CENTRO DE POSGRADOS DE LA UNIVERSIDAD DE LAS FUERZAS ARMADAS

PROGRAMA DE MAESTRÍA EN REDES DE INFORMACIÓN Y CONECTIVIDAD MRIC-II

TESIS PREVIO A LA OBTENCIÓN DEL TÍTULO DE MAGISTER

TEMA: ANÁLISIS DEL DESEMPEÑO DE ALGORITMOS DE DETECCIÓN DE EVENTOS VULCANOLÓGICOS BASADOS EN MACHINE LEARNING

AUTOR: JUAN FRANCISCO MOREJÓN

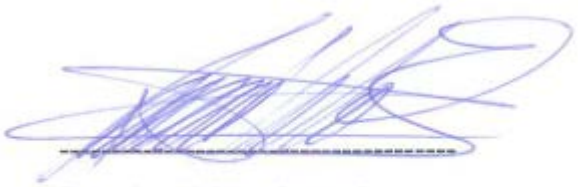
DIRECTOR: ROMÁN ALCIDES LARA CUEVA
CODIRECTOR: VINICIO CARRERA

SANGOLQUÍ
2015

CERTIFICACIÓN

Nosotros, Román Alcides Lara, tutor y Vinicio Carrera, oponente del proyecto: “ANÁLISIS DEL DESEMPEÑO DE ALGORITMOS DE DETECCIÓN DE EVENTOS VULCANOLÓGICOS BASADOS EN MACHINE LEARNING”, elaborado por Juan Francisco Morejón Patiño.

CERTIFICAMOS: Que el alumno antes mencionado ha cumplido con el proceso investigativo tendiente a satisfacer los requisitos para DEFENDER EL PRESENTE PROYECTO.



Román Alcides Lara Cueva
Sangolquí, abril de 2015

Vinicio Carrera
Sangolquí, abril de 2015

AUTORÍA DE RESPONSABILIDAD

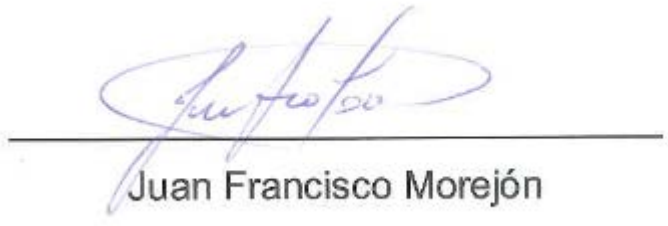
Juan Francisco Morejón

DECLARO QUE:

El proyecto de grado denominado “ANÁLISIS DEL DESEMPEÑO DE ALGORITMOS DE DETECCIÓN DE EVENTOS VULCANOLÓGICOS BASADOS EN MACHINE LEARNING”, ha sido desarrollado en base a una dedicada investigación, respetando derechos de autor, conforme las referencias que constan dentro del documento, cuyas fuentes se incorporan en la bibliografía, consecuentemente el presente trabajo es de mi autoría.

En virtud de esta declaración, me responsabilizo del contenido, veracidad y resultados científicos del proyecto de grado en mención.

Sangolquí, abril de 2015,

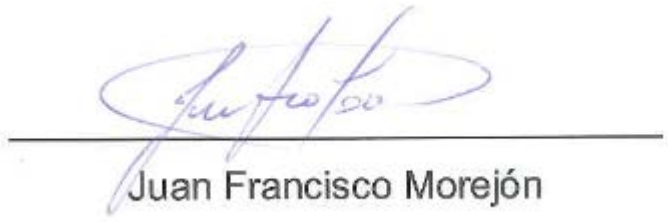


Juan Francisco Morejón

AUTORIZACIÓN

Yo, Juan Francisco Morejón Patiño, autorizo a la Universidad de las Fuerzas Armadas – ESPE, que proceda con la publicación del presente trabajo denominado “ANÁLISIS DEL DESEMPEÑO DE ALGORITMOS DE DETECCIÓN DE EVENTOS VULCANOLÓGICOS BASADOS EN MACHINE LEARNING”,

Sangolquí, abril de 2015,



Juan Francisco Morejón

DEDICATORIA

El presente trabajo de investigación y todo el tiempo y esfuerzo empleados para la consecución de este producto están inspirados en mi amada esposa. Este triunfo está dedicado a ella ya que con su amor ha logrado que mi alma y espíritu no tengan límites.

Una mención especial también la hago para mis padres y hermano, gran parte de este triunfo es por su ayuda.

Juan Fco. Morejón

AGRADECIMIENTOS

Quiero expresar mi más profundo agradecimiento al Director de este proyecto, Román Alcides Lara, ya que con su invaluable participación, guía y ayuda han contribuido a la consecución de este objetivo.

Sin duda Dios ha bendecido este trabajo y a todos sus participantes, por ello, ha culminado con éxito… Gracias.

Juan Fco. Morejón

INDICE

RESUMEN

La detección y caracterización de eventos vulcanológicos y en consecuencia su clasificación, puede ayudar a determinar el comportamiento de un volcán con la finalidad de prevenir a la sociedad frente a una eventual erupción. Es por ello, necesario encontrar un algoritmo con un alto desempeño, que sea capaz de identificar cada evento. El empleo de *machine learning* junto con la medición de parámetros de desempeño como la exactitud, precisión, sensibilidad, especificidad y el gasto computacional son métricas consideradas que permiten definir o establecer el mejor algoritmo. Los datos empleados en el presente trabajo comprenden un período de seis meses de monitorización del volcán Cotopaxi, entre enero y junio del año 2012, datos provistos por el Instituto Geofísico de la Escuela Politécnica Nacional. La base de datos contiene dos eventos principales conocidos como de largo período y terremotos volcano tectónicos. Con elementos de ambos grupos se formaron dos matrices denominadas: matriz de entrenamiento y matriz de prueba. Adicionalmente, se utilizaron 79 características en el dominio del tiempo, frecuencia y escala mediante la transformada rápida de Fourier y la transforma wavelet. Los clasificadores analizados fueron: vecinos más cercanos, árbol de decisiones y redes neuronales, los mismos que mediante el uso de las todas las características y la matriz de entrenamiento, obtuvieron un modelo cada uno. Con la matriz de prueba se obtuvo los parámetros de desempeño para cada modelo. Finalmente se efectuó una selección de características para encontrar aquellas que presentan mayor relevancia, determinando que con las primeras tres características, el modelo predictor que mejores resultados de desempeño presentó fue vecinos más cercanos con una exactitud del 98%

Palabras claves:

- **CLASIFICACIÓN**
- **MODELO PREDICTOR**
- ***MACHINE LEARNING***
- **VECINOS MÁS CERCANOS**
- **ÁRBOL DE DECISIONES**
- **REDES NEURONALES**

ABSTRACT

Identifying volcanological events and consequently classifying them can determine the behavior of a volcano. Finding an algorithm with a high performance that is capable of grouping each event can be very helpful for this purpose. The use of machine learning of supervised type by measuring performance parameters such as accuracy, precision, sensitivity, specificity and computational cost are analyzed to find the best algorithm. The data from the conducted study lasts a period of six months where the Cotopaxi Volcano is monitored between January and June 2012, conducted by the Instituto Geofísico de la Escuela Politécnica Nacional. Throughout the database used, two major events were identified: Long Period and Volcano Tectonic, which went to form two matrix, separate from each other, with the same number of elements, which we call training matrix and test matrix. Additionally, 79 features were used in time and frequency by the Fast Fourier Transforms and of Wavelet Transforms. The predictors that were analyzed were: k -Nearest Neighbors, Decision Trees and Neural Networks, the same ones that by using the full features and the training matrix obtained a model each. With the test matrix, performance parameters for each model was obtained. Finally, it was put into effect, a selection of features to find those having greater importance, determining that the first three features, the predictor model that presented best performance results was k -Nearest Neighbors.

Keywords:

- **CLASSIFICATION**
- **PREDICTOR**
- **MACHINE LEARNING**
- **K-NEAREST NEIGHBORS**
- **DECISION TREES**
- **NEURAL NETWORKS**

CAPITULO 1

INTRODUCCIÓN

1.1. Planteamiento del problema

La identificación de una posible erupción volcánica en tiempo real da la pauta para la preservación de la vida humana y por consiguiente la mitigación de los posibles riesgos y peligros ante el suceso del evento en mención.

Si bien es cierto, Ecuador cuenta con una gran biodiversidad de climas, especies vegetales y animales y una gran variedad de recursos naturales que representan un gran atractivo. Sin embargo, también se debe considerar que existen grandes amenazas precisamente de tipo natural, entre ellas se encuentra la Cordillera de los Andes que forma parte del Cinturón de Fuego del Pacífico en donde se asientan varios volcanes activos (IGEPN, 2010). Es evidente que los pobladores que viven en las zonas aledañas a estas elevaciones, específicamente volcanes activos, constantemente están expuesto al riesgo de una eminente erupción.

En los últimos tiempos, volcanes como el Reventador, el Cotopaxi y el volcán Tungurahua han presentado una gran actividad que ha ido desde la caída de piroplásticos, pasando por la emisión de ceniza y sismos, sobre todo el volcán Tungurahua. Estos comportamientos volcánicos han provocado la emisión de varias alertas que desembocaron con la evacuación de los pobladores asentados en las faldas y alrededores sin que hasta la presente fecha hagan erupción.

Al enfocarse en la predicción de los eventos volcánicos, la comunidad científica ha desarrollado varios trabajos de monitorización (Allen, 2006),

(Nassery, 1997), (Song, 2009), los cuales han empleado diversos algoritmos que van desde la determinación de umbrales (Song, 2009), pasando por enfoques Bayesianos (Tan, 2010) hasta el desarrollo de algoritmos de redes neuronales (Iyer, 2011), todos ellos con el único propósito de reducir al mínimo las falsas alarmas.

El Instituto Geofísico de la Escuela Politécnica Nacional tiene la misión "*Contribuir a través del conocimiento de las amenazas sísmicas y volcánicas a la reducción de su impacto negativo en el Ecuador, mediante la vigilancia permanente, la investigación científica, la formación académica de alto nivel y el desarrollo y aplicación tecnológica promoviendo la creación de una cultura de prevención*.". Es por ello que el Instituto en mención ha implementado en el volcán Cotopaxi, sendos controles que incluyen: la monitorización sísmica, geoquímica, visual, entre otras. Los citados controles en su mayoría son sistemas tradicionales de alto costo y que dependen de la inspección humana y de la habilidad de sus sentidos.

Adicionalmente y ratificando el inconveniente económico, no se ha podido desplegar controles, con el mismo nivel de detalle que se ha desplegado en el volcán Cotopaxi, en el resto de volcanes activos del Ecuador, ya sea por la falta de equipos especializados, infraestructura o personal que se encargue de la monitorización de los eventos volcánicos.

1.2. Justificación

Una de las políticas y lineamientos determinados en el Plan Nacional del Buen Vivir 2013-2017 es mejorar los sistemas de control y alerta temprana, monitorización y atención oportuna a la población, para identificar y mitigar las amenazas y vulnerabilidades sociales y ambientales ante los riesgos naturales y antrópicos (SENPLADES, 2009). Es por ello que la Secretaría de Gestión de

Riesgos emplea sus recursos con el fin de identificar, analizar, prevenir y mitigar riesgos para enfrentar y manejar eventos de desastre (SNGR, 2010), entre los cuales se encuentran los posibles desastres causados por las erupciones volcánicas.

Las investigaciones dentro del campo de la monitorización volcánica han sido de gran relevancia y han determinado que el empleo de redes de sensores inalámbricos representa una solución robusta como una solución global. Cabe mencionar que esta red se encuentra compuesta por varios bloques de funcionamiento, entre los cuales se encuentran los algoritmos de clasificación de eventos basados en *machine learning* (Nilsson, 1998), (Smola, 2008).

Al contar con varias opciones para escoger un algoritmo de clasificación de eventos y considerando que cada volcán presenta su comportamiento particular, es necesario un análisis de las variables involucradas. Parámetros de desempeño como el gasto computacional, exactitud (Ex), precisión (Pr), sensibilidad (Se) y especificidad (Es) pueden contribuir para seleccionar el mejor algoritmo que se adapte a las necesidades específicas del volcán Cotopaxi. Dicho algoritmo puede ser incorporarlo a una solución de bajo costo, insertándolo en una red de sensores inalámbricos para generar una alerta temprana, que permita la toma de decisiones oportunas de las autoridades pertinentes. La Universidad de las Fuerzas Armadas – ESPE ha contribuido con varios trabajos que contribuyen a la prevención y generación de alertas como se detallan en (Jaramillo, 2014), (Lara-Cueva, 2014) y (Lara, 2015).

1.3. Objetivos

1.3.1. General

Analizar el desempeño de algoritmos de clasificación de eventos vulcanológicos basados en *machine learning,* para su empleo en sistemas de supervisión y alerta temprana del volcán Cotopaxi.

1.3.2. Específicos

- Determinar los algoritmos de clasificación de eventos aplicados a la monitorización volcánica.
- Implementar a nivel de simulación tres algoritmos de clasificación.
- Probar tres algoritmos de clasificación con información real para determinar su desempeño.
- Validar tres algoritmos de clasificación con información real.
- Determinar el/los mejor/es algoritmo/s de clasificación de eventos volcánicos para el volcán Cotopaxi.

1.4. Trabajos Relacionados

Varios han sido los trabajos realizados por la comunidad científica para resolver el problema de la monitorización volcánica utilizando redes de sensores inalámbricos en conjunto con los algoritmos de clasificación de eventos entre los cuales podemos destacar los siguientes:

En (Shimshoni, 2002) se realiza un estudio con 380 eventos sísmicos donde se propone una máquina de clasificación integrada (ICM) compuesta por una red neuronal entrenada para clasificar formas de onda sísmicas. Del

conjunto de pruebas utilizadas, este método pudo clasificar el 92% de eventos de manera correcta, concluyendo que esta red neuronal puede ser utilizada para manejar problemas de clasificación de gran dimensión.

En (Scarpetta, 2005) se basa en el estudio de una red neuronal supervisada con una arquitectura de multicapa. Se emplea características espectrales de las señales y atributos parametrizados de la forma de onda. Como resultado se obtuvo un 100% de clasificaciones correctas de señales de tipo Volcano Tectónicas. Los datos que fueron utilizados pertenecen a cuatro estaciones de la red de monitorización de la elevación conocida como Mt. Vesuvius.

En (Curilem, 2009) se describe el estudio realizado en el volcán Villarrica de Chile, en donde se utilizó una red neuronal multi-capa para clasificar tres tipos distintos de eventos de origen volcánico: largo período, tremor volcánico y señales de tremor energéticos. Adicionalmente se realizó una optimización, mediante la selección de características más relevantes, obteniéndose como resultado más del 93% de exactitud en la identificación de señales de cada tipo.

En (Sharma, 2010) se menciona la variedad de algoritmos que existen y que van desde un simple umbral de amplitud, pasando por métodos adaptativos y redes neuronales, todos ellos basados en la amplitud o en la energía de la señal en el dominio del tiempo o de la frecuencia. Adicionalmente denotan que ninguno de los algoritmos de detección es óptimo bajo todas las situaciones. Finalmente se recalca que la variable del tiempo es especialmente crítica cuando existen altas tasas ya que puede existir el caso en el que un evento sea detectado como positivo, sin embargo, el sistema está guardando datos actuales procesados.

En (Iyer, 2011) se obtuvieron datos volcánicos para entrenar y probar el clasificador neuronal que fue desarrollado para distinguir la actividad volcánica de tres volcanes: Mount St. Helens-USA, Tungurahua-Ecuador, and Kasatochi-Alaska. Los resultados finales obtuvieron un algoritmo neuronal capaz de distinguir la actividad eruptiva de cada uno de los tres volcanes con una tasa de clasificación positiva de 97%.

Si bien el estudio efectuado en el monte Mt. Vesuvius alcanza un 100% de efectividad, el alcance es únicamente para señales de tipo volcano tectónicas. El resto de estudios sitúan su efectividad ente el 92 y 97% utilizando un solo clasificador. Este trabajo presenta un análisis de tres clasificadores distintos, los cuales trabajan con dos tipos de señales volcánicas, largo periodo y volcano tectónicas, con resultados superiores al 97% de efectividad.

1.5. Organización del Documento

El presente capítulo presenta una breve introducción del planteamiento del problema, su motivación y correspondiente justificación. Adicionalmente se puede encontrar el objetivo principal del estudio de investigación, así como los objetivos específicos del mismo. Finalmente se encuentra el estado del arte, el cual detalla varios trabajos relacionados con clasificación de eventos vulcanológicos.

El Capítulo II es dedicado enteramente a las máquinas de aprendizaje basadas en *machine learning*. Aquí se cita varias definiciones de algunos libros relacionados con el tema, las características que presenta una máquina de aprendizaje y los pasos que se deben seguir para la implementación de una de ellas. Adicionalmente se describen los tipos de máquinas, las cuales son divididas en algoritmos de tipo supervisado y no supervisados. La primera de

estas agrupaciones se divide en clasificación, la cual consiste en predecir la instancia a la que pertenece un determinado dato, y regresión, en donde el objetivo es predecir un valor numérico. Las técnicas no supervisadas encuentran valores estáticos para describir un dato. Por ello, todo el estudio se centrará únicamente en las técnicas supervisadas de clasificación.

El Capítulo III describe las herramientas que se van a emplear para el desarrollo de la investigación, del mismo modo se hace mención al origen de los datos con los cuales se pretende trabajar. Finalmente, se realiza una corta descripción de la metodología que se va a emplear, así como también, se detalla brevemente el procedimiento que se va a seguir para obtener los resultados propuestos.

El Capítulo IV en su primera parte se enfoca en el pre procesamiento de las señales, es decir, la obtención de las características en tiempo y frecuencia, la selección de los mejores algoritmos, la organización de los datos y la medición de los parámetros de desempeño que se van a utilizar para evaluar las máquinas de aprendizaje. En la segunda parte del capítulo se obtienen los resultados bajo las condiciones normales de funcionamiento, es decir, con el empleo de todas las características obtenidas en tiempo y frecuencia. Finalmente en la última sección se efectuó una selección de características con el fin de determinar aquellas que presentan mayor relevancia al momento de realizar la clasificación.

El Capítulo V contiene las conclusiones a las que se llegó una vez que se hizo el análisis de resultados obtenidos en el anterior capítulo. Adicionalmente se hace mención a los trabajos futuros que podrían realizaras a partir de la finalización del presente trabajo de investigación.

CAPÍTULO 2

FUNDAMENTO TEÓRICO

2.1 Introducción a *Machine Learning*

2.1.1 Definición

En (Witten, 2005) se define a *machine learning* como el aprendizaje a través de la observación a partir de un comportamiento pasado. A pesar de que esta definición se enfrasca en una discusión filosófica sobre el concepto de aprendizaje, ya que a decir del autor, el término entrenamiento estaría mejor aplicado cuando se refiere a una máquina o un ser que no puede pensar por sí solo.

En (Nilsson, 1998) *machine learning* es catalogado como los cambios en un sistema que mejora las tareas asociadas. Las tareas pueden envolver el reconocimiento, diagnóstico, planificación, control, clasificación, etc. De igual manera el autor centraliza su discusión alrededor de la palabra aprendizaje, la misma que la asocia a estudios realizados por psicólogos y zoólogos.

En (Michie, 1994) a *machine learning* se lo toma como los procedimientos automáticos de computación basados en las operaciones lógicas o binarias, que aprenden una tarea de una serie de ejemplos. Adicionalmente recalca la importancia de las expresiones, estas deben ser lo suficientemente compresibles para el ser humano.

2.1.2 Características

Machine learning tiene la capacidad de convertir los datos en información. Sus aplicaciones pueden estar involucradas en la informática, ingeniería, estadística y varias otras disciplinas. Cualquier campo, en donde se necesita de la interpretación y acción a partir de datos, puede beneficiarse de la aplicación de las técnicas de *machine learning*. Dos tipos de variables participan en el desarrollo dentro de la ejecución de los algoritmos: a) características o atributos, que son aquellas que se van a medir, y b) la variable objeto, que es aquella que se trata de predecir.

En razón que existen varias clases de algoritmos, con distintas características y usos, el primer paso para empezar, es escoger el algoritmo de aprendizaje. El segundo paso consiste en que el algoritmo debe ser entrenado con un conjunto de datos de alta calidad conocidos como el conjunto de entrenamiento. El algoritmo basado en *machine learning* aprende de la búsqueda de alguna relación entre los atributos y la variable objeto.

En (Harrington, 2012) las técnicas de aprendizaje son clasificadas en dos grandes grupos denominados: técnicas de aprendizaje supervisadas y no supervisadas. La primera de estas agrupaciones se divide en dos grandes partes: la primera es de clasificación, la cual consiste en predecir la instancia a la que pertenece un determinado dato, mientras que la segunda tarea se denomina de regresión, en donde el objetivo es predecir un valor numérico. El nombre de técnicas supervisadas es otorgado debido a que estos enfoques conocen lo que van a predecir.

Por otra parte, las técnicas no supervisadas no contienen etiquetas o valores objetivos dados para los datos y entre sus tareas se determina la necesidad de encontrar valores estáticos para describir un dato, esto es

conocido como estimación de la densidad, otra tarea de este grupo es la reducción de datos de un gran número a un número menor de características.

Para probar los algoritmos basados en *machine learning*, usualmente se separan dos grupos de datos: la primera asociación es denominada conjunto de datos de entrenamiento, mientras que el segundo es llamado conjunto de prueba. En principio, los datos de entrenamiento son ingresados al algoritmo, una vez que se encuentra adiestrado, se alimenta al algoritmo con el conjunto de pruebas, este grupo contiene los datos que son objeto de la investigación.

2.1.3 Desarrollo de Aplicaciones

Para el desarrollo de una determinada aplicación, en primer lugar, el investigador debe seleccionar el algoritmo de aprendizaje más idóneo que se ajuste a las necesidades particulares del estudio, de ahí que se puede considerar los siguientes aspectos:

- Considerar el objetivo. La meta que se quiere obtener es una predicción, una clasificación o una estimación
- Considerar el valor objeto. Para el caso de clasificación consiste en las diferentes clases en las que se encajaran los valores clasificados.
- Conocer los datos. Si los valores son nominales o continuos, si existen valores que no están en los atributos, si existen valores atípicos en los datos o si los eventos a buscar ocurren con muy poca frecuencia.

Una vez que se ha seleccionado el algoritmo más adecuado, aquel que se ajuste a las características específicas del trabajo de investigación, es recomendable tener en cuenta los siguientes pasos para la implementación de una aplicación:

- Colectar los datos.- el origen de los datos es indiferente, se puede utilizar todo dato que pueda ser medido.
- Preparar los datos de entrada.- los datos deben estar en un formato que pueda utilizarse para poder manipularlos.
- Certificar los datos.- verificar que la fuente de los datos sea confiable.
- Analizar los datos de entrada.- verificar que los datos no tengan algún patrón fácilmente visible, ubicar valores vacíos o ubicar valores atípicos serían las tareas principales.
- Entrenar el algoritmo.- alimentar al algoritmo para extraer la información
- Probar el algoritmo.- evaluar el algoritmo con la información obtenida en la predicción. En caso de que no se obtenga la precisión deseada, se puede optar por verificar el origen de los datos, su recolección y preparación.
- Utilizar la técnica.- Utilizar el algoritmo para la aplicación.

2.2 Algoritmos de Aprendizaje

Una vez que se ha descrito los diferentes tipos de algoritmos basados en *machine learning* y las consideraciones al momento de desarrollar una aplicación, se pertinente mencionar que el tipo que se utilizará para el presente trabajo es clasificación debido a las siguientes razones: se cuenta con las características o atributos (valores históricos detectados mediante métodos tradicionales de monitorización) y se conoce la variable objeto que se desea encontrar (se conoce el tipo de evento vulcanológico).

Adicionalmente, no se necesita predecir ningún valor, es decir, no es necesario los algoritmos de regresión. No se necesita encontrar valores estáticos para describir un dato ni tampoco se necesita la reducción de datos de un gran número a un número menor.

Por lo descrito, los algoritmos que se encuentran agrupados bajo el nombre de no supervisados, no serán utilizados para la tarea que representa este trabajo de investigación. Tampoco se necesita encontrar un determinado valor por lo que la regresión no será considerada.

Los algoritmos basados en machine learning que se van a emplear en el presente trabajo se investigación, pertenecen al grupo conocido como supervisados. Esta elección se debe a las particularidades de los datos, ya que se cuenta con las características y la variable que se requiere encontrar. Adicionalmente el trabajo que se realizará corresponde a una clasificación.

2.2.1 k-Nearest Neighbors (k-NN)

Las ventajas de este enfoque son: su alta precisión, la insensibilidad a los valores atípicos y la no suposición acerca de los datos. Sin embargo, también mantiene desventajas como un alto gasto computacional y necesita de una gran cantidad de memoria. Este algoritmo trabaja con valores numéricos y valores nominales.

Su funcionamiento es el siguiente: se parte de un conjunto existente de datos, el conjunto de entrenamiento. Se tienen etiquetas para todos estos datos y se conoce la ubicación de cada dato. Cuando al algoritmo ingresa un nuevo conjunto de datos, éstos son comparados con todos los conjuntos existentes. Una vez que se ha identificado los conjuntos de datos más parecidos, se toma los primeros *k* elementos cuyo valor mínimo puede ser definido como la raíz cuadrada del número de características, tal como se plante en (Duda, 2012). Finalmente el valor que se quería clasificar es asignado al conjunto que más se repite. En la Figura 1 se muestra un ejemplo del algoritmo en donde *k*=3.

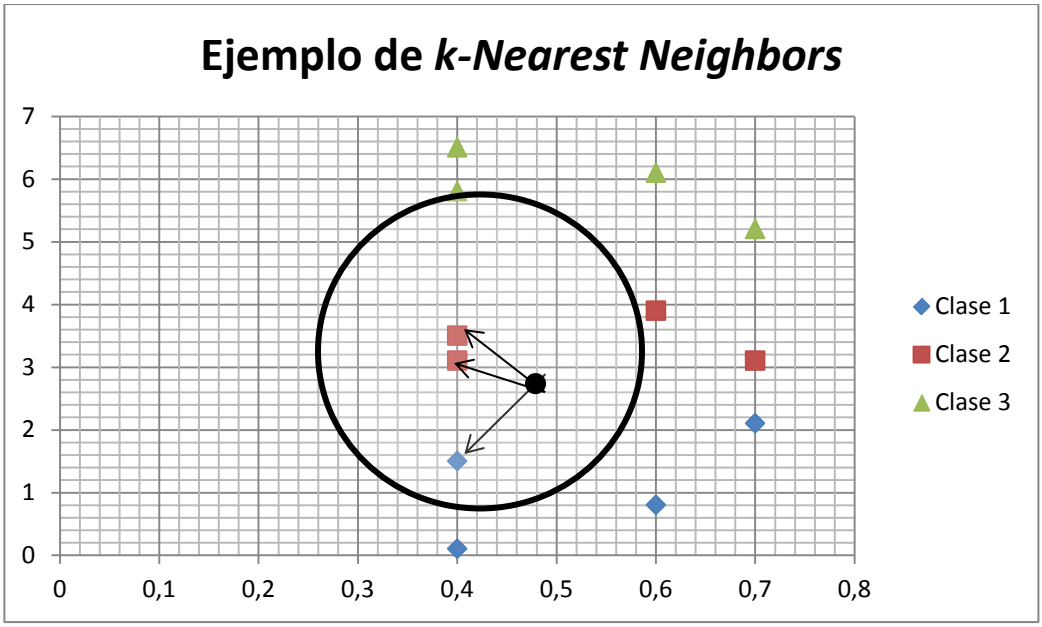


Figura 1. **Ejemplo de *k-Nearest Neighbors***

La selección del valor de k sin duda será determinante en los valores de desempeño del algoritmo, de este modo si k es muy pequeño puede ser sensible al ruido. En cambio de k es un muy grande los vecinos cercanos escogidos de seguro pertenecerán a otras clases. Para reducir las consecuencias de una mala elección del valor de *k*, se puede optar por la asignación de pesos a los objetos o características. El valor que se le asigne a cada peso corresponderá de forma directa a la distancia de los vecinos más relevantes.

2.2.2 Decision Trees (DT)

Es una técnica de clasificación muy sencilla y muy utilizada, entre sus ventajas está su bajo costo computacional y los resultados pueden ser fácilmente interpretados, adicionalmente este enfoque puede manejar características irrelevantes. Entre sus desventajas se puede mencionar que es propenso al sobreajuste, es decir el algoritmo puede sobre entrenarse desembocando en una respuesta exitosa con muestras de entrenamiento pero una clasificación pobre para muestras nuevas. Además no puede manejar valores atípicos y tiene inconvenientes cuando existen demasiadas ramificaciones.

En primer lugar se debe dividir el conjunto de datos a ser empleados en base a las características de los mismos. Para conseguir este fin se debe probar cada característica para obtener los mejores resultados. El resultado de esta prueba es la clasificación de los datos en conjuntos más pequeños. Estos subconjuntos deberán atravesar la primera rama de decisión y ubicarse correctamente, de no ser así se debe repetir el proceso de clasificación hasta que todos los datos hayan sido ubicados.

Para la obtención de los resultados esperados, es necesario cuantificar los datos para poder ordenarlos, esto es conocido como ganancia de la información. Esta ganancia es obtenida mediante la medición de la entropía (valor esperado de la información). En la Figura 2 se muestra un ejemplo del algoritmo en donde existen tres líneas de decisión.

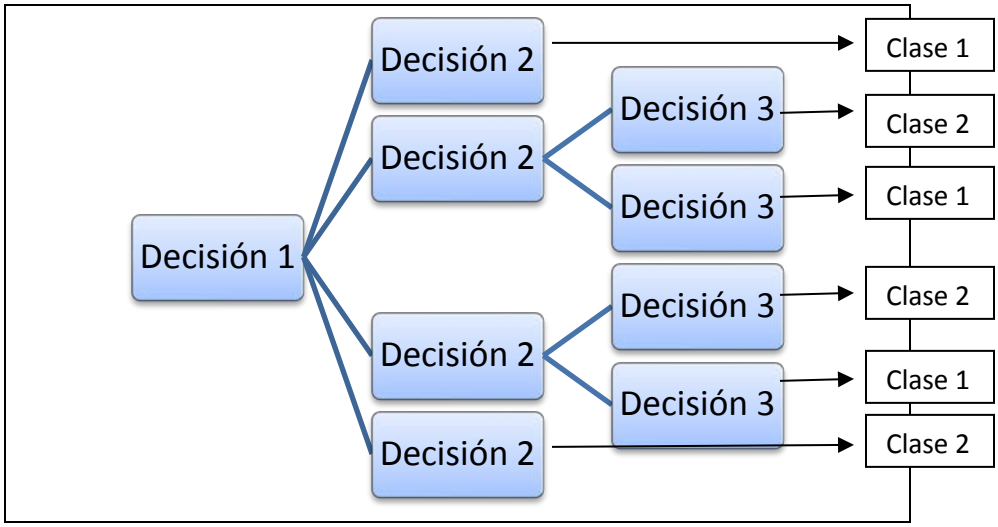


Figura 2. Ejemplo de *Decision Trees*

2.2.3 Neural Networks (NN)

En (McCaffrey, 2012) se refieren a un enfoque que trabaja con cualquier número de entradas para producir cualquier número de salidas, lo que lo hace sumamente útil. El alcance de esta técnica envuelve el reconocimiento de patrones, el mapeo de características, el agrupamiento y por su puesto la clasificación.

Existen varias definiciones como en (Haykin, 1994), es un proceso distribuido que trabaja de forma paralela y que tiene una propensión natural a almacenar conocimiento experimental para ser empleado, del mismo modo, (Nigrin, 1993) la describe como un circuito compuesto de un número muy grande de elementos de procesamiento simple en donde cada elemento opera de forma asíncrona y únicamente sobre la información local, finalmente (Zurada, 1992) lo define como un sistema físico celular el cual puede adquirir, almacenar y utilizar conocimiento experimental.

La estructura a ser utilizada se conoce como multicapa, la cual contiene una capa de entrada (características), una capa de salida (valores esperados) y una capa oculta en donde se debe determinar las “neuronas” que va a emplear el sistema, normalmente encontrar este número no obedece a un patrón específico, sino a la arquitectura del problema. En la Figura 3 se muestra un ejemplo del algoritmo en donde existen tres elementos en la capa de entrada, 5 elementos en la capa oculta y 2 elementos en la capa de salida.

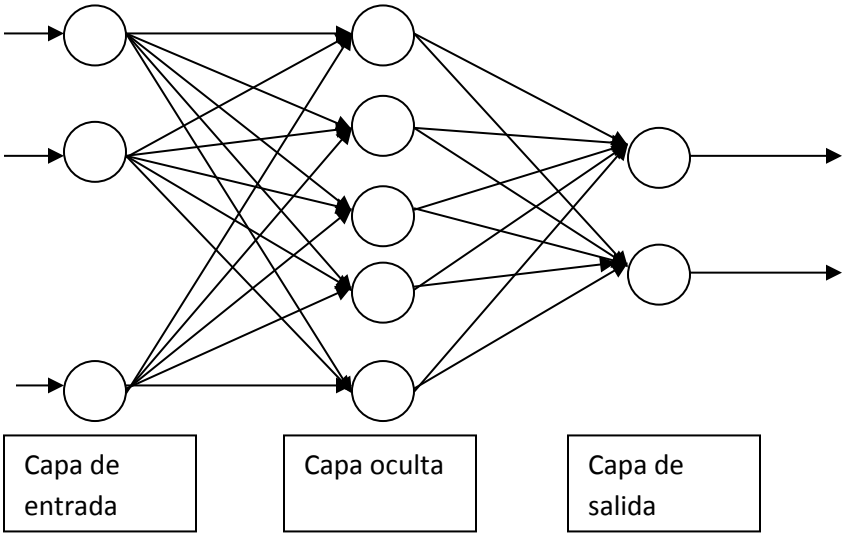


Figura 3. Ejemplo de *Neural Networks*

CAPÍTULO 3

MATERIALES Y MÉTODOS

3.1 Materiales

3.1.1 Ambientes de programación

De acuerdo al tipo de estudio que se plantea para el presente trabajo de investigación, es necesario establecer varios requisitos fundamentales que el lenguaje de programación debe cumplir, entre ellos, el que sea capaz de procesar señales que demanden un alto gasto computacional, claro sin dejar de lado: la legibilidad, facilidad de escritura, confiabilidad y costo, los mismos que son parte de los principales criterios al momento de escoger un lenguaje de programación, tal como se detalla en (Chiroque, 2012).

En (Harrington, 2012) se hace una breve comparativa entre cuatro lenguajes de programación de alto nivel que permitirían trabajar con *machine learning*, es decir, en la mayoría de los casos con grandes matrices. Entre estos lenguajes seleccionados se encuentran: Matlab, Python, Java y C.

Para el caso de C y Java menciona la existencia de librerías que podrían ser utilizadas para operaciones matemáticas, sin embargo, estos lenguajes perderían protagonismo al necesitar excesiva cantidad de código para poder ejecutar tareas que resultarían sencillas. Esto sin duda tendría repercusión directa al momento de medir el gasto computacional, ya que este vendría a ser uno de los parámetros de rendimiento a ser tomado en cuenta en el desempeño de los algoritmos.

Para el caso de Python, se lo menciona como un lenguaje de sintaxis clara, fácil de utilizarlo y por este sentido, mucha gente y organizaciones lo emplean, generando de este modo un amplio desarrollo y documentación. También utiliza librerías para operaciones con matrices y vectores conocidas como SciPy y NumPy y la librería Matplotlib para poder graficar en 2D y 3D. Finalmente se puede señalar que es un código abierto, no se necesita comprar licencias para su uso.

El último lenguaje es Matlab, a quien se lo menciona como de alto nivel con funciones integradas que permiten la implementación de algoritmos basados en *machine learning* de una manera más fácil y rápida. Sobre este punto, Matlab tendría gran ventaja sobre C y Java ya que estos dos últimos necesitaban de un gran gasto computacional para poder ejecutarse.

El único inconveniente con Matlab, el cual representa una desventaja ante Python, es que para ser utilizado necesita de una licencia, la misma que tiene un valor monetario. A pesar de ello, Matlab es el software líder desarrollado por MathWorks, el mismo que es utilizado por científicos e ingenieros para el cálculo matemático a nivel mundial.

En este sentido, Matlab cumpliría con cuatro de los cinco parámetros de selección de lenguajes: la legibilidad, facilidad de escritura, confiabilidad y gasto computacional, el único criterio que no cumple es el costo o valor monetario, claro está, que esto es considerando que no es un lenguaje de programación de carácter gratuito.

Para la programación y ejecución de los algoritmos se empleará un computador con las siguientes características de software y hardware:

- Sistema operativo Windows 7 Ultimate Copyright ©2009 Microsoft Corporation Service Pack 1
- Procesador Intel (R) Core (TM)2 Duo CPU T5800 @ 2.00 GHz 2.00 GHz
- Memoria RAM 4,00 GB
- Sistema operativo de 64 Bits
- Disco Duro de 300 GB con el 80% de capacidad utilizada

El lenguaje de programación empleado tiene las siguientes características: Matlab R2013a cuya versión es (8.1.0.604), la arquitectura es de 64-bit para sistema operativo Microsoft Windows. La fecha de la versión es febrero 15 de 2013 con su anuncio de Copyright 1984-2013 The MathWorks, Inc. Protected by U.S and International patents.

3.1.2 Base de datos de eventos reales

3.1.2.1 Clasificación de señales

Las señales sísmicas de origen volcánico pueden ser clasificadas por su contenido espectral, las frecuencias en las que tienen mayor presencia, sus fases y de todas estas características se puede desprender el fenómeno natural que provoca dicho movimiento. En (Cardenas, 2014) se propone la clasificación más utilizada para estas señales, la cual se detalla de la siguiente manera:

i. Volcano Tectónico

Existe movimiento paralelo al plano donde la duración de la señal es de carácter variable que va desde segundos hasta minutos para los movimientos más grandes. Presenta dos fases de onda, la primaria (P) y la secundaria (S). El rango de frecuencias se encuentra entre 5 - 15 Hz con un contenido

energético es considerable, a pesar que se han registrados casos en los que ha llegado hasta 30 Hz.

Presenta su actividad en una banda ancha de frecuencia con un amplio rango de profundidades que va desde decenas de kilómetros hasta la misma superficie. Adicionalmente se puede mencionar que este tipo se presente como enjambres sísmicos.

ii. Largo Período

Estos sismos no tienen fases, lo que torna complejo determinar el inicio de la señal. Su duración va desde segundos hasta un poco más de un minuto. El contenido espectral es limitado dentro de una banda de frecuencias estrechas.

El rango de frecuencias se encuentra entre 0.5 - 5 Hz con pico que se presentan entre 1 - 3 Hz. Su origen involucra interacción con fluidos volcánicos, lo que permite obtener información del estado interno del volcán.

iii. Híbridos

Este tipo de sismos comparte las características de eventos de largo período y volcano tectónicos. Comienza con señales de amplia banda espectral > 10 Hz que se interpreta por una fractura, seguida por una señal similar al largo período.

El espectro de estas señales se los puede dividir en dos regiones, la primera contenida en una banda ancha de frecuencias que van desde los pocos Hz hasta los 15 Hz. La segunda fase corresponde a bajas frecuencias con picos entre 0.5 y 3 Hz.

iv. Tremor volcánico

Esta señal sísmica es de amplitud constante y larga duración que puede ir de minutos hasta días. Presenta una banda de información estrecha. Adicionalmente se puede mencionar que tiene varias divisiones que se diferencian por la frecuencia y su origen.

v. Explosiones Volcánicas

Generan dos tipos de ondas, la primera asociada a la propagación en formas de ondas internas. La segunda onda conocidas como de aire, sonoras o de choque. Este tipo, junto a los tremores están presentes cuando ocurre un evento eruptivo. Regularmente se encuentran superpuestos a una señal de tremor.

3.1.2.2 Identificación de eventos reales

La base a utilizarse fue proporcionada por el Instituto Geofísico de la Escuela Politécnica Nacional (IGEPN). El volcán de estudio es el Cotopaxi, cuyas características se pueden encontrar en (IGEPN, 2015). La red de monitorización de este volcán es una de las más completas del Ecuador, teniendo como característica histórica que en 1976 se construyó la primera estación permanente de vigilancia en Sudamérica.

La información entregada por el IGEPN corresponde al periodo de 2009-2010 con un total de 914 eventos, los cuales se encuentran divididos en:

- 759 eventos de tipo Largo Período.
- 116 eventos de tipo Volcano Tectónico

- 30 eventos tipo Hibrido
- 9 eventos de tipo Tremor Volcánico

3.2 Métodos

3.2.1 Metodología de investigación

Para este trabajo de investigación se utilizará el método experimental en razón que se procederá a manipular la variable independiente (Datos reales de eventos vulcanológicos del volcán Cotopaxi) para observar su efecto sobre la variable dependiente (Algoritmos basados en *machine learning* utilizados para clasificación de eventos vulcanológicos) mediante experimentos controlados, posterior a lo cual se procedió a evaluar los parámetros de desempeño de cada predictor (exactitud, precisión, sensibilidad, especificidad y gasto computacional). (Zhu, 2010)

3.2.2 Recolección de la información

Para la obtención de información referente a la variable dependiente, es decir, los eventos del volcán Cotopaxi se recurrió al IGEPN, en razón que es el organismo especializado en el tema, el cual se encuentra encargado de la monitorización permanente de este y todos los volcanes presentes en el territorio ecuatoriano.

3.2.3 Procesamiento de la información

Previo al entrenamiento de los algoritmos de clasificación de eventos basados en *machine learning* aplicados a la monitorización volcánica, se debe efectuar un pre-procesamiento de las señales para determinar las mejores condiciones previo al entrenamiento de los mismos.

Para determinar el modelo de clasificación se toma una pequeña muestra de los eventos históricos del volcán Cotopaxi detectados por medios tradicionales, para que sean procesados por los algoritmos basados en *machine learning*. Una vez que los algoritmos se encuentren correctamente entrenados, es decir, se cuenta con un modelo predictor, la restante información histórica, se empleará para clasificar a cada uno de los eventos que se encuentran en la base de datos.

3.2.4 Evaluación de resultados y validación

Con los datos originales del volcán Cotopaxi, se tiene identificado a cada uno de los eventos, de acuerdo a las clasificación descrita en (Cardenas, 2014). Este trabajo fue realizado por el IGEPN, en tal sentido, se podría decir que la integridad de la información está garantizada. Las etiquetas de cada evento fueron colocadas por expertos vulcanólogos, por ello los resultados obtenidos de la investigación son confrontados con la información proporcionada.

Finalmente, una vez que los algoritmos hayan predicho los valores, se procederá con el análisis comparativo de los resultados mediante curvas que determinen el/los mejor/es algoritmos/s de clasificación de eventos volcánicos aplicados al volcán Cotopaxi, mediante parámetros de desempeño que están definidos en función de eventos catalogados como falsos positivos (FP), falsos negativos (FN), verdaderos positivos (VP) y verdaderos negativos (VN) (Zhu, 2010).

3.2.5 Medición de parámetros de desempeño

Los parámetros de desempeño utilizados para la medición de los resultados son: exactitud (Ex), precisión (Pr), sensibilidad (Se), especificidad

(Es) y gasto computacional. En (Zhu, 2010) se define cada uno de los mismos, los cuales están detallados en función de la Tabla1:

Tabla 1.

Términos para definir parámetros de desempeño

	Positivos	**Negativos**
Positivos	VP	FP
Negativos	FN	VN

$$Se = \frac{VP}{VP + FN}$$

$$Es = \frac{VN}{VN + FP}$$

$$Pr = \frac{VP}{VP + FP}$$

$$Ex = \frac{VN + VP}{VP + VN + FN + FP}$$

Adicionalmente a los parámetros de desempeño expuestos, falta mencionar el gasto computacional, el cual será medido en segundos y contemplará el tiempo empleado únicamente para el proceso de clasificación, es decir, se excluye del cálculo el pre procesamiento y la determinación del modelo predictor.

CAPÍTULO 4

IMPLEMENTACIÓN Y ANÁLISIS DE RESULTADOS

4.1 Manejo de la información

4.1.1 Obtención de características en tiempo y frecuencia

El pre-procesamiento, así como la obtención de las características de las señales de origen volcánico tanto en tiempo como en frecuencia se encuentra detallado en (Saltos, 2014). Aquí se detalla un filtrado de la señal para la remoción de errores provenientes de grietas llenas de fluido, así como la supresión de la media y tendencia lineal de las señales.

A continuación se detallan las características obtenidas en el dominio del tiempo, frecuencia y escala mediante la Transformada Rápida de Fourier y la Transforma Wavelet. En total existirán 79 características para cada uno de los eventos constantes en la base de datos, las cuales se encuentran detalladas en las tablas 2, 3 y 4.

Tabla 2

Características en el dominio del tiempo

TIEMPO	Duración
	Tiempo de Alcance Pico Máximo
	Pico Máximo
	Entropía
	Valor RMS
	Valor de Pico a Pico
	Relación Pico a RMS
	Energía en Tiempo
	Densidad de Cruces por Cero

CONTINUA →

	Kurtosis
	Densidad de Número de Picos sobre RMS

Fuente: (Saltos, 2014)

Tabla 3.

Características en el dominio de la frecuencia con FFT

FFT	Valor Pico FFT
	Frecuencia de Pico FFT
	Valor de Media FFT
	Valor en Umbral 10 Hz – 20 Hz
	Frecuencia Umbral Máxima (10 – 20 Hz)
	Valor en Umbral 20 Hz – 30 Hz
	Frecuencia Umbral Máxima (20 – 30 Hz)
	Valor RMS en FFT
	Relación de Pico a RMS en FFT
	Energía en FFT
	Densidad de Número de Picos sobre RMS
	Segundo Pico Máximo en FFT
	Frecuencia del segundo pico
	Tercer Pico Máximo en FFT
	Frecuencia del tercer pico
	Valor Máximo y Tiempo en pico de PSD
	Frecuencia en pico de PSD
	Valor Max en FFT en 6 niveles
	Frecuencia en pico FFT en 6 niveles
	Frecuencia media en FFT en 6 niveles

Fuente: (Saltos, 2014)

Tabla 4.

Características en el dominio de la frecuencia con WAVELET

WAVELET	Energía en *Wavelet*
	Porcentaje de Energía en 6 niveles
	Valor RMS en 6 niveles
	Valor de pico a pico en 6 niveles
	Relación de pico a RMS en 6 niveles

Fuente: (Saltos, 2014)

4.1.2 Elección de algoritmos

La selección de los algoritmos se enfoca en aquellos conocidos como supervisados, cuya tarea es la de predecir la instancia a la que pertenece un determinado dato o número, es decir, los clasifican en un grupo específico.

Por las características de cada uno de ellos, ventajas y desventajas que representa su implementación y sobre todo por la relación precisión versus la facilidad de implementación (gasto computacional), se ha determinado que los tres algoritmos que van a ser ejecutados son:

- *k-Nearest Neighbors*
- *Decision trees*
- *Neural Networks*

4.1.3 Organización de datos

En primer lugar se organizaron los eventos de Largo Período y Volcano Tectónico en dos matrices por separado para analizar su media y desviación estándar, obteniéndose los siguientes resultados:

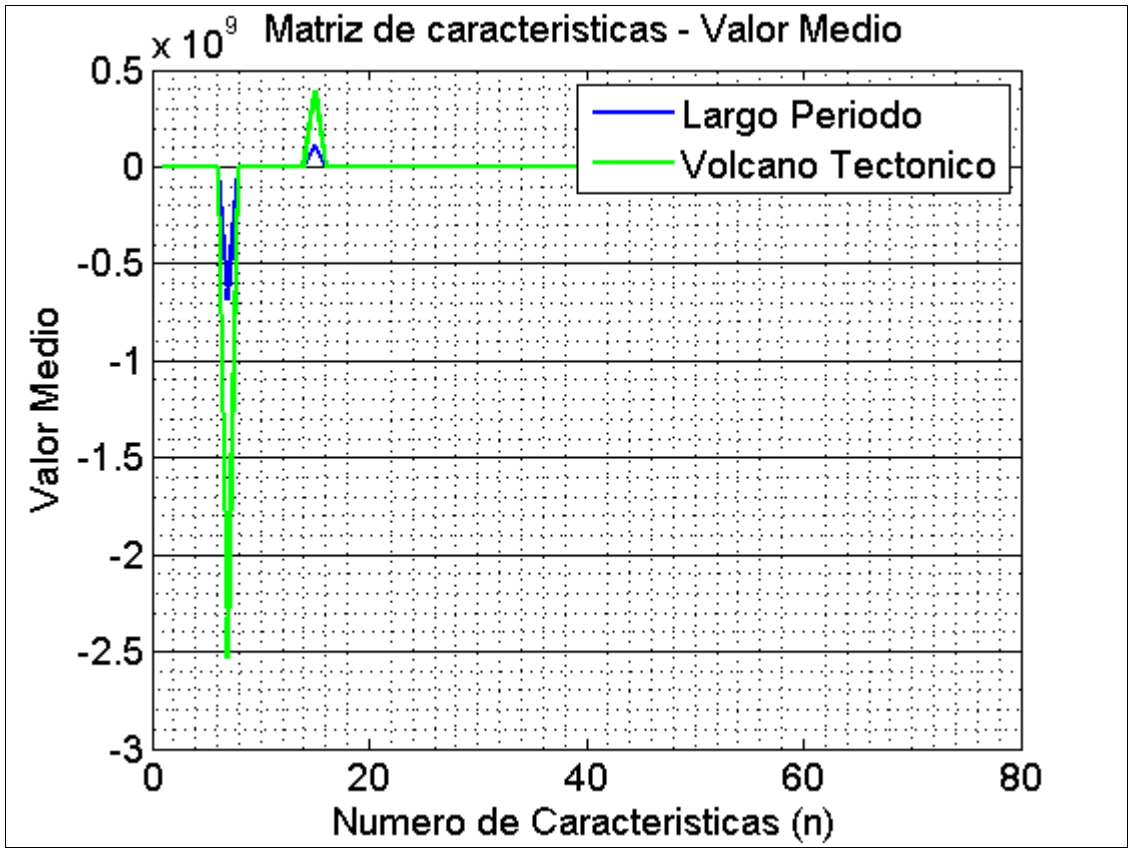


Figura 4. Matriz de Características – Valor Medio

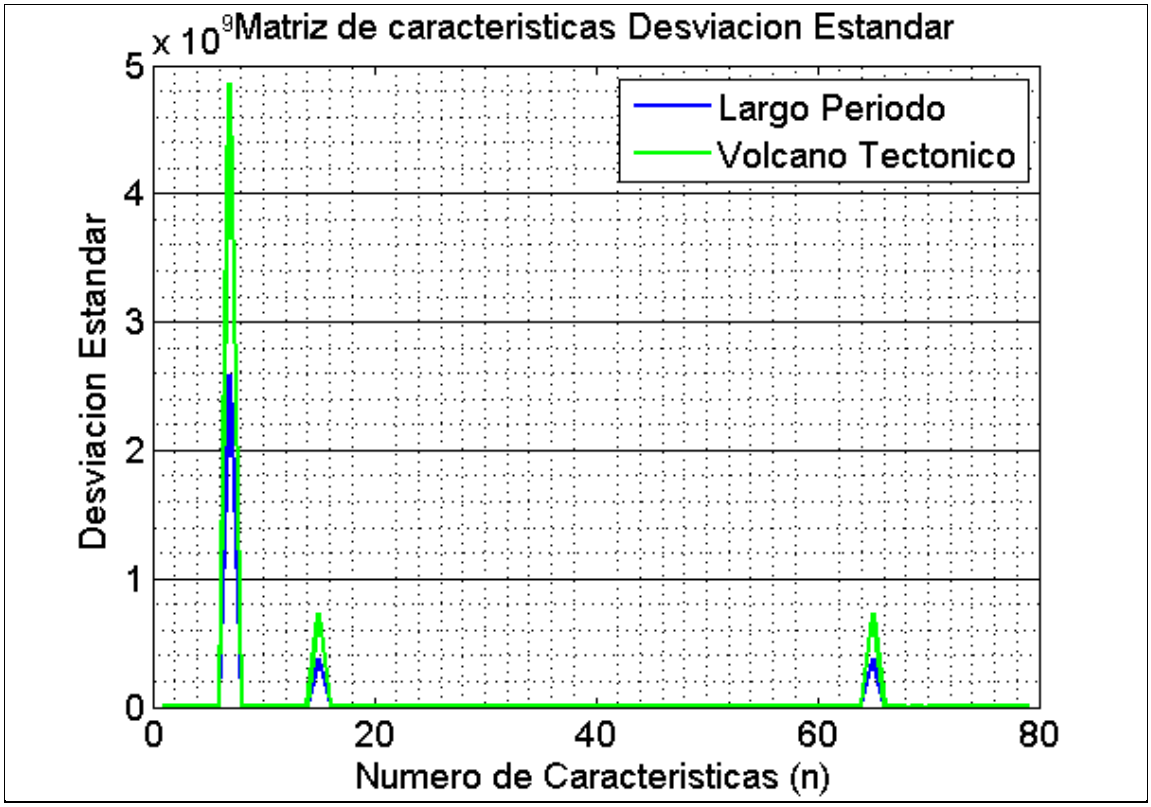


Figura 5 Matriz de Características – Desviación Estándar

Una vez efectuado el cálculo se puede apreciar que en la Figura 4 y Figura 5 las características pertenecientes a los eventos de Largo Período y Volcano Tectónico presentan valores muy similares, por lo que se procede a normalizar los vectores correspondientes a cada una de las características. Los resultados del valor medio y desviación estándar para este caso son los siguientes:

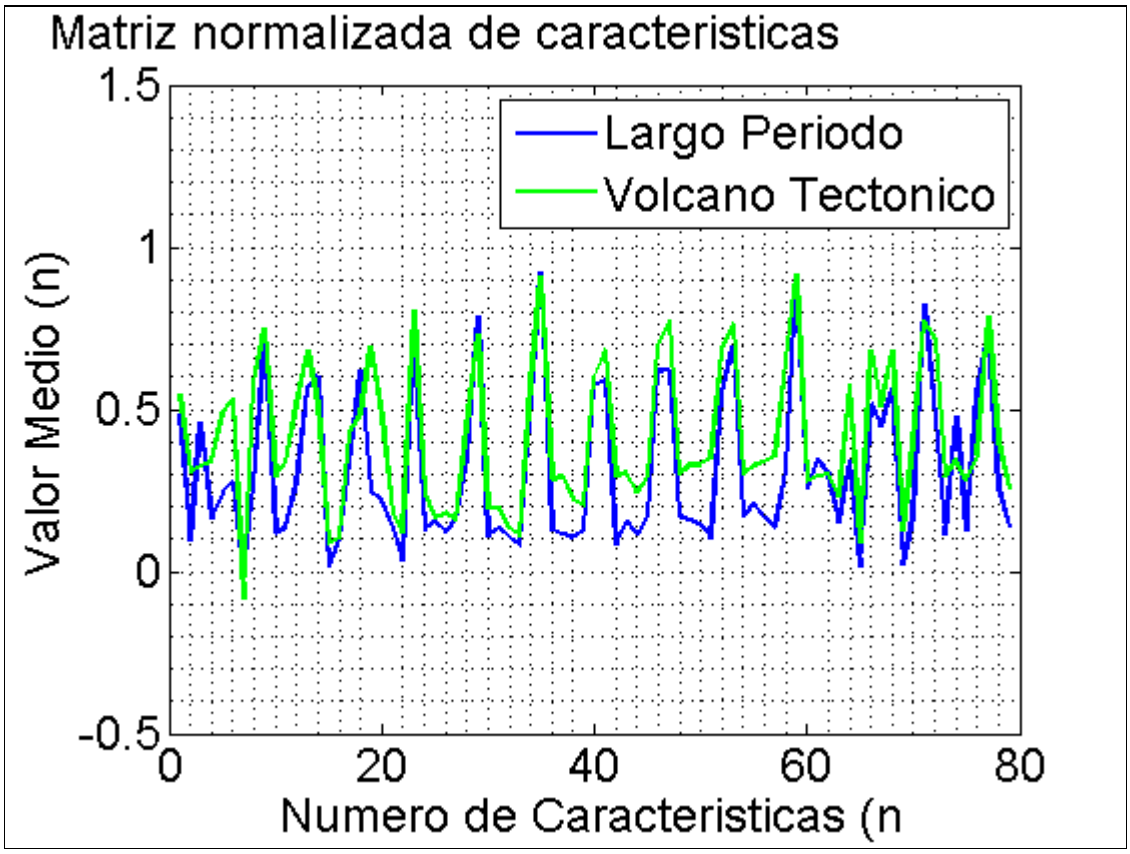


Figura 6. Matriz Normalizada de Características – Valor Medio

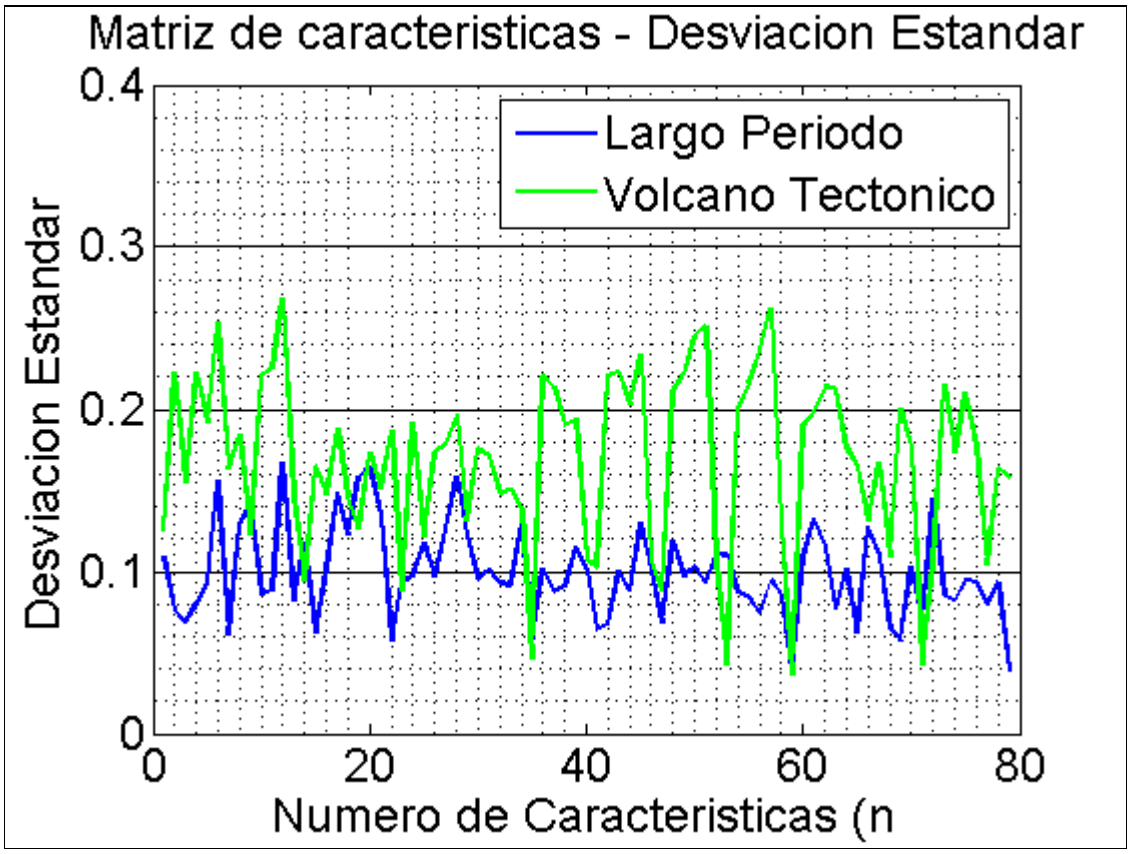


Figura 7. Matriz Normalizada de Características – Desviación Estándar

En la Figura 6 y Figura 7 se pueden apreciar visiblemente que existe una diferencia entre las características de los eventos de Largo Período y Volcano Tectónico. Por ejemplo, en la Figura 6, se pude apreciar que el valor de la característica 19, puede ser empleado para diferenciar claramente a los eventos de origen volcánico antes descritos.

Como se ha descrito en los párrafos anteriores, es conveniente trabajar con los valores normalizados de cada uno de los eventos, por lo que, todo cálculo que se efectúe a partir de este punto y toda conformación de grupos o matrices se lo hará con valores normalizados.

De los 914 eventos detectados, se procederá a trabajar con los 759 eventos de tipo Largo Período y los 116 eventos de tipo Volcano Tectónico, de tal manera que se formen dos matrices completamente independientes una de la otra, es decir, ningún evento estará presente en ambas matrices.

En razón que se cuenta con 116 eventos Volcano Tectónicos, este será el número máximo de elementos de cada matriz. En este sentido la primera matriz estará conformada por 58 elementos Volcano Tectónicos y 58 elementos de Largo Período. Esta matriz servirá para entrenar el algoritmo y determinar el modelo predictor.

La segunda matriz, tendrá el mismo número de elementos de la primera matriz, sin embargo, esta servirá para probar el modelo encontrado con anterioridad. Cada una de las matrices tendrá la dimensión de (116x79) ya que son 116 eventos de origen volcánico a los cuales se les extrajo 79 características en tiempo, frecuencia y escala.

Para mantener una total independencia entre la matriz de entrenamiento y de pruebas, se tomaron los primeros 58 elementos Volcano Tectónicos para entrenamiento y los siguientes 58 valores para la matriz de pruebas. Del mismo modo para eventos de Largo Período, se escogieron los 58 valores iniciales para la primera matriz y los 58 últimos valores para la segunda matriz.

El flujo de funcionamiento del estudio, se encuentra representado por la siguiente figura:

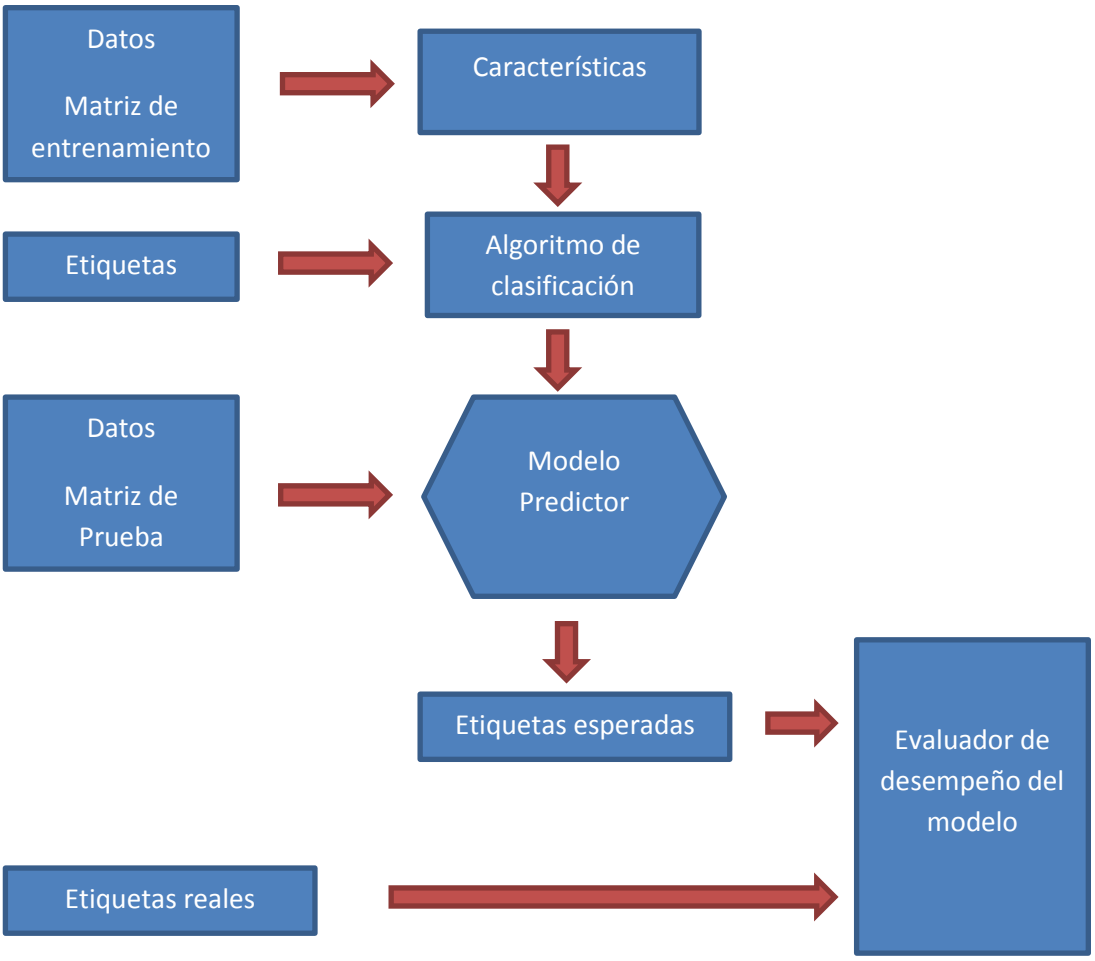


Figura 8. Flujo de trabajo de un modelo de aprendizaje supervisado

En la Figura 8, se puede apreciar que el proceso empieza con la matriz de entrenamiento, en donde se tiene identificadas a las características de cada evento. Del mismo modo cada tipo de sismo se encuentra etiquetado con su correspondiente nombre. Una vez que se cuenta con estos dos insumos, se desarrolla el clasificador, el cual se halla ajustado a los parámetros de la matriz de entrenamiento.

Posterior a ello, la matriz de pruebas ingresa en el modelo previamente creado con el afán de predecir las etiquetas de cada uno de los eventos que conforman la matriz. Esta predicción es comparada con las etiquetas reales

para determinar el desempeño del clasificador (exactitud, precisión, sensibilidad, especificidad y gasto computacional).

4.2 Resultados obtenidos

4.2.1 k-Nearest Neighbors

Siguiendo este modelo, el parámetro libre de configuración es el conocido como *k,* el cual determina el número de vecinos más cercanos. Si *k* tiene un valor muy bajo, los resultados pueden ser sensibles al ruido, en su lugar, si *k* es demasiado extenso, los vecinos pueden incluir puntos de otras clases (Wu, 2008).

En el presente trabajo de investigación, se implementó el algoritmo k-NN con las mismas matrices de entrenamiento y prueba, al ser *k* el único parámetro libre a definir, se modificó el valor de *k* desde 1 hasta 79, para obtener la gráfica de desempeño del predictor, una vez ingresada la matriz de pruebas, tal como se muestra en la Figura 9.

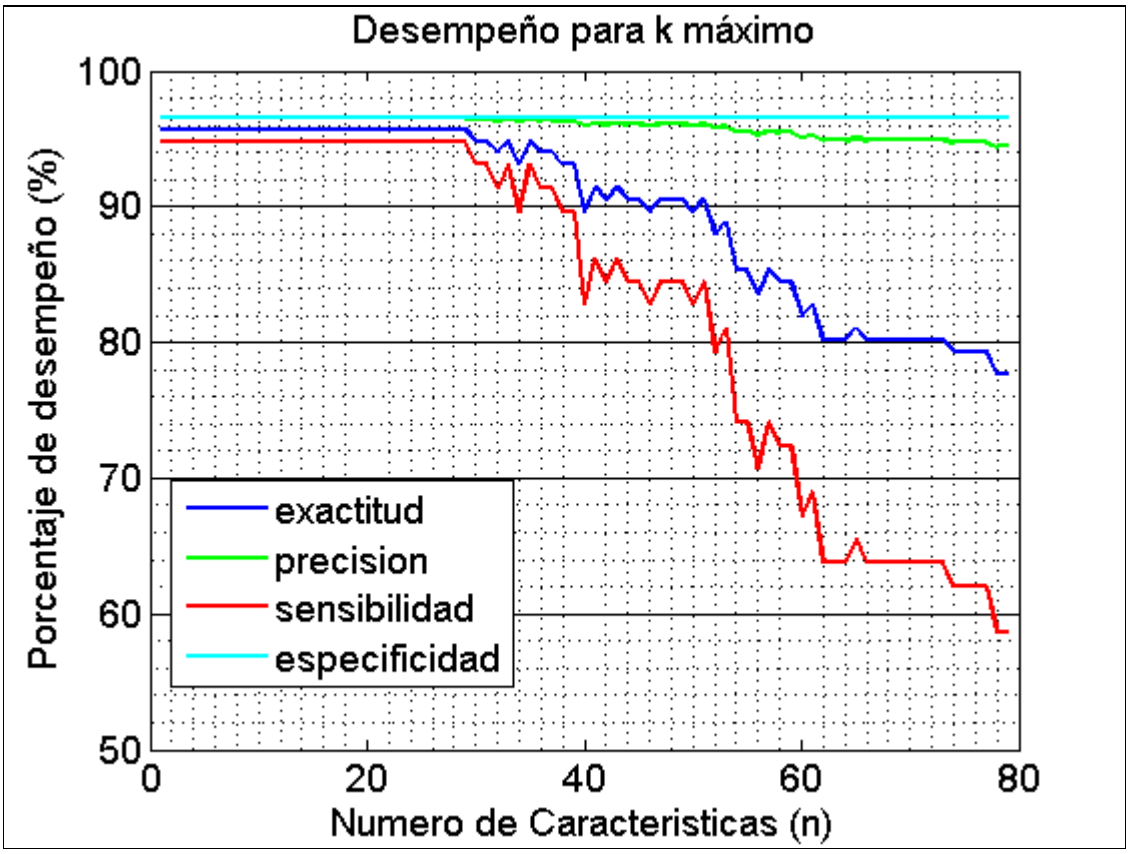


Figura 9. Desempeño del algoritmo k-NN variando *k*=1 hasta *k*=79

Los parámetros de desempeño *exactitud, precisión, sensibilidad y especificidad* se mantienen constantes hasta que *k* toma el valor de 29. Una vez que se supera este umbral, la exactitud y la sensibilidad comienzan a descender drásticamente, mientras que la precisión y sensibilidad lo hacen de manera menos pronunciada.

De ello, el cálculo de desempeño más alto se presenta en la Tabla 5, cuando *k* toma el valor de 29:

Tabla 5.

Parámetros de desempeño para k-NN

	Exactitud (%)	**Precisión (%)**	**Sensibilidad (%)**	**Especificidad (%)**	**Tiempo de procesamiento (ms)**
k-NN	96	97	95	94	143

4.2.2 Decision Trees

Para este algoritmo, el parámetro fundamental de configuración es la "frondosidad" del árbol, es decir, el nivel de profundidad de las ramas que tendrá el modelo predictor.

En un primer caso, cuando el árbol no se encuentra optimizado, su estructura se puede apreciar en la Figura 10, en tanto que las características de mayor relevancia que conforman este modelo predictor son las que a continuación se enlistan:

- Valor RMS en FFT
- Pico Máximo
- Tiempo de alcance pico máximo

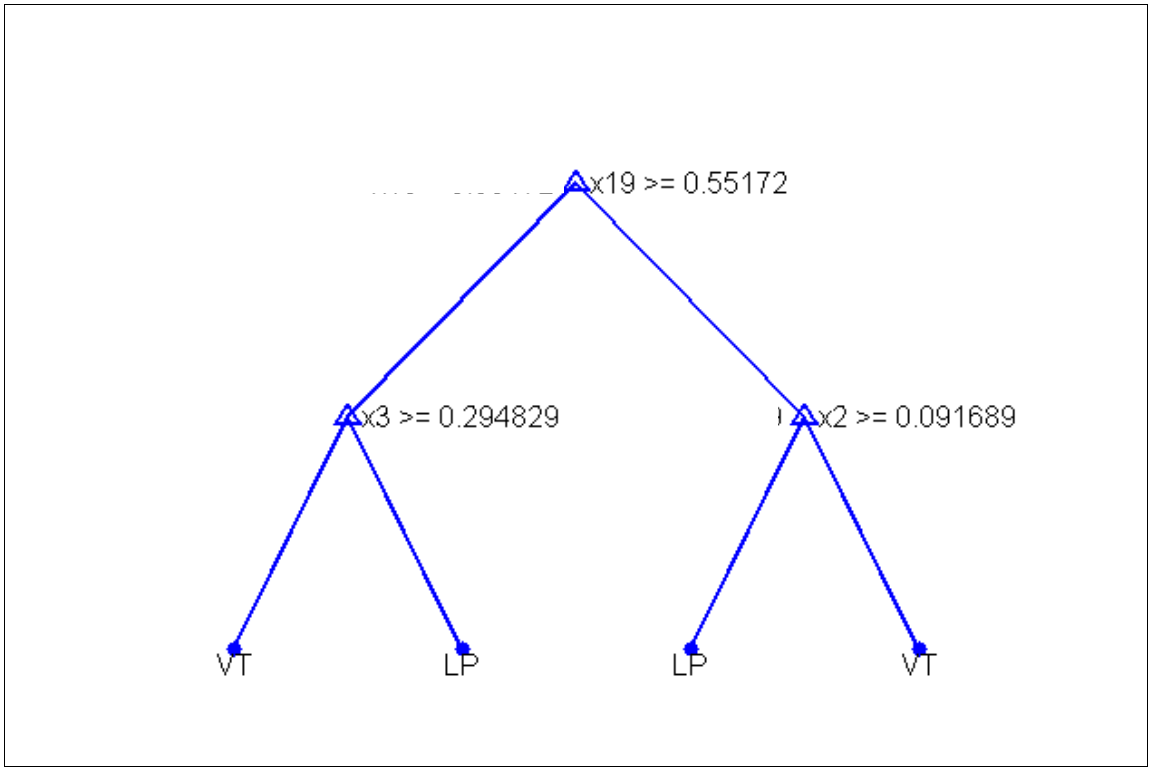


Figura 10. Modelo del árbol de decisiones

En la Figura 11 se puede apreciar un árbol que se encuentra optimizando, cuya característica fundamental es '"Valor RMS en FFT"'.

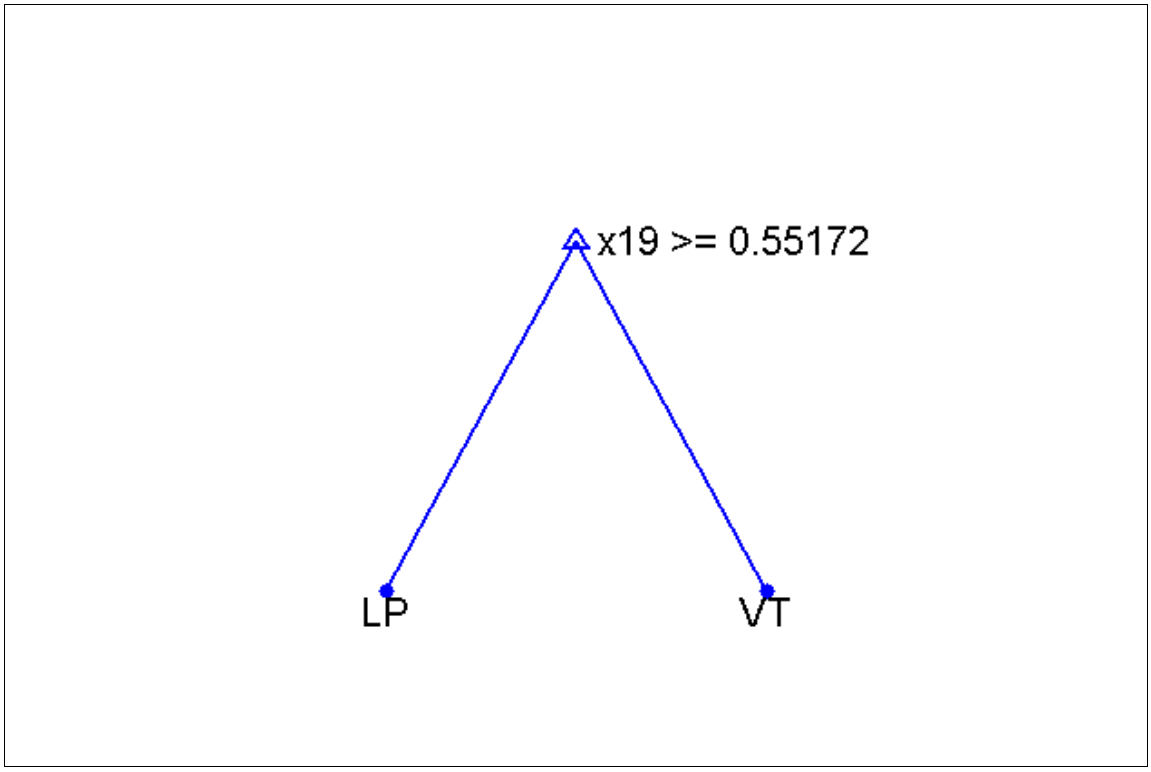


Figura 11. Modelo optimizado del árbol de decisiones

Una vez que se determinaron los modelos predictores con la matriz de entrenamiento, se pueden apreciar los valores de los parámetros de desempeño que resultaron de la matriz de pruebas, los mismos que se muestran en la Tabla 6.

Tabla 6.

Parámetros de desempeño para *Decision Trees*

	Exactitud (%)	**Precisión (%)**	**Sensibilidad (%)**	**Especificidad (%)**	**Tiempo de procesamiento (ms)**
Decision Trees	96	98	93	98	30
Decision Trees Optimal	89	94	83	95	9

4.2.3 Neural Networks

Las redes neuronales son altamente eficientes en el reconocimiento de patrones, cuando el algoritmo se encuentra correctamente configurado con un número adecuado de “neuronas”.

En la Figura 12 se puede apreciar los valores que se utilizaron este trabajo de investigación, existen 79 entradas (características), 8 neuronas y dos salidas (Volcano Tectónico y Largo Período).

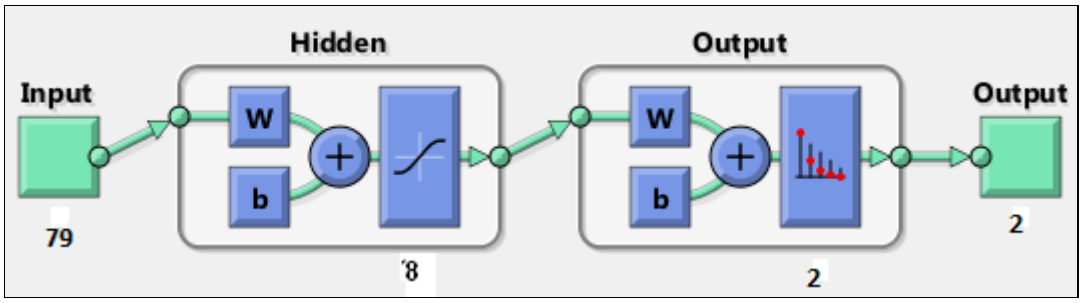


Figura 12. Modelo de redes neuronales: 79 características, 8 neuronas y 2 salidas

Fuente: (MathWorks, 2014)

En base al desempeño del predictor, una vez que se ingresó la matriz de pruebas, se varió el número de neuronas desde el valor de 1, hasta el valor de 79, siguiendo el mismo procedimiento que se utilizó para determinar el valor de *k* para k-NN. Los resultados se demuestran en la siguiente figura:

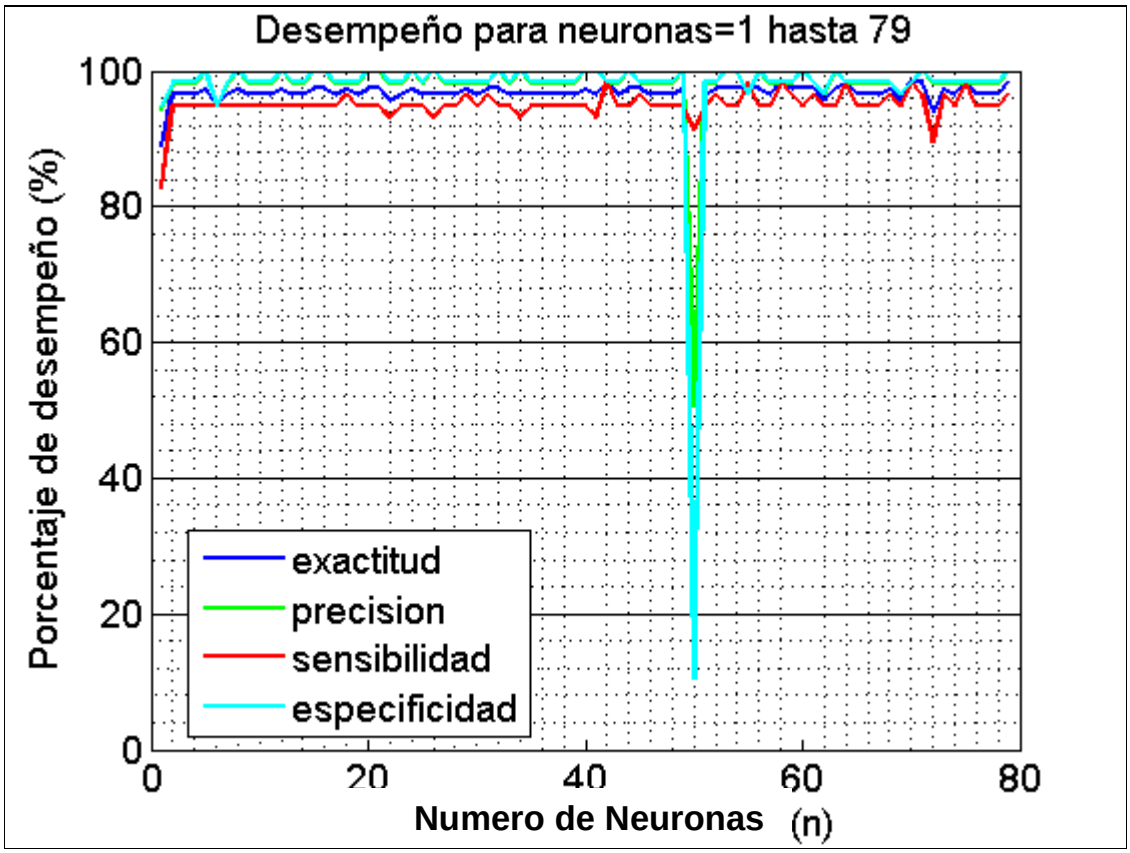


Figura 13. Desempeño de redes neuronales variando el valor de 1 hasta 79, para matriz de pruebas

La figura 13 muestra varios picos de un desempeño alto, por lo que la elección de número neuronas tendería a una elección semi aleatoria. Sin embargo, para evitar esta subjetividad, se adicionó un elemento más, con el fin de determinar un valor más fundamentado. En la Figura 14 se muestra el tiempo de procesamiento del predictor de redes neuronales, una vez que se ingresa la matriz de entrenamiento y prueba. De aquí se determina que el número adecuado de neuronas es 8.

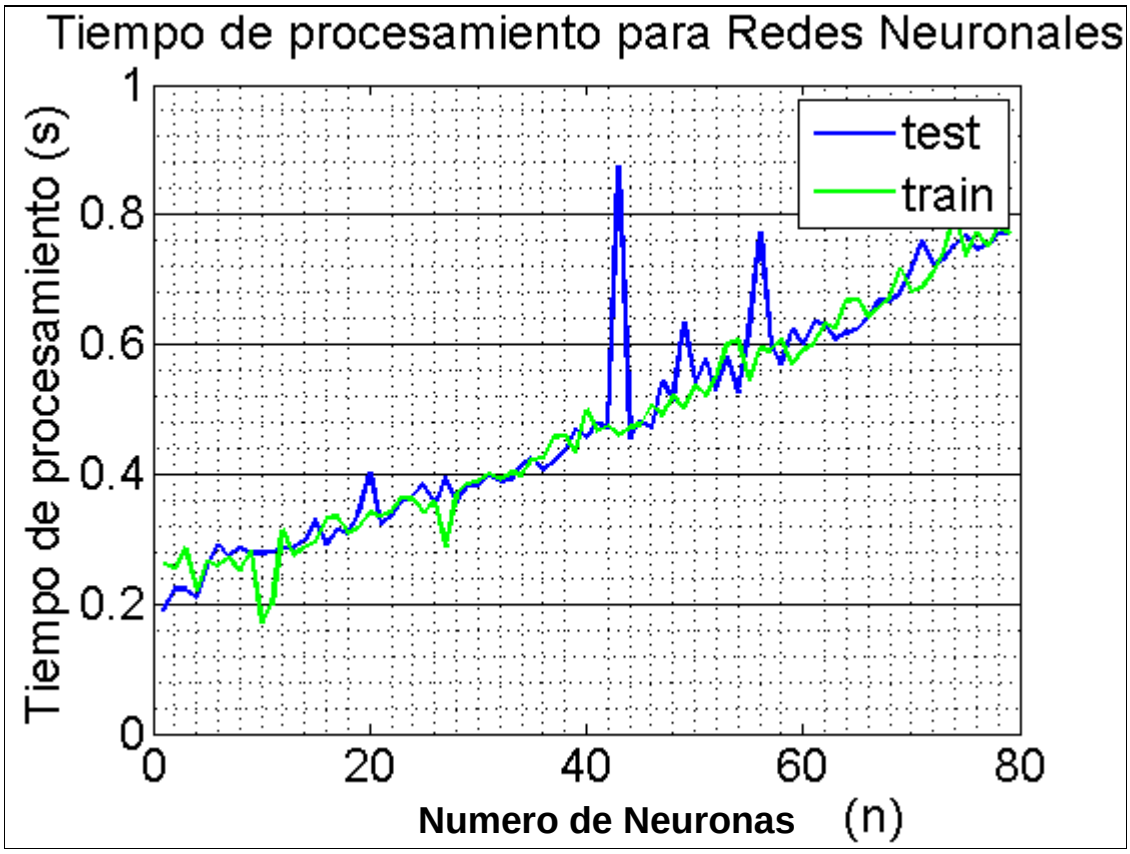


Figura 14. Tiempo de procesamiento del modelo de redes neuronales

En la Tabla 7, se muestran los parámetros de desempeño que se obtuvieron una vez que se configuro el algoritmo con 8 neuronas en la capa oculta:

Tabla 7.

Parámetros de desempeño para Redes Neuronales

	Exactitud (%)	**Precisión (%)**	**Sensibilidad (%)**	**Especificidad (%)**	**Tiempo de procesamiento (ms)**
Neural Network	97	98	95	98	269

En la Tabla 8 se demuestra que todos los parámetros de desempeño superan el 93%, siendo redes neuronales el más alto, seguido de árboles de decisión y finalmente se encuentra k-NN. Sin embargo, el tiempo de procesamiento denota que redes neuronales emplea más recursos para efectuar la clasificación, todo lo contrario sucede con árboles de decisión cuyo tiempo es 9 veces menor a redes neuronales. Cabe indicar que estos resultados fueron obtenidos a partir de la obtención de los modelos predictores y posterior ingreso de la matriz de pruebas.

Tabla 8.

Tabla de desempeño de algoritmos después de utilizar la matriz de pruebas

	Exactitud (%)	**Precisión (%)**	**Sensibilidad (%)**	**Especificidad (%)**	**Tiempo de procesamiento (ms)**
Decision Trees	96	98	93	98	31
Decision Trees Opt	89	94	83	95	9
k-NN	96	97	95	97	143
Neural Network	97	98	95	98	269

Adicionalmente a los resultados obtenidos en la Tabla 8, esto es considerando el criterio de normalización de las matrices de entrenamiento y prueba, se efectuó el mismo procedimiento para matrices sin haber pasado por el proceso de normalización. Aquí se puede apreciar un considerable descenso de los parámetros de desempeño para todos los algoritmos. Los resultados se muestran en la Tabla 9:

Tabla 9.

Tabla de desempeño de algoritmos sin normalizar las matrices de prueba y entrenamiento

	Exactitud (%)	Precisión (%)	Sensibilidad (%)	Especificidad (%)	Tiempo de procesamiento (ms)
Decision Trees	85	79	95	74	30
Decision Trees Opt	85	79	95	74	9
k-NN	66	67	60	71	146
Neural Network	94	98	90	98	289

4.3 Selección de Características

4.3.1 Métodos de Selección

Los resultados obtenidos en la sección previa contempla el empleo de las 79 características detalladas en (Saltos, 2014), sin embargo, en otros campos de investigación como por ejemplo, en la Bioinformática, los estudios tienen que sujetarse una base de datos corta en observaciones y amplia en el número de características, según se detalla en (MathWorks R. , 2014).

Los clasificadores basados en *machine learning* suelen enfrentarse a una disminución de la exactitud cuando tienen que trabajar con características que no son necesarias (Xiao, 2014), sin mencionar que, son propensos a caer en el sobreajuste al ruido por el excesivo número de características.

Por estos motivos, muchos de los clasificadores se someten a un pre-procesamiento con el fin de seleccionar las características más relevantes con el objetivo de aumentar su capacidad de predicción, disminuir el tiempo de procesamiento y aumentar la compresibilidad.

En varios artículos de investigación como (Somol, 2014), (Guyon, 2003) y (Kohavi, 1997) se describen dos principales métodos de selección de características conocidos como *filter* y *wrapper*, sin embargo, existe un tercer enfoque que es llamado *embedded* el cual se emplea en trabajos como (Lal, 2014) y (Xiao, 2014).

4.3.1.1 Métodos *Filter*

Selecciona el subconjunto de variables como un pre procesamiento independientemente del predictor escogido. La principal desventaja es que ignora por completo los efectos del subconjunto de características seleccionado sobre el desempeño del predictor.

Una vez que se emplea este método, se crea un parámetro de relevancia, el cual permite asignar un criterio de orden para las características que conformaran el subconjunto, sin embargo, este orden nunca toma en cuenta el modelo predictor que se va a utilizar.

4.3.1.2 Métodos *Wrapper*

En (Guyon, 2003) se plantea este método como una solución independiente a la máquina de aprendizaje, la cual es vista como una caja negra. El enfoque consiste en evaluar el desempeño de un modelo de predicción para cada uno de los subconjuntos seleccionados. Regularmente el

parámetro de evaluación es el número de predicciones correctamente realizadas.

Normalmente este enfoque requiere gran capacidad computacional, sin embargo, en estudios como (Reunanen, 2003) se denota que búsquedas pesadas pueden reducir el sobreajuste.

4.3.1.3 Métodos *Embedded*

Este método incorpora la selección de variables como parte del proceso de entrenamiento del modelo predictor, lo cual tiene varias ventajas como el evitar dividir los datos de entrenamiento en un conjunto de entrenamiento y validación.

Como ejemplo del uso de este enfoque, se puede mencionar a los árboles de decisión como CART, los cuales incorporan un mecanismo de selección de variables.

El método *filter* en comparación a *wrapper* puede ser más rápido, sin embargo, los métodos embedded pueden ser competitivos en este aspecto.

4.3.2 Desempeño de la Selección de Características

En la sección 4.2.2, en un primer paso, se desarrolló el modelo predictor para *Decision Trees*. Como segundo punto el modelo fue recalculado para el escenario en donde se aplicó el método *embedded* para determinar únicamente las características necesarias.

En la tabla 6. se detallan los parámetros de desempeño para ambos modelos de predictor, en donde se puede observar claramente la disminución

de la exactitud, precisión, sensibilidad y especificidad, y del tiempo de gasto computacional.

Como el método *embedded* está directamente ligado al modelo *Decision Trees*, se utilizó el método *wrapper* para ser empleado en los algoritmos de *k-Nearest Neighbors* y *Neural Networks.*

El método de selección empleado es conocido como Selección de Características Secuencial, el cual maneja dos componentes:

- Una función objeto llamada *criterio* en donde se maneja el error cuadrático medio para regresión y la tasa de error de clasificación justamente para criterios de clasificación.
- Un algoritmo de búsqueda secuencial que quite o suma características de un subconjunto, mientras el *criterio* es evaluado.

Como resultado del proceso de selección, se determina que las características 19 y 48 corresponden a las de más alta relevancia, siendo estas: Valor RMS en la FFT y Valor Máximo en FFT del nivel 1 de descomposición de la transforma Wavelet (Saltos, 2014). Adicionalmente se determinó que el proceso de selección de características toma un tiempo estimado de 5,61 segundos en ejecutarse.

Con el objetivo de determinar el comportamiento de cada uno de los modelos al realizar una selección de características, se formaron nuevas matrices de entrenamiento y prueba para determinar el desempeño del proceso de selección, sin embargo, para este paso se amplió a 16 el número de las características, las cuales se encuentra ordenadas de mayor a menor relevancia en la Tabla 10.

Tabla 10.

Características más relevantes

Características más relevantes			
1	Valor RMS en la FFT	9	Valor Máximo en FFT de A6
2	Pico Máximo	10	Porcentaje de energía en D5
3	Tiempo de alcance pico máximo	11	Valor Máximo en FFT de D6
4	Valor Máximo en FFT de D1	12	Porcentaje de energía en D3
5	Frecuencia umbral máxima (20-30 MHz)	13	Relación de pico RMS en D3
6	Entropía	14	Valor Máximo en FFT de D3
7	Valor RMS en D2	15	Frecuencia Pico FFT en D2
8	Porcentaje de Energía en D4	16	Energía en tiempo

Para ordenar las características de conformidad a su relevancia, se consideró como prioritarias aquellas que resultaron del método embedded que se encuentra inmerso en el proceso de clasificación de árboles de decisión. Como más prioritaria aparece aquella que se encuentra en la primera ramificación de decisión, seguida de la segunda y tercera rama respectivamente. Posterior a ello, se colocaron las características resultantes del método wrapper que no se repetían en el anterior método.

Este criterio de ordenamiento se tomó en base a los resultados de desempeño obtenidos en árboles de decisión, ya que el algoritmo emplea las características estrictamente necesarias para el proceso de clasificación. Y para trabajar con un número mayor de características, se completaron un total de 16 con el método denominado de Selección de Características Secuencial.

Los resultados para cada uno de las máquinas de aprendizaje se presentan en las figuras 16, 17 y 18. Cabe señalar que por cada grupo de características se calculó un modelo distinto, por ejemplo, para 1 característica se determinó un modelo 1, para 2 características se obtuvo un modelo 2 y así sucesivamente hasta completar el número total, que en este caso son 15.

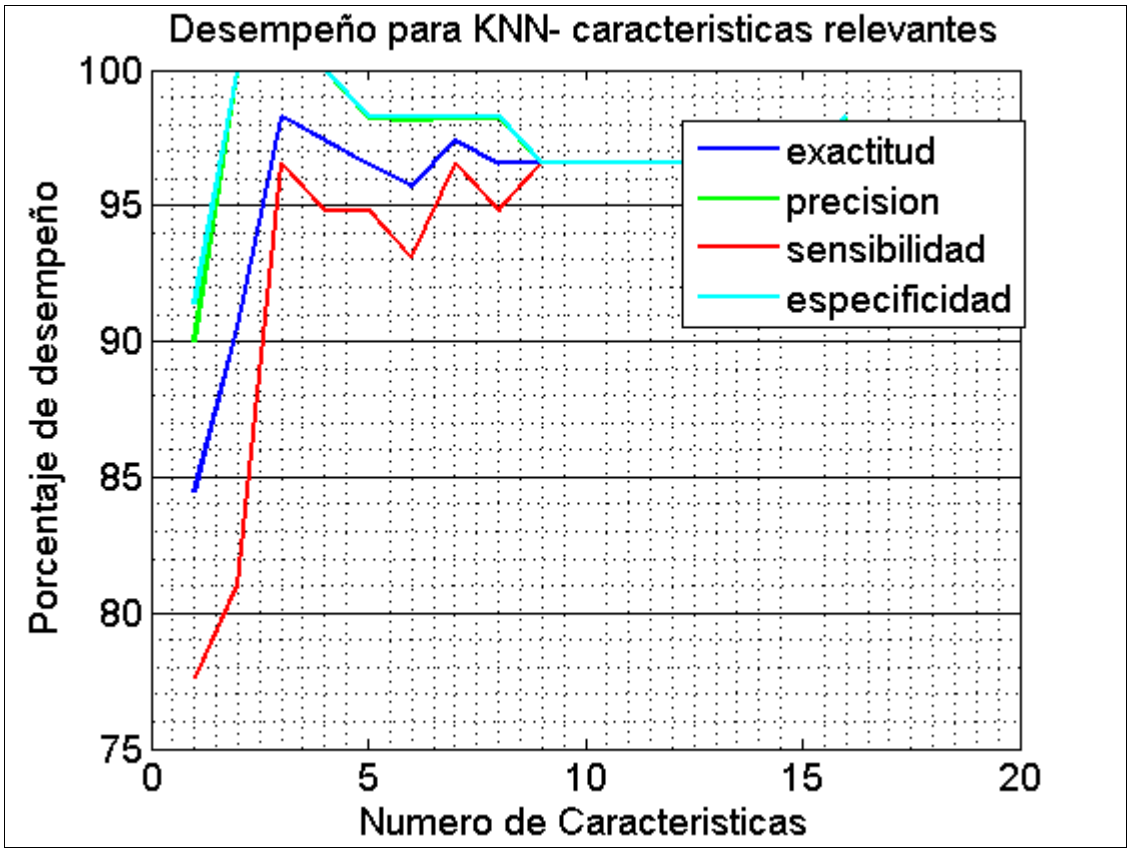


Figura 15. Desempeño del modelo k-NN variando de 1 a 16 características más representativas

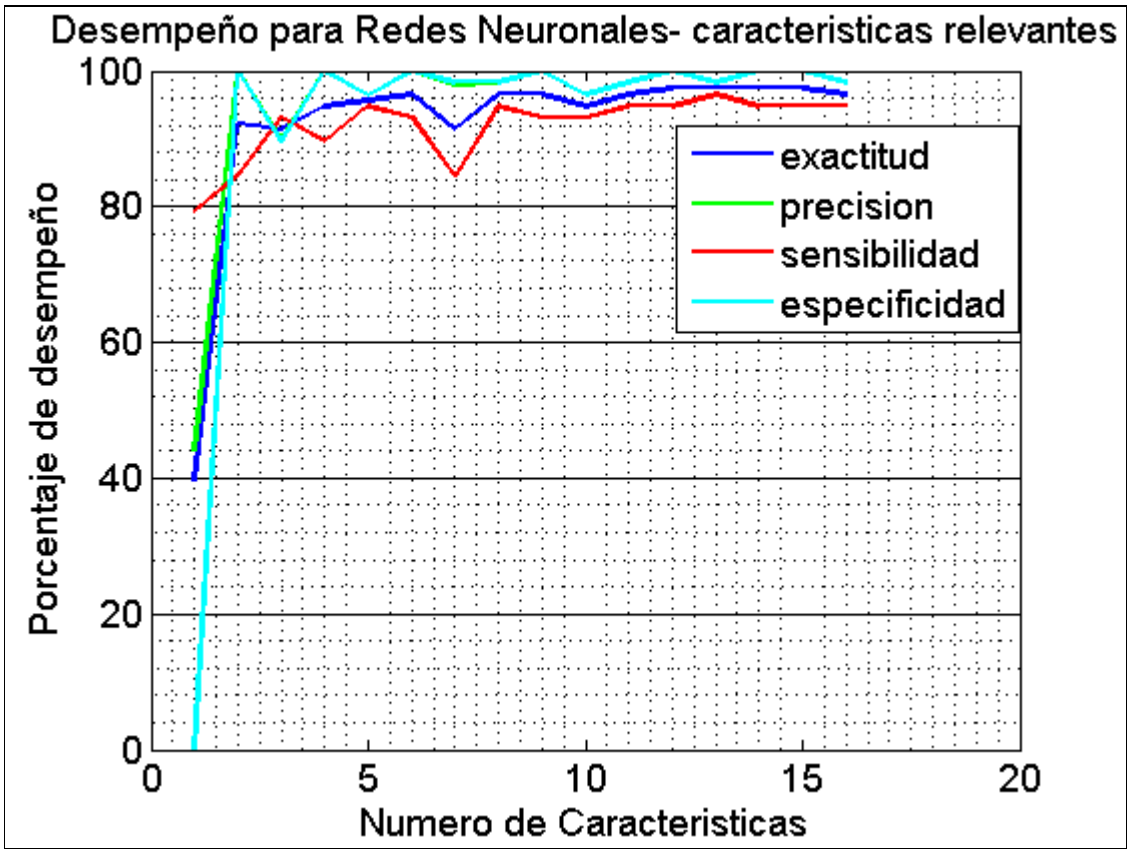


Figura 16. Desempeño del modelo de Redes Neurales variando de 1 a 16 características más representativas

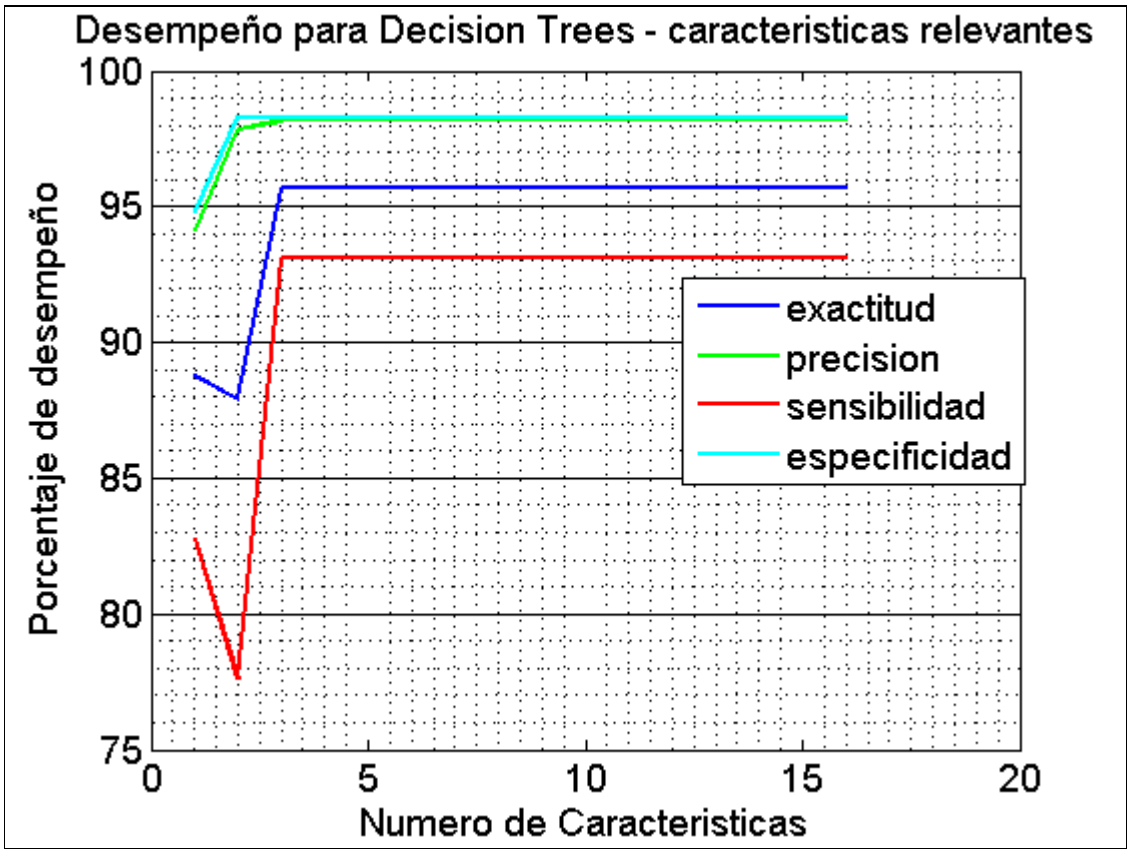


Figura 17. Desempeño del modelo DT variando de 1 a 16 características más representativas

Cabe señalar que con las 3 primeras características, los modelos de las máquinas virtuales de k-NN y DT alcanzaron su desempeño máximo, sin embargo NN para ese número de características no presenta su desempeño más alto, ya que esto lo hace cuando al sistema ingresan 6 características. Esto se puede apreciar en la Tabla 11:

Tabla 11.

Tabla de máximo desempeño de algoritmos cuando se ha utilizado selección de características.

	Exactitud (%)	**Precisión (%)**	**Sensibilidad (%)**	**Especificidad (%)**	**Tiempo de procesamiento (s)**
Decision Trees	96	98	93	98	3
k-NN	98	100	97	100	7
Neural Network[1]	91	90	93	90	190
Neural Network[2]	97	100	93	100	230

Finalmente, se confrontan los resultados obtenidos cuando al sistema ingresan todas las características y cuando se hace una selección de características, encontrando aquellas más relevantes para la clasificación. Los resultados se pueden observar en la Tabla 12:

[1] Desempeño con las primeras 3 características
[2] Desempeño con las primeras 6 características

Tabla 12.

Tabla de desempeño de algoritmos con y sin selección de características

	Exactitud (%)		**Precisión (%)**		**Sensibilidad (%)**		**Especificidad (%)**		**Tiempo de procesamiento (ms)**	
	Sin Selección	Con Selección	Sin Selección	Con Selección	Sin Selección	Con Selección	Sin Selección	Con Selección	Sin Selección	Con Selección
Decision Trees	96	96	98	98	93	93	98	98	31	3
k-NN	96	98	97	100	95	97	97	100	143	7
Neural Network	97	97	98	100	95	93	98	100	269	230

4.4 Análisis de resultados.

Los parámetros de desempeño: *exactitud, precisión, sensibilidad y especificidad,* bajo las condiciones descritas de normalización y como entrada, todas las características utilizadas, tienen valores relativamente altos, presentando mínimos superiores al 93% para cada uno de los modelos de las algoritmos seleccionados.

Cuando se utilizan las 79 características y los valores de las matrices se encuentran normalizados, los parámetros de desempeño determinan que los valores más altos los presenta *Neural Network*, seguido de *Decision Trees* y por último se encuentra *k-Nearest Neighbors.* Sin embargo, en lo que respecta al

tiempo de procesamiento, el predictor que más recursos utiliza es *Neural Network*, mientras que el que menos gasto genera es *Decision Trees*.

Cuando las matrices de entrenamiento y pruebas están formadas por valores que no se encuentran normalizados, los parámetros de desempeño sufren un considerable decremento, llegando a valores cercanos al 60% para el predictor *k-Nearest Neighbors* que es el modelo que más se ve afectado por esta situación. Todo lo contrario ocurre con *Neural Network* ya que sus valores de desempeño mínimos bordean el 89%.

El método de selección de características de tipo *wrapper*, emplea cerca de 5,6 segundos para determinar los parámetros más relevantes, lo que se traduce en un gasto computacional exigente. La selección de tipo embedded implementada en el modelo *Decision Trees* sin duda optimiza el uso del tiempo empleado en el proceso, ya que utiliza 31 milisegundos segundos para determinar la relevancia y clasificar.

El sistema de aprendizaje supervisado, implementado para cada uno de los modelos, puede ser catalogado de tipo off-line, puesto que, el gasto computacional calculado, únicamente contempla el tiempo empleado para la clasificación, es decir, deja de lado los recursos que se emplean en el pre procesamiento, en la obtención del modelo predictor y en la selección de características.

Con matrices normalizadas de entrenamiento y prueba, y utilizando el enfoque de selección de características, los modelos de las máquinas virtuales de *k-Nearest Neighbors* y *Decision Trees* alcanzaron su desempeño máximo, con tan solo 3 características, a pesar de ello, *Neural Networks* para ese número de características no presenta su desempeño más alto, ya que esto lo hace cuando al sistema ingresan 6 características.

El gasto computacional cuando se aplica la selección de características se reduce notablemente para *k-Nearest Neighbors* de 143 milisegundos s a 7 milisegundos y *Decision Trees* de 31 milisegundos a 3 milisegundos. Sin embargo, *Neural Networks* no reduce tan significativamente su tiempo ya que pasa de 269 milisegundos a 230 milisegundos.

Haciendo una comparación entre los resultados con y sin selección de características se determina que *Decision Trees* no sufre ninguna variante, ya que al ser un sistema de tipo *embedded* siempre seleccionará las variantes de mayor relevancia. *Neural Networks* muestra un ligero incremente en la especificidad y precisión, sin embargo, la sensibilidad disminuye. Finalmente *k-Nearest Neighbors,* es el modelo que más mejora, ya que sufre un incremento en los cuatro parámetros de desempeño.

El mejoramiento en el desempeño de *k-Nearest Neighbors,* se debe a la eliminación de características, ya que, esta máquina virtual al ser propensa al sobreajuste, trabajará de forma óptima cuando el proceso de entrenamiento no contenga parámetros irrelevantes.

Decision Trees es el modelo que menos gasto computacional emplea, cuando este predictor trabaja con el enfoque de selección de características, concretamente con 3 variantes. El modelo de predicción de *k-Nearest Neighbors* es la máquina virtual que presenta los valores más altos de *exactitud, precisión, sensibilidad y especificidad,* cuando trabaja con las 3 características más relevantes.

CAPÍTULO 5

CONCLUSIONES Y TRABAJOS FUTUROS

De los algoritmos de clasificación de eventos de origen vulcanológico basados en *machine learning: k-Nearest Neighbors, Decision Trees, Neural Networks,* se identificó que el mejor algoritmo es *k-Nearest Neighbors* presenta los valores más altos de desempeño en términos de exactitud (98%), precisión (100%), especificidad (100%), sensibilidad (97%) y gasto computacional (7 ms). Estos valores fueron posibles ya que en el pre-procesamiento se efectuó un filtrado de la señal para la remoción de errores, se suprimió la media y tendencia lineal de las señales, se normalizaron los valores y finalmente se realizó una selección de características, que al final permitió eliminar el sobreajuste en esta máquina virtual.

La selección de características que se efectuó previamente, contribuyó al mejoramiento de la exactitud, precisión, sensibilidad y especificidad de k-*Nearest Neighbors* y *Neural Networks.* El modelo clasificador obtenido a partir de uso de las características más relevantes: valor RMS en la FFT, pico máximo y tiempo de alcance del pico máximo permitió mejorar el desempeño del modelo clasificador conseguido con el empleo de todos los atributos de las señales.

Se identificaron que las características más relevantes que permiten mejorar el desempeño con en el dominio del tiempo son tiempo de alcance del pico máximo y pico máximo, mientras que en dominio de la frecuencia es el valor RMS en la FFT.

El tiempo de procesamiento empleado únicamente para la tarea de clasificación se reduce considerablemente cuando el modelo predictor es

obtenido únicamente con las características más significantes, de este modo *k-Nearest Neighbors* se redujo en 95%, *Decision Trees en 90% y Neural Networks* en 15%. A pesar de ello, es necesario llegar a un balance entre los resultados de los parámetros de desempeño y el tiempo total empleado, ya que mientras más eficientes son los algoritmos, más recursos son los empleados.

En términos generales, el tiempo medio que se emplea para la clasificación por cada evento en *k-Nearest Neighbors* es de 60 microsegundos, en *Decision Trees* es de 26 microsegundos y *Neural Networks* es de 2 milisegundos. Estos valores no consideran el tiempo empleado en pre procesamiento ni obtención del modelo predictor.

Finalmente estos resultados pueden ser tomados como plataforma de estudio para otros volcanes que presentan actividad volcánica vigente. Como se había manifestado, cada elevación presenta características propias, por tal motivo, es necesario ampliar esta investigación a los volcanes activos más representativos del país.

Por los altos valores de desempeño obtenidos en el presente trabajo, el mismo podría ser utilizado en el campo de la Geología para determinar patrones de comportamiento del Volcán Cotopaxi ya que al tener una correcta clasificación de los eventos, los expertos no tendrían que efectuar esta tarea, sino enfocarse únicamente en la conducta del volcán.

BIBLIOGRAFÍA

Allen, W. G. (2006). Deploying a Wireless Sensor Network on an Active Volcano. *IEEE Computer Society*, 18-25.

Cardenas, J. H. (2014, 11 24). *Universidad de Castilla - La Mancha.* Retrieved 11 24, 2014, from https://www.uclm.es/profesorado/egcardenas/sismos2.pdf

Curilem, G. V. (2009). Classification of seismic signals at Villarrica volcano (Chile) using neural networks and genetic algorithms. *Journal of Volcanology and Geothermal Research*, 1–8.

Chiroque, J. I. (2012, 9 21). *PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ .* Retrieved 02 16, 2015, from http://tesis.pucp.edu.pe/repositorio/bitstream/handle/123456789/4470/INCA_CHIROQUE_JULITA_HASKELL_LENGUAJES_PROGRAMACION.pdf?sequence=1

Duda, R. O. (2012). *Pattern classification.* John Wiley & Sons.

Guyon, I. E. (2003). An Introduction to Variable and Feature Selection. *Journal of Machine Learning Research 3*, 1157-1182.

Harrington, P. (2012). *Machine Learning in Action.* New York: Manning Publications Co.

Haykin, S. (1994). *Neural Networks: A Comprehensive Foundation.* NY: Macmillan.

Hazel, G. (2007). *Seismic event detection using arrays - A comparison of 2 approaches.* Utrecht University.

IGEPN. (2010). *Institulo Geofísico EPN*. Retrieved 03 19, 2012, from Institulo Geofísico EPN: http://www.igepn.edu.ec/index.php/red-de-observatorios-vulcanologicos-rovig

IGEPN. (2015, 02 02). *Instituto Geofísico de la Escuela Politécnica Nacional.* Retrieved 02 02, 2015, from http://www.igepn.edu.ec/index.php/cotopaxi

Iyer, A. S. (2011). Neural Classification of Infrasonic Signals Associated with Hazardous Volcanic Eruptions. *Proceedings of International Joint Conference on Neural Networks*, 336-341.

Jaramillo, C. L.-C. (2014). A new structure for sequential detection and maximum entropy spectral estimator for characterization of volcanic seismic signals. *Communications (LATINCOM), 2014 IEEE Latin-America Conference on*, 1-6.

Joevivek, V. C. (2010). Improving Seismic Monitoring System for Small to Intermediate. *International Journal of Computer Science and Security (IJCSS)*, 308-315.

Kohavi, R. J. (1997). Wrappers for feature subset selection. *Elsevier*, 273-324.

Lal, T. N. (2014, 12 12). *The Cyberneum.* Retrieved 12 11, 2014, from http://www.cyberneum.de/fileadmin/user_upload/files/publications/pdf3012.pdf

Lara, R. B.-A. (2015). On Real-Time Performance Evaluation of Volcano Monitoring Systems with Wireless Sensor Networks. *Sensors Journal, IEEE* , 2.

Lara-Cueva, R. B.-A. (2014). Performance evaluation of a volcano monitoring system using wireless sensor networks. *ommunications (LATINCOM), 2014 IEEE Latin-America Conference on* , 1-6.

MathWorks. (2014, 12 12). *Neural Network Classifier*. Retrieved from http://www.mathworks.com/help/nnet/examples/crab-classification.html

MathWorks, R. (2014). *Selecting Features for Classifying High-dimensional Data*. Retrieved 12 11, 2014, from http://www.mathworks.com/help/stats/examples/selecting-features-for-classifying-high-dimensional-data.html

McCaffrey, J. (2012). *Classification and Prediction Using Neural Networks*. Retrieved 1 27, 2015, from https://msdn.microsoft.com/en-us/magazine/jj190808.aspx

Michie, D. S. (1994). *Machine Learning, Neural and Statistical.*

Nassery, P. F. (1997). Real Time Seismic Signal Processing Using The ARMA Model Coefficients And An Intelligent Monitoring System. *IEEE TENCON*, 807-810.

Nigrin, A. (1993). *Neural Networks for Pattern Recognition.* Cambridge: The MIT Press.

Nilsson, N. J. (1998). *Introduction to Machine Learning.* Stanford.

Reunanen, J. (2003). Overfitting in making comparisons between variable selection methods. *JMLR*, 1371–1382.

Saltos, M. G. (2014). *Análisis de señales sísimicas del volcán Cotopaxi mediante las transformadas WAVELET y FOURIER.* Quito.

Scarpetta, S. F. (2005). Automatic Classification of Seismic Signals at Mt. Vesuvius Volcano, Italy, Using Neural Networks. *Bulletin of the Seismological Society of America*, 185–196.

SENPLADES. (2009). *Secretaría Nacional de Planificación y Desarrollo.* Retrieved 3 19, 2012, from Secretaría Nacional de Planificación y Desarrollo: http://www.senplades.gob.ec/web/senplades-portal/plan-nacional-para-el-buen-vivir

Sharma, B. K. (2010). Evaluation of Seismic Events Detection Algorithms. *JOUR.GEOL.SOC.INDIA, VOL.75*, 533-538.

Shimshoni, Y. I. (2002). Classification of seismic signals by integrating ensembles of neural networks. *Signal Processing, IEEE Transactions on (Volume:46 , Issue: 5)*, 1194 - 1201.

Smola, A. V. (2008). *Learning, Introduction to Machine.* Cambridge.

SNGR. (2010). *Secretaría Nacional de Riesgos.* Retrieved 03 19, 2012, from Secretaría Nacional de Riesgos: http://www.snriesgos.gob.ec/quienes-somos/informacion-institucional/mision-vision.html

Somol, P. B. (2014, 12 12). *Filter- versus Wrapper-based Feature Selection.* Retrieved from http://library.utia.cas.cz/separaty/historie/somol-filter-%20versus%20wrapper-based%20feature%20selection%20for%20credit%20scoring.pdf

Song, W. H. (2009). Air-dropped Sensor Network for Real-time High-fidelity. *The 7th Annual International Conference on Mobile Systems, Applications, and Services*, 305-318.

Tan, R. X. (2010). Quality-driven Volcanic Earthquake Detection using Wireless Sensor Networks. *31st IEEE Real-Time Systems Symposium*, 272-280.

Witten, I. H. (2005). *Data Mining - Practical Machine Learning tools and Technique.* San Francisco: Elsevier Inc.

Wu, X. K. (2008). Top 10 algorithms in data mining. *Springer*, 23.

Xiao, Z. D. (2014). *ESFS: A new embedded feature selection method based.*

Zhu, W. Z. (2010). Sensitivity, Specificity, Accuracy, Associated Confidence Interval and ROC. *NESUG*.

Zurada, J. (1992). *Introduction To Artificial Neural Systems.* Boston: PWS Publishing Company.

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

ANEXOS

Conformación de matrices

```
% %separacion de LP y VT
clear all
load matriz_caracteristica
% load etique
N=1;
countLPX=1;
countVTX=1;

while N < 876
a= strcmp (TIPO(N,1),'LP');
if a == 1;
   LPX(countLPX,:)= MATRIZ (N,:);
   countLPX=1+countLPX;
else
   VTX(countVTX,:)= MATRIZ (N,:);
   countVTX=1+countVTX;
end
N=N+1;
end


%matrices normalizadas
count=1;
LPXnor=LPX;
```

```
VTXnor=VTX;
while count <= size(LPX,2)
   LPXnor(:,count)=LPX(:,count)/max(abs(LPX(:,count)));
   VTXnor(:,count)=VTX(:,count)/max(abs(VTX(:,count)));
   count=count+1;
end
%
% %media con normalizar
% mediaLPX79nor=mean(LPXnor,1);     %79x1
% mediaVTX79nor=mean(VTXnor,1);     %79x1
% desLPX79nor=std(LPXnor,0,1);      %79x1
% desVTX79nor=std(VTXnor,0,1);      %79x1
%
%
%
% %matriz randomica de test y training
a1=size(VTX); % #VT
b1=size(LPX); % #LP

a2=randi(a1(1,1),1,a1(1,1));    %numeros aleatorios  VTX
b2=randi(b1(1,1),1,b1(1,1));    %numeros aleatorios  LPX

countrand=1;



while countrand<=size(a2,2)
   if countrand<=size(a2,2)/2
      Mtrain(countrand,:)=VTXnor(a2(1,countrand),:);
      Mtrain(countrand+size(a2,2)/2,:)=LPXnor(a2(1,countrand),:);
      etique(countrand,1)=TIPO(173,1);
      countrand=countrand+1;
```

```
   else
      Mtest(countrand-size(a2,2)/2,:)=VTXnor(a2(1,countrand),:);
      Mtest(countrand,:)=LPXnor(a2(1,countrand),:);
      etique(countrand,1)=TIPO(1,1);
      countrand=countrand+1;
   end

%
end


[predictree,predictreeop] = trees(Mtrain,Mtest,etique);
save Mtrain.mat
save Mtest.mat
% save etique1.mat
```

Obtención de Parámetros fundamentales *k* y *neuronas*

```
clear all
set_nice_plot_values

load Mtrain.mat
load Mtest.mat
load etique.mat

a=1;
while a<=79
%test
[predicknn,e] = knn(Mtrain,Mtest,etique,a);
[accknn, preknn,sensknn,specknn] = performance(etique,predicknn);
kneibor(a,:)=[accknn, preknn,sensknn,specknn];
time(a,:)=e;

%train
[predicknn2,e2] = knn(Mtrain,Mtrain,etique,a);
[accknn2, preknn2,sensknn2,specknn2] = performance(etique,predicknn2);
k(a,:)=[accknn2, preknn2,sensknn2,specknn2];
time2(a,:)=e2;

a=a+1;
end


figure(1)
```

```
plot(mean(kneibor,2),'b-')
grid minor
hold on
plot(mean(k,2),'g-')
legend('test','train')
title('Desempeño para k máximo')

figure(2)

plot(time,'b-')
grid minor
hold on
plot(time2,'g-')
legend('test','train')
title('Tiempo de procesamiento para KNN')



b=1;
while b<=79

%test
[predicneu,e3] = neunet(Mtrain,Mtest,etique,b);
[accneu, preneu,sensneu,specneu] = performance(etique,predicneu);
neuro(b,:)=[accneu, preneu,sensneu,specneu];
time3(b,:)=e3;

%train
[predicneu4,e4] = neunet(Mtrain,Mtrain,etique,b);
[accneu, preneu,sensneu,specneu] = performance(etique,predicneu4);
neuro1(b,:)=[accneu, preneu,sensneu,specneu];
time4(b,:)=e4;
```

```
b=b+1;
end


figure(3)
plot(mean(neuro,2),'b-')
grid minor
hold on
plot(mean(neuro1,2),'g-')
legend('test','train')
title('Desempeño para neuronas=1 hasta 79')

figure(4)

plot(time3,'b-')
grid minor
hold on
plot(time4,'g-')
legend('test','train')
title('Tiempo de procesamiento para Redes Neuronales')
```

Obtención de desviación estándar y media

```
clear all
% set_nice_plot_values

load Mtrain.mat
load Mtest.mat
load etique.mat

figure (1)
plot(mean(LPX,1),'b-')
grid minor
title('Matriz de caracteristicas - Valor Medio')
hold on
plot(mean(VTX,1),'g-')
legend('Largo Periodo','Volcano Tectonico')

figure (2)
plot(mean(LPXnor,1),'b-')
grid minor
title('Matriz normalizada de caracteristicas - Valor Medio')
hold on
plot(mean(VTXnor,1),'g-')
legend('Largo Periodo','Volcano Tectonico')

figure (3)
plot(std(LPX,1),'b-')
grid minor
title('Matriz de caracteristicas - Desviacion Estandar')
hold on
```

```
plot(std(VTX,1),'g-')
legend('Largo Periodo','Volcano Tectonico')

figure (4)
plot(std(LPXnor,0,1),'b-')
grid minor
title('Matriz de caracteristicas - Desviacion Estandar')
hold on
plot(std(VTXnor,0,1),'g-')
legend('Largo Periodo','Volcano Tectonico')
```

Cálculo de parámetros de desempeño – función performance

```
function [accuracy, precision,sens,spec] = performance(etique,Ypredictor)
a = length( find( (strcmp(etique,'VT'))&(strcmp(Ypredictor,'VT')) ) );
b = length( find( (strcmp(etique,'LP'))&(strcmp(Ypredictor,'VT')) ) );
c = length( find( (strcmp(etique,'VT'))&(strcmp(Ypredictor,'LP')) ) );
d = length( find( (strcmp(etique,'LP'))&(strcmp(Ypredictor,'LP')) ) );

accuracy    =
sum(strcmp(Ypredictor,'VT')==strcmp(etique,'VT'))/length(strcmp(etique,'VT'))*100;

precision   = a/(a+b)*100;
sens        = a/(a+c)*100;
spec        = (d/(b+d))*100;
ber         = (sens + spec )/2*100;
sumerror    = b+c;
end
```

Algoritmo k-Nearest Neighbors –funcion knn

```
function [Ypredictorknn,e] = knn(Mtrain,Mtest,etique,a)
t=cputime;
% tstart=tic;
%modelo y predictor

modelknn=ClassificationKNN.fit(Mtrain,etique,'NumNeighbors',a);
Ypredictorknn=predict(modelknn,Mtest);
e=cputime-t;
%view (modelknn,'Mode','GrapH')

%modelo y predictor optimizado
% modeloptknn=ClassificationKNN.fit(Mtrain,etique,'NumNeighbors',a);
% Ypredictoroptknn=predict(modeloptknn,Mtest);

%view (modelknn,'Mode','GrapH')

%view (modeloptknn,'Mode','GrapH')
% telapsed=toc(tstart);
End
```

Algoritmo Decision Trees – Función trees

```
function [Ypredictor,Ypredictoropt,e,e1] = trees(Mtrain,Mtest,etique)
randn('seed',0);
rand('seed',0);

%modelo y predictor
% tstart=tic;
t=cputime;
model=ClassificationTree.fit(Mtrain,etique);
Ypredictor=predict(model,Mtest);
% telapsed=toc(tstart);
e=cputime-t;
view (model,'Mode','GrapH');

%modelo y predictor optimizado
% tstart1=tic;
t1=cputime;
[~,~,~,bestlevel] = cvLoss(model,'SubTrees','All');
modelopt = prune(model,'Level',bestlevel);
Ypredictoropt=predict(modelopt,Mtest);
% telapsed1=toc(tstart1);
e1=cputime-t1;
view (modelopt,'Mode','GrapH');
end
```

Algoritmo Neural Networks – Función nuenet

```
function [Ypredictorneu,e] = neunet(Mtrain,Mtest,etique,b)

%modelo y predictor

% La matriz de etiquetas es de 116x2.
% En la columna 1, de la fila 1 a la 58 se coloca 1 para los VT
% En la columna 2, de la fila 59 a la 116 se coloca 1 para los LP
target=[ones(size(etique,1)/2,1),zeros(size(etique,1)/2,1);zeros(size(etique,1)/2,1),ones(size(eti
que,1)/2,1) ];
% tstart=tic;
t=cputime;
net = patternnet(b);
%view(net)
[net,tr] = train(net,Mtrain',target');
%plotperform(tr)
% Prueba con la matriz test
% Y tiene es una matriz de 116x2 y tiene la misma dimension
Y = net(Mtest');
Y = round(Y)';
% telapsed=toc(tstart);
e=cputime-t;
%Ypredictorneural=[Ypredictorneural(1:58,1);Ypredictorneural(59:116,2)];

a=1;

while a<=size(etique,1)/2
   if Y(a,1)== 1
```

```
      Ypredictorneu(a,1)=etique(1,1);
   else
      Ypredictorneu(a,1)=etique(116,1);
   end

   a=a+1;
end

while a<=size(etique,1)
   if Y(a,2)==1
      Ypredictorneu(a,1)=etique(116,1);
   else
      Ypredictorneu(a,1)=etique(1,1);
   end
     a=a+1;
end
end
```

Función principal para obtención de parámetros de calidad

```
clear all
set_nice_plot_values

load Mtrain.mat
load Mtest.mat
load etique.mat

a=1;
aa=1;
while a<=79

[predicknn,e] = knn(Mtrain,Mtest,etique,a);
[accknn, preknn,sensknn,specknn] = performance(etique,predicknn);
kneibor(a,:)=[accknn, preknn,sensknn,specknn];
time(a,:)=e;
a=a+1;
end

figure (1)
plot(kneibor(:,1),'b-')
grid minor
hold on
plot(kneibor(:,2),'g-')
hold on
plot(kneibor(:,3),'r-')
hold on
plot(kneibor(:,4),'c-')
legend('exactitud', 'precision' , 'sensibilidad' , 'especificidad')
title('Desempeño para k máximo')
```

```
figure (2)
plot(time)
grid minor
title('Tiempo de procesamiento para KNN')

b=1;
while b<=79

[predicneu,e] = neunet(Mtrain,Mtest,etique,b);

[accneu, preneu,sensneu,specneu] = performance(etique,predicneu);
kneibor1(b,:)=[accneu, preneu,sensneu,specneu];
time1(b,:)=e;
b=b+1;
end

figure (3)
plot(kneibor1(:,1),'b-')
grid minor
hold on
plot(kneibor1(:,2),'g-')
hold on
plot(kneibor1(:,3),'r-')
hold on
plot(kneibor1(:,4),'c-')
legend('exactitud', 'precision' , 'sensibilidad' , 'especificidad','tiempo')
title('Desempeño para neuronas=1 hasta 79')
figure(4)
plot(time1)
grid minor
title('Tiempo de procesamiento para Redes Neuronales')
```

```
%Obtencion de modelo de prediccion para KNN, DT, y NEURAL NETWORK
%Cálculo de parametros de desempeno, exactitud, precision, especificidad,
%sinsibilidad y gasto computacional

a=1;
[predicknn,eknn] = knn(Mtrain,Mtest,etique,a);
[predictree,predictreeop,etree,etreeop] = trees(Mtrain,Mtest,etique);
b=12;
[predicneu,eneu] = neunet(Mtrain,Mtest,etique,b);

%performance filas: tree, treeop,knn, knnopt - columnas: accuracy, precision,sens,spec
[acctree, pretree,senstree,spectree] = performance(etique,predictree);
[accutreeop, pretreeop,senstreeop,spetreeopt] = performance(etique,predictreeop);
[accknn, preknn,sensknn,specknn] = performance(etique,predicknn);
[accneu, preneu,sensneu,specneu] = performance(etique,predicneu);
% [accknnop, preknnop,sensknnop,specknnop] = performance(etique,predicknnop);

perform=[acctree, pretree,senstree,spectree,etree;accutreeop,
pretreeop,senstreeop,spetreeopt,etreeop;accknn, preknn,sensknn,specknn,eknn;accneu,
preneu,sensneu,specneu,eneu]
```