



Universidad Central de Venezuela
Facultad de Ciencias
Escuela de Computación
Laboratorio de Redes Móviles e Inalámbricas (ICARO)
Centro de Ingeniería de Software y Sistemas (Centro ISYS)

Diseño, Implementación y Evaluación de Técnicas Híbridas de Aprendizaje Automático en la Detección de Intrusos en Redes de Computadoras

Trabajo Especial de Grado
presentado ante la Ilustre
Universidad Central de Venezuela
Por el Bachiller
Deyban Andrés Pérez Abreu

Tutores:
Prof. Eugenio Scalise y Prof. Miguel Astor

Caracas, mayo de 2017



Universidad Central de Venezuela
Facultad de Ciencias
Escuela de Computación
Laboratorio de Redes Móviles e Inalámbricas (ICARO)
Centro de Ingeniería de Software y Sistemas (Centro ISYS)

Acta del veredicto

Quienes suscriben, Miembros del Jurado designado por el Consejo de la Escuela de Computación para examinar el Trabajo Especial de Grado, presentado por el Bachiller Deyban Andrés Pérez Abreu C.I.: 23110188, con el título “Análisis, Diseño e Implementación de Técnicas Híbridas de Aprendizaje Automático en la Detección de Intrusos en Redes de Computadoras”, a los fines de cumplir con el requisito legal para optar al título de Licenciado en Computación, dejan constancia de lo siguiente:

Leído el trabajo por cada uno de los Miembros del Jurado, se fijó el día **XX** de septiembre, a las **XX:XX**, para que su autor lo defendiera en forma pública en **LUGAR**, lo cual este realizó mediante una exposición oral de su contenido, y luego respondió satisfactoriamente a las preguntas que les fueron formuladas por el Jurado, todo ello conforme a lo dispuesto en la Ley de Universidades y demás normativas vigentes de la Universidad Central de Venezuela. Finalizada la defensa pública del Trabajo Especial de Grado, el jurado decidió aprobarlo.

En fe de lo cual se levanta la presente acta, en Caracas el **XX** de abril de 2017.

Prof. Miguel Astor (Tutor)

Prof. **Jurado 1**
(Jurado principal)

Prof. Eugenio Scalise
(Tutor)

Prof. **Jurado 2**
(Jurado principal)

Agradecimientos

Gracias a mis tutores, los profesores Eugenio Scalise y Miguel Astor, por su apoyo y asesoría.

Gracias a los profesores David Pérez y Karima Velasquez, quienes me han acompañado y aconsejado desde que comencé la licenciatura.

Gracias a mis padres, hermanos, abuelos y tíos por su cariño incondicional.

Gracias a mis amigos Fernando Crema, Eric Bellet y Leonardo Santella por sus ideas y sugerencias en las estrategias a adoptar en la presente investigación.

Dedicado a mis abuelos, José De Alcántara y María Freitas.

Resumen

Título

Diseño, Implementación y Evaluación de Técnicas Híbridas de Aprendizaje Automático en la Detección de Intrusos en Redes de Computadoras.

Autor:

Deyban Pérez.

Tutores:

Prof. Eugenio Scalise, Prof. Miguel Astor.

Las redes de computadoras han sido firmemente aceptadas en la sociedad y corresponden a un aspecto fundamental en la vida de millones de personas alrededor del mundo. El crecimiento de las mismas ha generado nuevos paradigmas tales como lo son la computación en la nube o el Internet de las cosas, que demandan nuevas estrategias de seguridad que garanticen a los usuarios un servicio confiable y seguro. Como estrategia revolucionaria en el área de seguridad informática ha surgido la tendencia de introducir el aprendizaje automático, buscando el aumento de la efectividad a la hora de detectar ataques. Por lo expuesto previamente, en el presente trabajo de investigación se realizó un estudio minucioso de las tendencias de investigación entre las áreas de ML y seguridad en redes de computadoras, seleccionando un conjunto de datos bien conocido como lo es el conjunto de datos NSL-KDD para analizar, diseñar e implementar modelos híbridos de ML combinando las técnicas de enfoque supervisado de redes neuronales y máquinas de vectores de soporte en conjunto con la técnica de enfoque no-supervisado K-Medias, obteniendo resultados que indican que el uso de ambos enfoques como complemento incrementa la cantidad de ataques detectados.

Palabras clave:

Redes de computadoras, seguridad en redes de computadoras, ataques, aprendizaje automático, modelos híbridos de aprendizaje automático, redes neuronales, máquinas de vectores de soporte.

Índice general

Índice de figuras	xii
Índice de tablas	xiii
1. Introducción	1
1.1. Objetivo general	3
1.2. Objetivos específicos	3
1.3. Justificación	3
1.4. Distribución del documento	4
2. Marco teórico	5
2.1. Aprendizaje automático (ML - <i>Machine Learning</i>)	5
2.2. Técnicas de aprendizaje automático	5
2.2.1. Flujo de trabajo	13
2.2.2. Herramientas de software	21
2.3. Estado del arte	22
2.3.1. NIDS basados en técnicas de ML	22
2.3.2. Objetivos de la aplicación de técnicas de ML en NIDS	26
2.3.3. Flujo general de trabajo en la implementación de un NIDS utilizando técnicas de ML	26
2.3.4. Herramientas utilizadas	35
2.3.5. Consideraciones en la utilización de técnicas de ML en la implementación de un NIDS	35
3. Marco aplicativo	39
3.1. Metodología	39
3.2. Consideraciones de diseño	39
3.2.1. Flujo de entrenamiento	41
3.2.2. Flujo de prueba	41
3.3. Consideraciones de implementación	42
3.4. Infraestructura de la solución	45

4. Análisis e interpretación de resultados	47
4.1. Recolección de los datos	47
4.2. Pre-procesamiento	47
4.2.1. Análisis exploratorio	48
4.2.2. Extracción de características	48
4.2.3. Renombramiento de columnas	50
4.2.4. Eliminación de características	50
4.2.5. Transformación de los datos	50
4.2.6. Generación de la vista minable	51
4.3. Implementación de modelos	51
4.3.1. Iteración 1	51
4.3.2. Iteración 2	60
4.3.3. Iteración 3	79
4.3.4. Iteración 4	87
4.4. Trabajos relacionados	96
5. Conclusiones	99
5.1. Contribuciones	101
5.2. Limitaciones	101
5.3. Trabajos futuros	102
Referencias	103

Índice de figuras

2.1. Modelo de SVM	8
2.2. Modelo de arquitectura de red neuronal	10
2.3. Agrupamiento en un conjunto con tres clases usando K-Medias	11
2.4. Codo de Jambu ideal	12
2.5. Flujo general del proceso de ML	13
2.6. Ejemplo de validación cruzada de K-Conjuntos	17
2.7. Ejemplo de curva ROC	21
2.8. Modelo genérico de NIDS supervisado	23
2.9. Modelo genérico de NIDS no-supervisado	25
2.10. Modelo genérico de NIDS híbrido	26
2.11. Flujo general de un NIDS basado en técnicas de ML	27
2.12. Características de KDD99	30
2.13. Distribución de aciertos en el conjunto de entrenamiento de KDD99	32
2.14. Distribución de aciertos en el conjunto de prueba de KDD99	33
3.1. Flujo de entrenamiento utilizado	42
3.2. Flujo de prueba utilizado	43
4.1. Distribución de clases en el conjunto de entrenamiento	49
4.2. Distribución de clases en el conjunto de prueba	50
4.3. Codo de Jambu sobre el conjunto de datos (Iteración 1)	53
4.4. Curvas ROC de los algoritmos del primer nivel de los modelos sobre el conjunto de datos de entrenamiento (Iteración 1)	57
4.5. Curvas ROC de los algoritmos del primer nivel de los modelos sobre el conjunto de datos de prueba (Iteración 1)	59
4.6. Varianza acumulada por componente principal en la aplicación de PCA (Iteración 2)	62
4.7. Graficación de las dos componentes principales del conjunto de datos de entrenamiento luego de la aplicación de PCA (Iteración 2)	63
4.8. Graficación de la desviación estándar y media de acierto por componente principal para NN (Iteración 2)	64
4.9. Graficación de la desviación estándar y media de acierto por componente principal para SVM (Iteración 2)	65

4.10. Graficación de las dos características principales del conjunto de datos de entrenamiento luego de la aplicación de GFR usando NN (Iteración 2)	66
4.11. Graficación de la desviación estándar y media de acierto por característica seleccionada para NN usando GFR (Iteración 2)	67
4.12. Graficación de las dos características principales del conjunto de datos de entrenamiento Luego de la Aplicación de GFR Usando SVM (Iteración 2) . .	68
4.13. Graficación de la desviación estándar y media de acierto por característica seleccionada para SVM usando GFR (Iteración 2)	69
4.14. Codo de Jambu sobre los conjuntos de datos (Iteración 2)	70
4.15. Curvas ROC de los algoritmos del primer nivel de los modelos sobre el conjunto de datos de entrenamiento (Iteración 2)	74
4.16. Curvas ROC de los algoritmos del primer nivel de los modelos sobre el conjunto de datos de prueba (Iteración 2)	77
4.17. Codo de Jambu sobre los conjuntos de datos (Iteración 3)	80
4.18. Curvas ROC de los algoritmos del primer nivel de los modelos sobre el conjunto de datos de entrenamiento (Iteración 3)	84
4.19. Curvas ROC de los algoritmos del primer nivel de los modelos sobre el conjunto de datos de prueba (Iteración 3)	86
4.20. Curva ROC de NN sobre el conjunto de datos entrenamiento (Iteración 4) . .	90
4.21. Curva ROC de NN sobre el conjunto de datos prueba (Iteración 4)	93

Índice de tablas

2.1. Ejemplo de matriz de confusión	19
2.2. NIDS basado en firma VS NIDS basado en anomalías	25
2.3. Aportes realizados en ML y NIDS en el período (2010 - 2015)	28
2.4. Registros redundantes en el conjunto de entrenamiento de KDD99	31
2.5. Registros redundantes en el conjunto de prueba de KDD99	31
2.6. Estadísticas de la selección de registros aleatorios del conjunto de entrenamiento de KDD99	33
2.7. Estadísticas de la selección de registros aleatorios del conjunto de prueba de KDD99	34
2.8. Algoritmos populares de ML en NIDS	35
2.9. Medidas de rendimiento más utilizadas	36
2.10. Herramientas de software más utilizadas	37
4.1. Resumen del conjunto de datos NSL-KDD	48
4.2. Distribución de las clases en el conjunto de datos NSL-KDD	49
4.3. Inercia inter-grupos del conjunto de datos (Iteración 1)	52
4.4. Matriz de confusión de cinco clases del mejor modelo K-Medias sobre el conjunto de datos de entrenamiento (Iteración 1)	53
4.5. Tasa de acierto de cinco clases del mejor modelo K-Medias sobre el conjunto de datos de entrenamiento (Iteración 1)	54
4.6. Matriz de confusión de dos clases del mejor modelo K-Medias de cinco clases sobre el conjunto de datos de entrenamiento (Iteración 1)	54
4.7. Matriz de confusión de dos clases del mejor modelo K-Medias de cinco clases sobre el conjunto de datos de entrenamiento (Iteración 1)	55
4.8. Comparación de las medidas de rendimiento binarias extraídas de los enfoques de cinco y dos grupos usando K-Medias sobre el conjunto de datos de entrenamiento (Iteración 1)	56
4.9. Tasas de acierto (5 Clases) del primer nivel de los modelos sobre el conjunto de datos de entrenamiento (Iteración 1)	56
4.10. Medidas de rendimiento binarias de los modelos sobre el conjunto de datos de entrenamiento (Iteración 1)	57
4.11. Tasas de acierto (5 Clases) del primer nivel de los modelos sobre el conjunto de datos de prueba (Iteración 1)	58

4.12. Medidas de rendimiento binarias de los modelos sobre el conjunto de datos de prueba (Iteración 1)	59
4.13. Ordenamiento de características para NN usando GFR (Iteración 2)	65
4.14. Ordenamiento de características para SVM usando GFR (Iteración 2)	67
4.15. Selección de parámetros para los modelos (Iteración 2)	69
4.16. Inercia inter-grupos de los conjuntos de datos (Iteración 2)	71
4.17. Tasa de acierto por clase de la matriz de confusión de cinco clases en el algoritmo K-Medias (Iteración 2)	72
4.18. Comparación de las medidas de rendimiento binarias extraídas de los enfoques de cinco y dos grupos usando K-Medias sobre el conjunto de datos de entrenamiento (Iteración 2)	72
4.19. Tasas de acierto (5 Clases) del primer nivel de los modelos sobre el conjunto de datos de entrenamiento (Iteración 2)	73
4.20. Medidas de rendimiento binarias de los modelos sobre el conjunto de datos de entrenamiento (Iteración 2)	75
4.21. Tasas de acierto (5 Clases) del primer nivel de los modelos sobre el conjunto de datos de prueba (Iteración 2)	76
4.22. Medidas de rendimiento binarias de los modelos sobre el conjunto de datos de prueba (Iteración 2)	78
4.23. Inercia inter-grupos de los conjuntos de datos (Iteración 3)	81
4.24. Tasa de acierto por clase de la matriz de confusión de cinco clases en el algoritmo K-Medias (Iteración 3)	82
4.25. Comparación de las medidas de rendimiento binarias extraídas de los enfoques de cinco y dos grupos usando K-Medias sobre el conjunto de datos de entrenamiento (Iteración 3)	82
4.26. Tasas de acierto (5 Clases) del primer nivel de los modelos sobre el conjunto de datos de entrenamiento (Iteración 3)	83
4.27. Medidas de rendimiento binarias de los modelos con parámetros seleccionados sobre el conjunto de datos de entrenamiento (Iteración 3)	84
4.28. Tasas de acierto (5 Clases) del primer nivel de los modelos sobre el conjunto de datos de prueba (Iteración 3)	85
4.29. Medidas de rendimiento binarias de los modelos sobre el conjunto de datos de prueba (Iteración 3)	87
4.30. Matriz de confusión (5 Clases) del mejor modelo de NN sobre el conjunto de datos de entrenamiento (Iteración 4)	89
4.31. Tasas de acierto (5 Clases) de NN sobre el conjunto de datos de entrenamiento (Iteración 4)	89
4.32. Matriz de confusión (2 Clases) del mejor modelo de NN sobre el conjunto de datos de entrenamiento (Iteración 4)	89
4.33. Matriz de confusión (2 Clases) de la aplicación de K-Medias sobre el conjunto de datos de entrenamiento (Iteración 4)	90

4.34. Matriz de confusión (2 Clases) del modelo híbrido GFR - NN - K-Medias sobre el conjunto de datos de entrenamiento (Iteración 4)	91
4.35. Medidas de rendimiento binarias de GFR - NN - K-Medias sobre el conjunto de datos de entrenamiento (Iteración 4)	91
4.36. Matriz de confusión (5 Clases) de NN sobre el conjunto de datos de prueba (Iteración 4)	92
4.37. Tasas de acierto (5 Clases) de NN sobre el conjunto de datos de prueba (Iteración 4)	92
4.38. Matriz de confusión (2 Clases) de NN sobre el conjunto de datos de prueba (Iteración 4)	93
4.39. Matriz de confusión (2 Clases) de la aplicación de K-Medias sobre el conjunto de datos de prueba (Iteración 4)	94
4.40. Matriz de confusión (2 Clases) del modelo híbrido GFR - NN - K-Medias sobre el conjunto de datos de prueba (Iteración 4)	94
4.41. Medidas de rendimiento binarias del modelo híbrido GFR - NN - K-Medias sobre el conjunto de datos de prueba (Iteración 4)	95
4.42. Resumen de la distribución de ataques detectados por nivel del modelo GFR - NN - K-Medias sobre el conjunto de datos de prueba (Iteración 4)	95

Capítulo 1

Introducción

Las redes de computadoras representan un aspecto fundamental en la vida de millones de personas alrededor del mundo. Servicios tales como: correo electrónico, banca, almacenamiento de información, entre otros; hacen uso de redes de computadoras como medio de comunicación para la conexión entre los diferentes nodos que componen a la red.

Con el crecimiento de las redes de computadoras surgieron algunas inquietudes con respecto al tema de seguridad de la información y de los servicios de los usuarios. En el año 1980, Anderson introdujo el concepto de seguridad informática [1]. La seguridad informática se refiere a la protección conferida a un sistema de información automatizado con el fin de alcanzar los objetivos principales de la preservación de la integridad, disponibilidad y confidencialidad de la información y los recursos del sistema [2]. Las violaciones a los principios básicos de seguridad descritos previamente son conocidas como ataques [3]. En particular, el término ataque hace referencia a las actividades que buscan dañar de manera directa a los sistemas informáticos.

Por otra parte, existen intrusos, estos buscan introducirse en la red para poder robar información. Los ataques e intrusos se basan en la búsqueda de vulnerabilidades en el sistema para poder lograr sus objetivos malintencionados. Ambas actividades son atípicas y suelen ser conocidas como anomalías [2, 4]. Los ataques pueden englobarse dentro de cuatro grandes clases:

- **DoS (*Denial of Service*):** clase de ataque donde el atacante busca saturar con sobreprocesamiento algún componente de hardware del sistema de cómputo para así deteriorar o detener los servicios ofrecidos por el mismo [5].
- **Probing:** esta clase de ataque busca escanear una red de computadoras para recolectar información o detectar vulnerabilidades conocidas. Un atacante con información acerca de las máquinas y los servicios que están disponibles puede usar esta información para explotar posibles vulnerabilidades [5].
- **R2L (*Remote to Local*):** clase de ataque que busca enviar ciertos paquetes a un

computador para reconocer cuentas de usuarios con la finalidad de poder tener acceso a la red de manera remota a una cuenta local [5].

- **U2R (*User to Root*):** esta clase de ataque inicia cuando un atacante accede a una cuenta local de manera remota (luego de un ataque R2L), luego mediante alguna vulnerabilidad del sistema es capaz de lograr permisos de administrador [5].

Para proteger a los sistemas informáticos de los ataques, se han propuesto e implementado diferentes mecanismos de seguridad. Uno de los artefactos de seguridad más populares dentro de una red de computadoras son los sistemas detectores de intrusos (IDS - *Intrusion Detection System*) [6]. Estos generalmente son utilizados como segunda línea de defensa por detrás de un corta fuegos (*firewall*). Las actividades principales de un IDS son las de examinar el tráfico entrante y saliente de una red, detectar anomalías y notificar en caso de que una anomalía haya sido detectada [3, 7].

Antiguamente, las reglas establecidas para la detección de anomalías en los IDS eran establecidas por expertos en el área de seguridad en redes de computadoras. Este enfoque trae como consecuencia que muchos de los ataques no fuesen detectados debido a la gran cantidad de datos generados en una red producto del número de elementos que la componen [8]. Los IDS pueden ser orientados a red (NIDS - *Network Intrusion Detection System*) u orientados a un computador en particular (HIDS - *Host Intrusion Detection System*). Este documento hará énfasis en los NIDS.

Recientemente se han realizado una gran cantidad de investigaciones que buscan automatizar el proceso de detección de intrusos en las redes de computadoras utilizando técnicas de aprendizaje automático (ML - *Machine Learning*) [9]. ML hace referencia a la habilidad que tiene un sistema de cómputo para generar conocimiento a partir de un conjunto de experiencias, sin que este comportamiento sea explícitamente programado [10]. Cualquier modelo de ML, sin importar el área al que esté orientado, está conformado por las siguientes fases en su flujo de trabajo: recolección de los datos, extracción de características, limpieza de los datos, ajuste de los datos, selección de características, validación del modelo, selección del modelo y evaluación del modelo [4].

Existen dos enfoques adoptados para los NIDS que utilizan aprendizaje automático. Por un lado están los NIDS basados en la firma del ataque [11]. Estos examinan las características particulares de los ataques, característica que los hace muy efectivos a la hora de detectar ataques conocidos, pero sin un buen acierto frente ataques no conocidos. Por otra parte, están los NIDS basados en anomalías. Estos establecen un perfil del comportamiento normal y analizan la desviación de los registros con respecto a dicho perfil para así poder clasificarlos. De esta manera, los NIDS basados en anomalías son capaces de lidiar con ataques conocidos y no conocidos; sin embargo, suelen tener altas tasas de falsas alarmas. De lo anterior se puede intuir que ambos enfoques pueden complementarse para disminuir sus carencias y mejorar la eficacia en la tarea de detección de intrusos en redes de computadoras.

Con lo descrito en el párrafo anterior surge la siguiente pregunta: ¿Es posible obtener un modelo de aprendizaje automático que pueda combinar técnicas basadas en la firma del ataque (aprendizaje supervisado) y técnicas basadas en anomalías (no supervisadas) con la meta de mejorar la eficacia en la tarea de detección de intrusos en redes de computadoras? Esta pregunta será respondida diseñando, implementando y analizando modelos híbridos que combinan ambos enfoques, utilizando como métrica base matrices de confusión que permitan medir el desempeño general y específico con respecto a cada clase de ataque.

1.1. Objetivo general

Analizar, Diseñar e implementar modelos híbridos basados en técnicas de aprendizaje automático mediante la evaluación de su desempeño en la tarea de detección de intrusos en redes de computadoras.

1.2. Objetivos específicos

- Indagar en el estado actual de la investigación del uso de técnicas de aprendizaje automático en la detección de intrusos en redes de computadoras.
- Diseñar, implementar modelos híbridos de detección de intrusos en redes de computadoras basados en técnicas de aprendizaje automático.
- Evaluar y comparar el desempeño general y específico de los modelos creados frente a los diferentes tipos de ataques descritos previamente haciendo uso de medidas de rendimiento.
- Evaluar la posibilidad de crear un modelo conjunto que combine los modelos creados e incremente el desempeño contra determinados tipos de ataque.
- Registrar los resultados y consideraciones encontradas durante la implementación de la propuesta.

1.3. Justificación

La justificación de este trabajo recae en la posibilidad de hacer investigación en un campo abierto de suma relevancia en la ciencia de la computación tal como lo es el uso de aprendizaje automático en la detección de intrusos en redes de computadoras. El aprendizaje automático ha tenido un auge en los últimos años y su aplicabilidad para la automatización de tareas es sumamente amplia. Adicionalmente, en el campo de redes de computadoras, existen tendencias tales como computación en la nube (CC - *Cloud Computing*) o el Internet de las cosas (IoT - *Internet of Things*), que demandan nuevas medidas de seguridad. Estas tendencias apuestan por la utilización de la nube (Internet) como plataforma principal de

servicios y de almacenamiento de información. Por esto, es necesario proveer servicios de seguridad acordes a las exigencias de los usuarios para así brindarles la confianza que los impulse a su uso. Por lo anterior, en los últimos años se ha propuesto el estudio de NIDS basados en ML, que permitan mejorar la seguridad en redes de computadoras.

Este trabajo puede servir como base para lineamientos de futuros Trabajos Especiales de Grado en la Escuela de Computación de la Facultad Ciencias en la Universidad Central de Venezuela relacionados a las áreas de Inteligencia Artificial, Minería de Datos, Seguridad en Redes de Computadoras y Redes de Computadoras. Adicionalmente, el mismo puede servir para implementar laboratorios en las materias mencionadas con anterioridad.

1.4. Distribución del documento

Una vez descrito el escenario, el presente documento busca responder las interrogantes planteadas previamente. Así mismo, el documento constará de cinco capítulos. El Capítulo 1, introduce el problema, los objetivos planteados y la justificación de la investigación. El Capítulo 2 detalla las bases teóricas necesarias para el correcto entendimiento del trabajo realizado, de tal manera, este incluye los conceptos básicos de ML y el correspondiente estado del arte entre ML y la seguridad en redes de computadoras. El Capítulo 3 establece la metodología y consideraciones a tomar en cuenta para la implementación de las actividades. El Capítulo 4 constituye el análisis e interpretación de los resultados de las actividades realizadas para alcanzar los objetivos planteados en la Secciones 1.1 y 1.2, adicionalmente, se presenta la comparación con trabajos relacionados. Por último, el Capítulo 5, presenta las conclusiones del trabajo realizado, describiendo los aportes logrados, limitaciones encontradas y planteamiento de trabajos futuros.

Capítulo 2

Marco teórico

Este capítulo presenta de manera concisa las bases teóricas necesarias para el correcto entendimiento de los siguientes capítulos que definen las actividades prácticas llevadas a cabo durante la investigación. El capítulo iniciará con la revisión de los conceptos básicos concernientes a ML y posteriormente se detallará el estado del arte entre ML y la seguridad en redes de computadoras.

2.1. Aprendizaje automático (ML - *Machine Learning*)

En el año 1959, Samuel [12] define el concepto de ML como el “campo de estudio que da a las computadoras la habilidad para aprender sin ser explícitamente programadas”. Otra definición popular para el área fue elaborada por Mitchell [10], quien lo definió de la siguiente manera, “se dice que un programa de computadora aprende de una experiencia **E** con respecto a alguna clase de tarea **T** donde su rendimiento es medido por **P** si su rendimiento para la tarea **T**, que es medido por **P**, mejora con la experiencia **E**”. ML es el estudio de los métodos para programar computadoras con la finalidad de aprender. Esta área utiliza los campos de la estadística, minería de datos y la psicología [4].

A continuación se presentarán las técnicas utilizadas en ML, los algoritmos populares, el flujo de trabajo general y las herramientas comúnmente usadas en el área.

2.2. Técnicas de aprendizaje automático

Los algoritmos de ML se clasifican en aprendizaje supervisado, aprendizaje no-supervisado y semi-supervisado [4]. La naturaleza y peculiaridad de cada uno de estos enfoques serán descritas a continuación.

Aprendizaje supervisado

Según este enfoque, el algoritmo necesita que los registros del conjunto de datos sean etiquetados por un maestro [4]. Sea X el conjunto total de registros en un conjunto de datos, donde x_i ($i = 1, \dots, n$) es el registro de la posición i de X y Y el conjunto de etiquetas en un conjunto de datos, donde y_i ($i = 1, \dots, n$) es la etiqueta de la posición i de Y . Entonces, para cada x_i debe existir un y_i ($i = 1, \dots, n$) perteneciente a Y tal que y_i etiquete a x_i [13].

Dentro de los métodos de aprendizaje supervisado existe una sub-división que depende de si el tipo de dato de las etiquetas del conjunto Y , son de tipo cualitativo o cuantitativo. Si Y es de tipo cualitativo el problema se trata de clasificación. Por otra parte, si Y es de tipo cuantitativo el problema se trata de regresión.

Algunas de las técnicas de aprendizaje supervisado más populares son las máquinas de vectores de soporte, las redes neuronales y los árboles de decisión para problemas de clasificación. Todas las técnicas descritas con anterioridad funcionan tanto para problemas de regresión como de clasificación. Este documento se centra en la aplicación de las técnicas de máquinas de vectores de soporte y de redes neuronales, específicamente en la aplicabilidad de los mismos para problemas de clasificación.

- **Máquina de vectores de soporte (SVM - *Support Vector Machine*)**: este algoritmo asocia los datos de entrada dentro de un espacio de características de dimensión superior y obtiene el hiperplano separador óptimo en el espacio de características de dimensión superior. El hiperplano separador es elegido maximizando la distancia entre los vectores de soporte, que corresponden a los puntos más cercanos al hiperplano separador generado [13]. Este enfoque hace al algoritmo robusto a registros inusuales (*outliers*) [14]. En la Figura 2.1 se puede apreciar como el hiperplano separa de manera óptima maximizando la distancia existente desde el hiperplano separador hasta los vectores de soporte.

En la Figura 2.1, se asume que la separación entre los conjuntos es lineal; sin embargo, este hecho no siempre es verdadero. Para manejar dichas situaciones, donde la separación de los datos pudiese tener un comportamiento curvo o circular, SVM utiliza una función de similaridad llamada kernel que permite conocer el resultado del producto punto entre los registros y el vector ortogonal sin que sea necesario conocer el valor individual entre cada par de registros [13]. Algunos de los kernel más populares son: lineal, radial, polinómico y sigmoide. Los mismos serán descritos a continuación.

- **Lineal**: establece separadores lineales entre los conjuntos, la función de este kernel viene dada por la Ecuación (2.1), donde u representa el vector de registros del conjunto de datos y v corresponde al vector ortogonal al vector u . Al hacer el producto punto entre dicho vectores se obtiene la proyección del vector v sobre el vector u que se usa como margen de separación entre las diferentes clases. Por

otra parte, c corresponde al parámetro de regularización que es el que controla el margen de error de la clasificación. Un valor pequeño de c genera un modelo flexible no sensible a ruido, mientras que un valor alto de c genera un modelo más rígido con mayor varianza [4, 13].

$$f(u, v, c) = \langle u^T, v \rangle + c \quad (2.1)$$

- **Polinómico:** establece separadores polinómicos que permiten establecer líneas curvas como separadores entre los conjuntos, la función de este kernel viene dada por la Ecuación (2.2), donde u , v y c tienen la misma representación explicada en la Ecuación (2.1). Por otra parte, el parámetro γ indica la distancia máxima a la que deben estar los registros para que estén relacionados entre si, $coef0$ funciona como parámetro para el escalamiento de los datos y $degree$ corresponde al grado del polinomio que se desea utilizar [4, 13].

$$f(\gamma, u, v, coef0, degree, c) = (\gamma * \langle u^T, v \rangle + coef0)^{degree} + c \quad (2.2)$$

- **Radial:** este kernel permite establecer fronteras circulares para la separación de conjuntos, la función de este kernel viene dada por la Ecuación (2.3), donde los parámetros u , v y γ tienen la misma representación explicada en la Ecuación (2.1) y en la Ecuación (2.2) [4, 13].

$$f(\gamma, u, v, c) = e^{-\gamma * \|u - v\|^2} + c \quad (2.3)$$

- **Sigmoide:** este kernel al igual que el radial permite establecer fronteras circulares entre los conjuntos, la función de kernel sigmoide viene dada por la Ecuación (2.4), donde los parámetros tienen la misma representación presentada en la Ecuación (2.1) y en la Ecuación (2.2) [4, 13].

$$f(u, v, coef0, c) = \tanh(\gamma * u^T * v + coef0) + c \quad (2.4)$$

Los algoritmos SVM sirven tanto para problemas de regresión como de clasificación y este uso dependerá de si la variable objetivo es de tipo cualitativo o cuantitativo.

- **Redes neuronales (NN - Neural Network):** la motivación para la utilización de las redes neuronales viene en el proceso del modelado del comportamiento del cerebro humano, el cual hace uso de estructuras llamadas neuronas, estas reciben y transmiten información a otras neuronas mediante un proceso de comunicación llamado sinapsis permitiéndole al cerebro ejecutar operaciones complejas a grandes velocidades basándose en conocimiento adquirido previamente [4].

El proceso de modelado de una NN en una computadora pasa por definir un conjunto de neuronas y ensamblarlas en un conjunto de capas. Elegir la arquitectura para una

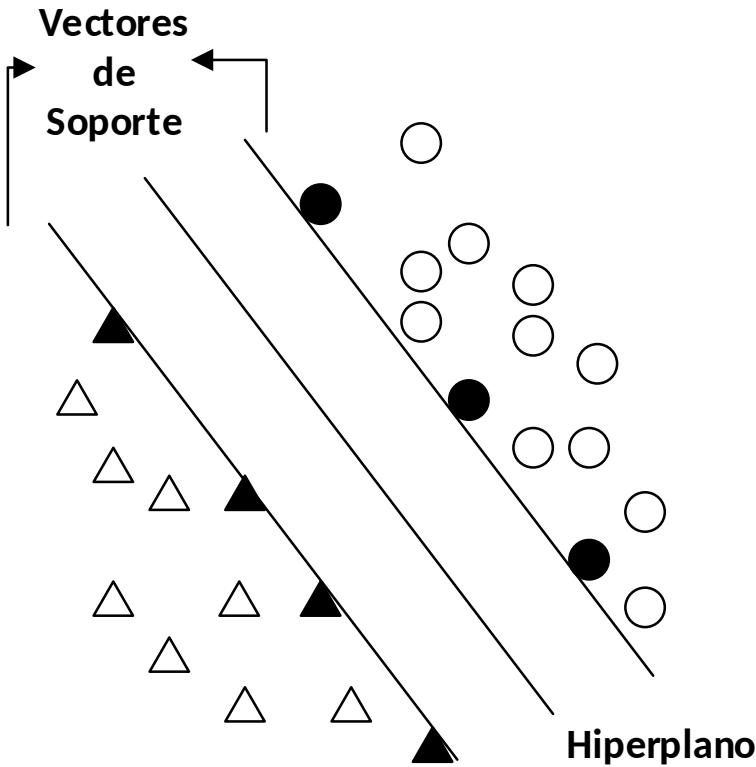


Figura 2.1: Modelo de SVM [4].

red neuronal es un proceso de ensayo y error que no está normado en ninguna bibliografía. Una arquitectura con una capa intermedia con el doble de neuronas de entrada (si el número de neuronas de entrada es menor a 20), o la mitad de las mismas (si el número de neuronas de entrada es superior a 20), es suficiente para resolver una amplia variedad de problemas. También se recomienda que al usar más de una capa intermedia se utilicen dos, esta segunda capa con la misma cantidad de neuronas que la primera capa intermedia. El uso de más de dos capas intermedias se considera poco práctico debido a que incrementará la complejidad de la red neuronal, situación que puede conllevar a situaciones de sobre ajuste e incremento de los tiempos de procesamiento en las fases de entrenamiento y de evaluación del modelo¹.

Las neuronas son entrenadas mediante un método compuesto de dos pasos que reciben los nombres de propagación hacia adelante y propagación hacia atrás. Estos métodos van ajustando los pesos de las relaciones entre neuronas para lograr el mejor desempeño posible. Entre mayor cantidad de neuronas haya, es posible realizar operaciones más complejas; sin embargo, esto repercute directamente en la complejidad del modelo y en el tiempo necesario para el entrenamiento [4]. Las redes neuronales son utilizadas para

¹Estas recomendaciones fueron realizadas por el profesor Andrew Ng en una de clases del curso de Machine Learning ofrecido en [Coursera](#).

problemas de clasificación y regresión; la aplicación de un enfoque u otro dependerá de si la salida del algoritmo es de tipo cualitativo o cuantitativo.

Algunos de los tipos de redes neuronales más populares son listados a continuación [15, 16]:

- **Red neuronal dinámica:**

- Red neuronal anticipativa (*FNN - Feedforward Neural Network*).
- Red neuronal probabilística (*PNN - Probabilistic Neural Network*).

- **Red neuronal estática:**

- Perceptron.
- Red neuronal modular.

- **Red de memoria:**

- Máquina neural de Turing.
- Memoria asociativa de un disparo.
- Memoria asociativa holográfica.

En la Figura 2.2 se presenta un ejemplo de una arquitectura de red neuronal con cinco capas, donde se señalan todos los elementos mencionados previamente. La cantidad de neuronas de entrada corresponde a la cantidad de variables predictoras a utilizar. Por otra parte la cantidad de capas ocultas y neuronas por capas ocultas y de salida, dependerá de la complejidad de la tarea que se desee realizar y del número de clases objetivo en el escenario para el cual fue modelada la red.

Las redes neuronales son muy versátiles y pueden ser usadas para múltiples propósitos tales como agrupamiento, clasificación, selección de características y detección de ataques en el contexto de seguridad en redes de computadoras.

Aprendizaje no-supervisado

A diferencia del aprendizaje supervisado, el aprendizaje no-supervisado no necesita de una etiqueta que identifique a cada registro para ser entrenado. Este enfoque se concentra en buscar patrones en el conjunto de datos que permitan describir al conjunto de datos. Se llama no-supervisado debido a la ausencia de una característica objetivo que ejerce el papel de maestro en el aprendizaje supervisado [13].

Los algoritmos de clasificación no-supervisados tienen como misión identificar patrones que permitan crear grupos separables en los conjuntos de datos. Uno de los enfoques es conocido como agrupamiento o análisis de grupos. En la Figura 2.3 se presenta un caso

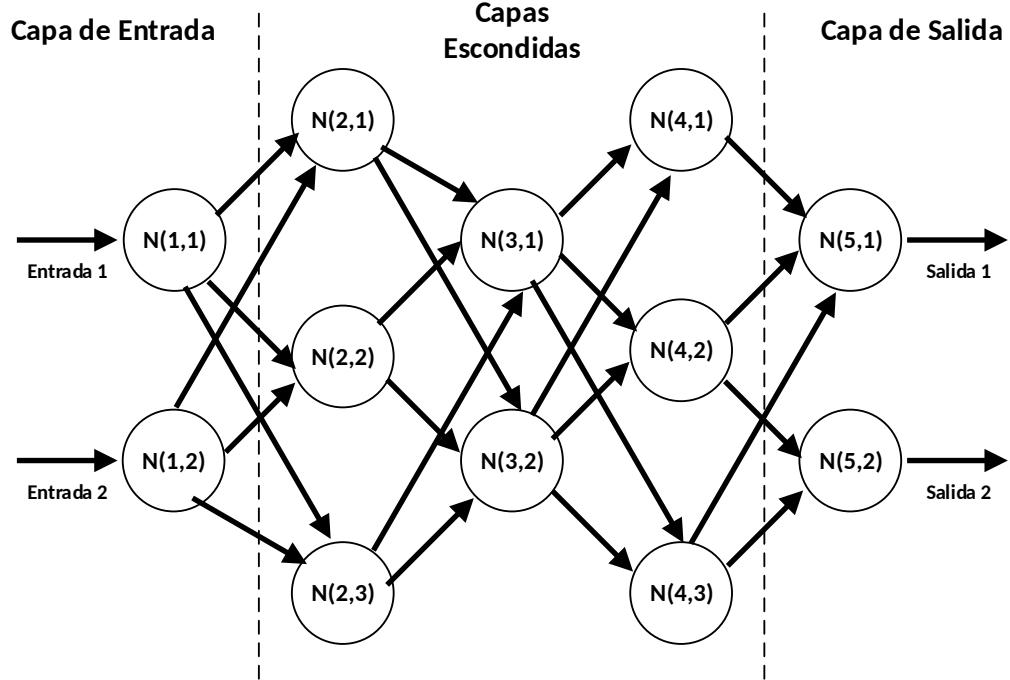


Figura 2.2: Modelo de arquitectura de red neuronal.

donde existen tres clases distintas que pueden ser identificadas mediante una figura circular. Existen diferentes algoritmos de agrupamiento que se dividen en la forma en la que los grupos son creados, estos son: basados en centroides, basados en medoides, basados en densidad, basados en cuadrículas y basados en jerarquía [4]. A continuación se describe el algoritmo K-Medias, que corresponde al enfoque basado en centroides.

- **K-Medias:** Es uno de los algoritmos más populares de agrupamiento basado en centroides. Divide los datos en un conjunto de K sub-conjuntos de tal manera que todos los puntos en un sub-conjunto están cercanos al mismo centroide. Al inicio, el algoritmo coloca aleatoriamente los K centroides y los datos son absorbidos por el centroide más cercano. La posición del centroide es recalculada sacando la media de las posiciones de los datos pertenecientes a un mismo centroide. El paso anterior es repetido hasta que se alcanza un criterio de convergencia [4]. En el Algoritmo 1 se ilustra el algoritmo de esta técnica.

El algoritmo K-Medias presenta una serie de propiedades y limitantes que fueron presentadas por Bhattacharyya de la siguiente manera [4]:

- **Propiedades**

- Es escalable y puede manejar grandes volúmenes de datos.
- Suele converger a un mínimo local.

- Es capaz de detectar grupos de forma esférica o convexa.

- **Limitantes**

- El número de grupos debe ser especificado como parámetro. Situación que no siempre es sabida de antemano.
- Sensible al ruido (*outliers*).
- Dependiendo de la posición inicial de los centroides se pueden obtener diferentes resultados.

Algoritmo 1 K-Medias

```

1: procedure KMEDIAS(Datos d, Clusters k)
2:   posicionCentroides = InicializarCentroides(k)
3:   while (!convergencia) do
4:     distancias = CalcularDistancias(d, posicionCentroides)
5:     clases = AsignarClusters(distancias)
6:     posicionCentroides = RecacularCentroides(d, clases)
7:   end while
8:   Retornar(clases)
9: end procedure

```

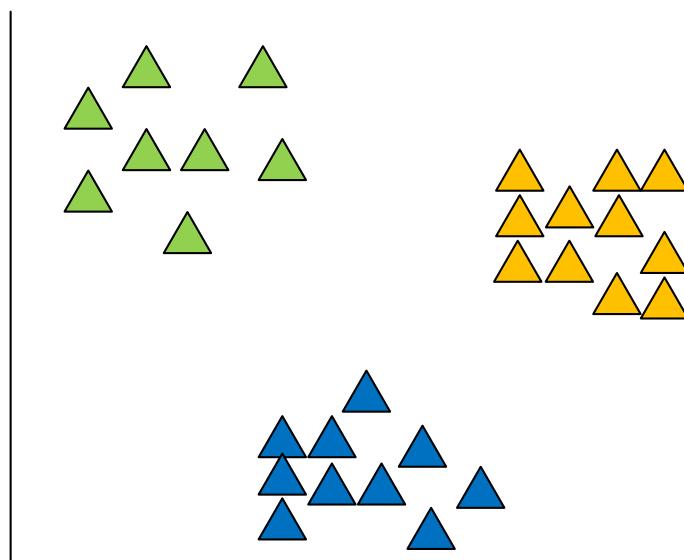


Figura 2.3: Agrupamiento en un conjunto con tres clases usando K-Medias.

- **Codo de Jambu**

Previamente se mencionó que una de las limitantes de K-Medias es que el número de grupos a ser agrupados debe ser pasado como parámetro. El Codo de Jambu es una de las técnicas más populares para la selección del número de grupos para el algoritmo de K-Medias [17]. Este método de selección de grupos itera probando diferentes números de grupos, calcula la distancia intra-grupos (Distancia entre los elementos pertenecientes a los grupos formados) y las grafica. Lo que se desea observar con esta estrategia es que a medida que se agregan mayor cantidad de clases, la distancia intra-grupos va disminuyendo hasta cierto punto en el que la misma se estabiliza formando una curvatura como la de un codo como la que es presentada en la Figura 2.4, donde se puede deducir que con cuatro clases se alcanza el número ideal de grupos a ser detectados. Con el criterio de la distancia intra-grupos se busca que la distancia entre los elementos de los diferentes grupos creados sea la más pequeña posible.

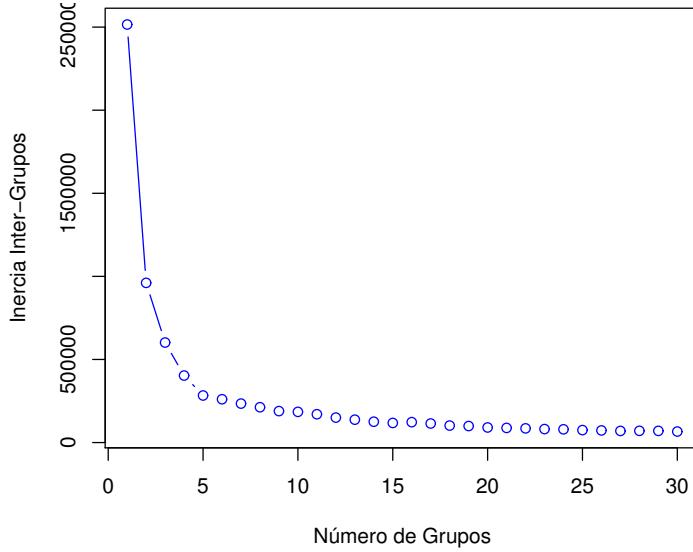


Figura 2.4: Codo de Jambu ideal.

Aprendizaje semi-supervisado

Este enfoque consta de datos etiquetados y no etiquetados. La cantidad de datos no etiquetados supone mayoría en el conjunto de datos y la cantidad de datos etiquetados corresponde a un pequeño sub-conjunto del mismo. La utilización de este enfoque generalmente involucra que los datos no etiquetados sean usados para el entrenamiento del algoritmo de manera similar al enfoque no-supervisado (ver Sección 2.2). Posteriormente, los datos etiquetados son utilizados para probar la eficacia del modelo [10, 4].

2.2.1. Flujo de trabajo

El proceso de ML consta de múltiples fases. A grandes rasgos estas fases son recolección de los datos, extracción de características, limpieza de los datos, ajuste de los datos, selección de características, construcción del modelo, validación del modelo y evaluación del modelo. En la Figura 2.5 se puede observar un flujo de trabajo ideal que recopila todas las fases mencionadas previamente. Es importante resaltar el hecho de que el flujo de ML es iterativo y por lo tanto es posible retroceder en cualquier punto hacia una fase previa. A continuación se describen las diferentes fases involucradas en el proceso mencionado.

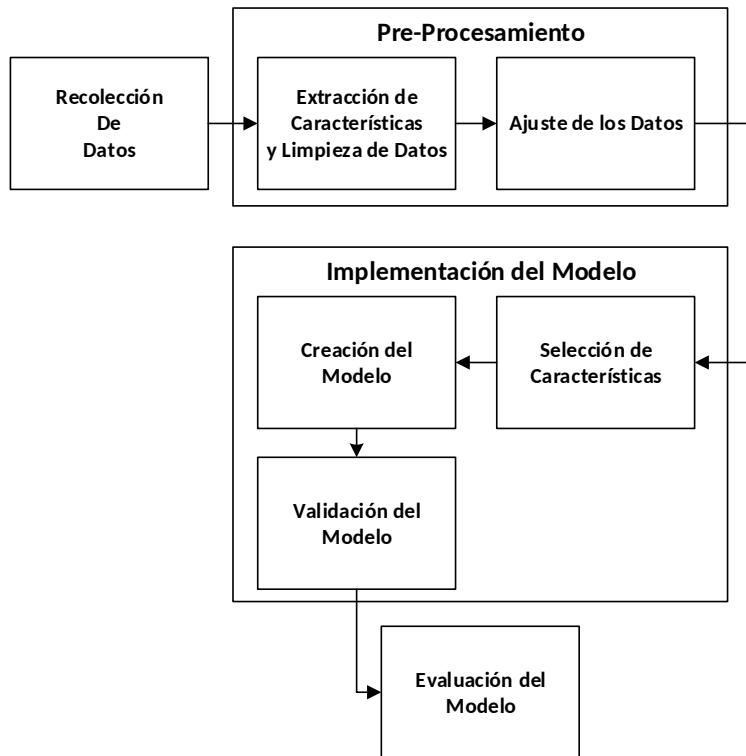


Figura 2.5: Flujo general del proceso de ML.

Recolección de los datos

Esta fase generalmente involucra el uso de un dispositivo especializado de hardware como un sensor de red, trabajo laboral como lo es la recolección de encuestas, o herramientas de software que permitan la recolección de información. Esta fase es altamente dependiente de la aplicación del modelo y usualmente es ajena al especialista de ML. Esta fase es crítica, debido a que buenas decisiones a la hora de recolectar los datos tendrán un impacto significativo en los resultados. Por lo general, luego de que los datos son recolectados se guardan en una base de datos o en un almacén de datos [18].

Extracción de características y limpieza de los datos

Por lo general cuando los datos son recolectados poseen un formato que no es el adecuado para realizar un proceso de ML. Estos se encuentran sin formato, poseen valores faltantes o contienen características de las que se puede derivar información. Una de las tareas más importantes en el proceso de ML es la generación de una vista minable que corresponde a una versión del conjunto de datos organizada y documentada que facilita la manipulación y entendimiento de la información [18].

En esta fase se hace reducción o expansión de las características presentes en el conjunto de datos, proceso que corresponde a la extracción de características; además, es necesario eliminar los registros que estén incompletos o completar los datos faltantes en los campos. Esta actividad se conoce como limpieza de los datos.

Ajuste de los datos

Una vez que la vista minable es creada y los datos tienen un formato adecuado es necesario hacer ciertos ajustes sobre los datos para garantizar que los mismos no causen ruido al modelo de ML durante su entrenamiento. El ajuste de los datos se refiere a la correcta organización de los datos con respecto a los rangos y tipos de datos de las variables predictoras y objetivo. Por ejemplo, los algoritmos de redes neuronales (ver Sección 2.2) sólo admiten variables predictoras de tipo numérico. Dependiendo del tipo de dato de la variable objetivo el algoritmo realizará regresión o clasificación [13]. A continuación se desarrollan los dos métodos más comunes para el ajuste de los datos que corresponden a transformación y estandarización de los datos.

- **Transformación:** esta técnica consiste en convertir los tipos de datos de las diferentes características presentes en el conjunto de datos a un tipo de dato diferente. Existen algoritmos que sólo admiten tipos de datos específicos y hacer este paso es obligatorio. Generalmente la situación más común es la de transformar las características de tipo cualitativo a tipo cuantitativo [19].
- **Estandarización:** esta técnica es utilizada en los tipos de datos cuantitativos donde los rangos entre las diferentes características poseen diferentes escalas. Mediante el proceso de estandarización, se busca llevar a todas las características a un mismo rango de valores donde la desviación estándar sea uno (1) y la media cero (0). Mediante la aplicación de esta técnica los algoritmos que hacen uso de medidas de distancia y de peso pueden ser más precisos, y acelerar los tiempos de entrenamiento y respuesta de los mismos [20].

Selección de características

En este punto los datos ya tienen un formato adecuado para ser utilizados por los diferentes algoritmos. A continuación se deben elegir las características a utilizar. Este paso es

importante debido a que las características que no aportan información suelen agregar ruido a los diferentes modelos, deteriorando su desempeño. Adicionalmente, seleccionando las características relevantes se decrementa la cantidad de características del conjunto de datos, reduciendo su dimensionalidad y acelerando el proceso de entrenamiento del mismo. A continuación se presentan dos de las técnicas más populares y efectivas a la hora de seleccionar las características relevantes de un conjunto de datos como lo son PCA y GFR.

- **Análisis de componentes principales (PCA - *Principal Component Analysis*):** este método rota el conjunto de datos para capturar la mayor cantidad de varianza en un conjunto de características reducido [18]. La matriz de covarianza es calculada para lograr este propósito. Las primeras columnas de esta matriz son las que acumulan mayor varianza y por ello son las más importantes. Por otra parte las últimas columnas son las que acumulan menor cantidad de varianza y son las menos relevantes.

PCA trata de representar el conjunto de los datos mediante la combinaciones lineales de las características. Esta técnica funciona bien cuando el número de dimensiones del conjunto de datos es grande y el conjunto de datos puede ser representado de buena manera por las primeras componentes de la matriz de covarianza. Un buen criterio para seleccionar el número de características es elegir el conjunto de componentes principales que acumulen varianza en un rango que va desde 95 % a 99 %. Este método corresponde a un técnica de aprendizaje no-supervisado y también es útil para hacer análisis exploratorio de los datos [13].

- **Eliminación gradual de características (GFR - *Gradually Feature Removal*):** este método toma el conjunto total de características y va eliminando de manera temporal alguna de ellas mientras evalúa el desempeño de cada combinación posible. Luego, se guarda la combinación de características que haya arrojado el mejor resultado y se repite el paso anterior con la mejor combinación obtenida. Este paso es repetido hasta que sólo quede una característica. Al final se tendrán N sub-conjuntos y debe evaluarse cada uno para elegir aquel que mejor se adapte a las necesidades del problema [21]. En el Algoritmo 2 se muestra un segmento de código donde se ilustra el funcionamiento del algoritmo. El algoritmo recibe como entrada un conjunto de datos D y retorna un vector con los nombres de las características más relevantes, donde las primeras posiciones representan las características más relevantes y las últimas posiciones representan las características menos relevantes.

Construcción del modelo

Esta fase es un reto debido a que no todos los problemas de ML son iguales. Que un problema sea de clasificación no significa que tenga la misma solución que algún otro problema de la misma índole, por lo que es imposible generalizar una solución y cada escenario es un reto diferente. La construcción del modelo amerita del análisis del problema y de la selección adecuada del algoritmo o de los algoritmos que mejor se adapten al problema actual

Algoritmo 2 GFR

```
1: procedure GFR(Datos D)
2:   vectorCaracteristicas = vector(tamaño = numeroColumnas(D))
3:   datosAuxiliar = D
4:   for (i = longitud(vectorCaracteristicas); i > 0; i = i-1) do
5:     peorResultado = ∞
6:     for (j = 1; i <= numeroColumnas(datosAuxiliar); j = j+1) do
7:       conjuntoTemporal = datosAuxiliar[,-j]
8:       resultadoTemporal = CalcularTasaAcierto(conjuntoTemporal)
9:       if (peorResultado > resultadoTemporal) then
10:         peorCaracteristica = j
11:         peorResultado = resultadoTemporal
12:       end if
13:     end for
14:     vectorCaracteristicas[i] = nombres(datosAuxiliar)[peorCaracteristica]
15:     datosAuxiliar = datosAuxiliar[,-peorCaracteristica]
16:   end for
17:   Retornar(vectorCaracteristicas)
18: end procedure
```

[18]. Para enfrentar esta fase es necesario saber la naturaleza de los algoritmos, así como sus ventajas y desventajas. Adicionalmente, este proceso muchas veces suele ser de ensayo y error, seleccionando algún algoritmo que se piensa que tendrá un buen desempeño frente a un escenario dado. Una vez que el modelo es seleccionado, se procede al entrenamiento del mismo.

Validación del modelo

La validación de los modelos juega un rol importante en el flujo de trabajo, debido a que no siempre se cuenta con datos de prueba para evaluar el desempeño del mismo. La validación del modelo permite tener una idea del desempeño que podría tener un modelo en la fase de entrenamiento. De esta manera se pueden hacer correcciones sobre los criterios adoptados de una forma más temprana [13].

El caso más común de uso de la validación de modelos viene dado para la selección de los parámetros que toman como entrada algunos algoritmos. Mediante un método de validación se puede tener una idea del desempeño de un modelo adoptando cierto parámetro, permitiendo así seleccionar la configuración que presente una mejor eficacia o desempeño. El método de validación de conjunto más popular es llamado validación cruzada de K-conjuntos (*K-Fold Cross Validation*), y será descrito a continuación.

- **Validación cruzada de K-conjuntos:** este método divide el conjunto de entrenamiento en K sub-conjuntos de aproximadamente el mismo tamaño. El modelo es en-

trenado con los datos pertenecientes a $K-1$ sub-conjuntos y se evalua el desempeño con el sub-conjunto que fue excluído inicialmente. En una siguiente iteración el sub-conjunto excluído pasa a formar parte del conjunto de entrenamiento y otro de los K sub-conjuntos que no haya sido excluído previamente para el entrenamiento es excluído en esta oportunidad. Este proceso es realizado K veces. Al finalizar el mismo, se retorna el modelo que haya presentado mejor desempeño a la hora de realizar la tarea, es decir, la solución que haya presentado menor error. En la Figura 2.6 se presenta un ejemplo de validación cruzada.

Para acelerar el tiempo de procesamiento de la aplicación de la técnica de validación cruzada, es común utilizar un sub-conjunto de los datos, es decir, utilizar una muestra de los datos que sea capaz de representar de buena manera al conjunto de datos original pero en una escala mucho menor con respecto a su tamaño. Las técnicas utilizadas para seleccionar sub-conjuntos de datos son llamadas técnicas de muestreo.

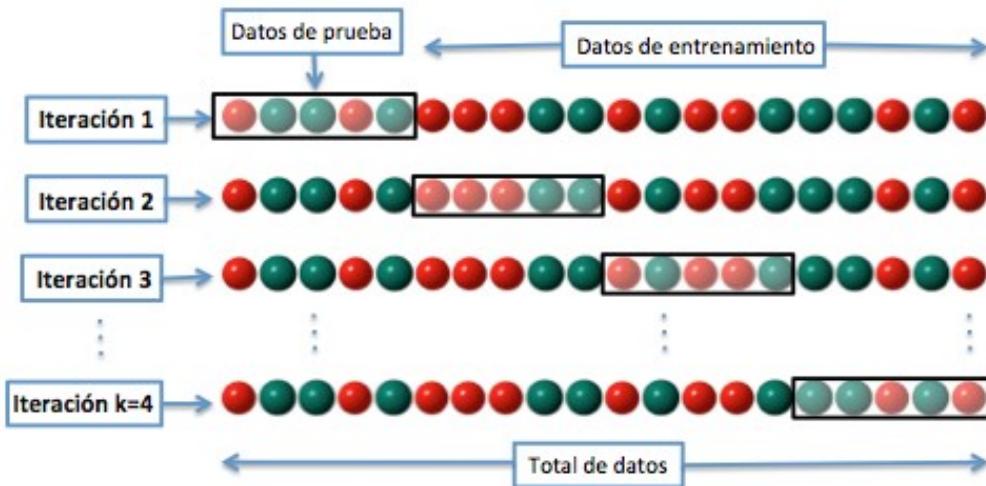


Figura 2.6: Ejemplo de validación cruzada de K-Conjuntos.
(Fuente: [Wikipedia](#)).

- **Técnicas de muestreo:** el entrenamiento de los datos es más costoso computacionalmente conforme el tamaño y las dimensiones del conjunto de datos incrementan. Previamente se vio que para reducir la cantidad de características se pueden utilizar algoritmos como PCA o GFR. Con las técnicas de muestreo se busca reducir la cantidad de registros presentes en un conjunto de datos seleccionando una cantidad reducida de registros que logren formar un sub-conjunto que represente de buena manera al conjunto de datos original. Algunas de las técnicas de muestreo son muestreo aleatorio y muestro estratificado.
 - **Muestro aleatorio:** se seleccionan los registros para el sub-conjunto de manera aleatoria. Este método suele ser efectivo cuando se tiene un conjunto de datos

balanceado; es decir, que hay una cantidad similar de registros pertenecientes a las diferentes clases en el conjunto de datos. Si el conjunto de datos no está balanceado, entonces en el sub-conjunto la relación entre las clases se verá deteriorada, situación no deseada a la hora de entrenar el modelo de ML debido a que este quedará sesgado por la(s) clase(s) dominante(s).

- **Muestreo estratificado:** con este método los registros son seleccionados de tal manera que la proporción entre las diferentes clases presentes en el conjunto de datos se mantenga en el sub-conjunto producto del muestreo. Por ejemplo, si un conjunto de datos contiene dos clases A y B , con una proporción de 3:1 de A sobre B (por cada tres registros de A hay uno presente de B), entonces en el sub-conjunto derivado de la aplicación del muestreo estratificado debe mantenerse la misma proporción entre las clases A y B . Esta técnica busca corregir las desventajas del muestreo aleatorio.

Anteriormente se mencionó el hecho de que el sub-conjunto de datos obtenido de la aplicación de alguna de las técnicas de muestreo debe representar de buena manera al conjunto total de los datos. Saber si un sub-conjunto cumple con la premisa elaborada anteriormente es complicado debido a que no hay manera de saber esto de antemano. Es por esto que el proceso de muestreo se repite una cantidad N de veces hasta que el promedio de los resultados de los modelos converja. Esta técnica es respaldada por la Ley de Grandes Números, que establece que al realizar un mismo experimento muchas veces se alcanza un promedio de resultados cercano al valor esperado [22].

Evaluación del modelo

Una vez que el modelo es seleccionado entonces se debe probar su desempeño. El error obtenido a la hora de evaluar el modelo con un conjunto de datos diferente al conjunto de entrenamiento es llamado error de prueba. Por otro lado, es posible evaluar el modelo con los mismos datos usados para el entrenamiento. El error obtenido en este caso es llamado error de entrenamiento. Este último enfoque no es una buena medida de desempeño del modelo debido a que por lo general se sobreestima el rendimiento debido a que los registros utilizados para evaluar fueron vistos con anterioridad por el modelo de ML. Utilizar un conjunto de datos diferente al de entrenamiento para evaluar el modelo de ML permite evaluar la capacidad de generalizar del mismo, y de esta manera se puede tener una noción más certera del desempeño del modelo en un ambiente real.

Para poder calcular el error de un modelo se debe comparar la salida predicha por el mismo contra el valor verdadero del registro. Esto es, comparar el valor estimado contra el valor real. Para dicha tarea es necesario conocer la salida real del problema y eso amerita que los datos sean etiquetados, situación que como se mencionó previamente es costosa en tiempo y dinero. Sin embargo, esto brinda la oportunidad de poder hacer cálculo de ciertas medidas de rendimiento que ayudan a la evaluación de los modelos. A continuación se presentan métodos para evaluar los modelos basados en matrices de confusión.

- **Matriz de confusión:** método que permite visualizar el desempeño de un modelo en el ámbito de problemas de clasificación a través de cuatro criterios representados en una tabla. Los cuatro criterios son verdaderos positivos, verdaderos negativos, falsos positivos, falsos negativos.

- **Verdaderos positivos (TP - True Positive):** aquellos registros que pertenecen a la clase objetivo y fueron clasificados por el modelo como la clase objetivo.
- **Verdaderos negativos (TN - True Negative):** aquellos registros que no pertenecen a la clase objetivo y que fueron clasificados por el modelo como la clase no objetivo.
- **Falsos positivos (FP - False Positive):** aquellos registros que no pertenecen a la clase objetivo que fueron clasificados como pertenecientes a la clase objetivo.
- **Falsos negativos (FN - False Negative):** aquellos registros que pertenecen a la clase objetivo que fueron clasificados como pertenecientes a la clase no objetivo.

Un ejemplo de como debe verse una matriz de confusión es presentando en la Tabla 2.1. En dicho ejemplo se puede ver como los elementos pertenecientes a la diagonal fueron el número de registros que fueron clasificados correctamente. Por otro lado los elementos que están fuera de la diagonal fueron clasificados de manera incorrecta. De la matriz de confusión se pueden derivar algunas medidas de rendimiento que serán listadas a continuación.

Real\Predicción	Positivo	Negativo
Positivo	Verdadero Positivo	Falso Negativo
Negativo	Falso Positivo	Verdadero Negativo

Tabla 2.1: Ejemplo de matriz de confusión [4].

- **Tasa de acierto:** la fórmula de la tasa de acierto viene dada por la Ecuación (2.5):

$$TasaAcierto = \frac{TP + TN}{N} \quad (2.5)$$

Donde TP = verdaderos positivos, TN = verdaderos negativos, N = número total de registros presentes en la evaluación. Esta medida de rendimiento, al multiplicarse por 100 representa el el porcentaje de acierto a la hora de clasificar, es decir, calcula el porcentaje de los elementos presentes en la diagonal de la matriz de confusión.

- **Tasa de error:** la fórmula de la tasa de error viene dada por la Ecuación (2.6):

$$TasaError = 1 - TasaAcierto \quad (2.6)$$

Esta medida al multiplicarse por 100 representa el porcentaje de registros clasificados de manera errónea, es decir, calcula el porcentaje de los elementos que están fuera de la diagonal.

- **Sensibilidad:** la fórmula de la sensibilidad viene dada por la Ecuación (2.7):

$$Sensibilidad = \frac{TP}{TP + FN} \quad (2.7)$$

Esta medida de rendimiento es usada para clasificadores binarios, al multiplicarse por 100 mide el porcentaje de elementos positivos que fueron clasificados de manera correcta de aquellos elementos que fueron detectados como positivos.

- **Especificidad:** la fórmula de la especificidad viene dada por la Ecuación (2.8):

$$Especificidad = \frac{TN}{FP + TN} \quad (2.8)$$

Esta medida de rendimiento es usada para clasificadores binarios, al multiplicarse por 100 mide el porcentaje de elementos negativos que fueron clasificados de manera correcta de aquellos elementos que fueron detectados como negativos.

- **Precisión:** la fórmula de la precisión viene dada por la Ecuación (2.9):

$$Precision = \frac{TP}{TP + FP} \quad (2.9)$$

Esta medida de rendimiento es usada para clasificadores binarios, al multiplicarse por 100 mide el porcentaje de elementos positivos que fueron clasificados de manera correcta del total de elementos que de verdad eran positivos.

- **Curva ROC:** es una técnica gráfica para la visualización, organización y selección de clasificadores basándose en su desempeño [23]. La curva es creada mediante la graficación de la sensibilidad contra la tasa de falsos positivos que viene dada por la Ecuación (2.10):

$$TasaFalsosPositivos = 1 - Especificidad \quad (2.10)$$

Se ordenan las predicciones de forma descendente usando cierto puntaje que suele corresponder al valor de certeza de una predicción realizada por el algoritmo. Esta técnica es útil para seleccionar los modelos óptimos y descartar aquellos que tengan menor desempeño [23].

En la Figura 2.7 se muestra un ejemplo donde se comparan dos modelos. La *línea ND* representan a la función de identidad que se interpreta como que los puntos que están sobre ella están teniendo un desempeño igual al de adivinar. Los puntos que pasen debajo de la línea ND tienen un desempeño peor que el de adivinar. El modelo será mejor conforme los puntos se encuentren más distanciados por encima de la función identidad y colocados hacia la parte superior izquierda (coordenadas (0,1)), es decir, aquella función que logre acumular mayor área bajo la curva. En el ejemplo se puede ver como el modelo *Test A* acumula mayor área bajo la curva

que el modelo *Test B* y de esta manera se puede interpretar que el modelo *Test A* posee mejor rendimiento que el modelo *Test B*.

Las curvas ROC son fáciles de interpretar, implementar y no son sensibles a los conjuntos de datos desbalanceados. Sin embargo, necesitan que los datos estén ordenados mediante el puntaje que se le asignado a cada predicción [23].

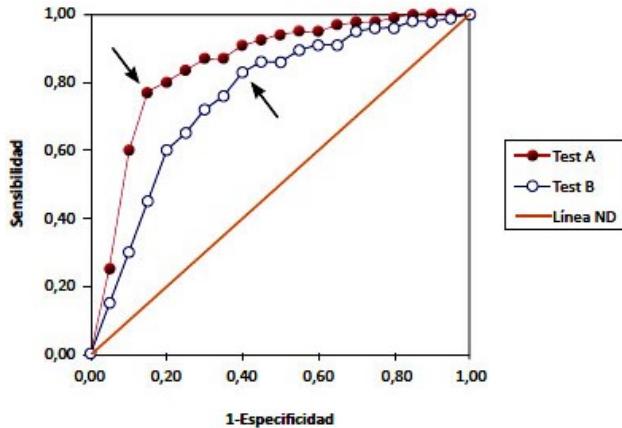


Figura 2.7: Ejemplo de curva ROC.
(Fuente: Scielo).

Las medidas de rendimiento mencionadas previamente son algunas de las muchas que existen, en las bibliografías [4, 13, 18, 23] se puede indagar más sobre las matrices de confusión y sobre otro tipo de medidas de rendimiento derivadas de ellas.

2.2.2. Herramientas de software

Muchas herramientas de software son utilizadas para la implementación de técnicas de ML. Algunas de las herramientas más populares son WEKA, Python, R, Matlab, C++, Java, etc. A continuación se presenta la herramienta utilizada para el presente trabajo, la cual es R.

R

Es un lenguaje de programación interpretado originalmente diseñado para usarse en programación estadística y para la generación de gráficos en documentos de investigación. Este lenguaje está escrito principalmente en el lenguaje C, Fortran y R. Es un software libre que se rige bajo las políticas de GNU.

Al ser un lenguaje interpretado hace uso de la cónsola de R, sin embargo, cuenta con un ambiente de desarrollo integrado llamado RStudio² de aceptación mundial en la comunidad

²<https://www.rstudio.com/>

científica. Con RStudio se pueden crear aplicaciones web, documentos en lenguaje markdown y proyectos compatibles con git. Aunque inicialmente R fue creado para un propósito estadístico, cuenta con una gran cantidad de paquetes para el área de ML que están disponibles en el CRAN³ (*The Comprehensive R Archive Network*).

2.3. Estado del arte

En esta sección se presenta un análisis del estado actual de la investigación de la seguridad en redes de computadoras utilizando aprendizaje automático. Específicamente se presentará la aplicabilidad de las técnicas de ML en el desarrollo de un NIDS. Se utilizarán como base las referencias bibliográficas [14, 24], que corresponden a *surveys* sobre el estado del arte entre NIDS y ML desde el año 2000 hasta el año 2015. En la el Capítulo 1 se introdujeron los conceptos de anomalía, ataque e intruso; todos estos conceptos se refieren a eventos no usuales en un sistema y serán utilizados de manera intercambiable.

La sección empieza introduciendo la necesidad del uso de técnicas de ML en la implementación de un NIDS; seguidamente se listan los tipos de enfoques presentes en el área, los objetivos, flujo general de trabajo que siguen la mayoría de las investigaciones y las tendencias en las fases de diseño e implementación de modelos de ML.

2.3.1. NIDS basados en técnicas de ML

Con el crecimiento de las redes de computadoras en la sociedad y su importancia dentro de la misma, ha surgido la necesidad de mejorar los diferentes mecanismos de seguridad mencionados en el Capítulo 1. Los métodos más recientes para el procesamiento de detección de ataques en redes de computadoras se han basado en utilizar técnicas de ML para automatizar el proceso de detección [9]. Esta iniciativa surgió debido a que los sistemas de seguridad manejados por analistas confían en reglas establecidas por expertos, que por lo general llevan a altas tasas de ataques no detectados [8].

La aplicación de técnicas de ML en el área de seguridad en redes de computadoras se centra en los NIDS y con esto se busca mejorar la eficacia al momento de detectar ataques y/o intrusos. Dependiendo del enfoque adoptado por el NIDS, estos se pueden clasificar en los siguientes modos: aprendizaje supervisado, aprendizaje no-supervisado, aprendizaje semi-supervisado, híbrido y conjunto. Estos serán tratados a continuación.

NIDS basados en aprendizaje supervisado

En la Sección 2.2 se define el concepto de aprendizaje supervisado en ML, donde se pudo observar que el aspecto más destacado es el de que los datos de entrenamiento deben ser

³<https://cran.r-project.org/>

previamente clasificados y etiquetados. Este concepto aplicado a los NIDS suele llevar el nombre de NIDS basado en la firma del ataque y será descrito a continuación.

- **NIDS basados en la firma del ataque:** este enfoque posee la habilidad de detectar ataques dependiendo de las características del mismo. Los sistemas almacenan las firmas de los ataques conocidos y las utilizan para compararlas contra las características de los nuevos registros que llegan, y de esta manera poder clasificarlos como un comportamiento normal o anómalo [25]. Este enfoque posee las siguientes ventajas y desventajas.

- **Ventajas**

- Alta tasa de aciertos para la detección de ataques conocidos.
- Sencillos de entrenar y de entender.

- **Desventajas**

- No generalizan de buena manera frente ataques no-conocidos.
- Los datos deben ser previamente etiquetados.
- La base de datos debe ser actualizada constantemente conforme surgen nuevos tipos de ataques.

La Figura 2.8 presenta un esquema general de la implementación de un NIDS utilizando técnicas de ML. Se observa que es necesario un proceso de retroalimentación para ir ajustando al modelo conforme va avanzando el tiempo y aparecen nuevos ataques.

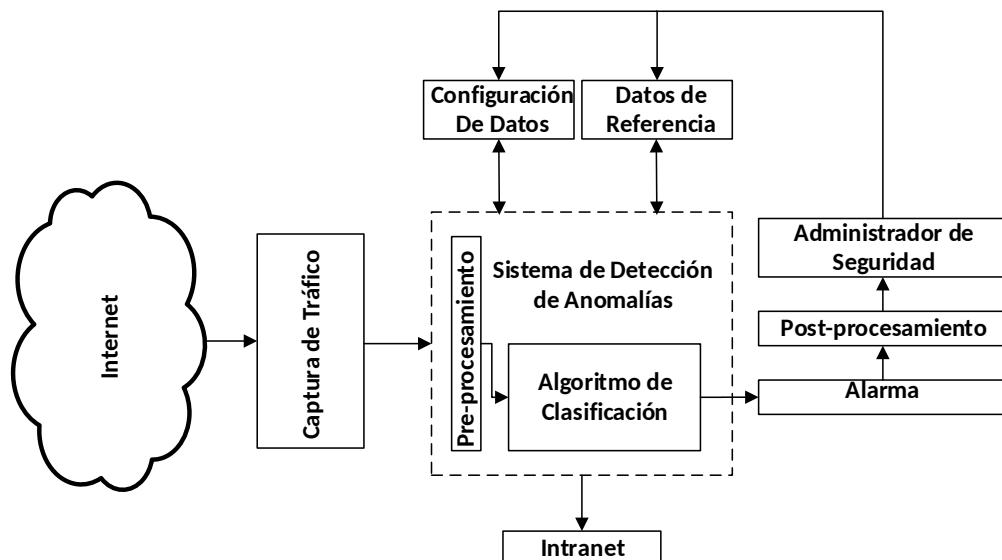


Figura 2.8: Modelo genérico de NIDS supervisado [4].

NIDS basados en aprendizaje no-supervisado

En la Sección 2.2 se define el concepto de aprendizaje no-supervisado en ML, donde se pudo observar que el aspecto más destacado es que no se necesita que los datos de entrenamiento sean previamente clasificados o etiquetados para que el algoritmo pueda ser entrenado. El concepto de aprendizaje no-supervisado aplicado a los NIDS suele llevar el nombre de NDIS basado en anomalías y será descrito a continuación.

- **NIDS basados en anomalías:** este enfoque define el comportamiento normal dentro de una red y trata de identificar los ataques e intrusos comparando la desviación de los registros con respecto al comportamiento normal [25]. Si el registro se desvía más de cierta medida establecida del comportamiento normal, entonces este registro es clasificado como una anomalía. Este enfoque presenta las siguientes ventajas y desventajas.

- **Ventajas**

- Capaces de detectar ataques conocidos y no conocidos.
- No es necesario que los datos sean previamente clasificados.

- **Desventajas**

- Suelen presentar altas tasas de falsas alarmas.
- Para describir el comportamiento normal, es necesario que los registros de entrenamiento se encuentren libres de anomalías.
- Complejidad al establecer las fronteras entre el tráfico normal y el anómalo.

La Figura 2.9 ilustra un modelo general de la implementación de un NIDS no-supervisado. Se puede observar como de igual manera que en el aprendizaje supervisado es necesaria la retroalimentación para el ajuste del modelo en el tiempo, y como se observa existe un proceso de asignación de puntuación que suele ser complicado de establecer.

La Tabla 2.2 muestra un resumen comparativo entre los NIDS basados en la firma del ataque contra los NIDS basados en anomalías. En la misma se puede observar el contraste entre un enfoque y otro.

NIDS basados en aprendizaje semi-supervisado

En la Sección 2.2 se define el concepto de aprendizaje semi-supervisado en ML, donde se pudo observar que el aspecto más destacado corresponde a que sólo se tiene una porción en los datos clasificados y la otra porción sin clasificar. El concepto de aprendizaje semi-supervisado aplicado en los NIDS suele utilizarse para entrenar modelos de la misma manera que en aprendizaje no-supervisado, con la diferencia de que los datos etiquetados son utilizados para medir el rendimiento del NIDS. Así mismo este enfoque es más flexible que el enfoque supervisado [4].

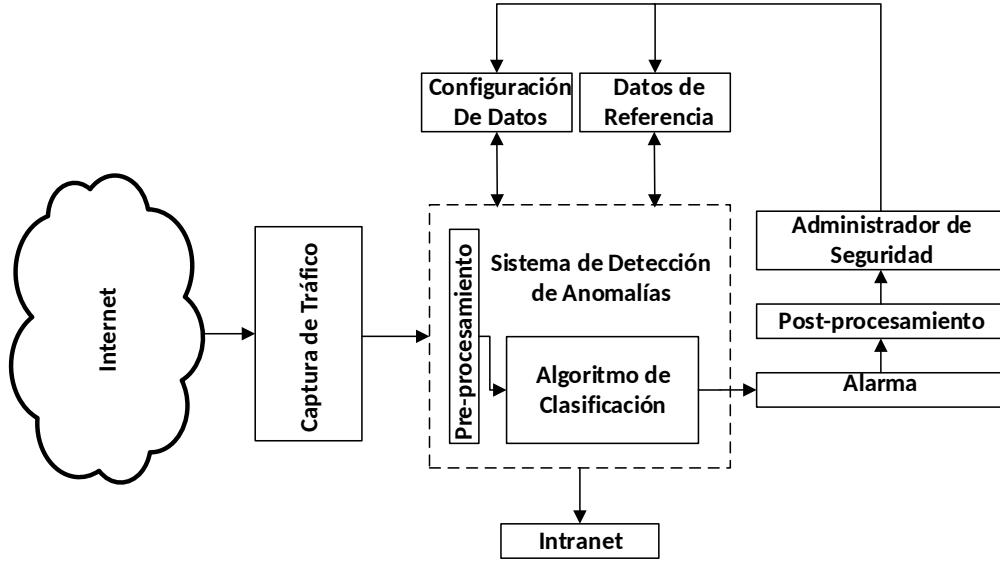


Figura 2.9: Modelo genérico de NIDS no-supervisado [4].

Basado en Firma	Basado en Anomalía
Funciona con ataques conocidos	Funciona con ataques conocidos y desconocidos
Alta tasa de aciertos	Alta tasa de falsas alarmas
Amerita de la previa clasificación de los datos de entrenamiento	Amerita que las anomalías sean removidas del conjunto de datos de entrenamiento
La base de datos debe ser actualizada frecuentemente	Puede ser complicado determinar las fronteras entre el tráfico normal y el anómalo

Tabla 2.2: NIDS basado en firma VS NIDS basado en anomalías.

NIDS híbridos

Este enfoque combina los modelos de aprendizaje supervisado y no-supervisado. Por un lado el método supervisado tiene la ventaja de ser capaz de detectar con alta eficacia los ataques conocidos. Por otra parte, el método no-supervisado tiene la habilidad de detectar nuevos ataques. Entonces, el enfoque híbrido de detección de intrusos es capaz de detectar tanto ataques conocidos como no conocidos [4]. En la Figura 2.10 se puede observar un modelo general de este enfoque, donde se puede observar que los registros son filtrados inicialmente por un clasificador supervisado en un primer nivel. Si el registro es clasificado como anómalo entonces este registro es reportado como tal. En caso contrario, si un registro es clasificado como normal, será pasado a un segundo nivel donde el mismo será revaluado por un segundo clasificador no-supervisado.

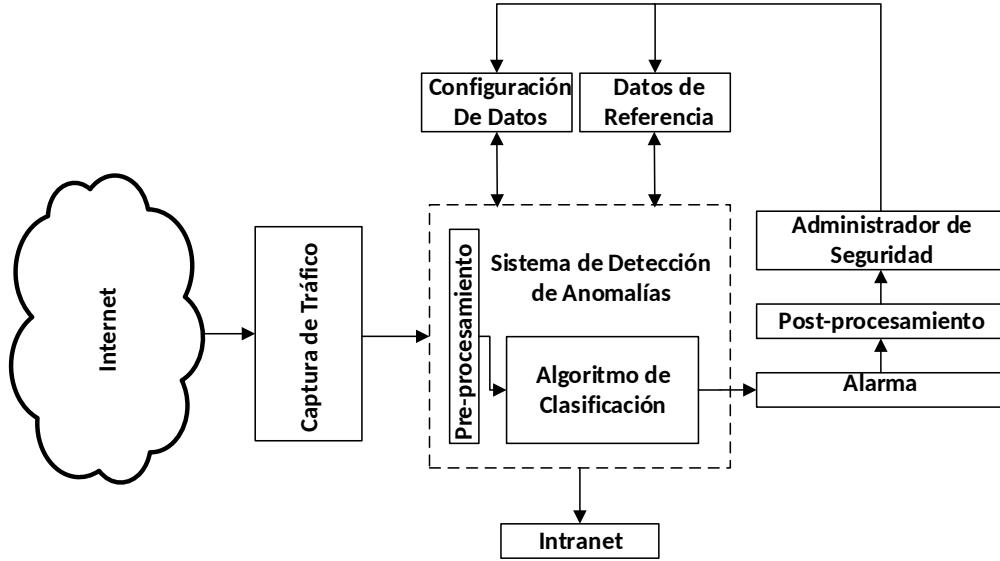


Figura 2.10: Modelo genérico de NIDS híbrido [4].

NIDS conjuntos

Son utilizados para mejorar el rendimiento de los clasificadores simples, que utilizan exclusivamente un algoritmo para la clasificación [26]. Este enfoque combina varios algoritmos de clasificación para luego tomar una decisión en conjunto, mediante un esquema de votación [27].

2.3.2. Objetivos de la aplicación de técnicas de ML en NIDS

Los objetivos que se buscan alcanzar al momento de utilizar técnicas de ML en la implementación de NIDS son listados a continuación.

1. Automatizar el proceso de detección de intrusos en redes computadoras.
2. Aumentar la tasa de aciertos de los NIDS al momento de detectar ataques, intrusos y anomalías.
3. Minimizar la tasa de falsos positivos y falsos negativos de los NIDS.

Cumpliendo los objetivos mencionados previamente se lograría una forma eficaz y versátil para detectar ataques, intrusos y anomalías en las redes de computadoras.

2.3.3. Flujo general de trabajo en la implementación de un NIDS utilizando técnicas de ML

El flujo general de trabajo en las investigaciones de los NIDS basados en técnicas de ML es apreciado en la Figura 2.11. En dicha figura se pueden identificar tres pasos principales y relevantes que corresponden a:

1. Extracción de características.
2. Pre-procesamiento.
3. Entrenamiento del modelo.

El paso más importante por lo general es el paso (1). Sin embargo, la mayoría de los investigadores hacen uso de conjuntos de datos minados e ignoran este paso debido a la complejidad del mismo, concentrándose en la propuesta e implementación de modelos de ML [24].

Las publicaciones más recientes hacen investigación en los pasos (2) o (3) del gráfico citado en el párrafo anterior. Especialmente en los pasos (2a), (2b), (2c), y (2d). En la Tabla 2.3 se presentan las tendencias de los aportes realizados en el período 2010 - 2015.

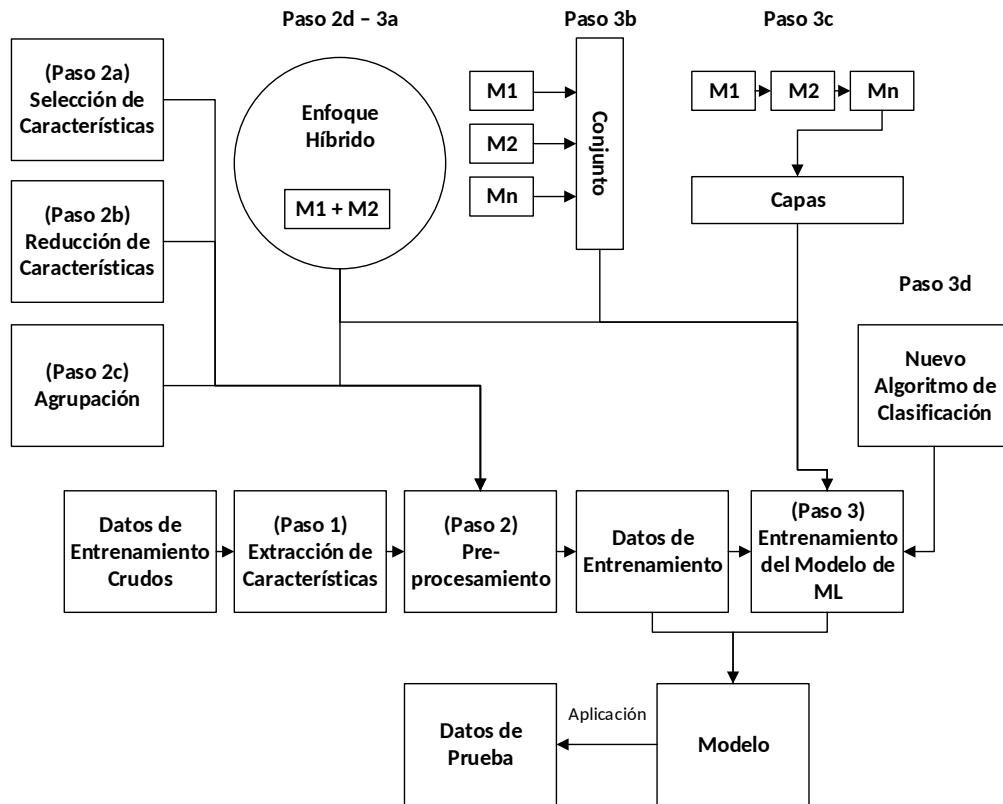


Figura 2.11: Flujo general de un NIDS basado en técnicas de ML [24].

Conjuntos de datos

Como se pudo observar en la Sección 2.2.1, la recolección de los datos es la fase más sensible de todo el proceso en las técnicas de ML. Dependiendo de la calidad de los datos recolectados se podrá tener un buen o mal modelo de ML. Dicho esto, en el área de los NIDS utilizando técnicas de ML es importante tener una base de conocimiento de registros

Contribución	Cantidad Artículos
Híbrido	50
Nuevo Clasificador	45
Reducción de Características	38
Selección de Características	34
Nuevo Algoritmo de Detección de Anomalías	33
Nuevo Algoritmo de Optimización	25
Capas	23
Nuevo Algoritmo de Agrupamiento	19
Conjunto	14
Basado en Agentes	12
Flujo de Datos	7

Tabla 2.3: Aportes realizados en ML y NIDS en el período (2010 - 2015) [24].

normales y de ataques que sea confiable. A continuación se describirán los conjuntos de datos públicos más populares en el área de los NIDS de acuerdo a las investigaciones realizadas.

- **DARPA 1998:** el conjunto de datos DARPA 1998 surgió de una competencia realizada en conjunto por el Instituto Técnico de Massachusetts (MIT - *Massachusetts Institute of Technology*) y la Agencia del Departamento de Defensa de los Estados Unidos (DARPA - *Defense Advanced Research Projects Agency*). Esta competencia que tuvo lugar en el año 1998 tuvo como finalidad motivar a investigadores e indagar en el área del estado del arte entre NIDS y ML con el propósito de generar conocimiento en la búsqueda de mejorar la seguridad en el ámbito de la detección de intrusos en redes de computadoras. Este conjunto de datos está disponible en el sitio web del MIT⁴ junto con otras versiones que corresponden a los años 1999 y 2000. Sin embargo, el conjunto de datos del año 1998 ha sido el más usado desde su creación hasta la actualidad [14, 24].

El conjunto de datos consta de alrededor cuatro Gigabytes de datos comprimidos de capturas de tráfico correspondiente a siete semanas. Estos datos pueden ser procesados en alrededor cinco millones de registros de conexión de alrededor 100 bytes cada uno. Los datos contienen la información de cada paquete transmitido entre computadores dentro y fuera de una base militar simulada [28].

Los datos contienen cuatro categorías principales de ataques: DoS, R2L, U2R y *Probing*. Los mismos fueron descritos en el Capítulo 1.

- **KDD99:** el conjunto de datos KDD99 nace del conjunto de datos de DARPA1998. Las iniciales KDD hacen referencia a *Knowledge Discovery in Databases*. Esto significa que

⁴<https://www.ll.mit.edu/ideval/data/>

el conjunto de datos ha sido sometido previamente a un proceso de minería de datos donde se generó nuevo conocimiento.

Este conjunto de datos fue creado por Lee y Stolfo en el año 1999 producto de su participación en la competencia DARPA 1998. El procesamiento de los datos utilizados para la extracción de características y el proceso de minado de los datos desde las capturas de tráficos proporcionadas en el conjunto de datos es descrita en una publicación presentada en el año 1999 [28]. En esa publicación Lee y Stolfo también establecen algunas métricas para el proceso de implementación de un NIDS, donde mencionan que uno de los pasos fundamentales es extraer la mayor cantidad de características de los datos crudos (capturas de tráfico de red) en un formato comprensible por los algoritmos de ML.

El conjunto de datos KDD99 ha sido el más utilizado desde el año 2000 como se menciona en las publicaciones [14, 24]. Lo anterior es debido a que los investigadores en el estado del arte entre NIDS y ML se ahorran el proceso de captura de los datos y extracción de características y trabajan sobre una base de datos ya minada donde pueden concentrarse en la propuesta de métodos y técnicas de ML correspondientes a las fases de pre-procesamiento, selección de características, selección del modelo, selección de parámetros, validación del modelo en fase de entrenamiento y evaluación del modelo en fase de prueba. Las distintas fases en el flujo del proceso implementación de técnicas de ML fueron definidas en la Sección 2.2.1.

Este conjunto de datos está disponible en el sitio web de la base de datos KDD⁵.

• Características

- Consta de 42 columnas, donde las primeras 41 corresponden a información de los distintos registros pertenecientes al conjunto de datos y la columna número 42 corresponde a la etiqueta del registro que puede adquirir los valores de normal, DoS, R2L, U2R y *Probing*.
- Las diferentes etiquetas por tipo de ataque son las siguientes, y están definidas individualmente en [4]:
 - ◊ **DoS:** *Smurf, Neptune, Back, Teardrop, Ping-of-death, Land.*
 - ◊ **R2L:** *FTP-write, Guess-password, Imap, Multihop, Phf, Spy, Warezclient, Warzmaster.*
 - ◊ **U2R:** *Buffer-overflow, Loadmodule, Perl, Rootkit.*
 - ◊ **Probing:** *Ipsweep, Nmap, Portsweep, SATAN.*
- Las 41 características se dividen en cuatro grandes grupos como se explica en [28, 29] y como se puede apreciar en la Figura 2.12.

⁵<http://kdd.ics.uci.edu/databases/kddcup99/>

- El conjunto de entrenamiento presenta 22 tipos de ataques divididos entre los cuatro tipos de ataques mencionados previamente. Por otro lado el conjunto de prueba posee 14 nuevos ataques que no están presentes en el conjunto de entrenamiento, con la finalidad de probar la capacidad de generalización que tienen los algoritmos de ML al momento de clasificar nuevos ataques [30].

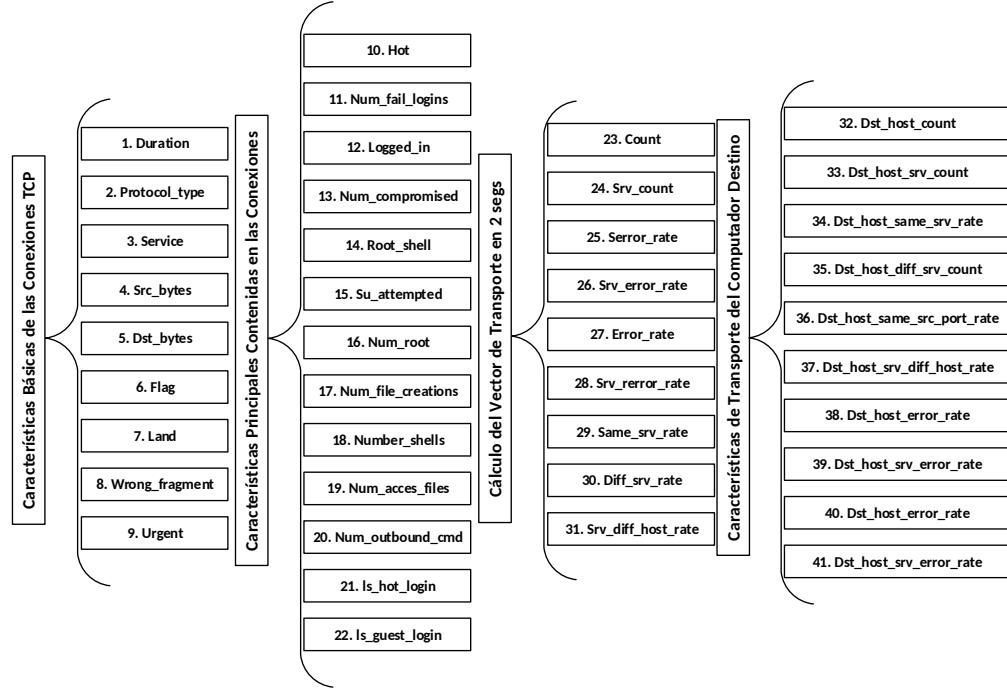


Figura 2.12: Características de KDD99 [21].

- **Problemas inherentes del conjunto de datos KDD99**
 - El principal problema es que el conjunto de datos posee una redundancia de alrededor del 78 % en el conjunto de datos de entrenamiento y de alrededor del 75 % en el conjunto de prueba [30]. La redundancia de los datos añade ruido a los algoritmos de ML y puede afectar negativamente el entrenamiento de los mismos. En la Tabla 2.4 y la Tabla 2.5 se ilustra este hecho.
 - La cantidad de datos es muy grande y conlleva a que los investigadores deban realizar sub-conjuntos de la información para poder manipularla.
 - Al estar derivado del conjunto de datos de DARPA 1998, KDD99 hereda los defectos de su predecesor [31].
 - Al ser los datos sintetizados, la carga de trabajo no es parecida al tráfico real de las redes de computadoras [30].
- **NSL-KDD:** este conjunto de datos es una mejora sobre el conjunto de datos KDD99. Esta mejora es explicada en detalle por Tavallae y sus colaboradores en [30]. El conjunto de datos es público y gratuito. Se puede obtener en el sitio web de la Universidad

	Registros Originales	Registros Diferentes	Tasa de Reducción
Ataques	3925650	262178	93.32 %
Normal	972781	812814	16.44 %
Total	4898431	1074992	78.05 %

Tabla 2.4: Registros redundantes en el conjunto de entrenamiento de KDD99 [30].

	Registros Originales	Registros Diferentes	Tasa de Reducción
Ataques	250436	29378	88.26 %
Normal	60591	47911	20.92 %
Total	311027	77289	75.15 %

Tabla 2.5: Registros redundantes en el conjunto de prueba de KDD99 [30].

de Nueva Brunswick⁶.

En su documento, Tavallae y colaboradores explican los problemas presentes en el conjunto de datos KDD99 que fueron tratados previamente, y utilizando este conjunto de datos como base, propusieron e implementaron una solución que será descrita a continuación.

- **Metodología utilizada**

A continuación se explicarán las actividades realizadas en el proceso de mejora del conjunto de datos KDD99.

- Eliminación de los registros repetidos, problema principal del conjunto KDD99 tratado previamente.
- Se entrenaron siete diferentes tipos de algoritmos de clasificación (J48, Red Bayesiana, Árbol de Decisión Bayesiano, Bosque Aleatorio, Árbol Aleatorio, Perceptron Multi-Capa, SVM). Por cada algoritmo se crearon tres modelos para un total de 21 modelos. A continuación se enumeran las actividades realizadas.
 1. Con el conjunto de entrenamiento sin registros duplicados se entrenaron los 21 modelos de clasificación.
 2. Se probó la efectividad de los 21 clasificadores sobre los conjuntos de entrenamiento y de prueba y se crearon los siguientes grupos **0-5, 6-10, 11-15, 16-20, 21**. Dichos rangos representan el número de clasificadores que acertaron al predecir un registro. En las Figuras 2.13 y 2.14, y en las Tablas 2.6 y 2.7 se refleja este hecho mediante gráficos y estadísticas, respectivamente.

⁶<http://www.unb.ca/research/iscx/dataset/iscx-NSL-KDD-dataset.html>

3. Una vez realizada la actividad descrita en el punto anterior, se agregó una columna al conjunto de datos de entrenamiento y de prueba que corresponde a la cantidad de clasificadores que acertaron al momento de clasificar cada registro. Este conjunto posee 43 columnas, las primeras 42 iguales al conjunto de KDD99 reflejadas en la Figura 2.12, y una adicional que corresponde a lo descrito previamente.

- **Ventajas sobre el conjunto de datos KDD99**

A continuación se presentan las ventajas sobre el conjunto de datos KDD99.

- No posee registros repetidos que añaden ruido a los algoritmos en la fase de entrenamiento.
- Tiene menos registros y se puede utilizar en su totalidad sin tener que realizar sub-conjuntos de los datos.
- Añade una característica que permite evaluar la efectividad de los modelos de clasificación con respecto a la dificultad de los diferentes registros presentes en el conjunto de datos.

- **Problemas inherentes del conjunto de datos NSL-KDD**

Al ser derivado del conjunto KDD99 hereda los mismos problemas intrínsecos. Sin embargo, este conjunto de datos es ampliamente utilizado por la comunidad científica y se ha convertido en el conjunto de datos de facto para el entrenamiento y prueba de modelos de ML en el área de NIDS [30].

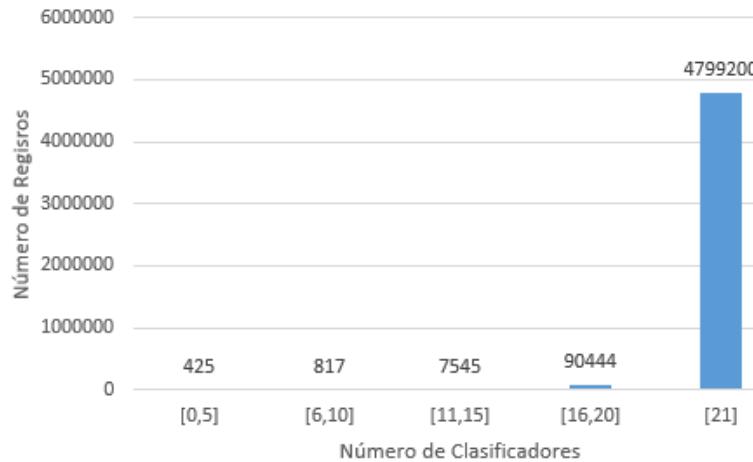


Figura 2.13: Distribución de aciertos en el conjunto de entrenamiento de KDD99 [30].

Pre-procesamiento de los datos

En la Sección 2.2.1 se mencionó el hecho de que los datos deben ser transformados a un formato que pueda ser entendible por los diferentes algoritmos de ML. Los métodos más populares son los de transformación, y estandarización. Bhavsar y Waghmare especifican en

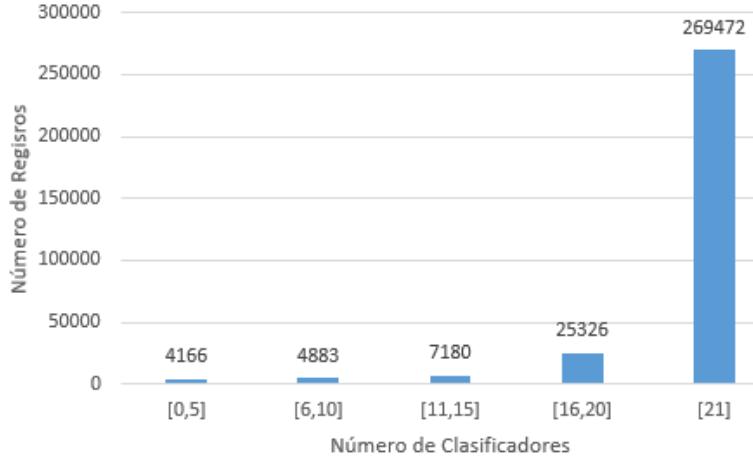


Figura 2.14: Distribución de aciertos en el conjunto de prueba de KDD99 [30].

	Registros Distintos	Porcentaje	Registros Seleccionados
0-5	407	0.04	407
6-10	768	0.07	767
11-15	6525	0.61	6485
16-20	58995	5.49	55757
21	1008297	93.80	62557
Total	1074992	100.00	125973

Tabla 2.6: Estadísticas de la selección de registros aleatorios del conjunto de entrenamiento de KDD99 [30]

[19] como las técnicas de pre-procesamiento de los datos mediante los métodos de transformación mencionados previamente pueden incrementar la tasa de aciertos de los modelos de ML y pueden reducir los tiempos de entrenamiento y de prueba utilizando un clasificador SVM.

Selección de características

En la Sección 2.2.1 se expusieron las técnicas PCA y GFR. Atilla y Hamit exponen que los algoritmos de selección de características más utilizados son PCA y algoritmos genéticos. Este hecho se puede apreciar en la Tabla 2.8. Algunos de los trabajos donde se utilizan estas técnicas fueron presentados por Thaseen quien utilizó PCA en conjunto con SVM [20]. El método GFR es implementado por Li y colaboradores, quienes adicionalmente utilizaron K-Medias y SVM en [21]. Por otra parte, Shon expone en [32] un trabajo en el que hace uso de algoritmos genéticos para la selección de características en el área de detección de intrusos en redes de computadoras.

	Registros Distintos	Porcentaje	Registros Seleccionados
0-5	589	0.76	585
6-10	847	1.10	838
11-15	3540	4.58	3378
16-20	7845	10.15	7049
21	64468	83.41	10694
Total	77289	100.00	22544

Tabla 2.7: Estadísticas de la selección de registros aleatorios del conjunto de prueba de KDD99 [30]

Selección del modelo

Una vez que los datos están en un formato que es entendible por los algoritmos de ML, es hora de seleccionar el modelo a utilizar y los algoritmos, tal como se especificó en la Sección 2.2.1. En este paso se deberá elegir un enfoque de los discutidos en la Sección 2.3.1 y las tendencias de investigación reflejadas en la Tabla 2.3.

Atilla presenta una tabla con los algoritmos más utilizados en publicaciones en el período 2010 - 2015 [24]. La Tabla 2.8 ilustra dicha investigación, donde se puede observar que los algoritmos de clasificación supervisada más utilizados son SVM, árbol de decisión y NN, ordenados por orden de importancia respectivamente. Por otro lado, tenemos que los algoritmos de clasificación no-supervisados más utilizados corresponden a análisis de concentración de partículas y K-Medias, de igual manera ordenados por orden de importancia respectivamente.

Validación del modelo

En la Sección 2.2.1 se presentó la necesidad de probar los modelos en la fase de entrenamiento con el objetivo de poder seleccionar el modelo que pueda hacer un mejor trabajo al momento de procesar datos no vistos previamente. En la implementación de un NIDS utilizando técnicas de ML, la técnica de validación de modelos más popular es la de validación cruzada, que fue mencionada en la misma Sección. Sin embargo, la mayoría de los trabajos no utilizan la validación de los modelos en la fase de entrenamiento para realizar las investigaciones, en el período entre 2010 y 2015, sólo el 21 % utilizaron técnicas de validación cruzada, el restante 79 % no usó técnicas de validación de modelos [24].

El uso de la validación cruzada es importante para la correcta selección de los parámetros. Bhavsar y Waghmare presentan un trabajo donde utilizan diferentes kernels en SVM y utilizan validación cruzada para seleccionar los mejores parámetros [19].

Nombre	Cantidad de Artículos
Máquina de Vectores de Soporte	24
Árbol de Decisión	19
Algoritmo Genético	16
Análisis de Componentes Principales	13
Análisis de Concentración de Partículas	9
K-Veinos	9
K-Medias	9
Probabilidad Ingenua de Bayes	9
Redes Neuronales (Perceptron Multi-Capa)	8
Programación Genética	6
Conjuntos Duros	6
Redes Bayesianas	5
Bosque Aleatorio	5
Sistemas Artificiales Inmunes	5
Lógica Difusa	4
Redes Neuronales (Mapa de Auto-Organización)	4

Tabla 2.8: Algoritmos populares de ML en NIDS [24].

Evaluación del modelo

Una vez que el modelo esté entrenado se procede a probarlo utilizando las métricas expuestas en la Sección 2.2.1. Los conjuntos de datos populares utilizados en el área de NIDS utilizando técnicas de ML poseen datos etiquetados que permiten evaluar los modelos haciendo uso de matrices de confusión y aplicando alguna de las medidas de rendimiento explicadas en la Sección mencionada. La Tabla 2.9 recopila las medidas de rendimiento más utilizadas en el período 2010 - 2015 [24].

2.3.4. Herramientas utilizadas

Existen herramientas de software que cuentan con paquetes, bibliotecas y otras funcionalidades que facilitan la implementación de las diferentes técnicas de ML. En la Tabla 2.10 se ilustran las herramientas más utilizadas en el período 2010 - 2015 [24].

2.3.5. Consideraciones en la utilización de técnicas de ML en la implementación de un NIDS

En esta Sección se presentarán los retos y problemas actuales en el diseño e implementación de un NIDS utilizando técnicas de ML.

Métrica de Rendimiento	Cantidad de Artículos
Tasa de Acierto	134
Falsos Positivos	70
Tiempo de Entrenamiento	44
Tiempo de Prueba	28
Curva ROC	24
Falsos Negativos	22
Matriz de Confusión (5 clases)	20
Verdaderos Positivos	20
Tasa de Error	13
Precisión	13
F-Score	13
Recall	12
Verdaderos Negativos	11
Número de Características Seleccionadas	10
Coeficiente de Correlación	9
Costo por Registros	9
Área Bajo la Curva ROC	8
Sensitividad	7
Especificidad	7
Error Cuadrado	6
Ninguno	5
Uso de Memoria	5

Tabla 2.9: Medidas de rendimiento más utilizadas [24].

Retos

- Mejorar el desempeño general de la tarea concerniente a la detección de anomalías en redes de computadoras mediante la utilización de técnicas de ML.
- Aumentar la cantidad de anomalías correctamente detectadas.
- Decrementar la cantidad de falsos positivos y falsos negativos.
- Crear un NIDS que sea versátil ante la evolución de los ataques y/o intrusos.
- Crear un NIDS que pueda detectar de forma rápida las anomalías presentes en la red.
- Optimizar el tiempo de los expertos en seguridad.

Problemas

- Recolección de datos confiables que puedan ser utilizados para entrenar a los modelos de ML.

Herramienta de Software / Paquete	Cantidad de Artículos
Sin Información (La herramienta usada no está clara)	78
Weka	34
Matlab	26
Libsvm	9
Java	7
C++	5
C#	3
Pascal, Hadoop, Python, MOA	2 c/u

Tabla 2.10: Herramientas de software más utilizadas [24].

- Complejidad para el pre-procesamiento de los datos crudos referentes a las capturas de tráfico en una red de computadoras con respecto a la generación de características y limpieza de los mismos.
- Dificultad para la implementación de un NIDS en tiempo real debido a la gran cantidad de datos que se generan en una red.
- La constante evolución de los ataques conlleva a la actualización constante de los modelos de ML para que estos puedan adaptarse a los cambios.

Capítulo 3

Marco aplicativo

En el presente capítulo se describen a detalle la metodología, consideraciones de diseño, consideraciones de implementación y arquitectura planteada para el desarrollo de actividades descritas en el Capítulo 4.

3.1. Metodología

El presente trabajo se guió por el flujo principal de trabajo de ML presentado en la Figura 2.5, recordando que el flujo presentado en dicha imagen refleja el flujo ideal del proceso de ML; sin embargo, es posible en cualquier punto del mismo, regresar a una fase previa para iterar sobre dicho flujo. De esta manera, la estrategia utilizada fue el uso de un enfoque iterativo y empírico que permitiera seleccionar el/los modelo(s) que mejor se ajustara(n) al escenario.

3.2. Consideraciones de diseño

Se crearon modelos híbridos de ML para la detección de ataques sobre el conjunto de datos NSL-KDD. Para la clasificación de los registros se utilizaron los algoritmos NN y SVM como representantes del enfoque supervisado y el algoritmo K-Medias como representante del enfoque no-supervisado. Por otra parte, para la reducción de características se usaron los algoritmos PCA y GFR. Dicho esto, se diseñaron, implementaron y analizaron seis modelos que serán descritos a continuación.

1. NN - K-Medias.
2. SVM (Radial) - K-Medias.
3. PCA - NN - K-Medias.
4. PCA - SVM (Radial) - K-Medias.
5. GFR - NN - K-Medias.

6. GFR - SVM (Radial) - K-Medias.

Los modelos (1) y (2) fueron creados utilizando el conjunto total de características o de variables predictoras del conjunto de datos NSL-KDD. Adicionalmente, fueron utilizados los parámetros por defecto de los diferentes algoritmos involucrados en ambos modelos. Con la creación de estos se buscó evaluar el desempeño de los diferentes esquemas para conocer rápidamente si estos se adaptaban de buena manera al escenario planteado, correspondiente al conjunto de datos NSL-KDD descrito en la Sección 2.3.3. Adicionalmente, los resultados obtenidos servirían como base comparativa con los resultados obtenidos en los modelos posteriores, donde se utilizaron técnicas de selección de características y de selección de parámetros buscando mejorar el desempeño de los modelos en la tarea de detección de intrusos.

Los modelos (3), (4), (5) y (6) corresponden a modelos donde se utilizaron algoritmos de selección de características y la posterior selección de parámetros. La selección de características y de parámetros fue llevada a cabo haciendo uso de la técnica de validación cruzada de 10-conjuntos descrita en la Sección 2.2.1. Con la implementación de estos modelos se buscó obtener mejoras con respecto a los modelos (1) y (2) desde un punto de vista de eficacia y cómputo.

Los modelos en el primer nivel tendrán que clasificar cinco clases de ataques: DoS, Normal, *Probing*, R2L y U2R. Por otra parte, los modelos en el segundo nivel correspondiente a K-Medias tendrán que clasificar dos clases: Ataque y Normal. La diferencia entre las salidas de los niveles se debe a la imposibilidad del segundo nivel correspondiente a K-Medias de clasificar de manera correcta teniendo cinco clases objetivo. De igual manera, la explicación del criterio adoptado será descrito a detalle en el Capítulo 4.

Con los diferentes esquemas planteados para los distintos modelos presentados con anterioridad, se buscó utilizar el algoritmo K-Medias como complemento de las técnicas de ML supervisadas NN y SVM con kernel radial con la finalidad de aumentar la tasa de aciertos en la tarea de detección de intrusos en redes de computadoras.

El algoritmo NN fue seleccionado debido a que es popularmente usado para la tarea de detección de intrusos [4] y a su flexibilidad para el ajuste de los modelos, característica discutida en la Sección 2.2. El algoritmo SVM fue elegido debido a que es la técnica de clasificación para la detección de intrusos en redes de computadoras más utilizada, tal como se ilustró en la Tabla 2.8. El kernel radial fue seleccionado utilizando como referencia el trabajo elaborado por Bhavsar y Waghmare [19], donde se observa que el mismo es más eficaz y rápido para el entrenamiento y predicción de registros que los kernel polinomial y sigmoide; característica de suma relevancia para los investigadores en el área. El algoritmo K-Medias fue elegido por su popularidad de uso en el área, descrita en la Sección 2.3.3 y por su capacidad de ajustarse a grandes volúmenes de datos, característica mencionada en la Sección 2.2. El algoritmo PCA fue elegido por su popularidad en el campo de ML, como se ilustra

en [4, 13], donde se explican generalidades del uso de PCA en la implementación de técnicas de ML. El algoritmo GFR fue elegido por la referencia tomada del trabajo publicado por Li, Xia y colaboradores [21], donde combinaron GFR con K-Medias y SVM. Por último, el conjunto de datos NSL-KDD fue elegido debido a que este conjunto de datos ya fue sometido a un pre-procesamiento previo referente a eliminación de registros repetidos y disminución de registros inconsistentes que aportan ruido a los diferentes algoritmos de ML, ya que como se trató en la Sección 2.3.3, este conjunto de datos corresponde a una versión mejorada del conjunto de datos KDD99, que es el más utilizado por la comunidad científica para el estudio de la aplicación de técnicas de ML en la detección de intrusos en redes de computadoras.

Previamente, en la Sección 3.1, se mencionó que el presente trabajo se rigió por el flujo general de trabajo presentado en la Figura 2.5. Específicamente dentro del módulo de Implementación del Modelo, se propusieron dos flujos de trabajo para el entrenamiento y prueba de los diferentes modelos. Los mismos serán presentados a continuación.

3.2.1. Flujo de entrenamiento

La Figura 3.1 ilustra el proceso de entrenamiento de los modelos que inicia en la Parte 1 donde se utiliza el conjunto de entrenamiento NSL-KDD. Posteriormente en la Parte 2 se hace el pre-procesamiento de los datos que corresponde a la reducción de dimensionalidad del conjunto de datos para reducir las columnas del mismo haciendo uso de las técnicas PCA o GFR. La Parte 3 corresponde a la fase de entrenamiento que inicia seleccionando los parámetros de los diferentes modelos haciendo uso de la técnica de validación cruzada de 10 conjuntos para la selección de la mejor combinación de los mismos. Luego, los mejores parámetros seleccionados son utilizados para el entrenamiento de los modelos, de nuevo, se utilizó la técnica de validación cruzada de 10 conjuntos para la validación de los diferentes modelos creados. Se culmina en la Parte 4 donde se realiza el análisis de los modelos entrenados en la fase anterior y finalmente se selecciona el modelo que haya presentado mejor desempeño, modelo que luego será introducido al flujo de prueba.

3.2.2. Flujo de prueba

La Figura 3.2 ilustra el proceso de prueba de los modelos que inicia en la Parte 1 utilizando conjunto de datos de prueba NSL-KDD. Luego, en la Parte 2 se aplica PCA o GFR y se utilizan aquellas características identificadas como más relevantes en la fase de entrenamiento. Posteriormente en la Parte 3 se aplica el enfoque híbrido, dando paso a una clasificación en dos niveles que inicia con SVM o NN como técnicas de aprendizaje supervisado. Si un registro que es evaluado por el primer nivel es detectado como anómalo, inmediatamente el registro es seleccionado para la posterior validación. Por otra parte, si un registro es clasificado como normal pasa al siguiente nivel, donde K-Medias (representante del enfoque no-supervisado) es el encargado de detectar algún ataque que no haya sido detectado en el nivel anterior. Se finaliza con el Paso 4, donde se utilizan las medidas de rendimiento presentadas en la Sección 2.2.1 para analizar el desempeño del modelo.

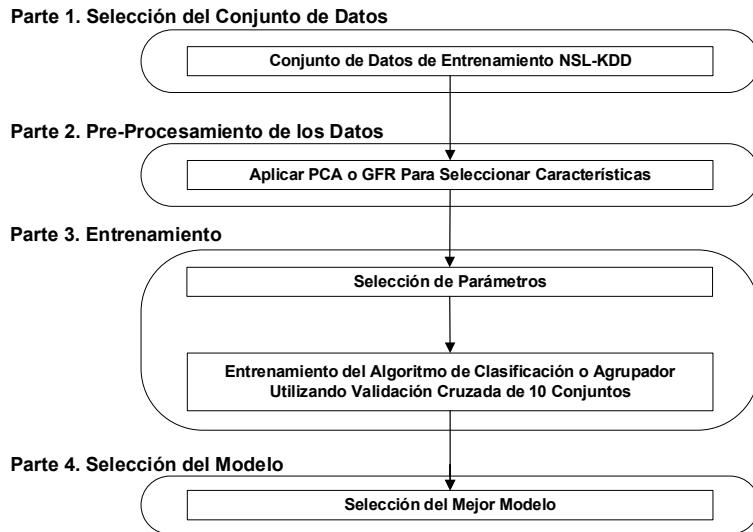


Figura 3.1: Flujo de entrenamiento utilizado.

3.3. Consideraciones de implementación

En esta sección se presentan todas las generalidades referentes a la implementación de la solución.

- Todas las variables predictoras de los diferentes modelos fueron sometidas a los procesos de transformación y estandarización tratados en la Sección 2.2.1. Específicamente, el proceso de transformación consistió en la transformación de variables de tipo categórico a tipo numérico. Por otra parte, en el proceso de escalamiento, el conjunto de variables predictoras fue normalizado para que las variables predictoras tuvieran media cero (0) y desviación estándar uno (1). Este paso es fundamental para el correcto entrenamiento de los modelos basados en distancias. Adicionalmente, acelera los tiempos de procesamiento para el entrenamiento y prueba de los diferentes modelos [13]. La Ecuación 3.1 presenta la fórmula de la normalización, donde X' representa el nuevo valor de un registro, X el valor actual, μ y σ representan la media y desviación estándar de los registros del conjunto de datos respectivamente.

$$X' = \frac{X + \mu}{\sigma} \quad (3.1)$$

- Uso de la técnica de validación cruzada de conjuntos para el análisis sobre el conjunto de entrenamiento, así como para la selección de parámetros y de características.
- Uso de la matriz de confusión como base para la evaluación de los distintos modelos creados, específicamente, las descritas en la Sección 2.2.1:
 - Matriz de confusión de cinco clases (Primer nivel).

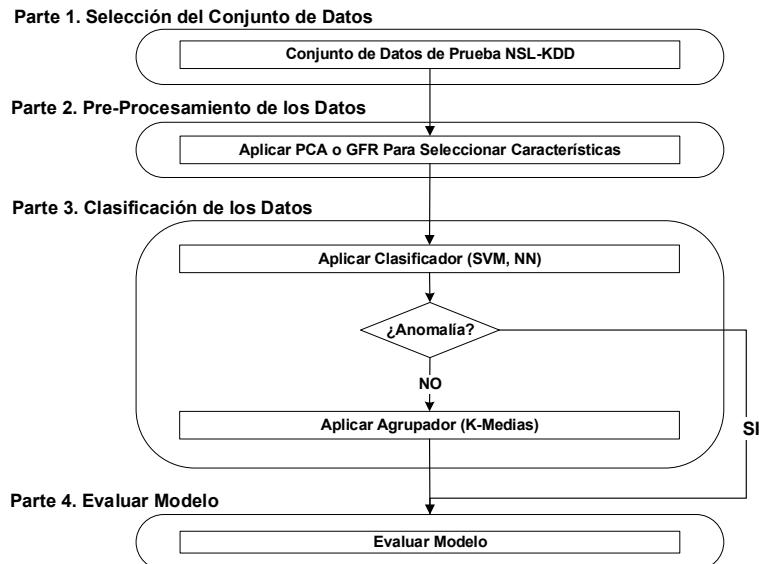


Figura 3.2: Flujo de prueba utilizado.

- Matriz de confusión de dos clases (Segundo Nivel).
- Especificidad.
- Sensibilidad
- Precisión.
- Curva ROC.
- Se comprimió la matriz de confusión de cinco clases a dos clases para poder generar las medidas de rendimiento binarias. La estrategia usada para llevar a cabo lo antes mencionado fue la compresión de las diferentes clases de ataques DoS, *Probing*, R2L y U2R a una sola clase llamada Ataque en conjunto con la etiqueta Normal que no sufrió ninguna modificación.
- La generación de la curva ROC fue implementada siguiendo como referencia el artículo escrito por Fawcett [23] con la variante de que en el eje *Y* se presenta la tasa de acierto tanto para la clase positiva como negativa, y en el eje *X* se presenta la tasa de error.
- Los tiempos de entrenamiento y de prueba fueron cronometrados únicamente para el análisis sobre el conjunto de prueba.
- Se utilizó la técnica de Codo de Jambu presentada en la Sección 2.2 para la selección de número de grupos a ser utilizados. Adicionalmente, se utilizó también la técnica de validación cruzada de 10 conjuntos como complemento para evaluar la técnica de K-Medias usando dos grupos y cinco grupos, números que corresponden a la cardinalidad de las etiquetas dependiendo del enfoque adoptado para la clasificación:
 - **Dos clases:** Ataque y Normal.

- **Cinco clases:** DoS, Normal, *Probing*, R2L y U2R.
- Para la implementación de NN en el lenguaje de programación R se probaron dos paquetes: *nnet*⁷ y *neuralnet*⁸. Por una parte, *nnet* es el paquete de NN con mayor documentación en la web. Este paquete soporta arquitecturas de NN de solo una capa intermedia que se entrena haciendo uso de las técnicas de propagación hacia adelante y propagación hacia atrás para el ajuste de los pesos entre las neuronas. Por otra parte, el paquete *neuralnet* soporta arquitecturas de NN con múltiples capas intermedias; sin embargo, para problemas de clasificación este paquete no funcionó de buena manera con respecto al paquete *nnet*. En algunas pruebas de funcionalidad usando la misma arquitectura de NN, el paquete *nnet* obtuvo mejores resultados con respecto al paquete *neuralnet*, específicamente, el modelo entrenado con *nnet* se entrenó haciendo uso del conjunto total de los datos con una arquitectura 40-20-5, donde 40 representa el número de neuronas de entrada, 20 el número de neuronas de la única capa intermedia y 5 el número de clases objetivo, obteniendo un tiempo de entrenamiento bastante aceptable y con buenos resultados. Por otra parte, se utilizó el mismo escenario planteado previamente para el paquete *nnet* pero con el paquete *neuralnet*, y el modelo no completó el entrenamiento del modelo en 24 horas, mientras que el creado con *nnet* duró apenas cuatro minutos en entrenarse. Por lo mencionado previamente y haciendo referencia a las recomendaciones mencionadas en la Sección 2.2, se utilizaron 20 neuronas en la capa intermedia de los modelos de NN por defecto.
- Para la implementación de modelos de SVM haciendo uso del lenguaje R se utilizó el paquete *e1071*⁹. Este paquete es descrito como la interfaz de la biblioteca *libsvm*¹⁰, que es la biblioteca más utilizada por la comunidad científica para la resolución de problemas que ameriten la utilización de SVM. En la Sección 2.2 se presentó que para el kernel radial de SVM los parámetros configurables son *cost* y *gamma*. En el paquete *e1071*, el parámetro *cost* tiene el valor por defecto uno (1), mientras que el valor del parámetro *gamma* es derivado de la Ecuación 3.2.

$$\gamma = \frac{1}{\#VariablesPredictoras} \quad (3.2)$$

- Todas las pruebas fueron realizadas usando semillas para la reproducibilidad de los resultados en otros ambientes. Los valores de las semillas están indicados explícitamente en el código fuente de la solución que se encuentra disponible en un repositorio en GitHub¹¹. Por defecto se usó el número entero 22; sin embargo, en ocasiones durante el proceso de validación cruzada de 10 conjuntos, el valor de la semilla corresponderá al

⁷<https://cran.r-project.org/web/packages/nnet/index.html>

⁸<https://cran.r-project.org/web/packages/neuralnet/index.html>

⁹<https://cran.r-project.org/web/packages/e1071/index.html>

¹⁰<https://www.csie.ntu.edu.tw/~cjlin/libsvm/>

¹¹<https://github.com/deybanperez/ML4NIDS>

número de la iteración. Adicionalmente, el repositorio cuenta con un archivo *README* que presenta la organización del repositorio, indicando la estructura del mismo.

3.4. Infraestructura de la solución

En esta sección se presentan los diferentes componentes de *hardware* y *software* utilizados para el desarrollo aplicativo de la investigación.

- ***Hardware***

- Procesador *Intel Core 2 Duo* de 2.4 GHz.
- 2 GB de memoria RAM DDR2.

- ***Software***

- Ubuntu Server 16.04.1 LTS de 64bits como plataforma de desarrollo.
- RStudio Server como entorno de desarrollo integrado (IDE).
- Lenguaje de Programación R en su versión 3.3.1.

Se eligió esta arquitectura basada en un servidor dedicado debido a que los tiempos de entrenamiento y prueba sobre el conjunto de prueba de NSL-KDD fueron cronometrados y de esta manera todos los recursos de cómputo estuvieron dedicados exclusivamente al procesamiento de dichas tareas. Por otra parte, el lenguaje R fue elegido debido a las características presentadas en la Sección 2.2.2.

Capítulo 4

Análisis e interpretación de resultados

El presente capítulo presenta el análisis e interpretación de los resultados obtenidos luego de la implementación de las actividades que fueron llevadas a cabo para el logro de los objetivos descritos en las Secciones 1.1 y 1.2. La misma se rige por la metodología y consideraciones planteadas previamente en el Capítulo 3.

4.1. Recolección de los datos

En la Sección 2.2.1, se mencionó el hecho de que un buen proceso de recolección de datos es fundamental para un buen proceso de ML. Adicionalmente, en la Sección 2.3.3 se presentó el flujo general de trabajo dentro de lo que es la implementación de un NIDS haciendo uso de técnicas de ML. En la misma sección, se menciona el hecho de que el proceso de recolección de los datos es omitido debido a la complejidad para transformar las capturas de tráfico de red a una vista minable que pueda alimentar a los diferentes algoritmos de ML. Por lo expuesto previamente, se decidió utilizar el conjunto de datos NSL-KDD presentado en la Sección 2.3.3, ya que este conjunto de datos es una mejora del conjunto de datos KDD99, que es el conjunto de datos más utilizado por la comunidad científica para la implementación y análisis de NIDS basados en técnicas de ML. Por último, en la mayoría de los trabajos referentes a la propuesta de modelos de ML en la implementación de un NIDS usando técnicas de ML, el primer paso referente a la recolección de los datos fue omitido, haciendo uso de una base de conocimientos ya minada y pre-procesada, en este caso, el conjunto de datos utilizado fue el conjunto de datos NSL-KDD [24].

4.2. Pre-procesamiento

El trabajo realizado en esta fase fue muy ligero debido a que el conjunto de datos NSL-KDD ya había pasado previamente por un proceso de pre-procesamiento, característica que impulsó su uso en la presente investigación. Dicho esto, se realizó un análisis exploratorio de los datos para observar la distribución de los registros y algunas otras tareas tales como: extracción de características, renombramiento de columnas, eliminación de características,

transformación de los datos y generación de la vista minable. Las mismas serán presentadas a continuación.

4.2.1. Análisis exploratorio

El conjunto de datos NSL-KDD presenta dos versiones: entrenamiento y prueba. La Tabla 4.1 presenta un resumen de las características de ambos conjuntos de datos donde se listan el número de registros, número de columnas, peso en MB, número de etiquetas y el número de ataques presentes en cada conjunto de datos. Es importante destacar que el número de ataques es siempre una unidad menor que el número de etiquetas, debido a que la etiqueta que difiere corresponde a la etiqueta Normal.

Algunas características relevantes encontradas durante el análisis exploratorio de los datos fue que el conjunto de prueba contiene 17 tipos de ataques no presentes en el conjunto de entrenamiento. A su vez, el conjunto de entrenamiento posee 2 tipos de ataques no presentes en el conjunto de prueba. La diferencia entre los conjuntos de entrenamiento y de prueba está justificada en el hecho de que con el conjunto de prueba tan diferente del conjunto de entrenamiento se busca probar la capacidad de generalización de los diferentes modelos creados a partir del conjunto de entrenamiento. Finalmente, existen 39 tipos de ataques presentes entre ambos conjuntos.

El conjunto de prueba fue analizado específicamente buscando la cantidad de registros que correspondían a nuevos ataques. Se encontró que el mismo consta de 12833 ataques, donde 3750 representan nuevos tipos de ataques no presentes en el conjunto de entrenamiento y 9083 representan tipos de ataques sí presentes en el conjunto de entrenamiento.

Conjunto	Número Filas	Número Columnas	Peso (MB)	Número Etiquetas	Número Ataques
Entrenamiento	125973	43	19.1	23	22
Prueba	22544	43	3.4	38	37

Tabla 4.1: Resumen del conjunto de datos NSL-KDD.

4.2.2. Extracción de características

En la Sección 3.2 se indicó que los diferentes modelos tendrían en el primer nivel el objetivo de clasificación usando cinco variables objetivo, correspondiente a las cinco etiquetas concernientes a: DoS, *Probing*, R2L y U2R, y Normal. El conjunto de datos original está etiquetado por tipo de ataque y no por clase de ataque, es por eso que fue necesario extraer las clases de ataques de los conjuntos de datos de entrenamiento y prueba.

Para realizar la extracción de las clases de ataques, se utilizó como referencia el trabajo realizado por Dhanal y Shantharajah [33], donde presentan la asociación de cada tipo de ataque presente en el conjunto de datos NSL-KDD a la correspondiente clase de ataque dentro del conjunto de clases de ataques mencionadas en el párrafo anterior. La Tabla 4.2 presenta la distribución del número de registros por clase en los conjuntos de entrenamiento y de prueba.

Conjunto	DoS	Normal	Probing	R2L	U2R
Entrenamiento	45927	67343	11656	995	52
Prueba	7458	9711	2421	2754	200

Tabla 4.2: Distribución de las clases en el conjunto de datos NSL-KDD.

La Figura 4.1 muestra gráficamente cómo en el conjunto de entrenamiento las clases DoS y la clase Normal son las clases que mayor cantidad de registros presentan. Por otra parte, las clases Probing, R2L y U2R presentan menor cantidad de registros mostrando un desbalance bastante marcado con respecto a las clases.

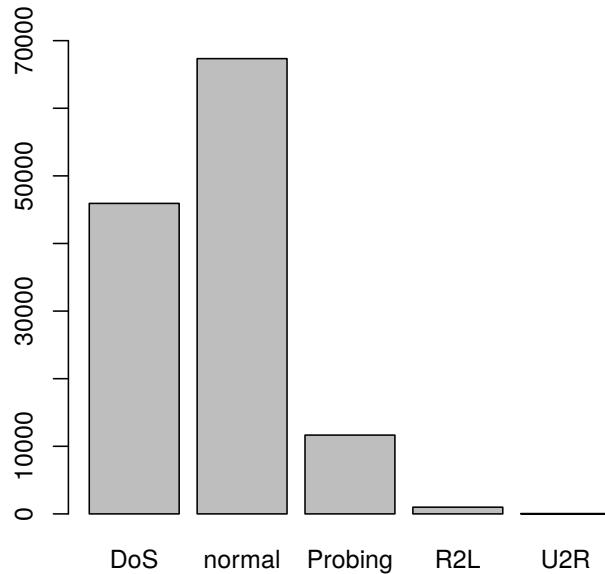


Figura 4.1: Distribución de clases en el conjunto de entrenamiento.

La Figura 4.2 presenta la distribución gráfica de los registros en el conjunto de prueba. En ella se puede observar como las diferentes clases están más equilibradas con respecto a la cantidad de registros que poseen, agregando mayor cantidad y variedad de ataques. Esto debido a que se desea que el escenario de prueba sea lo suficientemente riguroso para probar la capacidad de generalización de los distintos modelos entrenados haciendo uso del conjunto de entrenamiento.

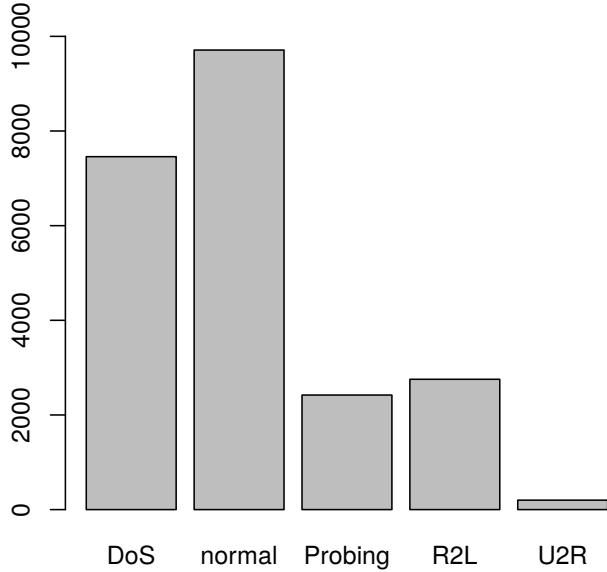


Figura 4.2: Distribución de clases en el conjunto de prueba.

Como última labor en el proceso de extracción de características, se comprimieron los tipos de ataques en una clase llamada Ataque y se agregó una nueva columna que etiqueta a los registros con las etiquetas Ataque o Normal.

4.2.3. Renombramiento de columnas

Las variables predictoras del conjunto de datos NSL-KDD carecían de nombres. Por tal motivo, se utilizó de nuevo como referencia el trabajo realizado por Dhanabl y Shantharajah [33] para asignarle el respectivo nombre a cada variable predictora.

4.2.4. Eliminación de características

Una de las validaciones realizadas fue la de que no hubiese ningún registro faltante; así mismo, también se validó que todas las variables predictoras tuvieran un valor no fijo; es decir, que no fueran una constante. Dentro de ese proceso de validación, se observó que la variable predictora *Num_outbound_cmd* poseía un único valor fijo correspondiente al valor entero cero (0) para todos los registros del conjunto de entrenamiento y del conjunto de prueba; es decir, esa variable predictora era una constante y por consecuente no aporta información. Por el motivo presentado previamente, esta columna fue eliminada de los conjuntos de entrenamiento y de prueba.

4.2.5. Transformación de los datos

Los algoritmos utilizados para la presente investigación correspondientes a SVM, NN y K-Medias no aceptan variables predictoras que no sean de tipo cuantitativo. Por tal motivo,

las variables predictoras correspondientes a: *Protocol_type*, *Service* y *Flag*, que eran originalmente de tipo cualitativo fueron transformadas a tipo cuantitativo. La estrategia adoptada para la transformación de las mismas pasó por ordenar alfabéticamente en orden ascendente los diferentes valores que podría adoptar cada variable predictora y asignar un número en el intervalo [1, longitud_vector] de igual manera de forma ascendente a cada una de las posiciones. Esta estrategia no tiene ninguna ganancia particular con respecto a los resultados posteriormente obtenidos, simplemente se adoptó por practicidad en la interpretación de la información.

4.2.6. Generación de la vista minable

Con las actividades descritas en las secciones previas se completó el proceso de preprocesamiento. Finalmente se generó una nueva vista minable para los conjuntos de entrenamiento y de prueba que constan de 44 columnas en total, donde 40 corresponden a variables predictoras y las restantes a las diferentes etiquetas mencionadas con anterioridad.

4.3. Implementación de modelos

Esta sección recopila las actividades realizadas para el entrenamiento, validación y análisis de los diferentes modelos planteados en la Sección 3.2. La presente sección está dividida en sub-secciones que recopilan las actividades realizadas en las diferentes iteraciones realizadas hasta alcanzar los resultados esperados, ya que como se mencionó en la Sección 3.1, se utilizó una metodología iterativa y empírica para elegir los modelos que mejor se adaptaran al escenario. Se realizaron cuatro iteraciones, las mismas serán presentadas a continuación.

4.3.1. Iteración 1

La Iteración 1 consiste en la prueba de rendimiento de los modelos:

1. NN - K-Medias.
2. SVM (Radial) - K-Medias.

Estos modelos fueron planteados en la Sección 3.2. Como se mencionó en dicha sección, con la implementación y análisis de ambos modelos se busca conocer si el diseño planteado se ajustaba de buena manera o no al problema. Por dicha razón, en esta primera iteración se utilizaron los parámetros por defecto y el conjunto de características total de las variables predictoras de los conjuntos de datos generados en la Sección 4.2.6.

El análisis empezará por la selección del número de grupos de K-Medias, luego se hará un análisis sobre el conjunto de entrenamiento, sobre el conjunto de prueba y se presentarán las conclusiones de la iteración.

Selección del número de grupos para K-Medias

El resultado obtenido luego de la aplicación del Codo de Jambu se muestra en la Figura 4.3. En la misma se puede observar que el gráfico del Codo de Jambu no se asemeja al presentando en la Figura 2.4. El motivo de este comportamiento se debe a que el conjunto de datos presenta múltiples formas de agrupación, los mismos pueden ser:

- Ataque vs normal.
- Clase de ataque vs normal.
- Tipo de ataque vs normal.

Por lo expuesto previamente la gráfica no se estabiliza en ningún punto y por tal motivo el Codo de Jambu no funciona de buena manera para la selección de grupos en este caso, donde además de presentarse muchas maneras de agrupar los registros, los mismos están solapados entre si.

Previamente se presentó que el Codo de Jambu no fue una buena medida de referencia para obtener el número de grupos a utilizar. Al tener el conjunto de datos etiquetado se conoce que hay dos maneras de clasificar el conjunto de datos, utilizando dos clases correspondiente a las etiquetas: Ataque o Normal y utilizando cinco clases correspondientes a las etiquetas: DoS, Normal, *Probing*, R2L y U2R.

Por otra parte, los nombres que aparecen en la leyenda de la Figura 4.3 corresponden a los diferentes algoritmos de distancias presentados para el algoritmo K-Medias. Para la elección del mejor algoritmo de distancia, se busca utilizar aquel que maximice la inercia inter-grupos, que se refiere a la maximización de la separación entre los diferentes grupos creados [17].

La Tabla 4.3 ilustra los resultados de la inercia inter-grupos utilizando dos y cinco grupos. En ambos casos se observa que el algoritmo Hartigan fue el que acumuló mayor inercia inter-grupos y por consecuente fue el elegido para ser utilizado para la clasificación con el algoritmo K-Medias.

Algoritmo	Inercia Inter-Grupos
Hartigan	726396
Lloyd	726393.5
Forgy	726393.5
MacQueen	726393.5

Tabla 4.3: Inercia inter-grupos del conjunto de datos (Iteración 1).

Para seleccionar la estrategia a utilizar se utilizó la técnica de validación cruzada de 10 conjuntos sobre los enfoques de dos y cinco clases, y usando el algoritmo que maximizara

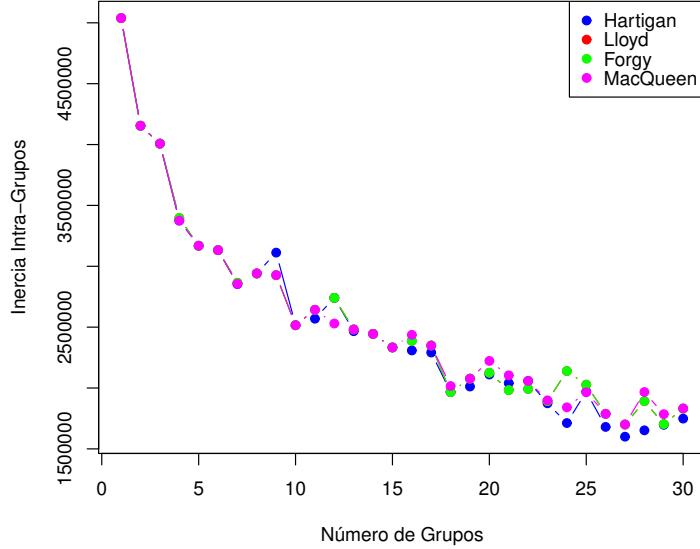


Figura 4.3: Codo de Jambu sobre el conjunto de datos (Iteración 1).

la inercia inter-grupos para cada caso (Ver Tabla 4.3). Adicionalmente se calcularon ciertas métricas basadas en matrices de confusión para analizar su eficacia y precisión en la clasificación del tráfico de red.

■ Análisis usando cinco grupos

La Tabla 4.4 muestra la matriz confusión de cinco clases del mejor modelo obtenido durante el proceso de validación cruzada de 10 conjuntos sobre el conjunto de datos de entrenamiento haciendo uso del algoritmo de clasificación K-Medias. En la misma se puede observar como esta está bastante desordenada, específicamente clasifica una gran cantidad de registros como pertenecientes a la clase *Probing* que en verdad pertenecen a la clase DoS. Otro aspecto a resaltar en dicha tabla es que las clases R2L y U2R no tuvieron ningún acierto. Este comportamiento no es bueno y da razones para pensar que cinco grupos no se ajusta de buena manera al escenario.

Real \ Predicción	DoS	Normal	Probing	R2L	U2R
DoS	34329	4691	68888	9	10
Normal	115	63828	119	425	2856
Probing	384	5845	869	4217	341
R2L	3	940	0	0	52
U2R	0	52	0	0	0

Tabla 4.4: Matriz de confusión de cinco clases del mejor modelo K-Medias sobre el conjunto de datos de entrenamiento (Iteración 1).

La Tabla 4.5 muestra la tasa de acierto por clase y la tasa de aciertos total de K-Medias. Se puede observar una muy buena tasa de acierto para las clases DoS y Normal; sin embargo, para el resto de las clases el rendimiento es bastante pobre. La tasa total de acierto es de 78.61 %, una tasa de aciertos bastante alta y engañosa, porque la mayoría de los registros se concentran en las clases DoS y Normal, pero la realidad es que para las otras tres clases el desempeño no fue bueno.

DoS	Normal	Probing	R2L	U2R	Total
74.75 %	94.78 %	7.46 %	0 %	0 %	78.61 %

Tabla 4.5: Tasa de acierto de cinco clases del mejor modelo K-Medias sobre el conjunto de datos de entrenamiento (Iteración 1).

La Tabla 4.6 presenta la matriz de confusión de dos clases del mejor modelo K-Medias usando cinco clases sobre el conjunto de datos de entrenamiento. En la misma se puede observar como en la diagonal se concentran la mayoría de los registros; es decir, la separación entre ataques y tráfico normal se realizó de buena manera, contrastando con la matriz de confusión original de cinco clases presentada previamente en la Tabla 4.4. Adicionalmente, se puede observar que el gran problema fue la gran cantidad de falsos negativos que se generaron, esta situación es poco deseable ya que si los ataques no son notificados no hay posibilidad de retro-alimentar el modelo para tomar acciones sobre dichos ataques y básicamente los ataques habrán logrado su cometido. Dicho de otra manera, se desea maximizar la tasa de verdaderos positivos, verdaderos negativos, y minimizar las tasas de falsos positivos y de falsos negativos. Específicamente, reducir la tasa de falsos negativos, ya que con los falsos positivos se puede retro-alimentar el modelo para que este pueda ser más preciso. En cambio, con los falsos negativos ninguna acción puede ser llevada a cabo.

Real \ Predicción	Ataque	Normal
Ataque	47102	11528
Normal	3515	63828

Tabla 4.6: Matriz de confusión de dos clases del mejor modelo K-Medias de cinco clases sobre el conjunto de datos de entrenamiento (Iteración 1).

■ Análisis usando dos grupos

Una vez obtenidas las métricas de rendimiento para el uso de K-Medias con cinco clases, se realizaron los mismos pasos para obtener las métricas para dos clases: Ataque y Normal.

La Tabla 4.7 presenta la matriz de confusión de dos clases del mejor modelo obtenido durante el proceso de validación cruzada de 10 conjuntos usando K-Medias. En la misma se puede observar una similitud con la matriz de confusión de dos conjuntos sobre la clasificación utilizando cinco clases presentada en la Tabla 4.6. En ambas matrices de confusión la mayoría de los registros están concentrados en la diagonal; es decir, la mayoría de los registros fueron clasificados de buena manera. Sin embargo, en esta ocasión de nuevo se observa la presencia de muchos falsos negativos, menos que en la matriz de confusión utilizando cinco clases pero siguen siendo muchos. Como punto favorable, en esta ocasión se observa una mejora notable con respecto a la correcta clasificación de ataques y del tráfico normal, presentando una poca generación de falsos positivos.

Real \ Predicción	Ataque	Normal
Ataque	47351	11279
Normal	681	66662

Tabla 4.7: Matriz de confusión de dos clases del mejor modelo K-Medias de cinco clases sobre el conjunto de datos de entrenamiento (Iteración 1).

■ Comparación entre los enfoques de dos y cinco grupos

La Tabla 4.8 presenta una tabla comparativa entre las matrices de dos clases de los enfoques de K-Medias utilizando cinco y dos grupos respectivamente. En la misma, se puede observar como el uso de dos grupos domina claramente sobre el uso de cinco grupos, incluso si el enfoque de cinco grupos es llevado a dos grupos. Por lo mismo, se utilizó en esta primera iteración el enfoque de K-Medias utilizando dos clases objetivo referentes a: Ataque y Normal.

Adicionalmente, en la Sección 2.3.1 se presentó que el enfoque de las técnicas no-supervisadas dentro de lo que es la implementación de un NIDS se basan en detectar aquellos registros que se desvían en cierta medida del comportamiento normal; es decir, estos teóricamente solo son capaces de definir si un registro es anómalo o normal, y no dan la oportunidad de catalogar explícitamente la clase del ataque al que el registro pertenece. El concepto presentado previamente se demuestra claramente con los resultados obtenidos en la evaluación presentada previamente.

■ Conclusiones parciales

El Codo de Jambu no fue de gran ayuda debido a las múltiples maneras de representar el conjunto de datos que hace imposible que la inercia intra-grupos se estabilice. Por otra parte, luego de seleccionar el algoritmo de distancia que mejor se ajustó a cada enfoque, se observó como el enfoque de dos grupos es claramente superior al de cinco grupos presentando mayor tasa de acierto y menor varianza en los resultados.

Modelo	Acierto Ataque	Acierto Normal	Acierto total	Acierto Promedio	Sensibilidad	Especificidad	Precisión
5 Grupos	80.34 %	94.78 %	88.06 %	71.87 %	80.34 %	94.78 %	93.06 %
2 Grupos	80.76 %	98.99 %	90.51 %	76.55 %	80.76 %	98.99 %	98.58 %

Tabla 4.8: Comparación de las medidas de rendimiento binarias extraídas de los enfoques de cinco y dos grupos usando K-Medias sobre el conjunto de datos de entrenamiento (Iteración 1).

Análisis de modelos sobre el conjunto de entrenamiento

En esta sección se presenta el análisis e interpretación de los resultados obtenidos del rendimiento de los modelos sobre el conjunto de datos de entrenamiento. En esta iteración los modelos harán uso del conjunto total de características que corresponde a 40 variables predictoras. Debido a esto la arquitectura utilizada para el modelo (1) NN - K-Medias será 40-20-5. Donde 40 corresponde al número de neuronas de la capa de entrada que corresponde al número de variables predictoras, 20 al número de neuronas de la capa intermedia que fueron seleccionadas siguiendo las consideraciones tratadas en la Sección 2.2 y 5 neuronas que corresponden a la capa de salida debido al uso de cinco clases objetivos correspondientes a las clases: DoS, Normal, *Probing*, R2L y U2R. Por otra parte, el modelo (2) SVM (Radial) - K-Medias tendrá como parámetros por defecto costo = 1 y gamma = 0.025, por lo expuesto en la Sección 3.3.

La Tabla 4.9 presenta la tasa de acierto por clase y la tasa de acierto total derivada de la matriz de confusión de cinco clases concerniente al primer nivel de clasificación de los dos modelos implementados en esta iteración. En la misma se puede observar como ambos modelos presentan un rendimiento excelente, donde NN destaca un poco más que SVM (Radial) pero al final ambos tienen un rendimiento similar.

Modelo	DoS	Normal	Probing	R2L	U2R	Total
(1) NN - K-Medias	99.91 %	99.78 %	99.00 %	93.24 %	37.50 %	99.67 %
(2) SVM (Radial) - K-Medias	99.80 %	99.46 %	98.59 %	83.78 %	25.00 %	99.36 %

Tabla 4.9: Tasas de acierto (5 Clases) del primer nivel de los modelos sobre el conjunto de datos de entrenamiento (Iteración 1).

En la Figura 4.4 se presentan las curvas ROC correspondientes al primer nivel de clasificación de ambos modelos. En esta se puede observar como el modelo (1) NN - K-Medias toma decisiones con mucha más certeza que el modelo (2) SVM (Radial) - K-Medias, donde la gráfica errática indica que se combinan muchos aciertos y fallos con niveles de certeza similares.

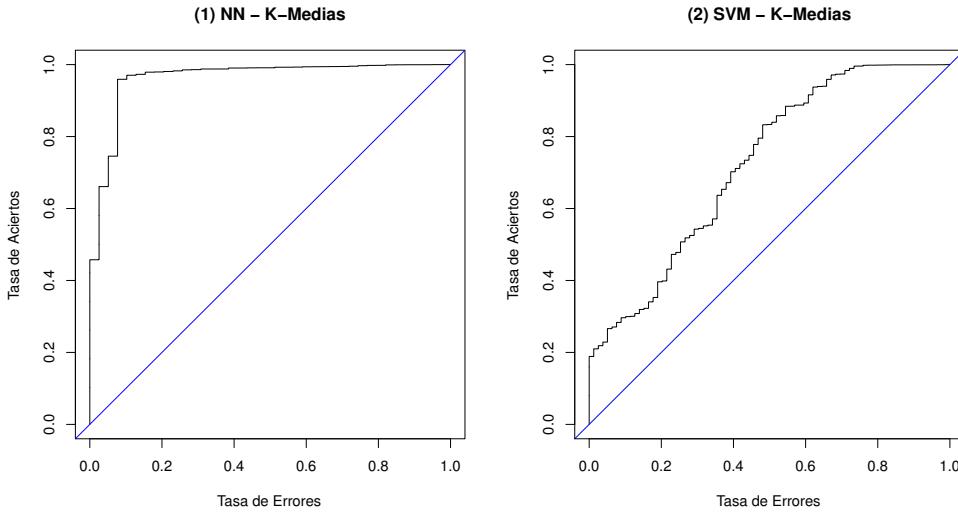


Figura 4.4: Curvas ROC de los algoritmos del primer nivel de los modelos sobre el conjunto de datos de entrenamiento (Iteración 1).

En la Tabla 4.10 se presenta una tabla comparativa con las medidas de rendimiento binarias de ambos modelos. En la misma destaca el muy buen desempeño del primer nivel correspondiente a los algoritmos de aprendizaje supervisado NN y SVM (Radial), y el poco aporte obtenido por parte del segundo nivel correspondiente a K-Medias que no logró una detección significativa de ataques; sin embargo, no generó una gran cantidad de falsos positivos y por consecuente no deterioró de gran manera las labores realizadas por el primer nivel. Es importante destacar que con la generación de falsos positivos se puede retro-alimentar el modelo para que estos aumenten su precisión.

Modelo	Parámetros	Tipo	Sensibilidad	Especificidad	Precisión	Tasa Acierto
(1) NN - K-Medias	Neuronas = 20	NN (2 Clases)	99.59 %	99.78 %	99.75 %	99.69 %
		NN \Rightarrow K-Medias	12.5 %	92.15 %	0.57 %	91.86 %
		NN + K-Medias	99.64 %	91.94 %	91.58 %	95.55 %
(2) SVM (Radial) - K-Medias	costo = 1 gamma = 0.025	SVM (2 Clases)	99.27 %	96.46 %	99.38 %	99.37 %
		SVM \Rightarrow K-Medias	13.95 %	95.08 %	1.80 %	94.56 %
		SVM + K-Medias	94.37 %	94.57 %	94.15 %	96.81 %

Tabla 4.10: Medidas de rendimiento binarias de los modelos sobre el conjunto de datos de entrenamiento (Iteración 1).

- **Conclusiones parciales** Se presentó un muy buen desempeño en la tasa de acierto por parte de los algoritmos de enfoque supervisado correspondientes al primer nivel, donde NN presentó una curva ROC mucho más certera que SVM (Radial).

Por otra parte, K-Medias no fue un buen complemento debido a la excelente labor del primer nivel y a la poca generación de falsos negativos. Como punto favorable se

tiene que no generó una gran cantidad de falsos positivos que deterioraran de manera significativa las labores realizadas por el primer nivel.

Análisis de modelos sobre el conjunto de prueba

En esta sección se presenta el análisis e interpretación de resultados obtenidos del rendimiento de los modelos sobre el conjunto de datos de prueba. En esta se tomaron las mismas consideraciones mencionadas en la Sección 4.3.1 con respecto a la implementación de los modelos.

En la Tabla 4.11 se presenta la tasa de acierto por clase y la tasa de acierto total del primer nivel de los modelos sobre el conjunto de datos de prueba. En la misma se observa como hubo un decremento considerable en la tasa de acierto debido a la cantidad de nuevos registros presentes en el conjunto de datos de prueba que no estuvieron presentes en el conjunto de datos de entrenamiento. Adicionalmente, destaca que en comparación a la tasa de acierto por clase presentada en la Tabla 4.9 correspondiente al análisis sobre el conjunto de datos de prueba, en esta ocasión el algoritmo SVM (Radial) presentó mayor tasa de acierto que el algoritmo NN. Resultado respaldado por la tasa de acierto en las clases DoS y Normal, debido a que en el resto de las clases NN obtuvo un mejor desempeño.

Tipo	DoS	Normal	Probing	R2L	U2R	Total
NN	79.67 %	96.16 %	69.35 %	12.85 %	2.00 %	76.81 %
SVM (Radial)	82.13 %	98.04 %	63.65 %	7.81 %	0.00 %	77.19 %

Tabla 4.11: Tasas de acierto (5 Clases) del primer nivel de los modelos sobre el conjunto de datos de prueba (Iteración 1).

En la Figura 4.5 se presenta la curva ROC de los modelos producto de la clasificación sobre el conjunto de datos de prueba. En la misma se puede observar como NN y SVM (Radial) decrementaron su rendimiento en comparación al obtenido sobre el conjunto de entrenamiento presentado en la Figura 4.4. A pesar de esto, el algoritmo NN sigue presentando mejor desempeño que el algoritmo SVM (Radial) debido a que la curva ROC del primero se despegó mucho más de la línea diagonal correspondiente la línea del azar.

En la Tabla 4.12 se presenta una tabla comparativa con las medidas de rendimiento binarias por nivel de ambos modelos. En la misma destaca el hecho de que la tasa de acierto al pasar de la matriz de confusión de cinco clases a dos clases aumentó. Este comportamiento se debe a que con la clasificación de cinco clases se cometieron errores de clasificación entre clases de ataques. Sin embargo, la varianza entre la tasa de acierto entre cinco y dos clases es muy poca y despreciable. Por otra parte, para el modelo (1) NN - K-Medias el complemento del algoritmo K-Medias para el primer nivel correspondiente a NN fue mucho mejor que para el modelo (2) SVM (Radial) - K-Medias desde un punto de vista de sensibilidad, que indica que se detectaron muchos más ataques. Por otra parte, el modelo (2) SVM (Radial) presenta

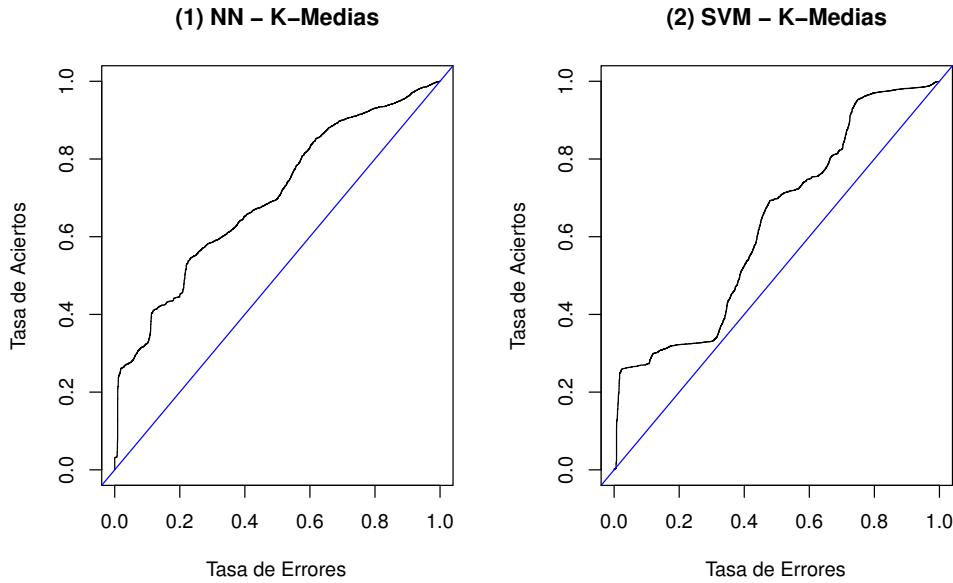


Figura 4.5: Curvas ROC de los algoritmos del primer nivel de los modelos sobre el conjunto de datos de prueba (Iteración 1).

menos cantidad de falsos positivos, haciendo más certeras las detecciones de ataques. Que la clasificación de SVM (Radial) y K-Medias haya sido similar para el modelo (2) SVM (Radial) K-Medias tiene que ver con que ambos algoritmos de clasificación se concentran en la búsqueda de circunferencias en espacios n-dimensionales. Por lo tanto, el complemento de ambos algoritmos no es del todo bueno debido a que ambos algoritmos podría decirse que hacen lo mismo. Finalmente, los tiempos de entrenamiento y de prueba de NN son muchos menos que los de SVM (Radial). De esta manera se observa no solo que el algoritmo de NN es más certero sino también mucho más rápido.

Modelo	Parámetros	Tipo	Sensibilidad	Especificidad	Precisión	Tasa Acierto	Tiempo Entrenamiento	Tiempo Prueba
(1) NN - K-Medias	Neuronas = 20	NN (2 Clases)	65.56 %	96.16 %	95.75 %	78.74 %	266.68 segs	0.38 segs
		NN \Rightarrow K-Medias	52.81 %	82.21 %	58.42 %	72.76 %	7.97 segs	0.08 segs
		NN + K-Medias	83.75 %	79.05 %	84.09 %	81.72 %	274.65 segs	0.46 segs
(2) SVM (Radial) - K-Medias	costo = 1 gamma = 0.025	SVM (2 Clases)	63.53 %	98.04 %	97.72 %	78.40 %	1216.12 segs	20.70 segs
		SVM \Rightarrow K-Medias	11.58 %	99.15 %	87.14 %	70.30 %	9.44 segs	0.07 segs
		SVM + K-Medias	67.76 %	97.21 %	96.98 %	80.45 %	1225.56 segs	20.77 segs

Tabla 4.12: Medidas de rendimiento binarias de los modelos sobre el conjunto de datos de prueba (Iteración 1).

■ Conclusiones parciales

En la evaluación sobre el conjunto de datos de prueba se apreció como la eficacia de los modelos disminuyó considerablemente. Este aspecto es reflejado en la tasa de acierto

y en la curva ROC. A pesar de la gran disminución en la tasa de acierto, los modelos presentaron un muy buen desempeño considerando que clasificaban registros no presentes en el conjunto de entrenamiento.

Un aspecto resaltante fue el de que K-Medias sirvió de buena manera como complemento del modelo (1) NN - K-Medias elevando su rendimiento alrededor de un 3 %. Mientras que por otra parte, para el modelo (2) SVM (Radial) - K-Medias, el segundo nivel correspondiente a K-Medias no fue un buen complemento debido a que ambos algoritmos presentan el mismo enfoque de separación buscando circunferencias en espacios n-dimensionales.

Conclusiones de la Iteración 1

Los modelos base (1) NN - K-Medias y (2) SVM (Radial) - K-Medias presentan un muy buen desempeño a la hora de detectar ataques conocidos. En este caso, K-Medias no presenta un buen desempeño debido a que el primer nivel genera muy poca tasa de falsos negativos y el segundo nivel de K-Medias genera altas tasas de falsos positivos que podrían ayudar al modelo a ser más preciso ya que los mismos servirían como retro-alimentación para el modelo.

Sobre el conjunto de prueba, el rendimiento bajó considerablemente con respecto a los resultados obtenidos en la evaluación sobre el conjunto de entrenamiento. Este comportamiento es entendible debido a la nueva cantidad de registros nuevos agregados en este conjunto de datos. Sin embargo, el comportamiento del modelo híbrido es muy bueno en el primer nivel en especial para NN que presenta una curva ROC superior en rendimiento que la presentada para SVM (Radial). Adicionalmente, el modelo (1) NN - K-Medias se complementa de buena manera demostrando adaptarse de mejor forma que el modelo (2) SVM (Radial) - K-Medias en la tarea de la detección de ataques y correcta clasificación del tráfico normal.

Por lo expuesto previamente, se refleja que los enfoques híbridos planteados en la Sección 3.2 son acordes al problema de implementación de un NIDS basado en técnicas de aprendizaje automático y en futuras iteraciones se pueden hacer modificaciones de características y parámetros sobre estos para obtener mejores resultados.

4.3.2. Iteración 2

En la Iteración 1 presentada en la Sección 4.3.1 se llegó a la conclusión de que la combinación de técnicas de enfoque supervisado tales como NN y SVM en conjunto con técnicas de aprendizaje no-supervisado, en este caso K-Medias es efectiva. Una vez demostrado lo anterior, la Iteración 2 corresponde a modificaciones sobre los modelos (1) NN - K-Medias y (2) SVM (Radial) - K-Medias buscando el aumento de la eficacia a la hora de detectar intrusos y/o anomalías en redes de computadoras.

En esta iteración se utilizaron las técnicas de selección de parámetros y selección de características para la siguiente combinación de estrategias y modelos.

1. (1) NN - K-Medias.
2. (2) SVM (Radial) - K-Medias.
3. (3) PCA - NN - K-Medias.
4. (4) PCA - SVM (Radial) - K-Medias.
5. (5) GFR - NN - K-Medias.
6. (6) GFR - SVM (Radial) K-Medias.

Las consideraciones de implementación ya fueron presentadas a detalle en el Capítulo 3. Sin embargo, en resumen en esta iteración se seleccionó el mejor conjunto de parámetros haciendo uso de la técnica de validación de modelos concerniente a la validación cruzada de 10 conjuntos. Adicionalmente, para los modelos (3) PCA - NN - K-Medias, (4) PCA - SVM (Radial) - K-Medias, (5) GFR - NN - K-Medias y (6) GFR - SVM (Radial) K-Medias, se utilizaron las técnicas de selección de características PCA y GFR que son identificadas en los prefijos de los diferentes modelos buscando una solución que quitara ruido a los modelos, los hiciera más rápidos y precisos.

Por lo expuesto previamente, esta sección presentará la selección de características, selección de parámetros, selección del número de grupos para K-Medias, evaluación de los modelos sobre los conjuntos de datos de entrenamiento y de prueba, y las conclusiones obtenidas al final de la iteración.

Selección de características

La selección de características fue la primera actividad realizada en esta iteración. Como se mencionó previamente se utilizaron las técnicas referentes a PCA y GFR. Las mismas fueron descritas a detalle en la Sección 2.2.1 y su uso es justificado en el Capítulo 3.

▪ PCA

Como se explicó en la Sección 2.2.1, la técnica de reducción de características PCA consiste en rotar el conjunto de datos para capturar la mayor cantidad de varianza en un conjunto de datos reducido usando como herramienta la matriz de covarianza. En la Figura 4.6 se muestra la proporción de varianza acumulada por número de componentes principales. En la misma se puede observar como con aproximadamente 27 o 28 componentes principales se logra acumular la mayoría de la varianza del conjunto de datos. Los resultados obtenidos muestran que con 24 componentes principales se logra la acumulación del 95.19 % de la varianza acumulada del conjunto de datos y con 30

componentes principales el 99.16 % de varianza acumulada sobre el mismo. Guiándonos por lo presentado en la Sección 2.2.1, se debería utilizar un número de componentes principales en el rango [24, 30].

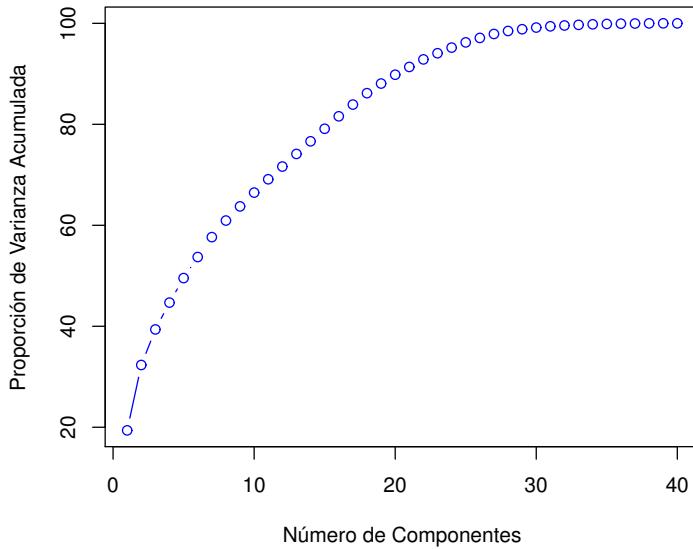


Figura 4.6: Varianza acumulada por componente principal en la aplicación de PCA (Iteración 2).

Un paso muy común de análisis exploratorio haciendo uso de PCA es la graficación de las dos primeras componentes principales, esto buscando encontrar alguna distribución bien marcada de los diferentes grupos o clases en dichas componentes principales. La Figura 4.7 presenta dicha imagen, en ella se puede observar que parece haber regiones donde la separación de las clases Normal, DoS y *Probing* están bien definidas; sin embargo, también se observa claramente como muchos registros son solapados entre sí hacia el medio de la gráfica, haciendo imposible el correcto establecimiento de fronteras separadoras de clases en dos dimensiones.

Se realizó una actividad extra que fue la de aplicar la técnica de validación cruzada de 10 conjuntos iterando con el número de componentes principales en el rango [1, 40]; es decir se fueron agregando componentes principales hasta llegar a las 40. En cada iteración se realiza el proceso de validación cruzada de 10 conjuntos, se calculó la desviación estándar de los resultados y se graficaron las medias de los resultados por número de componentes principales. A continuación se presentan los resultados obtenidos para NN y SVM. Como aspecto a destacar, la implementación de esta técnica de agregación de componentes combinada con validación cruzada de 10 conjuntos tomó alrededor de 1 día para su completación tanto para NN como para SVM.

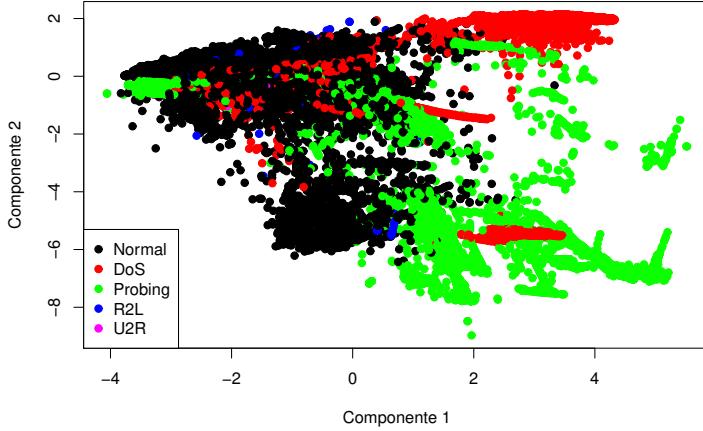


Figura 4.7: Graficación de las dos componentes principales del conjunto de datos de entrenamiento luego de la aplicación de PCA (Iteración 2).

- **NN**

En la Figura 4.8 se presenta el gráfico con la desviación estándar y la media de acierto por número de componentes principales. En la misma se puede observar como la media de aciertos para siete componentes principales forma una articulación donde se alcanza alrededor del 99 % de acierto y a partir de allí los resultados producto de agregar más componentes principales incrementan despreciablemente la tasa de acierto. Por otra parte, se puede ver como con siete componentes principales la desviación estándar de los resultados fue muy baja, solo de 0.0010, que corresponde al 0.1 %. Dicho esto, aparentemente con siete componentes principales se podrían tener muy buenos resultados reduciendo en un 82.5 % el número de componentes principales a ser utilizadas.

- **SVM**

En la Figura 4.9 se presenta el gráfico de la desviación estándar y la media de acierto por número de componentes principales. En esta se puede observar como al igual que para NN con siete componentes se alcanza una articulación en la cual agregar mayor cantidad de componentes principales no aumenta de forma significativa la tasa de aciertos. Adicionalmente, se observa también como con siete componentes se alcanza una desviación estándar menor a 0.1 % que representa una de las desviaciones más bajas entre todas las características y que adicionalmente es más baja que la presentada en el estudio sobre NN presentado en la Tabla 4.8.

- **Conclusiones parciales**

Por los resultados presentados anteriormente ilustrados gráficamente en las Figuras 4.8 y 4.9, los modelos en esta iteración basados en PCA constarán de siete

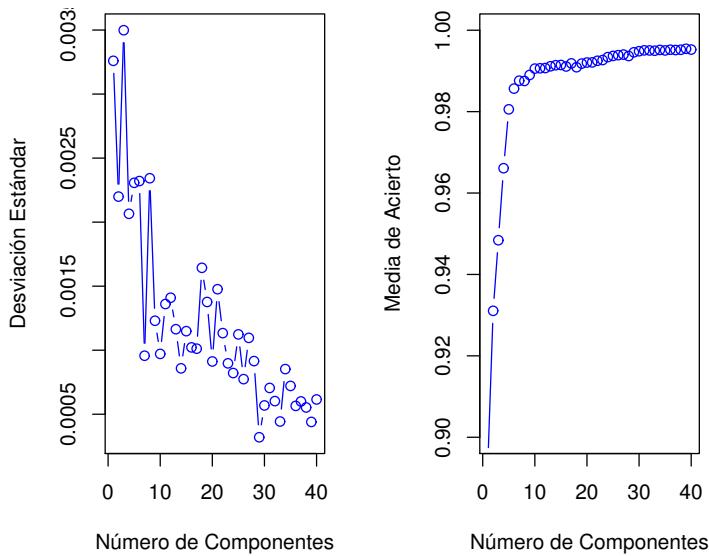


Figura 4.8: Graficación de la desviación estándar y media de acierto por componente principal para NN (Iteración 2).

componentes principales tanto para SVM como para NN, ya que con este número de componentes se observa como sobre el conjunto de entrenamiento se obtuvieron altas tasas de acierto con baja desviación estándar.

■ GFR

En la Sección 2.2.1 se presentó a detalle la técnica de GFR. Las actividades realizadas con GFR consistieron en seleccionar las características tanto para NN como para SVM de manera independiente. A continuación se presentan los resultados obtenidos para cada algoritmo. Es importante destacar que este procesamiento fue realizado usando la técnica de validación cruzada de 10 conjuntos y los parámetros por defecto de 20 neuronas para la capa intermedia de NN, y para SVM el costo = 1 y $\gamma = 1 / \#VariablesPredictoras$. Adicionalmente, el tiempo total invertido para la selección de características fue de 18 días tanto para NN como para SVM.

• NN

En la Tabla 4.13 se presenta la tabla con las características ordenadas por orden de importancia para NN. En la misma destaca el hecho del orden de las características debido a que se le da mayor importancia para la correcta clasificación de los registros a los detalles de conexión tales como el contador de conexiones realizadas al mismo *host* en los últimos dos segundos (count) y al tipo de protocolo (Protocol_type) que a otras características que se pensaría tendrían más relevancia como por ejemplo el número de terminales abiertas (Num_shells) o el

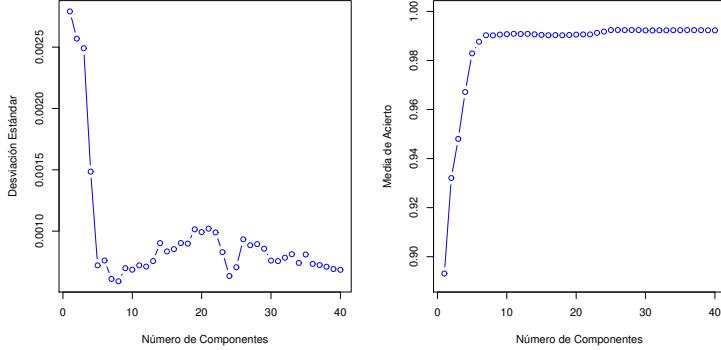


Figura 4.9: Graficación de la desviación estándar y media de acierto por componente principal para SVM (Iteración 2).

número de terminales abiertas en modo administrador (Num_root). Esto es debido a que si un atacante buscar tener permisos de administrador sobre el sistema tratará solo de abrir una conexión para pasar desapercibido.

1	Count	11	Service	21	Su_attempted	31	Urgent
2	Protocol_type	12	Logged_in	22	Land	32	Src_bytes
3	Dst.host_srv_count	13	Is_guest_login	23	Rerror_rate	33	Dst.host_srv_rerror_rate
4	Dst.host_same_src_port_rate	14	Dst.host_srv_diff.host_rate	24	Num.access_files	34	Num_file_creations
5	Hot	15	Same_srv_rate	25	Root_shell	35	Srv_count
6	Dst.host_count	16	Duration	26	Num_failed_logins	36	Srv_error_rate
7	Dst.host_serror_rate	17	Srv_rerror_rate	27	Is_host_login	37	Num_compromised
8	Dst.host_count	18	Srv_diff_host_rate	28	Num_shells	38	Diff_srv_rate
9	Wrong_fragment	19	Dst.host_same_srv_rate	29	Serror_rate	39	Dst_bytes
10	Dst.host_diff_srv_rate	20	Flag	30	Num_root	40	Dst.host_srv_serror_rate

Tabla 4.13: Ordenamiento de características para NN usando GFR (Iteración 2).

La Figura 4.10 presenta el gráfico de las dos características seleccionadas como más importantes para NN referentes a Count y Protocol_type. Con este gráfico se buscó identificar posibles patrones separadores en dos dimensiones. La gráfica muestra una gran diferencia con el gráfico de las dos componentes principales de PCA presentada previamente en la Figura 4.7 donde se muestra una nube de puntos, en esta ocasión la gráfica presenta un enfoque más analítico donde dependiendo del rango en el que se encuentren los puntos se asocian a una clase u a otra.

La Figura 4.11 presenta el gráfico de la desviación estándar y media de acierto por número de características. En la misma se puede observar como con nueve características se alcanza una articulación donde agregar mayor cantidad de características resulta en una mejora de acierto poco significativa. Adicionalmente, se puede ver como la desviación estándar para dicho número de características es ligeramente superior a 0.1%, resultado muy bajo que indica que los resultados obtenidos durante la fase de validación cruzada de 10 conjuntos fueron bastante consistentes y por lo tanto son una buena medida de referencia.

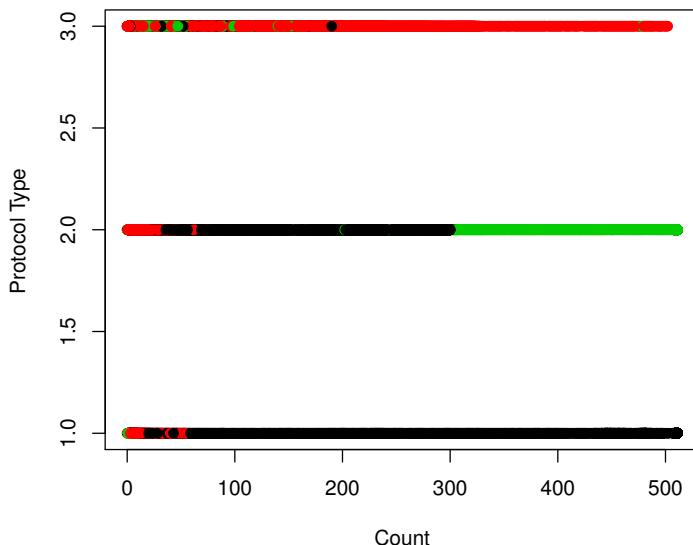


Figura 4.10: Graficación de las dos características principales del conjunto de datos de entrenamiento luego de la aplicación de GFR usando NN (Iteración 2).

Leyenda: Rojo = DoS, Negro = Normal, Verde = *Probing*, Azul = R2L y Magenta = U2R.

- **SVM**

En la Tabla 4.14 se presenta la tabla con las características ordenadas por orden de importancia para SVM. En la misma se puede observar como las características tienen un orden de importancia diferente al de NN (Ver Tabla 4.13). Esto indica que el enfoque para tomar decisiones difiere entre un método y otro. Sin embargo, es importante destacar que siguen teniendo más importancia las características de la conexión de red que por ejemplo el número de terminales abiertas. Adicionalmente, también se puede observar que las características seleccionadas tanto para SVM como para NN a pesar de tener un orden diferente, al hacer la unión de las primeras 10 características de ambos ordenamientos, se observa como muchas características coinciden y el conjunto resultante luego de la operación unión tiene una cardinalidad inferior a 20 que sería el resultado si las características seleccionadas de ambos conjuntos fueran totalmente diferentes.

La Figura 4.12 presenta el gráfico de las dos características seleccionadas como más importantes para SVM que corresponden a Flag y Service. Al igual que con NN, con este gráfico se busca identificar en dos dimensiones posibles patrones o fronteras que ayuden a la detección o identificación de grupos o clases. Esta gráfica se asemeja en cierta forma a la presentada en el estudio de NN en la Figura 4.10. Sin embargo, se muestran mayor cantidad de líneas y además estas están rotadas en forma vertical. Por la interpretación matemática de SVM que traza fronteras dibujando regiones separadoras en planos n-dimensionales, el

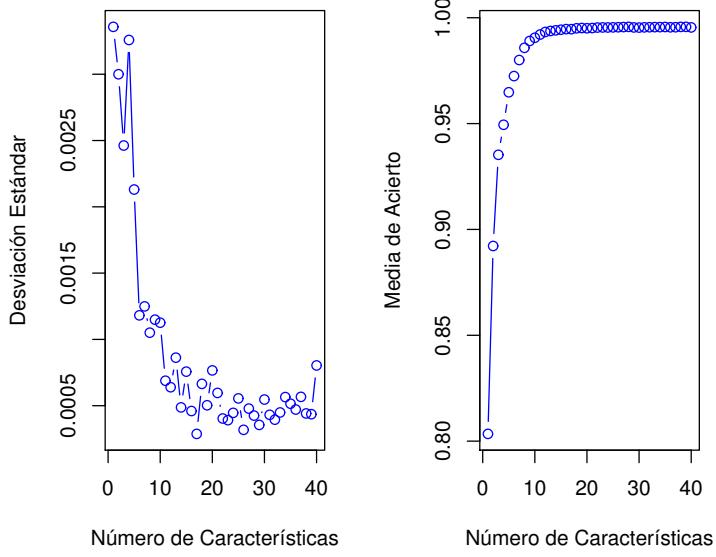


Figura 4.11: Graficación de la desviación estándar y media de acierto por característica seleccionada para NN usando GFR (Iteración 2).

1	Flag	11	Wrong_fragment	21	Dst_bytes	31	Num_compromised
2	Service	12	Dst_host_serror_rate	22	Land	32	Serror_rate
3	Dst_host_same_src_port_rate	13	Srv_diff_host_rate	23	Dst_host_rerror_rate	33	Is_host_login
4	Dst_host_diff_srv_rate	14	Srv_error_rate	24	Num_file_creations	34	Num_root
5	Protocol_type	15	Same_srv_rate	25	Diff_srv_rate	35	Dst_host_srv_error_rate
6	Hot	16	Dst_host_srv_count	26	Su_attempted	36	Num_shells
7	Count	17	Duration	27	Root_shell	37	Urgent
8	Dst_host_count	18	Logged_in	28	Num_failed_logins	38	Dst_host_srv_error_rate
9	Dst_host_same_srv_rate	19	Is_guest_login	29	Num_access_files	39	Src_bytes
10	Dst_host_srv_diff_host_rate	20	Rerror_rate	30	Srv_error_rate	40	Srv_count

Tabla 4.14: Ordenamiento de características para SVM usando GFR (Iteración 2).

comportamiento presentado en la gráfica se justifica en la búsqueda de circunferencias que puedan agrupar algunas nubes de puntos y asociarlas a una clase en particular, comportamiento que no se ve reflejado en la figura de NN presentada previamente donde se ve que el criterio separador es más analítico que descriptivo.

La Figura 4.13 presenta el gráfico de la desviación estándar y media de acierto por número de características. En la misma se puede observar como con nueve características se alcanza una articulación donde la agregación de mayor cantidad de características aumenta de forma poco significativa la media de aciertos. Adicionalmente, se observa como este resultado es respaldado por una desviación estándar menor a 0.1% que indica que los resultados obtenidos fueron bastante consistentes durante la fase de validación cruzada de 10 conjuntos y por lo tanto son resultados fiables.

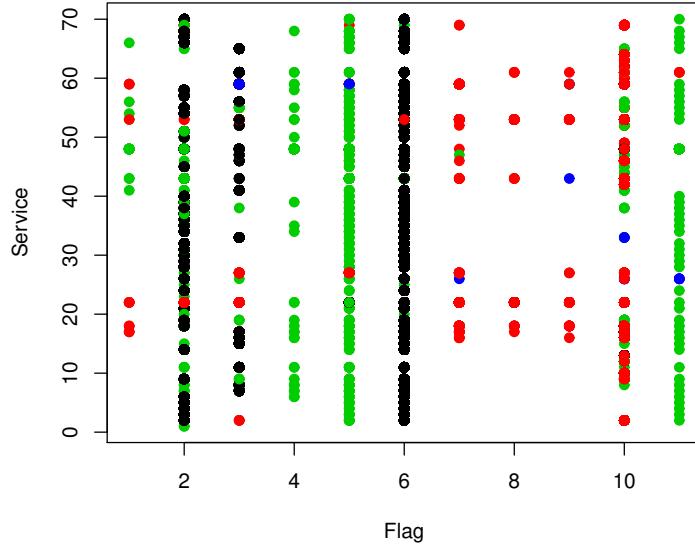


Figura 4.12: Graficación de las dos características principales del conjunto de datos de entrenamiento Luego de la Aplicación de GFR Usando SVM (Iteración 2)
 Leyenda: Negro = Normal, Rojo = DoS, Verde = *Probing*, Azul = R2L y Magenta = U2R.

- **Conclusiones parciales**

Los resultados obtenidos tanto para NN y SVM sugieren la utilización de nueve características. Esto debido a que se alcanza una media de acierto bastante alta con muy poca desviación estándar, reduciendo de esta manera la dimensionalidad del conjunto de datos en un 77.5 %.

Selección de parámetros

Los resultados de la selección de parámetros se presentan en la Tabla 4.15. En ella se pueden observar los parámetros por defecto, el rango de parámetros que fueron probados, los parámetros seleccionados en conjunto con la mejor tasa de acierto encontrada y la dispersión que hubo en los resultados.

Es importante destacar que la validación de la prueba de los parámetros se hizo haciendo uso de la técnica de validación cruzada de 10 conjuntos. Adicionalmente, en el caso de SVM (Radial), como hay dos parámetros ajustables, se realizó el producto cartesiano entre los dos conjuntos de parámetros para obtener todos los pares ordenados posibles y así considerar todas las combinaciones entre los dos conjuntos posibles.

Como aspecto a resaltar durante la selección de parámetros, se puede observar que para NN siempre se seleccionó la mayor cantidad de neuronas posibles; es decir, la selección de parámetros opta por modelos más complejos que permitan realizar operaciones más com-

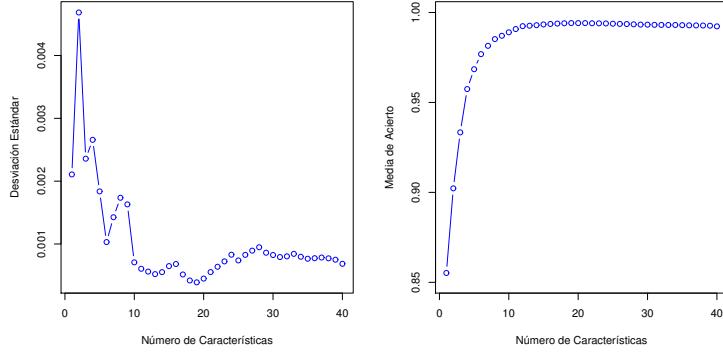


Figura 4.13: Graficación de la desviación estándar y media de acierto por característica seleccionada para SVM usando GFR (Iteración 2).

plejas en la capa intermedia para de esta manera, tener una mejor eficacia a la hora de clasificar los registros. Por otra parte, en SVM se observa que de igual manera la selección de parámetros sugiere modelos con mayor cantidad de costo (Cantidad de error permitida por vector de soporte) y un gamma más grande (Regiones con mayor diámetro), de esta manera se obtienen regiones separadoras más grandes que las por defecto y más flexibles permitiendo mayor cantidad de error.

Modelo	Parámetros Defecto	Rango Valores	Parámetros Seleccionados	Tasa Acierto	Dispersión Resultados
(1) NN - K-Medias	Neuronas = 20	Neuronas = [17,21]	Neuronas = 21	99.56 %	0.035 %
(2) SVM (Radial) - K-Medias	costo = 1 gamma = 0.025	costo = [1,6] gamma = {0.01, 0.025, 0.03, 0.04, 0.05, 0.06, 0.07, 0.08}	costo = 6 gamma = 0.08	99.62 %	0.050 %
(3) PCA - NN - K-Medias	Neuronas = 20	Neuronas = [17, 30]	Neuronas = 30	99.00 %	0.063 %
(4) PCA - SVM (Radial) - K-Medias	costo = 1 gamma = 0.14	costo = [1,6] gamma = {0.06, 0.07, 0.08, 0.14, 0.2, 0.3, 0.4}	costo = 6 gamma = 0.4	99.45 %	0.076 %
(5) GFR - NN - K-Medias	Neuronas = 20	Neuronas = [17, 30]	Neuronas = 30	99.04 %	0.098 %
(6) GFR - SVM (Radial) - K-Medias	costo = 1 gamma = 0.11	costo = [1, 6] gamma = {0.06, 0.07, 0.08, 0.11, 0.2, 0.3, 0.4}	costo = 6 gamma = 0.4	99.27 %	0.067 %

Tabla 4.15: Selección de parámetros para los modelos (Iteración 2).

Selección de grupos para K-Medias

En esta iteración se volvió a hacer la selección de grupos para K-Medias debido a que como los conjuntos de datos fueron reducidos era posible obtener diferentes resultados a la hora de aplicar el Codo de Jambú.

El resultado obtenido luego de la aplicación del codo de Jambu es presentado en la Figura 4.14. En la misma se puede observar como al igual que en la Figura 4.3 correspondiente a la selección de grupos para K-Medias en la Iteración 1, la inercia intra-grupos no se estabiliza en ningún punto por los mismos motivos expuestos en dicha sección (Ver Sección 4.3.1).

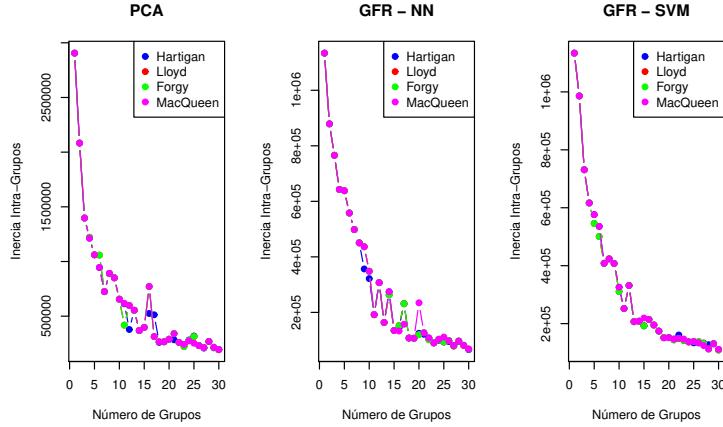


Figura 4.14: Codo de Jambu sobre los conjuntos de datos (Iteración 2).

En la Tabla 4.16 se presentan los resultados obtenidos de la inercia inter-grupos de los diferentes algoritmos de distancia de K-Medias usando dos y cinco grupos para los diferentes conjuntos de datos de esta iteración. En la misma se observa como para dos grupos todos los algoritmos arrojan los mismos resultados para cada uno de los conjuntos. Como en la Iteración 1 se utilizó el algoritmo Hartigan, se seleccionó el mismo para el enfoque con dos grupos. Por otra parte para cinco grupos, el algoritmo que maximiza la distancia inter-grupos difiere dependiendo del conjunto de datos. Para el conjunto de datos PCA (7 componentes) el algoritmo que maximizó la inercia inter-grupos fue Hartigan, mientras que para los otros conjuntos de datos correspondientes a GFR - NN (9 características) y GFR - SVM (Radial) (9 características) los algoritmos Lloyd y Forgy fueron los que maximizaron la inercia inter-grupos. Finalmente, para el conjunto de datos PCA (7 componentes) se utilizó el algoritmo Hartigan y para los conjuntos de datos GFR - NN (9 características) y GFR - SVM (Radial) (9 características) se utilizó el algoritmo Lloyd. El algoritmo Lloyd fue elegido de manera arbitraria y no hay ninguna razón que justifique su uso sobre el algoritmo Forgy, ya que teóricamente ambos deberían presentar el mismo rendimiento.

Una vez seleccionados los algoritmos a ser utilizados por conjunto de datos, se procedió a hacer uso de la técnica de validación cruzada de 10 conjuntos sobre los enfoques de dos grupos y cinco grupos para derivar algunas medidas de rendimiento que permitieran seleccionar el enfoque que mejor se adaptara al escenario. A diferencia del enfoque adoptado en la Sección 4.3.1, en este caso como se presentan múltiples conjuntos de datos, la Sección “Análisis usando dos grupos” fue omitida debido a que en la Sección “Comparación entre los

Conjunto	Algoritmo	Inercia Inter-Grupos (2 Grupos)	Inercia Inter-Grupos (5 Grupos)
PCA (7 Componentes)	Hartigan	752913.5	1761230
	Lloyd	752913.5	1753649
	Forgy	752913.5	1753649
	MacQueen	752913.5	1757997
GFR - NN (9 Características)	Hartigan	199740.4	546907.7
	Lloyd	199740.4	545839.6
	Forgy	199740.4	545839.6
	MacQueen	199740.4	545441.9
GFR - SVM (9 Características)	Hartigan	259975.9	550666
	Lloyd	259975.9	551107.9
	Forgy	259975.9	551107.9
	MacQueen	259975.9	550500.1

Tabla 4.16: Inercia inter-grupos de los conjuntos de datos (Iteración 2).

enfoques de dos y cinco grupos” se presenta una tabla comparativa que muestra las medidas de rendimiento binarias de los enfoques de dos y cinco grupos.

- **Análisis usando cinco grupos**

En la Tabla 4.17 se presenta la tasa de acierto por clase y la tasa de acierto total de los mejores modelos seleccionados por conjunto de datos reducido durante el proceso de la validación cruzada de 10 conjuntos. En la misma se puede observar como los mismos presentan un desempeño bastante aceptable dentro de lo que es la detección de las diferentes clases sobre los diferentes conjuntos de datos. Si se comparan los resultados obtenidos con la tasa de acierto presentada en el análisis de cinco grupos para K-Medias ilustrada en la Tabla 4.5, se puede observar como en los conjuntos de datos reducidos hay un desempeño similar incluso con el uso de una poca cantidad de componentes principales y de características.

El conjunto de datos de PCA (7 características) fue el que presentó mejor desempeño. Esto se debe a que las componentes son creadas mediante combinaciones lineales de las características originales [13]. Es por esto, que estas componentes son capaces de acumular mayor cantidad de información que haciendo uso de un sub-conjunto de características del conjunto de características original.

- **Comparación entre los enfoques de dos y cinco grupos**

Al igual que en la Sección 4.3.1, las matrices de confusión de cinco clases obtenidas luego de la aplicación de K-Medias sobre los diferentes conjuntos de datos fue compri-

Conjunto	DoS	Normal	Probing	R2L	U2R	Total
PCA (7 Componentes)	74.73 %	89.24 %	44.91 %	2.31 %	7.69 %	79.12 %
GFR - NN (9 Características)	76.09 %	84.52 %	45.85 %	58.39 %	0.00 %	77.63 %
GFR - SVM (Radial) (9 Características)	89.62 %	76.63 %	9.39 %	86.93 %	1.92 %	75.19 %

Tabla 4.17: Tasa de acierto por clase de la matriz de confusión de cinco clases en el algoritmo K-Medias (Iteración 2).

mida a dos clases.

En la Tabla 4.18 se presenta una tabla con las medidas de rendimiento binarias extraídas de las matrices de confusión de dos clases de los enfoques de dos y cinco grupos sobre los distintos conjuntos de datos reducidos. En la misma se puede observar como existe una gran variación entre la tasa de acierto de los resultados de la matriz de confusión de cinco clases al ser comprimida en dos clases. Esta situación indica que se cometieron muchos errores clasificación entre las diferentes clases de ataques. Por otra parte, la tasa de acierto con dos clases fue muy superior a la de cinco clases, donde también resalta que la tasa de acierto promedio presentó mucha menor varianza, situación que sugiere que la convergencia hacia dos clases es mucho más natural y certera con dos clases que con cinco.

Conjunto	Modelo	Sensibilidad	Especificidad	Precisión	Acierto Total	Acierto Promedio
PCA (7 Componentes)	5 Grupos	91.57 %	89.24 %	88.11 %	90.32 %	74.70 %
	2 Grupos	80.90 %	98.76 %	98.27 %	90.45 %	80.61 %
GFR - NN (9 Características)	5 Grupos	96.61 %	84.52 %	84.46 %	90.15 %	40.37 %
	2 Grupos	76.98 %	98.65 %	98.03 %	88.57 %	71.84 %
GFR - SVM (Radial) (9 Características)	5 Grupos	97.54 %	76.63 %	77.27 %	86.36 %	30.42 %
	2 Grupos	85.03 %	92.92 %	91.27 %	89.25 %	71.74 %

Tabla 4.18: Comparación de las medidas de rendimiento binarias extraídas de los enfoques de cinco y dos grupos usando K-Medias sobre el conjunto de datos de entrenamiento (Iteración 2).

▪ Conclusiones parciales

Al igual que en la Iteración 1 expuesta en la Sección 4.3.1, el Codo de Jambu no fue de gran ayuda por las mismas razones mencionadas en dicha Iteración. Adicionalmente, de nuevo el enfoque usando dos grupos fue superior al enfoque con cinco grupos, presentando mayor tasa de acierto y menor varianza en los resultados obtenidos durante

el proceso de validación cruzada de 10 conjuntos, comportamiento que refleja que con dos grupos se alcanza la convergencia más rápida que con cinco grupos.

Análisis de modelos sobre el conjunto de entrenamiento

En esta sección se presenta el análisis e interpretación de los resultados obtenidos de los modelos sobre el conjunto de datos de entrenamiento.

En esta iteración se hará uso de los conjuntos de datos reducidos con las características seleccionadas en la Sección 4.3.2 y los parámetros seleccionados en la Sección 4.3.2 para los diferentes enfoques.

La Tabla 4.19 presenta las tasas de acierto por clase y la tasa de acierto total derivada de la matriz de confusión de cinco clases del mejor modelo seleccionado de la fase de validación cruzada de 10 conjuntos de los diferentes modelos sobre el conjunto de datos de entrenamiento. Esta tabla refleja el excelente desempeño general de todos los modelos a la hora de detectar registros conocidos. Como aspecto destacado, se puede observar como la reducción de características no perjudicó de manera notable el desempeño de los modelos del primer nivel, ya que la diferencia de tasa de acierto total de los modelos usando el conjunto total de características con respecto a los modelos que usan características reducidas es de apenas 0.5 % aproximadamente. Este resultado es bastante aceptable debido a que los conjuntos de datos reducidos poseen solo 7 componentes y 9 características para los modelos de PCA y GFR respectivamente. Adicionalmente, se puede observar como para los modelos (1) NN - K-Medias y (2) SVM (Radial) - K-Medias la tasa de acierto incrementó con respecto a los resultados presentados en la Iteración 1 en la Tabla 4.9 debido a la selección de parámetros realizada en esta iteración.

Modelo	DoS	Normal	Probing	R2L	U2R	Total
(1) NN - K-Medias	99.80 %	99.66 %	99.39 %	90.12 %	50.00 %	99.61 %
(2) SVM (Radial) - K-Medias	99.91 %	99.79 %	99.39 %	90.18 %	50.00 %	99.69 %
(3) PCA - NN - K-Medias	99.80 %	99.28 %	97.72 %	87.25 %	0.00 %	99.21 %
(4) PCA - SVM (Radial) - K-Medias	99.87 %	99.66 %	98.61 %	87.65 %	50.00 %	99.54 %
(5) GFR - NN - K-Medias	99.78 %	99.60 %	97.01 %	72.97 %	0.00 %	99.20 %
(6) GFR - SVM - K-Medias	99.70 %	99.31 %	99.48 %	92.59 %	0.00 %	99.38 %

Tabla 4.19: Tasas de acierto (5 Clases) del primer nivel de los modelos sobre el conjunto de datos de entrenamiento (Iteración 2).

En la Figura 4.15 se presentan las curvas ROC de los diferentes modelos sobre el conjunto de entrenamiento. En la misma se puede apreciar como al igual que en la Iteración 1, los modelos concernientes al modelo NN presentan una curva ROC mucho más uniforme y acumulan mayor cantidad de área bajo la curva que los modelos basados en SVM (Radial). Este comportamiento sugiere que los modelos basados en NN son muchos más certeros a la hora

de tomar decisiones. Por otra parte, destaca el hecho de que la reducción de características no decrementó significativamente el desempeño de la curva ROC con respecto a los modelos que usan el conjunto total de las mismas. Adicionalmente, si se comparan los resultados de los modelos (1) NN - K-Medias y (2) SVM (Radial) - K-Medias con los presentados en la Figura 4.4 concerniente a las curvas ROC sobre el conjunto de entrenamiento en la Iteración 1, se aprecia que en esta iteración las curvas ROC presentaron un mejor desempeño, reflejando que la selección de parámetros funcionó positivamente.

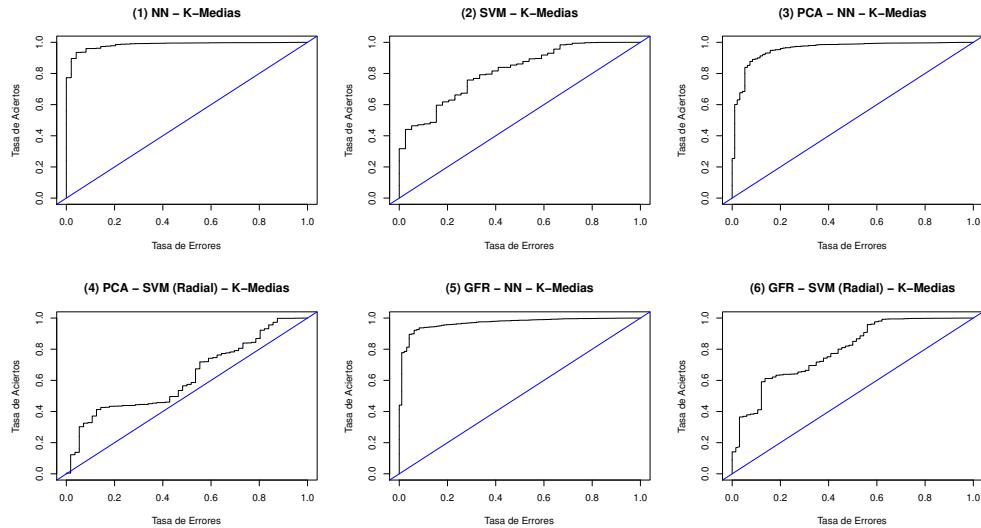


Figura 4.15: Curvas ROC de los algoritmos del primer nivel de los modelos sobre el conjunto de datos de entrenamiento (Iteración 2).

En la Tabla 4.20 se presenta una tabla comparativa de las medidas de rendimiento binarias de los diferentes modelos referentes a la Iteración 2 sobre el conjunto de entrenamiento. En esta se puede observar como para el primer nivel de clasificación correspondiente a los algoritmos de aprendizaje supervisado el desempeño fue bastante grande; sin embargo, el desempeño del segundo nivel de clasificación referente a K-Medias se vió deteriorado en los modelos con conjuntos de características reducidos. Específicamente, es claro como GFR presenta peor desempeño que PCA, debido a que el cambio del espacio de dimensiones en PCA consta en utilizar combinaciones lineales, hay mayor cantidad de información comprimida en las componentes seleccionadas. Como el complemento de K-Medias para los modelos (1) NN - K-Medias y (2) SVM (Radial) - K-Medias fue bueno y se vió deteriorado para los modelos que usaron la reducción de características GFR, es de esperar que con una mayor cantidad de características en dichos modelos K-Medias pueda mejorar en su tarea de complemento del primer nivel de clasificación en dichos modelos.

Modelo	Parámetros	Tipos	Sensibilidad	Especificidad	Precisión	Tasa Acierto
(1) NN - K-Medias	Neuronas = 21	NN (2 Clases)	99.55 %	99.66 %	99.61 %	99.61 %
		NN \Rightarrow K-Medias	0.00 %	99.74 %	0.00 %	99.36 %
		NN + K-Medias	99.55 %	99.41 %	99.32 %	99.47 %
(2) SVM (Radial) - K-Medias	costo = 6 gamma = 0.08	SVM (2 Clases)	99.57 %	99.79 %	99.76 %	99.69 %
		SVM \Rightarrow K-Medias	0.00 %	99.99 %	0.00 %	99.61 %
		SVM + K-Medias	99.57 %	99.78 %	99.74 %	99.68 %
(3) PCA - NN - K-Medias	Neuronas = 30	NN (2 Clases)	99.20 %	99.28 %	99.19 %	99.25 %
		NN \Rightarrow K-Medias	38.30 %	89.81 %	2.59 %	89.45 %
		NN + K-Medias	99.51 %	89.17 %	89.01 %	94.01 %
(4) PCA - SVM (Radial) - K-Medias	costo = 6 gamma = 0.4	SVM (2 Clases)	99.43 %	99.66 %	99.61 %	99.56 %
		SVM \Rightarrow K-Medias	9.09 %	95.76 %	1.04 %	95.33 %
		SVM + K-Medias	99.49 %	95.43 %	94.95 %	97.31 %
(5) GFR - NN - K-Medias	Neuronas = 30	NN (2 Clases)	98.81 %	99.60 %	99.53 %	99.23 %
		NN \Rightarrow K-Medias	30.00 %	52.04 %	0.65 %	51.08 %
		NN + K-Medias	99.17 %	51.83 %	64.44 %	73.99 %
(6) GFR - SVM (Radial) - K-Medias	costo = 6 gamma = 0.4	SVM (2 Clases)	99.67 %	99.31 %	99.20 %	99.48 %
		SVM \Rightarrow K-Medias	31.58 %	52.17 %	0.19 %	52.17 %
		SVM + K-Medias	99.78 %	51.81 %	64.11 %	74.03 %

Tabla 4.20: Medidas de rendimiento binarias de los modelos sobre el conjunto de datos de entrenamiento (Iteración 2).

■ Conclusiones parciales

El ajuste de los parámetros aumentó la tasa de acierto y la certeza de los modelos en el primer nivel. Por otra parte, la reducción de características fue efectiva para el primer nivel, pero no para el segundo nivel, en especial para los modelos basados en GFR. Adicionalmente, se puede observar que K-Medias para los modelos (1) NN - K-Medias y (2) SVM (Radial) - K-Medias no deteriora de gran manera el desempeño, es por ello que quizás haga falta mayor cantidad de características para lo que son los conjuntos de datos reducidos por GFR. Para PCA, recordemos que las componentes son derivadas de combinaciones lineales entre las diferentes características, y es por esto que estas comprimen mayor cantidad de información en las mismas.

Análisis de modelos sobre el conjunto de prueba

En esta sección se presentarán los resultados obtenidos en la evaluación de los diferentes modelos sobre el conjunto de prueba. Esta tomará las mismas consideraciones de implementación presentadas en la Sección 4.3.2 correspondientes al análisis de modelos sobre el conjunto de entrenamiento.

En la Tabla 4.21 se presentan las tasas de acierto por etiqueta y la tasa de acierto total de cada uno de los modelos sobre el conjunto de prueba. En la misma resalta el deterioro sufrido por los modelos del primer nivel referentes a NN y a SVM a la hora de generalizar sobre nuevos tipos de registros en comparación a los resultados obtenidos en la Iteración 1 y presentados en la Tabla 4.11. Esta situación refleja que al hacer ajuste de los parámetros se sobre-ajusta el modelo al conjunto de entrenamiento y de esta manera pierde poder de

generalización ante nuevos tipos de registros. Por otra parte, también destaca la baja tasa de acierto de los modelos basados en PCA, donde NN y SVM tuvieron un pobre desempeño luego de haber mostrado un excelente desempeño sobre el conjunto de entrenamiento. Esto indica que puede faltar mayor cantidad de componentes principales o que simplemente los conjuntos de prueba y de entrenamiento son muy diferentes entre si y por lo tanto las matrices de correlación creadas luego de aplicar PCA también lo son. Por último, los modelos basados en GFR presentaron un desempeño ligeramente inferior al de los modelos (1) NN - K-Medias y (2) SVM (Radial) - K-Medias que utilizan el conjunto de características completo. Específicamente, en este caso los modelos de SVM presentaron un desempeño bastante bueno, mucho mejor que el desempeño de NN y se puede pensar que con una mayor cantidad de características el desempeño podría incrementar positivamente en ambos casos.

Modelo	DoS	Normal	Probing	R2L	U2R	Total
(1) NN - K-Medias	51.18 %	92.73 %	65.84 %	1.20 %	4.00 %	64.13 %
(2) SVM (Radial) - K-Medias	81.35 %	98.29 %	52.58 %	0.25 %	0.00 %	74.93 %
(3) PCA - NN - K-Medias	6.72 %	48.26 %	20.69 %	24.36 %	0.00 %	28.21 %
(4) PCA - SVM (Radial) - K-Medias	3.24 %	28.48 %	0.00 %	0.00 %	0.00 %	13.34 %
(5) GFR - NN - K-Medias	34.20 %	95.41 %	63.40 %	1.42 %	0.50 %	59.40 %
(6) GFR - SVM (Radial) - K-Medias	75.02 %	97.79 %	58.53 %	8.64 %	1.50 %	74.29 %

Tabla 4.21: Tasas de acierto (5 Clases) del primer nivel de los modelos sobre el conjunto de datos de prueba (Iteración 2).

En la Figura 4.16 se presentan las curvas ROC de los diferentes modelos sobre el conjunto de prueba. En la misma se puede observar como en comparación a las presentadas en el análisis sobre el conjunto de entrenamiento correspondiente a la Figura 4.15, el desempeño general de las mismas decreció de manera notable, incluso algunas están por debajo de la línea que delimita el azar. Adicionalmente, las curvas ROC de los modelos (1) NN - K-Medias y (2) SVM (Radial) - K-Medias en comparación con las presentadas en la Iteración 1 en la Figura 4.5 respectivamente son mucho menos certeras, situación que indica que el ajuste de los parámetros sobre-ajusta el modelo al conjunto de entrenamiento y de esta manera el mismo pierde generalidad para clasificar los nuevos registros pertenecientes al conjunto de prueba. Por otra parte, el resto de los modelos muestra un pobre desempeño salvo el modelo (5) GFR - NN - K-Medias que muestra un desempeño incluso mayor que los alcanzados en la Iteración 1 por los modelos (1) NN - K-Medias y (2) SVM (Radial) - K-Medias presentados en la Figura 4.5, esto sugiere que la reducción de características quitó algo de ruido y lo hizo tomar mejores decisiones; sin embargo, los resultados obtenidos en la tasa de aciertos presentados en la Tabla 4.21 sugieren una mayor cantidad de características a ser elegidas, por lo tanto, con un mayor número de características se puede lograr aumentar la precisión manteniendo un buen desempeño de la curva ROC que refleje tomas de decisiones con alta certeza por parte del modelo a la hora de clasificar el tráfico de red.

En la Tabla 4.22 se presenta una tabla comparativa de las medidas de rendimiento binarias de los diferentes modelos referentes a la Iteración 2 sobre el conjunto de prueba. En esta se

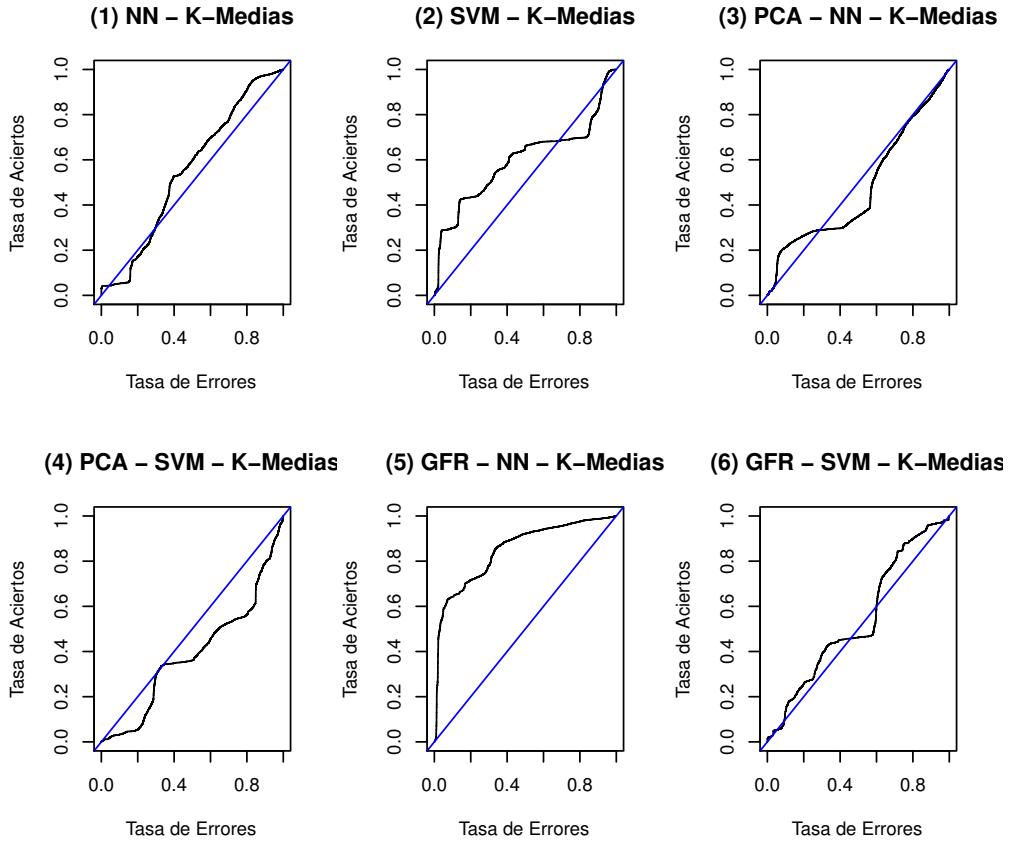


Figura 4.16: Curvas ROC de los algoritmos del primer nivel de los modelos sobre el conjunto de datos de prueba (Iteración 2).

puede observar como el modelo (1) NN - K-Medias es el que posee mayor tasa de acierto con un 85.07 %. Este resultado es engañoso debido a que en la Tabla 4.21 se reflejó una tasa de acierto de 64.13 % y en esta tabla al pasar la matriz de confusión de cinco clases a dos clases se presenta una tasa de acierto de 76.20 %. Se presenta una variación de resultados de alrededor del 12 %, situación que indica que hubo muchos errores en la clasificación entre ataques y una varianza tan grande entre los resultados no es deseable. La misma situación es presentada para el modelo (5) GFR - NN - K-Medias. Por otra parte, los resultados de los modelos (2) SVM (Radial) - K-Medias y (4) PCA - SVM (Radial) - K-Medias, indican que el complemento entre ellos es poco efectivo si se compara con los resultados a la hora de usar NN en conjunto con K-Medias. Esta situación lleva a pensar cada vez más que el uso de dos técnicas que se basan en el mismo principio de separación es poco efectivo para un modelo híbrido dentro de el escenario actual de la presente investigación. Se refleja el pobre desempeño tenido de PCA que sugiere falta de información o que esta técnica no es adecuada para el escenario debido a la diferencia entre los conjuntos de datos. Finalmente, un aspecto a destacar es la reducción de los tiempos de entrenamiento y de prueba para los

modelos cuyas características fueron reducidas. Para todos estos, los tiempos fueron menores que para los modelos que usaron el conjunto de características en su totalidad, cumpliendo así el objetivo de hacer modelos más rápidos. Especialmente, se puede observar una diferencia notable con respecto a los modelos basados en SVM en el primer nivel, donde los tiempos de entrenamiento y de prueba se vieron reducidos a casi la mitad en comparación con los tiempos obtenidos en el modelo haciendo uso del conjunto total de características.

Modelo	Parámetros	Tipo	Sensibilidad	Especificidad	Precisión	Tasa Acierto	Tiempo Entrenamiento	Tiempo Prueba
(1) NN - K-Medias	Neuronas = 21	NN (2 Clases)	63.70 %	92.73 %	92.05 %	76.20 %	275.05 segs	0.22 segs
		NN \Rightarrow K-Medias	66.71 %	87.68 %	73.70 %	80.53 %	7.78 segs	0.06 segs
		NN + K-Medias	87.91 %	81.31 %	86.14 %	85.07 %	282.83 segs	0.28 segs
(2) SVM (Radial) - K-Medias	costo = 6 gamma = 0.08	SVM (2 Clases)	59.77 %	98.29 %	97.88 %	76.36 %	1593.09 segs	10.64 segs
		SVM \Rightarrow K-Medias	0.95 %	99.22 %	39.84 %	64.73 %	8.32 segs	0.07 segs
		SVM + K-Medias	60.15 %	97.53 %	96.98 %	76.25 %	1601.41 segs	10.71 segs
(3) PCA - NN - K-Medias	Neuronas = 30	NN (2 Clases)	31.02 %	48.26 %	44.21 %	38.45 %	210.13 segs	0.14 segs
		NN \Rightarrow K-Medias	42.58 %	99.15 %	98.95 %	62.16 %	1.70 segs	0.01 segs
		NN + K-Medias	60.39 %	47.85 %	60.48 %	54.99 %	211.83 segs	0.15 segs
(4) PCA - SVM (Radial) - K-Medias	costo = 6 gamma = 0.4	SVM (2 Clases)	12.21 %	28.48 %	18.41 %	19.22 %	816.24 segs	6.08 segs
		SVM \Rightarrow K-Medias	19.42 %	99.82 %	99.77 %	35.27 %	1.82 segs	0.02 segs
		SVM + K-Medias	29.26 %	28.43 %	35.08 %	28.90 %	818.06 segs	6.10 segs
(5) GFR - NN - K-Medias	Neuronas = 30	NN (2 Clases)	64.37 %	95.41 %	94.56 %	75.46 %	216.53 segs	0.13 segs
		NN \Rightarrow K-Medias	24.03 %	53.44 %	22.07 %	43.01 %	2.48 segs	0.02 segs
		NN + K-Medias	69.89 %	50.98 %	65.33 %	61.75 %	219.02 segs	0.15 segs
(6) GFR - SVM (Radial) - K-Medias	costo = 6 gamma = 0.4	SVM (2 Clases)	59.74 %	97.79 %	97.27 %	76.13 %	796.57 segs	6.43 segs
		SVM \Rightarrow K-Medias	17.83 %	53.12 %	17.14 %	40.68 %	2.43 segs	0.02 segs
		SVM + K-Medias	66.92 %	51.94 %	64.79 %	60.47 %	799.00 segs	6.45 segs

Tabla 4.22: Medidas de rendimiento binarias de los modelos sobre el conjunto de datos de prueba (Iteración 2).

■ Conclusiones parciales

El ajuste de parámetros perjudicó negativamente a la generalización de los modelos (1) NN - K-Medias y (2) SVM (Radial) - K-Medias, que disminuyeron las tasas de aciertos y también la certeza en las decisiones tomadas. Sumado a eso, la reducción de características no fue buena para PCA, situación que sugiere que quizás este método no es efectivo para el escenario estudiado en la investigación.

Como aspecto positivo, se tiene que el modelo (5) GFR - NN - K-Medias presentó buenos resultados con la reducción de características al presentar una buena curva ROC; sin embargo, se amerita mayor cantidad de componentes principales para reducir la varianza entre los resultados entre cinco y dos clases, y también aumentar el aporte de K-Medias como complemento de NN. Por último, la reducción de características redujo sustancialmente los tiempos de entrenamiento y de prueba, especialmente para los modelos basados en SVM.

Finalmente, al igual que en la Iteración 1, se pudo observar como el complemento entre los algoritmos SVM (Radial) y K-Medias no se complementan de buena manera debido a que ambos presentan el mismo enfoque de clasificación.

Conclusiones de la Iteración 2

El ajuste de los parámetros sobre-ajusta los modelos al conjunto de entrenamiento, restándole capacidad de generalización frente a los nuevos registros presentes en el conjunto de prueba. Por lo anterior, se sugiere utilizar parámetros por defecto.

Adicionalmente, los modelos basados en PCA y GFR ameritan mayor cantidad de componentes y características que aporten más información para obtener modelos más precisos. En el caso de PCA se recomienda utilizar 24 componentes principales, siguiendo las recomendaciones tratadas en la Sección 2.2.1 referentes a la sección de PCA. Por otra parte, para los modelos basados en GFR, se recomienda utilizar 19 características tomando como referencia el trabajo publicado por Li, Xia y colaboradores [21], donde presentan que con 19 características obtuvieron el número óptimo de características seleccionadas en una investigación similar a la presentada en este documento.

Al igual que en la Iteración 1, se notó el hecho de que el complemento entre los algoritmos SVM (Radial) y K-Medias no es del todo bueno debido a que ambos algoritmos utilizan el mismo enfoque de separación basado en la búsqueda de circunferencias en espacios n-dimensionales.

4.3.3. Iteración 3

En la Iteración 2 (Ver Sección 4.3.2) se presentaron los resultados de hacer selección de parámetros y de características sobre los diferentes modelos concernientes a dicha iteración. Los resultados obtenidos sugirieron el uso de los parámetros por defecto usados en la Iteración 1 debido a que la selección de parámetros causa sobre-ajuste de los modelos sobre el conjunto de entrenamiento haciendo que los mismos pierdan poder de generalización frente a nuevos tipos de registros presentes en el conjunto de datos de prueba. Adicionalmente, se observó como la reducción de características en dicha iteración no fue muy efectiva y es por eso que las conclusiones de la misma sugieren el uso de mayor cantidad de componentes y de características en la presente iteración con la finalidad de que los modelos cuenten con mayor cantidad de información que los ayude a mejorar el desempeño a la hora de clasificar el tráfico de red.

En esta iteración se entrenaron cuatro modelos referentes a:

3. PCA - NN - K-Medias.
4. PCA - SVM (Radial) - K-Medias.
5. GFR - NN - K-Medias.
6. GFR - SVM (Radial) - K-Medias.

Los modelos (1) NN - K-Medias y (2) SVM (Radial) - K-Medias fueron omitidos en esta iteración debido a que todos los estudios pertinentes a dichos modelos fueron realizados previamente y ahora el estudio fue centralizado en los modelos con características reducidas. Los modelos (3) PCA - NN - K-Medias y (4) SVM (Radial) K-Medias contaron en esta iteración con 24 componentes principales, debido a que es el número de características que hacen que la varianza acumulada en la matriz de covarianza alcance el 95 % de la misma. Por otra parte, los modelos (5) GFR - NN - K-Medias y (6) GFR - SVM (Radial) - K-Medias contarán con 19 características siguiendo las recomendaciones de un trabajo publicado por Li, Xia y colaboradores [21] que indica que con ese número de características se encontró el número óptimo de características en un trabajo similar al presentado en el presente documento. La estructura en esta iteración es igual a las iteraciones previas.

Selección del número de grupos para K-Medias

Al igual que en las Iteraciones 1 y 2, se utilizó el codo de Jambu para tratar de seleccionar el número óptimo de grupos a ser pasados al algoritmo de K-Medias. En esta ocasión fueron agregadas más componentes principales para el conjunto de datos basado en PCA y mayor cantidad de características para los conjuntos de datos basados en GFR, por lo tanto podría haber diferentes resultados a los obtenidos en las iteraciones previas donde se reflejó que la técnica de Codo de Jambu no fue efectiva y que el uso de dos grupos predominó sobre el de cinco grupos.

En la Figura 4.17 se presenta el resultado de la aplicación del Codo de Jambu sobre los diferentes conjuntos de datos. En dicha figura se presentan los mismos resultados obtenidos en las iteraciones previas. El Codo de Jambu es errático y se hace imposible encontrar un punto en el que la inercia intra-grupos se estabilice.

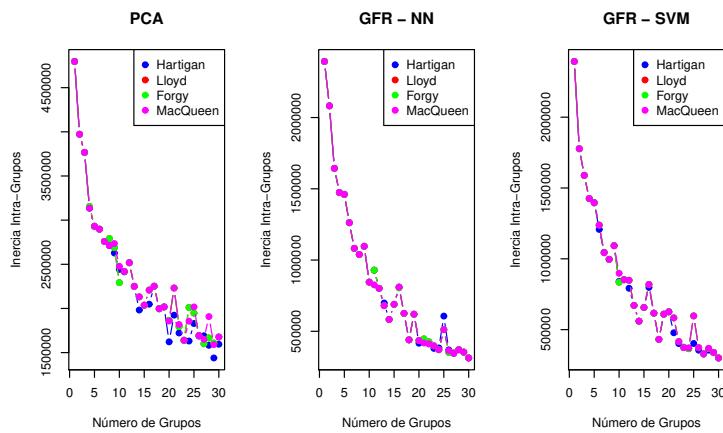


Figura 4.17: Codo de Jambu sobre los conjuntos de datos (Iteración 3).

La Tabla 4.23 presenta los resultados de la distancia inter-grupos para cada conjunto de datos utilizando dos y cinco grupos. En la misma se puede apreciar como para el conjunto de datos PCA (24 Componentes) usando dos grupos todos los algoritmos presentan el mismo desempeño; sin embargo, usando cinco grupos, el algoritmo Hartigan es el que maximiza la inercia inter-grupos y es por esto que para este conjunto de datos se utilizaó dicho algoritmo para ambos enfoques. Para el conjunto de Datos GFR - NN (19 Características) usando dos grupos todos los algoritmos presentan el mismo desempeño y usando cinco grupos Lloyd y Forgy son los que maximizan la inercia inter-grupos, por lo tanto los algoritmos seleccionados para este conjunto de datos fueron Hartigan y Lloyd para dos y cinco grupos respectivamente. Por último, para el conjunto de datos GFR - SVM (19 Características) el algoritmo MacQueen y Hartigan presentan el mismo desempeño para dos grupos, mientras que los algoritmos Lloyd y Forgy presentan el mismo desempeño para cinco grupos, por lo tanto se usaron los algoritmos Hartigan y Lloyd para dos y cinco grupos respectivamente.

Conjunto	Algoritmo	Inercia Intra-Grupos (2 Grupos)	Inercia Intra-Grupos (5 Grupos)
PCA (24 Componentes)	Hartigan	677575.4	1735919
	Lloyd	677573	1726413
	Forgy	677573	1726413
	MacQueen	677573	1731375
GFR - NN (19 Características)	Hartigan	439293.1	1030473
	Lloyd	439293.1	1032120
	Forgy	439293.1	1032120
	MacQueen	439293.1	1028519
GFR - SVM (19 Características)	Hartigan	515383.8	1038039
	Lloyd	506995.6	1041416
	Forgy	506995.6	1041416
	MacQueen	515383.8	1039260

Tabla 4.23: Inercia inter-grupos de los conjuntos de datos (Iteración 3).

Una vez seleccionados los algoritmos a ser utilizados por conjunto de datos, se procedió a hacer uso de la técnica de validación cruzada de 10 conjuntos sobre los enfoques de dos grupos y cinco grupos para derivar algunas medidas de rendimiento que permitieran seleccionar el enfoque que mejor se adaptara al escenario. Al igual que en la Iteración 2, la Sección correspondiente a “Análisis usando dos grupos” fue omitida (Ver Sección 4.3.2).

■ Análisis usando cinco grupos

En la Tabla 4.24 se presenta la tasa de acierto por clase y la tasa de acierto total del mejor modelo seleccionado durante la fase de validación cruzada de 10 conjuntos. En la misma se puede ver un desempeño bastante aceptable para la clasificación de las

diferentes clases. Esta selección mejora a la presentada en la Iteración 2 ilustrada en la Tabla 4.17.

Modelo	DoS	Normal	Probing	R2L	U2R	Total
PCA (24 Componentes)	74.76 %	95.10 %	16.76 %	0.00 %	13.46 %	79.65 %
GFR - NN (19 Características)	77.01 %	87.00 %	45.50 %	47.64 %	3.85 %	79.18 %
GFR - SVM (19 Características)	76.10 %	89.98 %	31.37 %	0.40 %	11.54 %	78.76 %

Tabla 4.24: Tasa de acierto por clase de la matriz de confusión de cinco clases en el algoritmo K-Medias (Iteración 3).

■ Análisis para dos y cinco grupos

En la Tabla 4.25 se presenta una tabla con las medidas de rendimiento binarias de los diferentes enfoques de dos y cinco grupos sobre los distintos conjuntos de datos reducidos. En la misma se puede observar como existe una gran varianza con respecto a los resultados obtenidos en la matriz de confusión de cinco clases con los resultados obtenidos al comprimir dicha matriz en dos clases. Esto indica que hubo muchos fallos a la hora de clasificar entre ataques. Adicionalmente, se puede observar como los resultados con dos grupos presentan menor varianza durante la fase de validación cruzada de 10 conjuntos y que adicionalmente los resultados de dos grupos son muy similares a la comprensión de la matriz de confusión de cinco clases en dos. Por lo expuesto previamente de nuevo se decidió que el número óptimo de grupos es dos.

Conjunto	Modelo	Sensibilidad	Especificidad	Precisión	Acierto Total	Acierto Promedio
PCA (24 Componentes)	5 Grupos	88.21 %	95.10 %	93.45 %	88.17 %	71.64 %
	2 Grupos	80.75 %	98.91 %	98.47 %	90.46 %	78.54 %
GFR - NN (19 Características)	5 Grupos	96.77 %	87.01 %	86.64 %	91.55 %	52.27 %
	2 Grupos	85.17 %	93.48 %	91.92 %	89.61 %	78.32 %
GFR - SVM (19 Características)	5 Grupos	92.21 %	89.98 %	91.90 %	91.02 %	54.36 %
	2 Grupos	83.46 %	94.37 %	92.80 %	89.29 %	83.92 %

Tabla 4.25: Comparación de las medidas de rendimiento binarias extraídas de los enfoques de cinco y dos grupos usando K-Medias sobre el conjunto de datos de entrenamiento (Iteración 3).

■ Conclusiones parciales

Con dos clases se puede obtener una convergencia más precisa y con menor cantidad de varianza. Por el mismo motivo se utilizará al igual que las Iteraciones 1 y 2, dos grupos

para hacer el estudio en el segundo nivel de los diferentes modelos que corresponden a K-Medias.

Análisis de modelos sobre el conjunto de entrenamiento

En esta sección se presenta el análisis e interpretación de los resultados obtenidos de los modelos sobre el conjunto de datos de entrenamiento.

En la Tabla 4.26 se presentan las tasas de acierto por clase y la tasa de acierto total de los diferentes modelos sobre el conjunto de entrenamiento. Estos resultados fueron extraídos del mejor modelo obtenido durante la fase de validación cruzada de 10 conjuntos usando la matriz de confusión de cinco clases. En la misma se refleja un excelente desempeño de todos los modelos, siendo el modelo (5) GFR - NN - K-Medias el que presenta mejor desempeño total y por clases. Más allá de eso no hay ningún resultado relevante que no haya sido presentado en iteraciones previas.

Modelo	DoS	Normal	Probing	R2L	U2R	Total
(3) PCA - NN - K-Medias	99.91 %	99.61 %	98.84 %	81.08 %	37.50 %	99.50 %
(4) PCA - SVM (Radial) - K-Medias	99.80 %	99.34 %	98.26 %	87.84 %	25.00 %	99.29 %
(5) GFR - NN - K-Medias	99.85 %	99.66 %	99.96 %	90.18 %	50.00 %	99.56 %
(6) GFR - SVM (Radial) - K-Medias	99.87 %	99.64 %	98.42 %	89.19 %	25.00 %	99.50 %

Tabla 4.26: Tasas de acierto (5 Clases) del primer nivel de los modelos sobre el conjunto de datos de entrenamiento (Iteración 3).

En la Figura 4.18 se presentan las curvas ROC de los diferentes modelos sobre el conjunto de datos de entrenamiento. En la misma se puede observar un excelente desempeño de los modelos basados en NN, un desempeño aceptable pero bastante errático del modelo (6) GFR - SVM (Radial) - K-Medias y un pobre desempeño del modelo (2) PCA - SVM - K-Medias. Al final, los modelos basados en NN son más precisos y certeros a la hora de clasificar que los modelos basados en SVM. Si se comparan las curvas ROC presentadas en la Figura 4.18 con sus homólogos en la Iteración 2 expuestas en la Figura 4.15, se puede observar que se obtiene un comportamiento bastante similar donde quizás se pueda apreciar con mayor claridad una mejora en esta tercera Iteración en el modelo (4) PCA - SVM (Radial) - K-Medias en la curva ROC. Por otra parte, los comportamientos de los otros modelos son bastante parecidos.

En la Tabla 4.27 se comparan las medidas binarias de los diferentes modelos implementados en esta Iteración 3. En la misma se observa que los algoritmos de NN y SVM (Radial) tuvieron un desempeño excelente en el primer nivel. Sin embargo, el segundo nivel de K-Medias al igual que en la Iteración 2 (Ver Tabla 4.20) vuelve a deteriorar la tasa de acierto del primer nivel correspondiente al enfoque supervisado por la alta generación de falsos positivos que se ve reflejada en la medida de precisión. Esta situación no es del todo preocupante debido a que como se mencionó previamente, la generación de falsos positivos permite retroalimentar el modelo. Por otra parte, se puede observar como K-Medias afecta menos a los modelos basados en SVM detectando menor cantidad de ataques, pero clasificando mayor

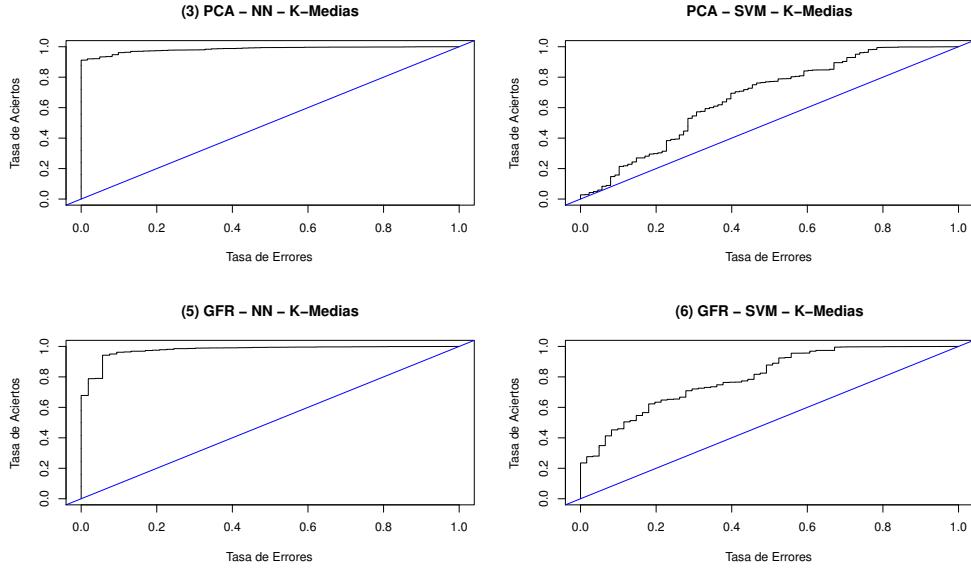


Figura 4.18: Curvas ROC de los algoritmos del primer nivel de los modelos sobre el conjunto de datos de entrenamiento (Iteración 3).

cantidad registros pertenecientes a la clase Normal correctamente. Esta situación hace que se reafirme la conducta de que el uso de SVM (Radial) y K-Medias no se complementan de buena manera al tener el mismo enfoque de clasificación basados en circunferencias en planos n-dimensionales.

Modelo	Parámetros	Tipo	Sensibilidad	Especificidad	Precisión	Tasa Acierto
(3) PCA - NN - K-Medias	Neuronas = 20	NN (2 Clases)	99.41 %	99.61 %	99.56 %	99.52 %
		NN \Rightarrow K-Medias	48.57 %	55.85 %	0.57 %	55.65 %
		NN + K-Medias	99.69 %	55.64 %	66.43 %	76.26 %
(4) PCA - SVM (Radial) - K-Medias	costo = 1 gamma = 0.041	SVM (2 Clases)	99.25 %	99.34 %	99.25 %	99.30 %
		SVM \Rightarrow K-Medias	22.73 %	92.92 %	2.08 %	92.46 %
		SVM + K-Medias	99.42 %	92.31 %	91.93 %	95.64 %
(5) GFR - NN - K-Medias	Neuronas = 20	NN (2 Clases)	99.49 %	99.66 %	99.61 %	99.58 %
		NN \Rightarrow K-Medias	30.00 %	72.92 %	0.49 %	72.73 %
		NN + K-Medias	99.64 %	72.67 %	76.11 %	85.25 %
(6) GFR - SVM (Radial) - K-Medias	costo = 1 gamma = 0.053	SVM (2 Clases)	99.37 %	99.64 %	99.59 %	99.53 %
		SVM \Rightarrow K-Medias	5.41 %	89.75 %	0.29 %	89.29 %
		SVM + K-Medias	99.41 %	89.43 %	89.23 %	94.10 %

Tabla 4.27: Medidas de rendimiento binarias de los modelos con parámetros seleccionados sobre el conjunto de datos de entrenamiento (Iteración 3).

■ Conclusiones parciales

Al aumentar el número de componentes y de características dejando los parámetros por defecto, los resultados sobre el conjunto de entrenamiento se asemejaron mucho a

los obtenidos en la Iteración 2. No se encontró ningún resultado destacado además de que de nuevo SVM y K-Medias no parecen ser un buen complemento.

Análisis sobre el conjunto de prueba

En esta sección se presenta el análisis e interpretación de los resultados obtenidos de los modelos sobre el conjunto de datos de prueba.

En la Tabla 4.28 se presentan las tasas de acierto por clase y la tasa de acierto total de los diferentes modelos sobre el conjunto de prueba. En la misma se puede observar como al igual que en la Iteración 2, los modelos basados en PCA presentaron un pobre desempeño incluso añadiendo el número de componentes principales que hacían que se acumulara el 95 % de varianza. De esta manera, se puede ya dar por descartada la aplicabilidad de PCA sobre el escenario ya que claramente las matrices de covarianza del conjunto de entrenamiento y de prueba son muy diferentes y la reducción de características usando PCA no es efectiva por tal motivo.

Por otro lado, los modelos basados en GFR presentan un muy buen resultado. De hecho, el modelo (5) GFR - NN - K-Medias presentó un incremento bastante notable de alrededor 15 % con respecto a su homólogo en la Iteración 2 presentado en la Tabla 4.21. El modelo (6) GFR - SVM (Radial) presenta más o menos el mismo desempeño que el obtenido en la Iteración 2 y que fue presentado en la tabla referenciada previamente.

Modelo	DoS	Normal	Probing	R2L	U2R	Total
(3) PCA - NN - K-Medias	2.40 %	72.77 %	7.97 %	0.04 %	0.00 %	33.00 %
(4) PCA - SVM (Radial) - K-Medias	0.50 %	59.06 %	9.42 %	0.00 %	0.00 %	26.61 %
(5) GFR - NN - K-Medias	78.59 %	96.90 %	60.18 %	12.45 %	3.00 %	75.75 %
(6) GFR - SVM (Radial) - K-Medias	81.09 %	97.92 %	54.19 %	1.27 %	1.00 %	74.99 %

Tabla 4.28: Tasas de acierto (5 Clases) del primer nivel de los modelos sobre el conjunto de datos de prueba (Iteración 3).

En la Figura 4.19 se presentan las curvas ROC de los diferentes modelos sobre el conjunto de prueba. En la misma se puede observar como al igual que en la Figura 4.16 correspondiente a la Iteración 2, los modelos basados en PCA poseen un pobre desempeño. Por otra parte, el modelo (5) GFR - NN - K-Medias presenta una curva ROC bastante buena; sin embargo, en la Iteración 2 se logró una mejor curva ROC. Finalmente, la curva ROC de SVM no muestra ninguna novedad, sigue siendo errática sobre el conjunto de prueba al igual que en las iteraciones previas.

En la Tabla 4.29 presenta una tabla comparativa de las medidas de rendimiento binarias de los diferentes modelos referentes a la Iteración 3 sobre el conjunto de prueba. En la misma se puede observar como el modelo (6) GFR - SVM (Radial) - K-Medias es el modelo que presenta mayor tasa de acierto total, este modelo combina un buen resultado con poca varianza entre las matrices de confusión de cinco y dos grupos y un buen complemento de

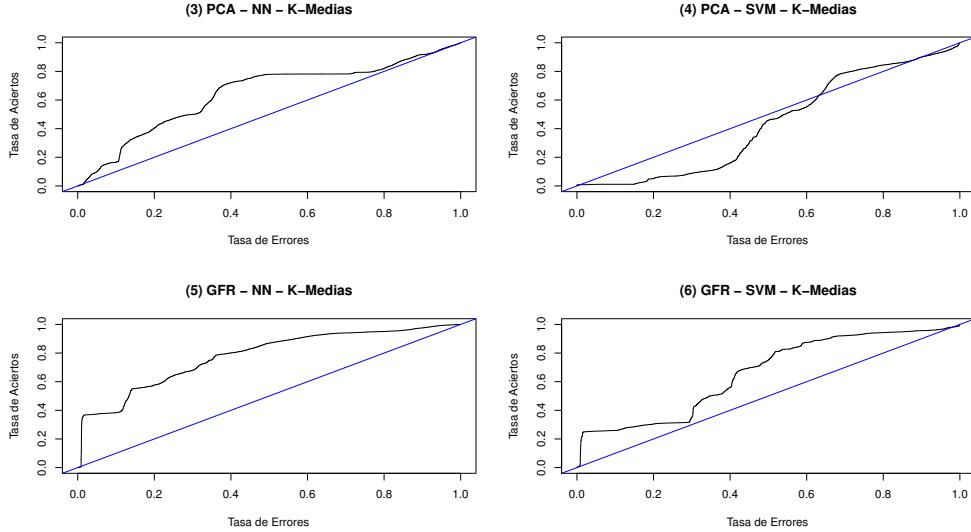


Figura 4.19: Curvas ROC de los algoritmos del primer nivel de los modelos sobre el conjunto de datos de prueba (Iteración 3).

K-Medias que se justifica en una buena selección de características que redujo el ruido para la intervención de K-Medias. Por otra parte, el modelo (5) GFR - NN - K-Medias presenta una varianza bastante significante con el paso de la matriz de confusión de cinco clases a dos clases, adicionalmente el complemento de K-Medias fue poco efectivo también justificado por las características seleccionadas que no están optimizadas para la separación en un plano n-dimensional sino mediante el establecimiento de reglas analíticas para la clasificación de los registros. Los algoritmos basados en PCA poseen una varianza muy alta con respecto a la tasa de aciertos obtenida en la matriz de confusión de cinco clases. Como aspecto relevante, se puede destacar la efectividad de K-Medias como complemento, que también está asociada al pobre desempeño presentado por el primer nivel para la clasificación de los registros. Por último, los tiempos de prueba y de entrenamiento siguen siendo menores que los presentados en la Iteración 1 donde se usó el conjunto total de características para el entrenamiento de los modelos, pero estos son un poco más lentos que los presentados en la Iteración 2 debido a que en esta Iteración se usaron mayor cantidad de características.

■ Conclusiones parciales

Los modelos basados en PCA presentaron de nuevo un pobre desempeño y con esto se reafirma que la reducción de dimensionalidad usando PCA no es productiva para el escenario debido a que las matrices de covarianza de los conjuntos de datos de entrenamiento y de prueba son muy diferentes entre si. Por otra parte, al agregar mayor cantidad de características en los modelos basados en GFR se obtuvo un resultado mucho más preciso para el modelo (5) GFR - NN - K-Medias; sin embargo, K-Medias no fue un buen complemento para este modelo debido a que las características seleccionadas

Modelo	Parámetros	Tipo	Sensibilidad	Especificidad	Precisión	Tasa Acierto	Tiempo Entrenamiento	Tiempo Prueba
(3) PCA - NN - K-Medias	Neuronas = 20	NN (2 clases)	37.51 %	72.77 %	64.55 %	52.70 %	214.65 segs	0.18 segs
		NN \Rightarrow K-Medias	60.57 %	98.40 %	97.73 %	78.29 %	5.21 segs	0.05 segs
		NN + K-Medias	75.36 %	71.61 %	77.82 %	73.74 %	219.86 segs	0.23 segs
(4) PCA - SVM (Radial) - K-Medias	costo = 1 gamma = 0.041	SVM (2 clases)	28.22 %	59.06 %	47.67 %	41.51 %	1009.38 segs	13.45 segs
		SVM \Rightarrow K-Medias	46.13 %	99.29 %	99.04 %	66.53 %	5.02 segs	0.05 segs
		SVM + K-Medias	61.33 %	58.63 %	66.21 %	60.17 %	1014.40 segs	13.50 segs
(5) GFR - NN - K-Medias	Neuronas = 20	NN (2 clases)	69.86 %	96.90 %	96.75 %	81.51 %	189.88 segs	0.32 segs
		NN \Rightarrow K-Medias	11.17 %	99.55 %	91.14 %	73.81 %	3.79 segs	0.03 segs
		NN + K-Medias	73.23 %	96.47 %	96.49 %	83.24 %	193.67 segs	0.35 segs
(6) GFR - SVM (Radial) - K-Medias	costo = 1 gamma = 0.053	SVM (2 clases)	62.13 %	97.92 %	97.53 %	77.55 %	687.22 segs	9.98 segs
		SVM \Rightarrow K-Medias	53.35 %	91.16 %	75.51 %	78.37 %	3.80 segs	0.03 segs
		SVM + K-Medias	82.33 %	89.26 %	91.01 %	85.32 %	691.02 segs	10.01 segs

Tabla 4.29: Medidas de rendimiento binarias de los modelos sobre el conjunto de datos de prueba (Iteración 3).

no son optimizadas para la búsqueda de circunferencias en el espacios n-dimensionales sino para el establecimiento de reglas que ayuden a la clasificación de las clases. Por otra parte, SVM mantuvo un rendimiento similar a la Iteración 2, pero en esta ocasión K-Medias si funcionó como un buen complemento, por lo mismo, podría ser favorable utilizar dichas características en K-Medias como complemento de NN.

Conclusiones de la Iteración 3

Agregar mayor cantidad de características fue productivo para los modelos basados en GFR ya que NN logró ser más preciso y SVM mantuvo el mismo comportamiento que el presentado en la Iteración 2. En el modelo (6) GFR - SVM (Radial) K-Medias se pudo observar como K-Medias funcionó de buena manera como complemento del primer nivel debido a la reducción de características. Es por esto que dicha selección de características puede funcionar de buena manera para ser usada con NN y así K-Medias lograr ser un mejor complemento del mismo.

Por otra parte, los modelos basados en PCA en esta iteración fueron totalmente descartados debido a los malos resultados obtenidos nuevamente para la generalización con registros provenientes del conjunto de datos de prueba debido a la gran diferencia entre las matrices de covarianza entre los conjuntos de datos y de prueba.

4.3.4. Iteración 4

En la Iteración 3 presentada en la Sección 4.3.3 se pudo observar como el modelo (5) GFR - NN - K-Medias presenta muy buenos resultados para el primer nivel. Sin embargo, en el segundo nivel K-Medias no fue un gran complemento. Por otra parte, en la misma iteración, K-Medias logró buenos resultados para el modelo (6) GFR - SVM (Radial) - K-Medias debido a que las características fueron seleccionadas teniendo como principio fundamental la búsqueda de circunferencias en planos n-dimensionales.

Adicionalmente, en la Iteración 2 presentada en la Sección 4.3.2, el modelo (5) GFR - NN

- K-Medias presentó una mejor curva ROC sobre el conjunto de prueba (Ver Figura 4.16) usando 30 neuronas en la capa intermedia que en la Iteración 3 referenciada previamente donde la curva ROC del análisis sobre el conjunto de prueba presentada en la Figura 4.19 con 20 neuronas en la capa intermedia.

Basándose en los expuesto previamente que se refiere a información obtenida de las iteraciones previas. En esta iteración se creó un único modelo:

5. GFR - NN - K-Medias

Este modelo tuvo en el primer nivel concerniente a NN las mismas 19 características seleccionadas por GFR para NN utilizadas en la Iteración 3, adicionalmente contó con 30 neuronas en la capa intermedia. Adicionalmente, para el segundo nivel de K-Medias se usaron las 19 características seleccionadas por GFR para SVM usadas también en la Iteración 3. De esta manera, se buscó utilizar los mejores factores encontrados en las iteraciones previas para la creación del modelo que presente mejor desempeño a la hora de clasificar el tráfico de red.

En esta sección no se hará el análisis sobre K-Medias para seleccionar el número óptimo de grupos debido a que el mismo ya fue realizado en la Iteración 3 en la Sección 4.3.3 ya que se reutilizaron las características seleccionadas por GFR para SVM de dicha iteración y en la cual se seleccionaron dos grupos como el número óptimo para la aplicación de K-Medias. Adicionalmente el análisis sobre los conjuntos de entrenamiento y de prueba serán muchos más amplios en esta iteración debido a que solo hay un modelo y por consecuente se presentarán las matrices de confusión de cinco y de dos clases que ilustran gráficamente el desempeño del modelo.

Análisis sobre el conjunto de entrenamiento

En esta sección se presenta el análisis e interpretación de los resultados obtenidos sobre el conjunto de entrenamiento en la Iteración 4. Esta iniciará por el análisis del primer nivel concerniente a NN, luego el análisis del segundo nivel correspondiente a K-Medias, y se culminará con el análisis grupal de ambos enfoques.

■ NN

En la Tabla 4.30 se presenta la matriz de confusión de cinco clases del rendimiento del mejor modelo de NN obtenido del proceso de validación cruzada de 10 conjuntos sobre el conjunto de datos de entrenamiento. En la misma se puede observar que el resultado es excelente, acumulando la mayoría de los elementos en la diagonal y presentando muy pocos registros fuera de ella. Un aspecto a resaltar es que logra clasificar una proporción bastante alta de registros pertenecientes a la clase U2R. La tasa de acierto promedio en la fase de validación cruzada fue de 99.57 %.

Real\Predicción	DoS	Normal	Probing	R2L	U2R
DoS	4603	5	1	1	0
Normal	3	6676	14	4	2
Probing	0	9	1197	0	0
R2L	0	5	0	69	0
U2R	0	4	0	1	3

Tabla 4.30: Matriz de confusión (5 Clases) del mejor modelo de NN sobre el conjunto de datos de entrenamiento (Iteración 4).

En la Tabla 4.31 se presenta la tasa de acierto por etiqueta y la tasa de acierto total de la matriz de confusión de cinco clases presentada en la Tabla 4.30. La tasa de acierto de 99.61 % refleja el buen desempeño observado en la matriz de confusión de cinco clases. Este rendimiento está respaldado con tasas de acierto muy altas en las diferentes clases salvo en U2R, clase que cuenta con muy pocos registros en el conjunto de entrenamiento.

Tipo	DoS	Normal	Probing	R2L	U2R	Total
NN	99.85 %	99.66 %	99.25 %	93.24 %	37.50 %	99.61 %

Tabla 4.31: Tasas de acierto (5 Clases) de NN sobre el conjunto de datos de entrenamiento (Iteración 4).

En la Tabla 4.32 se presenta la matriz de confusión de dos clases derivada de la matriz de confusión de cinco clases presentada en la Tabla 4.30. En la misma se observa como se produjeron solo 46 errores de clasificación divididas en 23 falsos positivos y 23 falsos negativos. De esta forma es mucho más notorio el rendimiento del modelo a la hora clasificar registros conocidos presentes en el conjunto de entrenamiento.

Real\Predicción	Ataque	Normal
Ataque	5875	23
Normal	23	6676

Tabla 4.32: Matriz de confusión (2 Clases) del mejor modelo de NN sobre el conjunto de datos de entrenamiento (Iteración 4).

La Figura 4.20 presenta la curva ROC derivada de la matriz de confusión de dos clases presentada en la Tabla 4.32. Esta gráfica presenta el excelente desempeño de NN desde un punto de vista de certeza, indicando que las decisiones tomadas por NN son muy precisas ya que la gran mayoría de los aciertos son realizados con altas tasas de certeza. Este resultado es similar al presentado por los modelos (5) GFR - NN - K-Medias en las Iteraciones 2 y 3 presentadas en la Figuras 4.15 y 4.18 respectivamente.

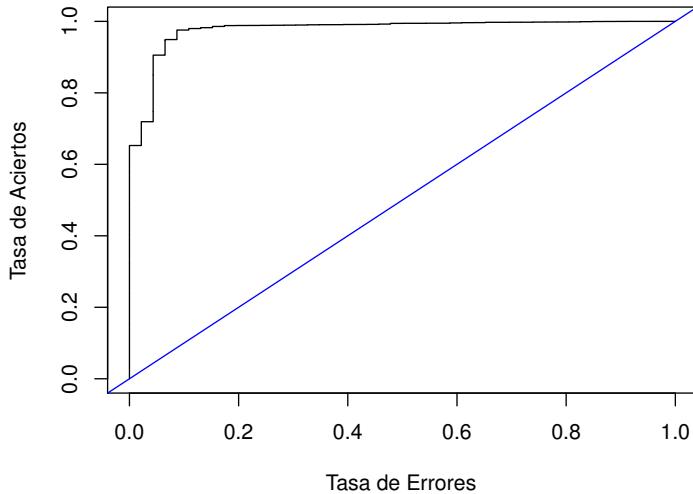


Figura 4.20: Curva ROC de NN sobre el conjunto de datos entrenamiento (Iteración 4).

- **NN ⇒ K-Medias**

En la Tabla 4.33 se presenta la matriz de confusión de dos clases de K-Medias sobre los registros clasificados como Normal por el primer nivel correspondiente a NN. En la misma se puede observar como de los 23 falsos negativos solo se detectaron 5 ataques y se generaron 1385 nuevos falsos positivos. Como se ha venido mencionando a lo largo del documento, la generación de falsos positivos no corresponde un llamado de atención muy grande desde el punto de vista de que estos pueden interpretarse como registros inusuales que ameriten de una revisión y que adicionalmente servirán para la retro-alimentación del modelo.

Real \ Predicción	Ataque	Normal
Ataque	5	18
Normal	1385	5291

Tabla 4.33: Matriz de confusión (2 Clases) de la aplicación de K-Medias sobre el conjunto de datos de entrenamiento (Iteración 4).

- **NN + K-Medias**

En la Tabla 4.34 se presenta la matriz de confusión de dos clases producto de la unificación de los resultados de las matrices de confusión de dos clases de los niveles correspondientes a NN y a K-Medias. En la misma se puede observar como en comparación con la matriz de confusión del nivel de NN presentada en la Tabla 4.32, la matriz de confusión final presenta mayor cantidad de falsos positivos debido a la generación

de falsos positivos generada en el nivel de K-Medias. Como se mencionó previamente, la generación de falsos positivos no es deseada; sin embargo, presentan la oportunidad para retro-alimentar el modelo.

Real \ Predicción	Ataque	Normal
Ataque	5880	18
Normal	1408	5291

Tabla 4.34: Matriz de confusión (2 Clases) del modelo híbrido GFR - NN - K-Medias sobre el conjunto de datos de entrenamiento (Iteración 4).

En la Tabla 4.35 se presenta un resumen de las medidas de rendimiento binarias en todos los niveles del modelo. En la misma se refleja el excelente desempeño presentado por el primer nivel concerniente a NN y como K-Medias deteriora el desempeño presentado por el nivel previo con la generación de falsos positivos. Al final la tasa de aciertos total es de 88.68 % que es una tasa de acierto bastante buena pero inferior a la presentada en la Iteración 1.

Modelo	Parámetros	Tipo	Sensibilidad	Especificidad	Precisión	Tasa Acierto
(5) GFR - NN - K-Medias	Neuronas = 30	NN (2 Clases)	99.63 %	99.61 %	99.66 %	99.61 %
		NN \Rightarrow K-Medias	21.74 %	79.25 %	0.36 %	79.06 %
		NN + K-Medias	99.69 %	78.98 %	80.68 %	88.68 %

Tabla 4.35: Medidas de rendimiento binarias de GFR - NN - K-Medias sobre el conjunto de datos de entrenamiento (Iteración 4).

▪ Conclusiones parciales

El rendimiento de NN es excelente; sin embargo, K-Medias agrega una cantidad considerable de falsos positivos al modelo. Este comportamiento ya fue presentado en las Iteraciones 2 y 3. Únicamente en la Iteración 1 fue cuando en el conjunto de entrenamiento usando todas las características el rendimiento del primer nivel no se vio deteriorado por la aplicación de K-Medias.

Aanálisis sobre el conjunto de prueba

En esta sección se presenta el análisis e interpretación de los resultados obtenidos sobre el conjunto de prueba en la Iteración 4. Esta iniciará con el análisis correspondiente al primer nivel concerniente a NN, luego el análisis del segundo nivel correspondiente a K-Medias, y se culminará con el análisis grupal de ambos enfoques.

- NN

En la Tabla 4.36 se presenta la matriz de confusión de cinco clases producto de la clasificación del mejor modelo seleccionado de NN mediante la validación cruzada de 10 conjuntos sobre el conjunto de prueba. Esta matriz de confusión se ve mucho más desordenada que la presentada sobre el conjunto de entrenamiento reflejada en la Tabla 4.30 debido a que el conjunto de prueba presenta una gran cantidad de nuevos registros; sin embargo, la matriz de confusión de cinco clases sobre el conjunto de prueba acumula la mayoría de los registros en la diagonal, reflejando de esta manera un muy buen desempeño en la clasificación de los registros con una tasa de acierto de 74.60 %.

Real\Predicción	DoS	Normal	Probing	R2L	U2R
DoS	5626	1545	273	14	0
Normal	69	9476	147	16	3
Probing	294	701	1426	0	0
R2L	9	2391	71	283	0
U2R	2	118	70	4	6

Tabla 4.36: Matriz de confusión (5 Clases) de NN sobre el conjunto de datos de prueba (Iteración 4).

En la Tabla 4.37 se presenta la tasa de acierto por etiqueta y la tabla de acierto total de la matriz de confusión de cinco clases presentada en la Tabla 4.36. La tasa de acierto de 74.60 % se respalda en una muy correcta clasificación de los registros pertenecientes a la clase Normal y una gran cantidad de ataques detectados pertenecientes a las clases DoS y Probing. Por otra parte, las clases R2L y U2R no presentaron una gran cantidad de aciertos debido a que estas clases cuentan con pocos registros en el conjunto de entrenamiento.

Tipo	DoS	Normal	Probing	R2L	U2R	Total
NN	75.44 %	97.58 %	58.90 %	10.28 %	3.00 %	74.60 %

Tabla 4.37: Tasas de acierto (5 Clases) de NN sobre el conjunto de datos de prueba (Iteración 4).

La Tabla 4.38 presenta la matriz de confusión de dos clases derivada de la matriz confusión de cinco clases presentada en la Tabla 4.36. En esta se puede observar como la gran mayoría de ataques y de tráfico normal son clasificados de buena manera con una baja generación de falsos positivos pero una alta generación de falsos negativos.

La Figura 4.21 presenta la curva ROC referente al modelo de NN. En esta se observa como su desempeño es bastante bueno ya que se separa bastante de la diagonal, acumulando gran cantidad de área bajo la curva y adicionalmente presentando mejor

Real \ Predicción	Ataque	Normal
Ataque	8078	4755
Normal	235	9476

Tabla 4.38: Matriz de confusión (2 Clases) de NN sobre el conjunto de datos de prueba (Iteración 4).

desempeño que los modelos homólogos de las iteraciones previas presentados en las Figuras 4.5, 4.16 y 4.19. Este comportamiento indica que la selección de características y el aumento del número de neuronas de la capa intermedia fue fructífero.

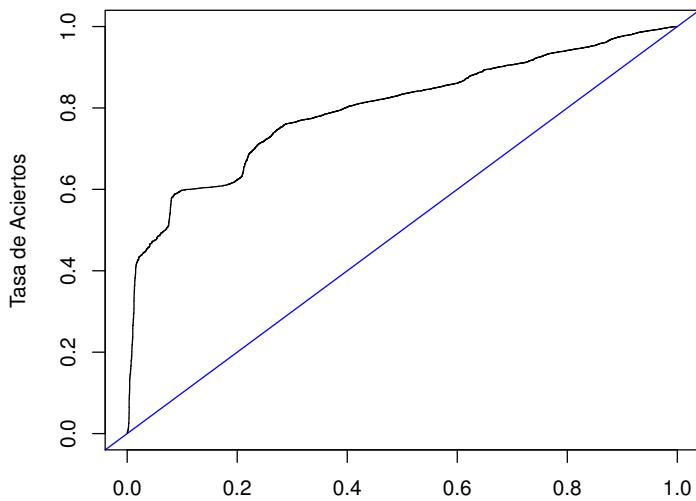


Figura 4.21: Curva ROC de NN sobre el conjunto de datos prueba (Iteración 4).

- **NN \Rightarrow K-Medias**

En la Tabla 4.39 se presenta la matriz de confusión de dos clases derivada de la clasificación de K-Medias. En esta tabla se puede observar como se detectaron 2138 nuevos ataques, se disminuyó el número de falsos negativos y se incrementó el número de falsos positivos. Como se ha mencionado previamente, el número de falsos positivos representa un hecho que a pesar de no ser deseado permite que registros considerados inusuales puedan ser analizados y de esta manera se pueda retro-alimentar el modelo para que el mismo sea más preciso. Así mismo, el objetivo principal que era el de incrementar el número de ataques detectados y decrementar el número de falsos negativos presentes en la clasificación fue logrado de manera satisfactoria, indicando que K-Medias fue un buen complemento de NN en esta iteración usando las 19 características más importantes seleccionadas por GFR para SVM (Radial).

Real \ Predicción	Ataque	Normal
Ataque	2138	2617
Normal	1814	7636

Tabla 4.39: Matriz de confusión (2 Clases) de la aplicación de K-Medias sobre el conjunto de datos de prueba (Iteración 4).

- **NN + K-Medias**

La Tabla 4.40 presenta la matriz de confusión de dos clases producto de la unificación de los resultados de las matrices de confusión de dos clases de los niveles correspondientes a NN y K-Medias. En esta se puede observar que a diferencia de la matriz de confusión presentada en la Tabla 4.34 correspondiente al rendimiento del modelo híbrido sobre el conjunto de entrenamiento, en esta ocasión K-Medias si fue un buen complemento para NN aumentando el número de ataques detectados y distribuyendo más o menos de manera equitativa los fallos en los falsos positivos y falsos negativos. Recordando que la generación de falsos positivos además de permitir examinar registros considerados anómalos o inusuales, también brindan la oportunidad de retro-alimentar el modelo.

Real \ Predicción	Ataque	Normal
Ataque	10216	2617
Normal	2075	7636

Tabla 4.40: Matriz de confusión (2 Clases) del modelo híbrido GFR - NN - K-Medias sobre el conjunto de datos de prueba (Iteración 4).

La Tabla 4.41 presenta las medidas de rendimiento binarias de cada uno de los niveles de manera individual y luego de manera conjunta junto con los tiempo de entrenamiento y de prueba. En la misma se puede observar como la tasa de acierto de NN con dos clases presenta muy poca varianza con respecto a la tasa de acierto con cinco clases, esta situación refleja que los resultados obtenidos por NN para 5 clases fueron bastante precisos a la hora de clasificar entre ataques. Por otra parte a diferencia de otras iteraciones, esta ocasión se presenta una tasa de acierto de 68.68 % detectando el 44.96 % de los falsos negativos generados por el primer nivel correspondiente a NN. De esta manera, el rendimiento en la tasa de aciertos se incrementó en un 2 % aproximadamente. Por último se puede apreciar como en comparación con la Tabla 4.29, los tiempos de entrenamiento y de prueba son similares; sin embargo, K-Medias presenta mejor tasa de acierto debido al uso de las características seleccionadas por GFR para SVM (Radial). Finalmente los tiempo de entrenamiento y de prueba fueron similares para el nivel de K-Medias presentado en la Tabla 4.29 referente al análisis de modelos sobre el conjunto de prueba en la Iteración 3. Por otra parte, para NN los tiempos fueron mayores que los indicados en la Iteración 3 debido a que se usaron mayor cantidad de neuronas.

Parámetros	Tipos	Sensibilidad	Especificidad	Precisión	Tasa Acierto	Tiempo Entrenamiento	Tiempo Prueba
Neuronas = 30	NN (2 Clases)	62.95 %	97.58 %	97.17 %	77.87 %	398.17 segs	0.50 segs
	NN ⇒ K-Medias	44.96 %	80.58 %	53.75 %	68.68 %	3.89 segs	0.04 segs
	NN + K-Medias	79.61 %	78.63 %	83.12 %	79.19 %	402.06 segs	0.54 segs

Tabla 4.41: Medidas de rendimiento binarias del modelo híbrido GFR - NN - K-Medias sobre el conjunto de datos de prueba (Iteración 4).

En la Tabla 4.42 se presenta la proporción de ataques conocidos y no-conocidos detectados por el modelo híbrido (5) GFR - NN - K-Medias en esta Iteración 4. En la misma destaca que el primer nivel concerniente a NN es capaz de detectar el 28 % y el 69.16 % de los ataques conocidos sobre el conjunto de datos de prueba. De esta manera se demuestra la eficacia del enfoque supervisado a la hora de detectar ataques conocidos. Por otra parte, el segundo nivel correspondiente a K-Medias detecta el 67.91 % de los nuevos ataques de los registros que fueron pasados al segundo nivel de clasificación y el 24.40 % de los ataques conocidos de los mismos. También esto refleja lo tratado en la Sección 2.3.1 donde se trataron las fortalezas y debilidades del enfoque supervisado y no-supervisado de aprendizaje automático en la detección de intrusos en las redes de computadoras. Finalmente, al combinar ambos resultados se logra detectar con el modelo híbrido presentado en esta iteración un 68.93 % y 75.90 % de tasa de acierto a la hora de detectar ataques no-conocidos y conocidos respectivamente. Resultado que indica un muy buen desempeño del modelo y un buen complemento entre los enfoques supervisado y no-supervisado, donde el primero detecta ataques conocidos y el segundo los ataques no-conocidos.

Tipo	Nuevos Ataques Detectados	% Acierto Nuevos Ataques	Ataques Conocidos Detectados	% Acierto Ataques Conocidos
NN	1059	28.24 %	6282	69.16 %
NN ⇒ K-Medias	1526	67.91 %	612	24.40 %
NN + K-Medias	2585	68.93 %	6894	75.90 %

Tabla 4.42: Resumen de la distribución de ataques detectados por nivel del modelo GFR - NN - K-Medias sobre el conjunto de datos de prueba (Iteración 4).

■ Conclusiones parciales

El rendimiento del modelo (5) GFR - NN - K-Medias sobre el conjunto de datos fue muy bueno, combinando de buena manera los enfoques supervisado y no-supervisado para crear un modelo híbrido que detecta tanto ataques conocidos como nuevos ataques.

Este comportamiento está respaldado por una certeza en la toma de decisiones muy buena reflejada en la Figura 4.21 correspondiente a la curva ROC y altas tasas de aciertos para cada una de las clases.

Conclusiones de la Iteración 4

La Iteración 4 presenta el modelo (5) GFR - NN - K-Medias. Este presenta un desempeño muy bueno tanto para la detección de registros conocidos como para la detección de los nuevos registros presentes en el conjunto de prueba. En este modelo se combinaron las características resaltantes encontradas en iteraciones anteriores con la finalidad de encontrar el modelo que recopilara todas las fortalezas encontradas para el aumento de la eficacia a la hora de detectar ataques y clasificar de manera correcta el tráfico normal. El modelo creado en esta iteración es un muy buen candidato para una implementación en un ambiente de producción de un NIDS basado en aprendizaje automático, donde la principal característica de este modelo es que al ser híbrido y combinar los enfoques supervisado y no-supervisado permite detectar una gran cantidad de ataques conocidos y no-conocidos con bajas tasas de falsos positivos que permiten retro-alimentar el modelo. Adicionalmente, cuenta con una poca generación de falsos negativos.

4.4. Trabajos relacionados

Li, Kia y colaboradores [21] presentan un trabajo donde se usa GFR en combinación con SVM (Radial) usando la técnica de colonias de hormigas para el muestreo sobre el conjunto de datos KDD99. Se logró una tasa de acierto de 98 %, haciendo uso de validación cruzada de 10 conjuntos sobre el conjunto de entrenamiento. En este trabajo de investigación se logró una tasa superior a la presentada en la publicación referenciada previamente concerniente a 99 % para la combinación de GFR y SVM (Radial). Adicionalmente, en este trabajo se presentó una comparación de SVM (Radial) contra NN, indicando que NN es superior en la tasa de acierto y en la curva ROC tanto para el conjunto de entrenamiento como para el conjunto de prueba. Por otra parte, se reflejó la eficacia de GFR como método reducción de características para la creación de modelos que generalicen frente a nuevos tipos de registros.

Thaseen y Kumar [20] presentan una investigación que refleja la eficacia de PCA para reducir el ruido sobre un modelo de SVM optimizado. Los resultados presentados en esta investigación reflejan una tasa de aciertos de 99 % sobre el conjunto de entrenamiento, resultados que fueron igualados en el presente trabajo de investigación; sin embargo, en dicha publicación obvian las pruebas de la aplicabilidad de PCA frente a nuevos tipos de registros. Estas pruebas fueron realizadas en la presente investigación, indicando que bajo el escenario de nuevos registros, PCA no es una técnica efectiva debido a que si las matrices de covarianza de los conjuntos de datos de entrenamiento y de prueba son muy diferentes, entonces la efectividad de los modelos se verá notablemente deteriorada.

El trabajo presentado por Mukkamala, Janoski y Sung [5] ilustra la comparación entre diferentes arquitecturas de NN con múltiples capas intermedias contra SVM (Radial); indicando que SVM (Radial) es mucho más rápido, preciso y escalable que NN usando MATLAB como lenguaje de programación (Bibliotecas omitidas). En el presente trabajo de investigación usando el lenguaje de programación R y los paquetes *nnet* y *e1071* para NN y SVM (Radial) respectivamente, los resultados indicaron que con una sola capa intermedia con menor cantidad de neuronas que las usadas en la implementación de la investigación referenciada previamente, NN alcanza mejor tasa de acierto que SVM (Radial). Así mismo, NN tiene mayor certeza en las predicciones realizadas, se entrena más rápido y realiza predicciones de forma más rápida que SVM (Radial).

Dentro de las publicaciones sobre modelos híbridos, el trabajo realizado por Tahir, Hasan y Said [25] presenta la combinación de SVM (Kernel no indicado) con K-Medias usando el conjunto de datos NSL-KDD de entrenamiento. Los resultados obtenidos en esta publicación fueron mejorados en el presente trabajo de investigación no solo con la combinación de SVM (Radial) y K-Medias, sino también por los modelos basados en NN que presentaron mejor desempeño que los basados en SVM (Radial).

Adicionalmente, Veeramachaneni [8] presenta un modelo de ML con retro-alimentación bajo un escenario diferente al conjunto de datos NSL-KDD ya que este utilizó *logs* de páginas web; sin embargo, los resultados obtenidos en dicha publicación separando el conjunto de entrenamiento del de prueba tuvieron una mejora significativa al incluir un sistema de retro-alimentación, alcanzando una tasa de aciertos de 86 % contra el 79 % alcanzado en el mejor modelo obtenido en la presente investigación. Situación que sugiere que la inclusión de un módulo de retro-alimentación en el enfoque presentado a lo largo del documento podría mejorar los resultados obtenidos.

Tal como se presenta en la publicación realizada por Atilla y Hamit [24], la mayoría de los trabajos de investigación no presentan de buena manera los escenarios usados en los experimentos y tampoco presentan de buena manera los resultados obtenidos, haciendo engoroso el proceso de comparación de resultados de los modelos obtenidos con otras publicaciones al omitir datos relevantes referentes a parámetros, número de registros usados para entrenamiento y prueba, características seleccionadas y las medidas de rendimiento oportunas para la comparación entre modelos. Así mismo, la publicación referenciada previamente presenta un conjunto de sugerencias a adoptar para facilitar los aspectos mencionados previamente a otros investigadores en el área, las mismas fueron tomadas en cuenta a lo largo de esta investigación, detallando todas las consideraciones de diseño e implementación necesarias para que los resultados puedan ser reproducidos por otro investigador, facilitando el proceso de comparación con otros modelos.

Capítulo 5

Conclusiones

En el presente trabajo de investigación se diseñaron, implementaron y analizaron diferentes enfoques de modelos basados en técnicas híbridas de ML para probar su aplicabilidad en la implementación de un NIDS. Para esto se realizó una amplia revisión bibliográfica correspondiente al estado del arte en el uso de aprendizaje automático en el área de seguridad en redes de computadoras. Con este proceso se logró conocer las tendencias de investigación en el área, logrando así identificar los algoritmos, enfoques y herramientas usadas por la comunidad científica dentro de este campo de investigación. Una vez estudiado el campo de estudio y luego de haber detectado las herramientas disponibles, la investigación se centró en el estudio de las técnicas híbridas de ML que combinan técnicas de enfoque supervisado en conjunto con técnicas de enfoque no-supervisado con la finalidad de que al combinar ambos enfoques se pudieran complementar para así mejorar la tasa de acierto en la labor de detección de intrusos en redes de computadoras. Las peculiaridades de ambos enfoques fue presentada en la Tabla 2.2.

Las actividades fueron llevadas a cabo siguiendo el flujo general de ML presentado en la Figura 2.5 y donde todas sus estaciones fueron definidas en la Sección 2.2.1. Adicionalmente, adoptando las buenas prácticas identificadas en [24] para la investigación en el uso de aprendizaje automático en el área de seguridad en redes de computadoras. En la búsqueda del modelo que mejor se ajustara al problema, se usó un enfoque empírico e iterativo, usando la matriz de confusión como medida de rendimiento fundamental, ya que de esta se pudieron derivar otras métricas que permitieron medir el desempeño de los modelos creados. Se realizaron cuatro iteraciones. La metodología, consideraciones de diseño, consideraciones de implementación y arquitectura adoptada para la solución fue presentada en el Capítulo 3.

Se combinaron los algoritmos de enfoque supervisado NN y SVM (Radial), y el algoritmo de enfoque no supervisado K-Medias como complemento de los mismos. Adicionalmente, se usaron las técnicas de selección de características PCA y GFR, y se realizó selección de parámetros. Los diferentes modelos creados a partir de la combinación de los algoritmos mencionados previamente que fueron definidos en la Sección 3.2 reciben como entrada en el primer nivel correspondiente a aprendizaje supervisado registros etiquetados con cinco

clases divididas en cuatro clases de ataques (DoS, *Probing*, R2L y U2R) y una etiqueta que identifica al tráfico normal (Normal). Así mismo, este primer nivel clasificará los registros dentro de las cinco clases presentadas previamente. Por otra parte, el segundo nivel correspondiente al enfoque no-supervisado tomará como entrada los registros clasificados por el primer nivel como pertenecientes a la clase Normal y tratará de identificar en los mismos los ataques que no fueron detectados en el primer nivel. En este segundo nivel, la entrada y salida se realizó usando dos clases correspondientes a Ataque o Normal. Esto debido a que la naturaleza del enfoque no-supervisado identifica las anomalías calculando la desviación de los registros con respecto a un comportamiento definido como normal y no es capaz de detectar específicamente una clase de ataque particular.

Durante la implementación y análisis de los diferentes modelos, se pudo observar como los dos objetivos principales en la implementación de NIDS basados en técnicas de ML son maximizar la cantidad de ataques detectados y minimizar la cantidad de falsos negativos. Adicionalmente se observó como los falsos positivos permiten retro-alimentar los modelos para que los mismos puedan ser más precisos. Basado en lo expuesto previamente, los algoritmos basados en enfoque supervisado tuvieron un desempeño muy bueno en la detección de ataques conocidos; específicamente, NN tuvo un mejor desempeño que SVM (Radial) que quizás no fue reflejado en la tasa de acierto de manera considerable pero si en las curvas ROC presentadas en las diferentes iteraciones, donde se observó que la toma de decisiones presentaron más certeza, por lo tanto, al SVM (Radial) presentar una curva ROC tan errática, no se recomienda el uso de un modelo conjunto que combine a los algoritmos NN y SVM (Radial). Por otra parte, el algoritmo K-Medias tuvo un alto desempeño en la detección de ataques no conocidos, resultando ser un buen complemento para NN si la cantidad de falsos negativos generados por este primer nivel de clasificación es alta, de otra manera, el aporte de K-Medias es poco significativo. Como aspecto a destacar, se identificó que K-Medias no fue un buen complemento de SVM (Radial) debido a que ambos algoritmos presentan el mismo enfoque de separación basado en la búsqueda de circunferencias en planos n-dimensionales y es por esto que K-Medias no lograba detectar gran cantidad de ataques como complemento de SVM (Radial), ya que se aplicaba dos veces el mismo criterio de clasificación.

En las técnicas de reducción de características se pudo observar que PCA no es una buena estrategia en este escenario, debido a que las matrices de covarianza de los conjuntos de datos de entrenamiento y de prueba son muy diferentes y por lo tanto los modelos tuvieron un pobre desempeño sobre el conjunto de datos de prueba. Por otra parte, la selección de características usando GFR sí fue bastante fructífera, eliminando ruido de los modelos, haciéndolos más rápidos y precisos. En cuestiones de tiempo, se pudo apreciar también que NN es mucho más rápido que SVM (Radial) tanto para el entrenamiento como para las predicciones. Por otra parte, se observó que la selección de parámetros siempre apostó por modelos más complejos sobre el conjunto de datos de entrenamiento que hicieron que los mismos se sobre-ajustaran al mismo, perdiendo de esta manera poder de generalización frente a los nuevos registros presentes en el conjunto de datos de prueba.

El proceso de ML es un proceso empírico y es importante tener varias medidas de rendimiento que permitan al experto de ML interpretar las fortalezas y debilidades de los mismos con el fin de seleccionar el algoritmo que se adapte de mejor forma a un escenario en particular. Adicionalmente, es importante la interacción de un experto con los modelos para retro-alimentarlos de manera periódica.

Luego de cuatro iteraciones se encontró un modelo que se considera lo bastante bueno como para considerarse candidato en la implementación de un NIDS basado en técnicas de ML en un ambiente de producción. El modelo (5) GFR - NN - K-Medias de la Iteración 4, presenta un excelente rendimiento en ambos niveles demostrando que la combinación de los enfoques supervisado y no supervisado fue fructífera.

Con lo expuesto previamente, concluimos que se cumple a cabalidad con los objetivos presentados en las Secciones 1.1 y 1.2, demostrando que es posible crear un modelo híbrido que combine técnicas de ML supervisadas y no-supervisadas (Híbridas) que automaticen el proceso de detección de intrusos en redes de computadoras de una forma rápida y eficaz.

5.1. Contribuciones

Las contribuciones de este trabajo de investigación son las siguientes:

- Se realizó una revisión amplia sobre las tendencias de la implementación de NIDS basados en técnicas de ML que pueden servir como punto de partida para futuros trabajos de investigación.
- Se diseñó una metodología genérica para la implementación de NIDS basados en técnicas de ML que recopila buenas prácticas en lo que respecta a la investigación dentro de esta área.
- Se realizó investigación en un área de sumo interés por la comunidad científica en la actualidad, demostrando la aplicabilidad de los NIDS basados en técnicas híbridas de ML. Adicionalmente, se expusieron otras consideraciones particulares durante la implementación que pueden ser de ayuda para otras personas que realicen investigación en el mismo campo de estudio.
- La investigación realizada puede ser utilizada como material para materias tales como: Inteligencia Artificial, Minería de datos, Seguridad en Redes de Computadoras y Redes de Computadoras.

5.2. Limitaciones

La principal limitación de este trabajo fue la de no contar con un conjunto de datos reciente, motivo que condujo al uso de un conjunto de datos tan antiguo como lo es el

conjunto de datos NSL-KDD que es derivado del popular conjunto de datos KDD99 generado en el año 1999. Sin embargo, elaborar una nueva base de conocimientos es una tarea bastante laboriosa que se escapa del alcance de esta investigación. Adicionalmente, como expusieron Atilla y Hamit [24], el conjunto de datos de KDD99 es el conjunto más utilizado por la comunidad científica a la hora de realizar investigación en el uso de aprendizaje automático en el área de seguridad en redes de computadoras.

5.3. Trabajos futuros

Se proponen los siguientes trabajos futuros.

- Utilizar otra combinación de algoritmos supervisados y no-supervisados. Particularmente algoritmos de diferente naturaleza que puedan complementarse de buena manera, para así evitar la situación presentada durante este trabajo de investigación entre los algoritmos SVM (Radial) y K-Medias, y así poder comparar los resultados con los obtenidos en el presente trabajo.
- Diseñar una estación de retro-alimentación para los modelos, específicamente para el modelo obtenido en la Iteración 4. Se propone la retro-alimentación del primer nivel correspondiente al enfoque supervisado a partir de los ataques detectados por el segundo nivel correspondiente al enfoque no-supervisado creando un modelo híbrido condicionado que considere la probabilidad con la que el primer nivel realizó la predicción de los registros. Esto aprovecharía la fortaleza del enfoque supervisado referente a la alta tasa de aciertos sobre registros conocidos para así reducir la cantidad de falsos positivos y también aprovecharía la característica del primer nivel para detectar cinco clases en vez de dos. Por último, sería interesante evaluar el impacto de la retroalimentación realizando un estudio en el que se pueda evaluar hasta qué punto sea beneficioso retro-alimentar el modelo, sin que la actualización del mismo ocasione una pérdida de memoria con respecto al conocimiento previamente adquirido.
- Conseguir o elaborar una base de conocimientos más reciente. Se podrían usar las características seleccionadas por GFR para NN y SVM (Radial), intersectar las primeras 19 características y tratar de extraerlas de capturas de tráfico de red.
- Implementar un *parser* de capturas de tráfico de red a una vista minable que permita poner en producción los diferentes modelos en un ambiente real.

Referencias

- [1] J. Anderson, “Computer Security Threat Monitoring and Surveillance,” Technical report, James P. Anderson Company, Fort Washington, Pennsylvania, Tech. Rep., 1980.
- [2] W. Stallings, *Cryptography and Network Security Principles and Practice*. Pearson, 2014.
- [3] P. Ning y S. Jajodia, “Intrusion Detection Techniques,” *The Internet Encyclopedia*, 2003.
- [4] D. Bhattacharyya y J. Kalita, *Network Anomaly Detection: A machine Learning Perspective*. CRC Press, 2013.
- [5] S. Mukkamala, G. Janoski, y A. Sung, “Intrusion Detection Using Neural Networks and Support Vector Machines,” in *Proceedings of the 2002 International Joint Conference on Neural Networks, 2002. IJCNN’02.*, vol. 2. IEEE, 2002, pp. 1702–1707.
- [6] J. Sundus, M. Zaiton, M. Ma, y Y. Warusia, “Machine Learning Techniques for Intrusion Detection System: A Review,” *Journal of Theoretical & Applied Information Technology*, vol. 72, n.º 3, 2015.
- [7] T. Anantvalee y J. Wu, “A survey on intrusion detection in mobile ad hoc networks,” in *Wireless Network Security*. Springer, 2007, pp. 159–180.
- [8] K. Veeramachaneni, “AI 2: Training a Big Data Machine to Defend,” in *International Conference in Big Data security on Cloud*. IEEE, 2016.
- [9] D. Upadhyaya y S. Jain, “Hybrid Approach for Network Intrusion Detection System Using K-Medoid Clustering and Naive Bayes Classification,” *International Journal of Computer Science Issues (IJCSI)*, vol. 10, n.º 3, pp. 231–236, 2013.
- [10] T. Mitchell, *Machine Learning*. McGraw Hill, 1997.
- [11] K. Leung y C. Leckie, “Unsupervised Anomaly Detection in Network Intrusion Detection Using Clusters,” in *Proceedings of the Twenty-eighth Australasian conference on Computer Science- Volume 38*. Australian Computer Society, Inc., 2005, pp. 333–342.

- [12] A. Samuel, “Some Studies in Machine Learning Using the Game of Checkers,” *IBM Journal of research and development*, vol. 3, n.º 3, pp. 210–229, 1959.
- [13] G. James, D. Witten, T. Hastie, y R. Tibshirani, *An Introduction to Statistical Learning*. Springer, 2013.
- [14] C. Tsai, Y. Hsu, C. Lin, y W. Lin, “Intrusion Detection by Machine Learning: A Review,” *Expert Systems with Applications*, vol. 36, n.º 10, pp. 11 994–12 000, 2009.
- [15] L. Fausset, *Fundamentals of Neural Networks: Architectures, Algorithms, and Applications*. Prentice-Hall, Inc., 1994.
- [16] S. Samarasinghe, *Neural Networks for Applied Sciences and Engineering: From Fundamentals to Complex Pattern Recognition*. CRC Press, 2006.
- [17] R. Tibshirani, G. Walther, y T. Hastie, “Estimating the Number of Clusters in a Data Set Via the Gap Statistic,” *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 63, n.º 2, pp. 411–423, 2001.
- [18] C. Aggarwal, *Data Mining: The Textbook*. Springer, 2015.
- [19] Y. Bhavsar y K. Waghmare, “Intrusion Detection System Using Data Mining Technique: Support Vector Machine,” *International Journal of Emerging Technology and Advanced Engineering*, vol. 3, n.º 3, pp. 581–586, 2013.
- [20] S. Thaseen y C. Kumar, “Intrusion Detection Model Using fusion of PCA and optimized SVM,” in *International Conference on Contemporary Computing and Informatics (IC3I)*. IEEE, 2014, pp. 879–884.
- [21] Y. Li, J. Xia, S. Zhang, J. Yan, X. Ai, y K. Dai, “An Efficient Intrusion Detection System Based on Support Vector Machines and Gradually Feature Removal Method,” *Expert Systems with Applications*, vol. 39, n.º 1, pp. 424–430, 2012.
- [22] G. Canavos, *Applied Probability and Statistical Methods*. Little, Brown, 1984.
- [23] T. Fawcett, “An Introduction to ROC Analysis,” *Pattern recognition letters*, vol. 27, n.º 8, pp. 861–874, 2006.
- [24] O. Atilla y E. Hamit, “A Review of KDD99 Dataset Usage in Intrusion Detection and Machine Learning Between 2010 and 2015,” *PeerJ Preprints*, vol. 4, p. e1954v1, 2016.
- [25] H. Tahir, W. Hasan, y A. Said, “Hybrid Machine Learning Technique for Intrusion Detection System.” 5th International Conference on Computing and Informatics (ICOPI) 2015, 2015.
- [26] J. Kittler, M. Hater, R. Duin, y J. Matas, “On Combining Classifiers,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 20, n.º 3, pp. 226–239, 1998.

- [27] F. Majidi, H. Mirzaei, T. Iranpour, y F. Foroughi, “A Diversity Creation Method for Ensemble Based Classification: Application in Intrusion Detection,” in *7th IEEE International Conference on Cybernetic Intelligent Systems, 2008. CIS 2008.* IEEE, 2008, pp. 1–5.
- [28] W. Lee y S. Stolfo, “A Framework for Constructing Features and Models for Intrusion Detection Systems,” *ACM Transactions on Information and System Security (TISSEC)*, vol. 3, n.º 4, pp. 227–261, 2000.
- [29] W. Zhang, Q. Yang, y Y. Geng, “A Survey of Anomaly Detection Methods in Networks,” in *International Symposium on Computer Network and Multimedia Technology, 2009. CNMT 2009.* IEEE, 2009, pp. 1–3.
- [30] M. Tavallaei, E. Bagheri, W. Lu, y A. Ghorbani, “A Detailed Analysis of the KDD CUP 99 Data Set,” in *Proceedings of the Second IEEE Symposium on Computational Intelligence for Security and Defence Applications 2009*, 2009.
- [31] J. McHugh, “Testing Intrusion Detection Systems: A Critique of the 1998 and 1999 DARPA Intrusion Detection System Evaluations as Performed by Lincoln Laboratory,” *ACM Transactions on Information and System Security (TISSEC)*, vol. 3, n.º 4, pp. 262–294, 2000.
- [32] T. Shon y J. Moon, “A Hybrid Machine Learning Approach to Network Anomaly Detection,” *Information Sciences*, vol. 177, n.º 18, pp. 3799–3821, 2007.
- [33] L. Dhanabal y S. Shanthyarajah, “A Study on NSL-KDD Dataset for Intrusion Detection System Based on Classification Algorithms,” *International Journal of Advanced Research in Computer and Communication Engineering*, vol. 4, n.º 6, pp. 446–452, 2015.