos de procesamiento y clasificación, en el que se consideran los 52.000 datos de entrada y se seleccionan únicamente 100 muestras que cumplan con haber sido registradas por la estación BRR. Estas muestras pasan por el proceso de selección de ventanas de 200 muestras de onda P y ruido, obviando procesos de filtrado, normalización y re-muestreo de las señales registradas. Una vez obtenidas las ventanas, se extrañen dos atributos: el DOP y el RV2T, dando paso a las fases de entrenamiento, validación y prueba de la red mediante entrenamientos simples. La métrica de salida es la exactitud que, al finalizar el proceso, da como resultado un valor de 80%.

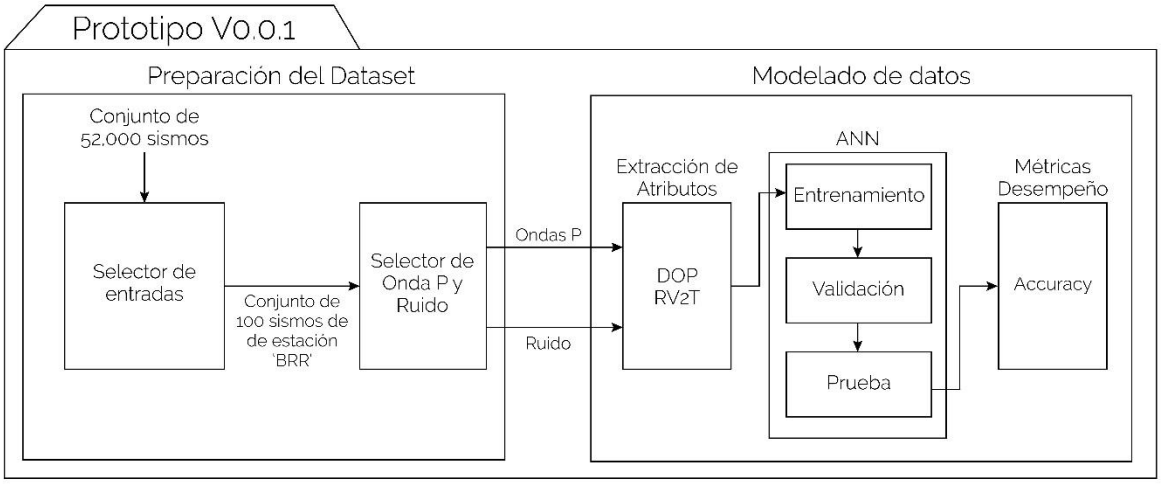


Figura F1. Diagrama de bloques del Prototipo V.0.0.1.

Debido a que se observan variaciones inesperadas en las magnitudes de las señales de entrada, se propone la adición de las etapas de filtrado, normalización, re-muestreo y algunos atributos adicionales en el proceso de extracción, con el fin de aumentar el desempeño del clasificador. Estas mejoras fueron propuestas para el prototipo V0.0.2.

2. PROTOTIPO V0.0.2

En el Prototipo V0.0.2 mostrado en la Figura F2 se ejecutan las siguientes mejoras frente a la versión 0.0.1:

- Inclusión del proceso de preprocesamiento en donde se hace un filtrado, normalización y re-muestreo de las señales entrantes.
- Ampliación de la cantidad de atributos, considerando el DOP, RV2T, la entropía, la Kurtosis, la Asimetría y la Dimensión de Correlación.
- Modificación de los criterios de inclusión de eventos que son considerados en el selector de entradas, de tal forma que todo evento que tenga trazas en cero, ausencia en el registro del tiempo de inicio, fin o de la onda P, sean removidos del conjunto de datos de entrada.

- Inclusión de una validación cruzada por K-fold en el proceso de clasificación de los eventos símicos.

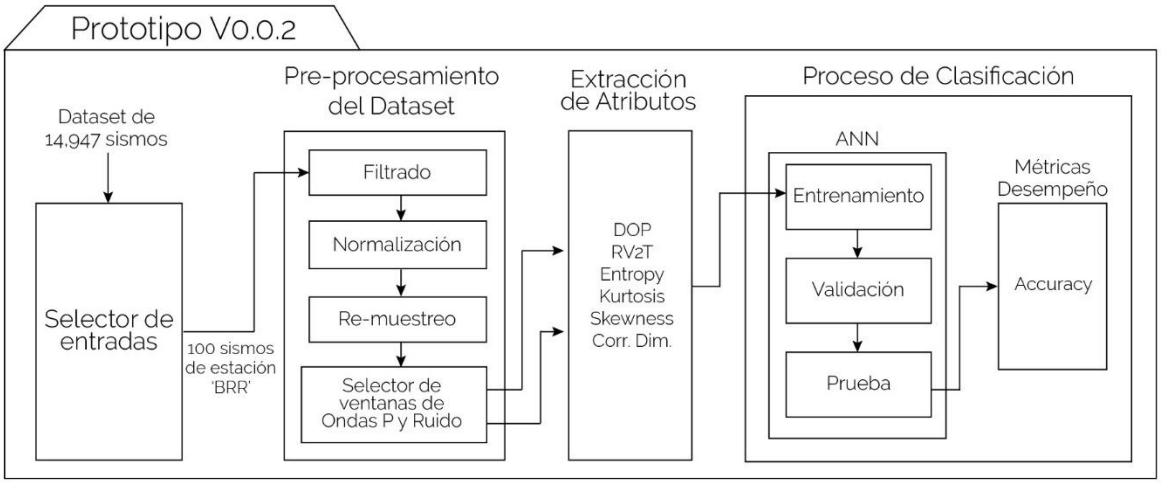


Figura F2. Diagrama de bloques del Prototipo V.0.0.2.

La métrica de salida de exactitud da como resultado al finalizar el proceso un valor de 84%. Sin embargo, aunque se ejecutó un proceso de preprocesamiento que involucra el filtrado, normalización y re-muestreo de las señales, algunas de ellas carecen de la anotación de la onda P y los tiempos de inicio y fin, aunque están especificados, no coinciden entre componentes. Estas consideraciones fueron tenidas en cuenta para el desarrollo del prototipo V0.1.0.

3. PROTOTIPO V0.1.0

En el Prototipo V0.1.0 mostrado en la Figura F3 se ejecutan las siguientes mejoras frente a la versión 0.0.2:

- Modificación de los criterios de inclusión de eventos que son considerados en el selector de entradas, de tal forma que se consideren eventos de las estaciones BRR, RUS y PAM con el fin de identificar mejoras en el desempeño del clasificador.

- Inclusión en el proceso de preprocesamiento de los sub-procesos de anotación de la onda P, sincronización de las señales y selección de las ventanas de onda y ruido, teniendo en cuenta un porcentaje de corrimiento de la onda P.

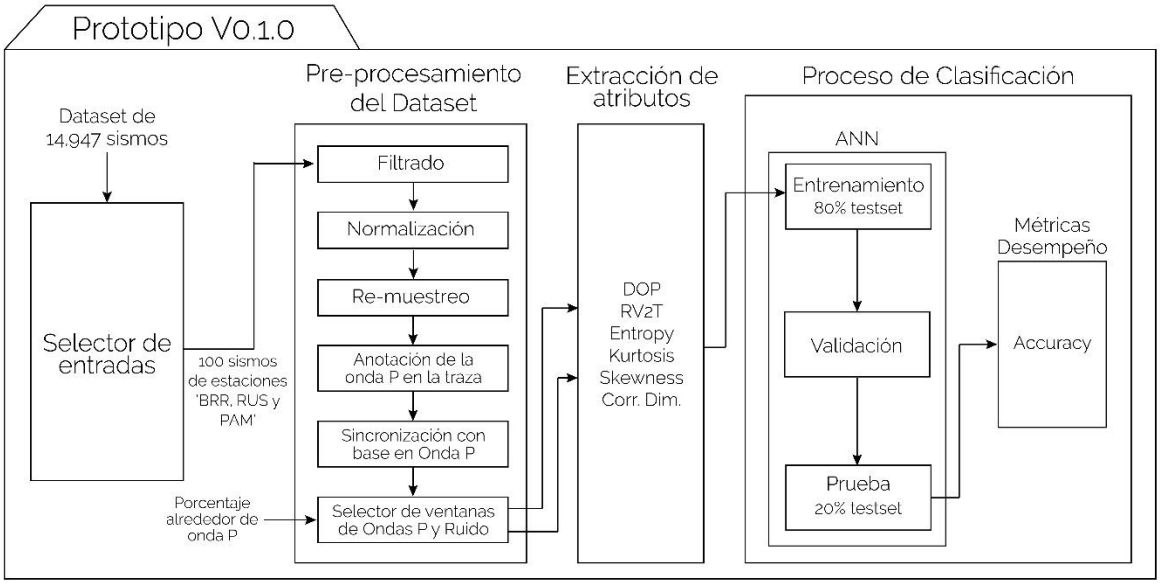


Figura F3. Diagrama de bloques del Prototipo V.0.1.0.

La métrica de salida de exactitud da como resultado al finalizar el proceso un valor de 89%. Sin embargo, aunque el clasificador fue evaluado con una métrica válida de desempeño, la exactitud puede generar un sesgo en la interpretación, debido a que no se puede discriminar el desempeño del clasificador frente a la tasa de rechazo de los eventos. Esta consideración fue tenida en cuenta para el desarrollo del prototipo V0.1.0.

4. PROTOTIPO V0.1.1

En el Prototipo V0.1.1 mostrado en la Figura F4 se ejecutan las siguientes mejoras frente a la versión 0.1.0:

- Cambio de la métrica de validación del entrenamiento de exactitud a F1-score, incluyendo la sensibilidad, especificidad, precisión y Recall como métricas adicionales del clasificador.
- Modificación de los criterios de inclusión de eventos que son considerados en el selector de entradas, de tal forma que se consideren eventos de las estaciones BRR, RUS y PAM o BRR, RUS, PAM y PTB, según sea el caso, con el fin de identificar diferencias en el desempeño del clasificador.

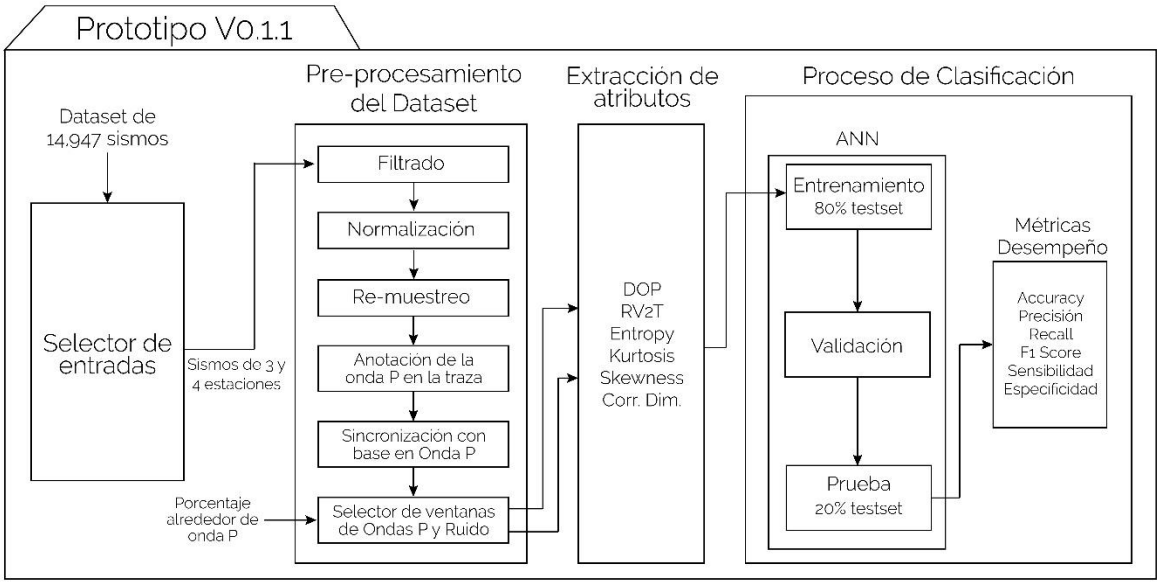


Figura FF4. Diagrama de bloques del Prototipo V.0.1.1.

El prototipo presenta dos derivaciones que son descritas a continuación y que varían en los criterios de inclusión del selector de entradas. Sin embargo, aunque el clasificador fue evaluado con métricas válidas de desempeño, el proceso podría no ser generalizable. Esta consideración fue tenida en cuenta para el desarrollo del prototipo V0.2.0.

4.1. PROTOTIPO V0.1.1-3STATS

En el Prototipo V0.1.1-3STATS mostrado en la Figura F5 se hace una modificación a los criterios de inclusión de eventos que son considerados en el selector de entradas, de tal forma que se consideren eventos de las estaciones BRR, RUS y PAM, ventanas de onda y ruido con dos variaciones en el porcentaje de ubicación de la onda P: 50 y 90%. Se consideraron métricas de desempeño adicionales (sensibilidad, especificidad, precisión, Recall y F1-score), validando el entrenamiento con la métrica F1-score.

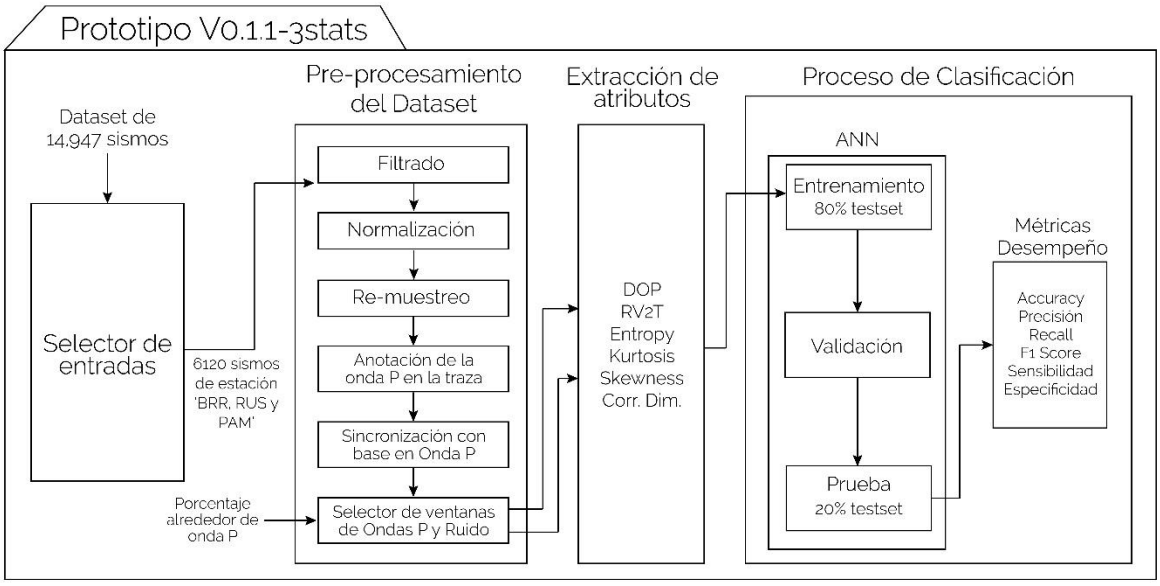


Figura F5. Diagrama de bloques del Prototipo V0.1.1-3STATS.

La métrica de salida de F1-score da como resultado al finalizar el proceso de clasificación un valor de 97.3% para la onda al 50% y 94.9% para la onda al 90%.

4.2. PROTOTIPO V0.1.1-4STATS

En el Prototipo V0.1.1-4STATS mostrado en la Figura F6 se hace una modificación a los criterios de inclusión de eventos que son considerados en el selector de entradas, de tal forma que se consideren eventos de las estaciones BRR, RUS, PAM y PTB, ventanas de onda y ruido con dos variaciones en el porcentaje de ubicación de la onda P: 50 y 90%. Se consideraron métricas de desempeño adicionales (sensibilidad, especificidad, precisión, Recall y F1-score), validando el entrenamiento con la métrica F1-score.

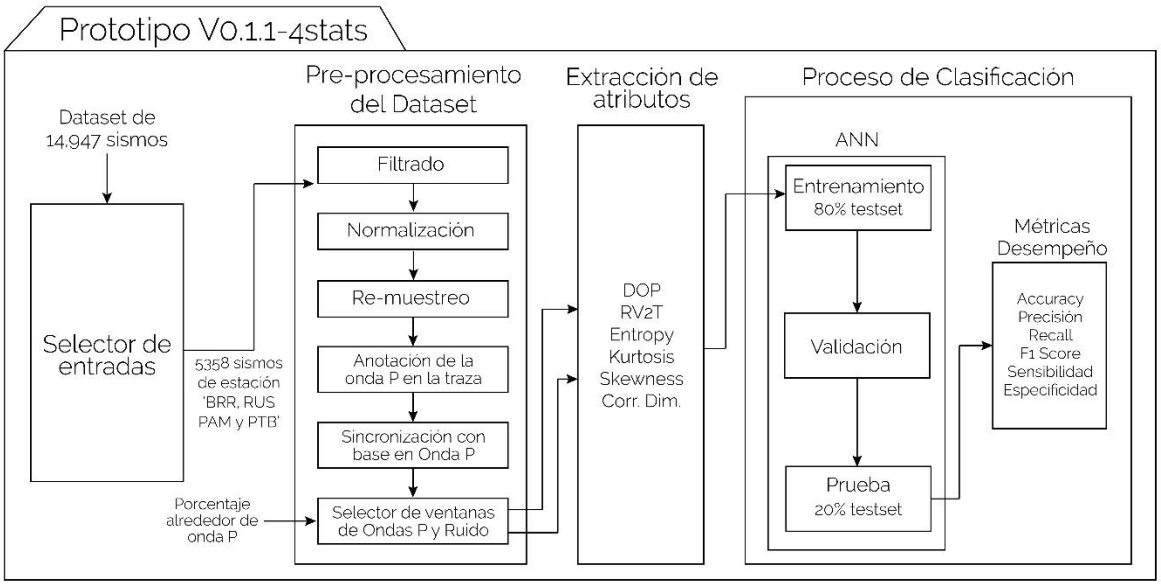


Figura F6. Diagrama de bloques del Prototipo V0.1.1-4STATS.

La métrica de salida de F1-score da como resultado al finalizar el proceso de clasificación un valor de 98.5% para la onda al 50% y 95.6% para la onda al 90%.

ANEXO G – MÉTRICAS DE DESEMPEÑO RESULTANTES

En las Tablas G1 y G2 se detallan las características de la arquitectura de las cuatro redes neuronales desarrolladas para el prototipo V0.2.0, así como las métricas promedio resultantes del proceso de validación cruzada por Monte Carlo, ejecutando variaciones en la cantidad de estaciones y en la posición de la onda P respecto a la ventana de observación.

Tabla G1. Métricas promedio para el *Test set* para el prototipo V0.2.0 con variación en la cantidad de estaciones y la onda P al 90% de la ventana.

	1 estación	2 estaciones	3 estaciones	4 estaciones
Característica	Valor			
Arquitectura				
Número de capas ocultas	2	2	2	2
Número de neuronas por capa oculta	7 y 4	14 y 7	21 y 10	29 y 14
Número de neuronas de entrada	14	28	42	55
Función de entrenamiento	Adadelta	Adadelta	Adadelta	Adadelta
Topología	14-7-4-1	28-14-7-1	42-21-10-1	55-29-14-1
Batch Size	8	16	8	8
Número de epochs	100	10	10	10
Cantidad de folds	10	10	10	10
Métricas de desempeño				
Sensibilidad	0.6967	0.7893	0.9093	0.9272
Especificidad	0.9285	0.9630	0.9828	0.9816
Precisión	0.9084	0.9556	0.9816	0.9806
Recall	0.6967	0.7893	0.9093	0.9272
F1-Score	0.7884	0.8644	0.9440	0.9531
Accuracy	0.8119	0.8757	0.9458	0.9543

Tabla G2. Métricas promedio para el *Test set* para el prototipo V0.2.0 con variación en la cantidad de estaciones y la onda P al 50% de la ventana.

	1 estación	2 estaciones	3 estaciones	4 estaciones
Característica	Valor			
Arquitectura				
Número de capas ocultas	2	2	2	2
Número de neuronas por capa oculta	7 y 4	14 y 7	21 y 10	29 y 14
Número de neuronas de entrada	14	28	42	55
Función de entrenamiento	Adadelta	Adadelta	Adadelta	Adadelta
Topología	14-7-4-1	28-14-7-1	42-21-10-1	55-29-14-1
Batch Size	32	8	128	64
Número de epochs	500	50	50	10
Cantidad de folds	10	10	10	10
Métricas de desempeño				
Sensibilidad	0.9242	0.9537	0.9862	0.9903
Especificidad	0.9736	0.9738	0.9897	0.9938
Precisión	0.9727	0.9733	0.9896	0.9938
Recall	0.9242	0.9537	0.9862	0.9903
F1-Score	0.9477	0.9634	0.9879	0.9921
Accuracy	0.9487	0.9637	0.9879	0.9921