APLICACIÓN DE SELECCIÓN DE CARACTERISTICAS, METRICAS DE APRENDIZAJE Y REDUCCION DE DIMENSION EN SISTEMAS DE DETECCION DE INTRUSOS

FABIO MENDOZA PALECHOR
INGENIERO DE SISTEMAS, M.Sc.(C)

UNIVERSIDAD TECNOLOGICA DE BOLIVAR
CARTAGENA DE INDIAS
AGOSTO 2013

APLICACIÓN DE SELECCIÓN DE CARACTERISTICAS, METRICAS DE APRENDIZAJE Y REDUCCION DE DIMENSION EN SISTEMAS DE DETECCION DE INTRUSOS

FABIO MENDOZA PALECHOR
INGENIERO DE SISTEMAS, M.Sc.(C)

TRABAJO DE TESIS PARA OPTAR AL TITULO DE
MAGISTER EN INGENIERIA CON ENFASIS EN INGENIERIA DE SISTEMAS

DIRECTOR
EDUARDO DE LA HOZ CORREA
DOCTOR OF PHILOSOPHY (PH.D) (C) TECHNOLOGY AND COMMUNICATION
MASTER'S DEGREE IN COMPUTER ENGINEERING AND NETWORKS
MASTER OF COMPUTER AND SYSTEMS ENGINEERING
SPECIALIST COMPUTERS NETWORKS

UNIVERSIDAD TECNOLOGICA DE BOLIVAR
CARTAGENA DE INDIAS
AGOSTO 2013

TITULO EN ESPAÑOL

Aplicación De Selección De Características, Métricas De Aprendizaje Y Reducción De Dimensión En Sistemas De Detección De Intrusos

TITLE IN ENCLISH

Applying Features Selection, Learning Metrics and Dimension Reduction in Intrusion Detection Systems

RESUMEN

La seguridad de las redes de computadores es un punto crítico en las diferentes empresas tanto a nivel nacional como internacional, garantizar la confidencialidad, integridad y disponibilidad de la información es una tarea ardua la cual requiere de mucho tiempo y muchas veces genera grandes costo a nivel económico para las diferentes organizaciones, por ello se ha intentado dar solución a los problemas de seguridad de las redes con diferentes alternativas tanto de hardware como de software que muchas veces son poco eficientes a la hora de detectar los ataques dentro de una red. En esta tesis se define un prototipo que permite identificar los ataques realizados a una red, dicho prototipo se basa en los datos pertenecientes al data-set NSL-KDD el cual es tratado a través de diferentes técnicas de Selección De Características, Métricas De Aprendizaje Y Reducción De Dimensión tales como FEAST, SVM y ISOMAP, la utilización de dichas técnicas tiene como objetivo lograr resultados confiables para la detección de ataques en redes de computadores, dichos resultados serán validados a partir de los datos que arrojados por la aplicación de técnica SOM y GHSOM bajo el mismo condiciones de prueba.

ABSTRAC

The security of computer networks at a critical point in different companies both nationally and internationally, to ensure confidentiality, integrity and availability of information is an arduous task which is time-consuming and often generates high-level cost economic for different organizations, so it has been tried to solve the problems of network security with different alternatives both hardware and software are often inefficient when detect attacks within a network. This thesis defines a prototype for identifying attacks made to a network, the prototype is based on data pertaining to the data-set NSL-KDD which is treated through different techniques Features Selection, Learning and Metrics Dimension reduction such as FEAST, SVM and ISOMAP, the use of these techniques aims to achieve reliable results for detection of attacks on computer networks, such results will be validated from the data thrown by the technical implementation of SOM and GHSOM under the same test conditions.

Palabras Claves: Data-Set, Seguridad, Redes, informacion.

Keyword: Data-Set, security, networks, information.

Nota de aceptación
Trabajo de tesis
"mención"

_____

Jurado

_____

Jurado

_____

Director
Eduardo De la Hoz Correa
DOCTOR OF PHILOSOPHY (PH.D) (C) TECHNOLOGY AND COMMUNICATION
MASTER'S DEGREE IN COMPUTER ENGINEERING AND NETWORKS
MASTER OF COMPUTER AND SYSTEMS ENGINEERING
SPECIALIST COMPUTERS NETWORKS

Cartagena de indias agosto 2013

AGRADECIMIENTOS

Agradezco a DIOS por llenarme de sabiduría y paciencia para culminar de manera exitosa este proceso que inició hace 2 años, sin él no hubiese sido posible alcanzar esta gran meta. A DIOS padre doy gracias por tan bella familia Yulieth Diago Hernandez, Fabio Andres Mendoza Diago, quienes han sido un soporte para el desarrollo exitoso de este proceso, agradezco su paciencia y comprensión.

Agradezco a mis padres Alvaro Mendoza Arevalo, Janeth Palechor Espitia, quienes me han brindado todo su apoyo para culminar esta etapa de mi vida de forma exitosa.

Agradezco también a mi director de tesis Eduardo de la hoz Correa, por la gran orientación que le dio a este trabajo, sin su colaboración este proceso no hubiese sido posible, gracias por su tiempo y sobre todo gracias por el conocimiento trasmitido a lo largo del desarrollo de esta tesis.

Agradezco a la **Corporación Universidad de la Costa, CUC**, por haber confiado en mí y por brindarme la oportunidad para alcanzar esta tan anhelada meta.

Agradezco en general a todos las personas que de una u otra forma colaboraron para que este proyecto saliera adelante, a todos muchísimas gracias.

TABLA DE CONTENIDO

LISTA DE TABLAS

LISTA DE GRAFICOS

GLOSARIO DE TERMINOS

- **NSL-KDD:** Conjunto de datos que contiene conexiones de diferentes tipos “Normal, DOS, PROBING, U2R, R2L”, generado por University of new Brunswick.
- **FEAST:** Feature Selection Toolbox
- **MIN:** Mutual Information Maximisation
- **MRMR:** Max-Relevance Min-Redundancy
- **MIFS:** Mutual Information Feature Selection
- **CMIM:** Conditional Mutual Info Maximisation
- **JMI:** Joint Mutual Information
- **DISR:** Double Input Symmetrical Relevance
- **CIFE**: Conditional Infomax Feature Extraction
- **ICAP**: Interaction Capping
- **FCBF**: Fast Correlation Based Filter
- **ISOMAP**: Isometric Feature Mapping
- **PCA**: Principal Component Analysis
- **KPCA**: Kernel Principal Component Analysis
- **SVM**: Support Vector Machine
- **SOM**: Self-Organizing Map
- **GHSOM**: Growing Hierarchical Self-Organizing Map
- **IDS**: intrusion detecion systems
- **DOS**: Denial of Services
- **R2L**: Remote to Local
- **U2R**: User to Root
- **RNA**: Artificial Neural Network
- **MPI:** Message Passing Interfac

Capítulo 1: Descripción del Proyecto

En este capítulo se expone de forma clara, los objetivos a alcanzar con el desarrollo de esta investigación, socializando de forma previa los antecedentes y razones que hacen necesaria la temática a desarrollar.

1.1 INTRODUCCIÓN

Las redes de computadores inicialmente fueron diseñadas para una cantidad limitada de usuarios, hoy día se presentan como una necesidad para los hogares, pequeñas, medianas y grandes organizaciones. Los malos diseños de estructura de las redes de computadores han generado brechas de seguridad para mantener la integralidad, confidencialidad y disponibilidad de la información que es transferida por dicho medio, por ello existe la necesidad de proponer nuevas estrategias que permitan la identificación de ingresos no autorizados a las redes de computadores.

El desarrollo de esta investigación tiene como propósito la **aplicación de técnicas de selección de características, métricas de aprendizaje y reducción de dimensión en sistemas de detección de intrusos**, utilizando los datos almacenados en el dataset NSL-KDD, el cual contiene 225.000 registros de conexiones en una red de computadores con 41 características.

La aplicación de diferentes técnicas o métodos de disciplinas como **Inteligencia Artificial**, **Minería de Datos** y **Machine Learning** hoy día es muy común para el análisis de grandes volúmenes de datos, para el desarrollo de esta investigación se aplicó como técnica de selección de características la Toolbox de Matlab "**FEAST**" la cual contiene 13 métodos entre ellos "**mim, mrmr, mifs, cmim, jmi, disr, cife, icap, condred, cmi, relief, fcbf, betagamma**" que permiten la selección de características en los grandes volúmenes de datos. Por otro lado se utiliza **ISOMAP** y **Maquinas de Soporte Vectorial** (**SVM**), como técnica de reducción de dimensión y métricas de aprendizaje respectivamente, para obtener como resultado un prototipo de Matlab basado en las técnicas antes mencionadas garantizando resultados óptimos para la detección de los ataques pertenecientes a las registros de conexiones de redes de computadores del DataSet NSL-KDD.

Para la selección del mejor método del FEAST fue necesario realizar diferentes pruebas utilizando validación cruzada, cada método fue ejecutado de forma paralela con un número de características definidas (5, 10, 15, 20, 25, 30, 35) para posteriormente analizar los mayores niveles de precisión arrojado por cada método. Las pruebas fueron realizadas en el laboratorio de redes de la Corporación Universidad de la Costa, CUC, la cual está compuesta por 20 equipos de cómputo con amplias configuraciones de hardware.

Una vez seleccionado el mejor método del **FEAST** se procedió añadir a los script el código necesario para la ejecución de las técnicas de reducción de características y métricas de aprendizaje que en este caso son **ISOMAP** y **SVM**. Luego de haber desarrollado el prototipo en matlab se somete dicha información a un proceso de depuración que permita minimizar el uso de recursos por parte del prototipo generado y para finalizar se procede a validar la información obtenida en comparación con los datos arrojados en estudios similares donde se aplica SOM - GHSOM .

1.2 OBJETIVOS

Objetivo General:

- Diseñar un prototipo de detección de intrusos basado en técnicas de selección de características, métricas de aprendizaje y reducción de dimensión apoyados en los datos contenidos en el "DATA-SET" NSL-KDD, para aumentar la seguridad de una red.

Objetivos Específicos:

- Analizar las técnicas de Selección de Características, utilizadas en los conjuntos de datos "DATA-SET", para la preparación de los datos.
- Analizar las técnicas de Reducción de Dimensión y Métricas de Aprendizaje, utilizadas en los conjuntos de datos "DATA-SET", para la preparación de los datos.
- Construir y entrenar el prototipo en Matlab, basados en las mejores técnicas selección de características, métricas de aprendizaje y reducción de dimensiones.
- Validar de prototipo creado a partir de las técnicas SOM Y GHSOM.

Capítulo 2: Contexto y Motivación

El desarrollo de este capítulo tiene como propósito dar conocimiento de la necesidad existente relacionada con el estudio de la seguridad en las redes de computadores, teniendo en cuenta las diferentes disciplinas que han intentado dar solución a los problemas existentes que tiene que ver con las falencias de seguridad o tipos de ataques que se puede presentar en una red de computadores. Tener el conocimiento técnico acerca de: seguridad informática, ¿Qué se debe proteger en un sistema informático?, mecanismos de prevención, mecanismos de detección, mecanismos de recuperación, técnicas de selección de características, técnicas de reducción de dimensión y métricas de aprendizaje es el propósito de este capítulo.

2.1 CONTEXTO Y DESCRIPCIÓN DEL PROBLEMA

La falta de medidas que permitan garantizar la seguridad y acceso a los diferentes equipos de cómputo y dispositivos de red es común dentro de pequeñas, medianas y algunas grandes empresas, por ellos es común que se presenten alteraciones de la información en las distintas organizaciones que podrían ser prevenidas con el cumplimiento de políticas de seguridad que restrinjan la ejecución de actividades que atenten contra el buen uso y seguridad de la red.

Los ataques en redes de computadores representan una constante vulnerabilidad para las diferentes organizaciones tanto a nivel local, regional, nacional e internacional, por ello se hace necesaria la detección de ataques que permitan prevenir la copia ilegal, la perdida de la integridad y la pérdida total de la información. En este mundo cambiante contar con herramienta de información y sistemas que admitan contrarrestar dichas ataques permitiría mantener la seguridad en las redes.

Poseer sistemas pasivos dentro representa una vulnerabilidad para cualquier organización debido a que dichos sistemas en algunas ocasiones solo registran la detección de la violación de la seguridad de la red, dejando a un lado la protección y ejecución de procedimientos que puedan detener las acciones ejecutadas por dicha ataque. Por otro lado existen organizaciones que poseen sistemas activos los cuales en el momento de registrar un ataque ejecutan de forma inmediata acciones para

detenerlo, en ocasiones representan algún tipo de errores de registros en el momento especial que en al momento de ejecutar alguna tarea diferente a lo que hace durante un día normal de trabajo.

Lo expuesto anteriormente ha llevado al planteamiento del siguiente interrogante:

- ¿Qué tan efectivas son las técnicas **selección de características, métricas de aprendizaje y reducción de dimensión para la detección de intrusos en la redes**?

2.2 MARCO TEÓRICO

2.2.1 SEGURIDAD INFORMATICA

2.2.1.1 ¿QUÉ ES LA SEGURIDAD INFORMÁTICA?

Para hablar de seguridad informática primero debemos hablar de la definición de cada uno de los términos que conforman dicha frase. El termino seguridad se define como: Cualidad de seguro (http://www.rae.es/drae/srv/search?id=wwcs1Hw7LDXX28FVsWTI), siendo "seguro" el adjetivo que se define como: "Libre y exento de todo peligro, daño o riesgo" (http://www.rae.es/drae/srv/search?id=hUBFUTTeqDXX2RfCAYIz).

Por otro lado el termino informática es definido como: "Conjunto de conocimientos científicos y técnicas que hacen posible el tratamiento automático de la información por medio de ordenadores" (http://lema.rae.es/drae/srv/search?key=inform%C3%A1tica). Teniendo claro los términos que engloban el término seguridad informática, pasamos ahora a esclarecer que es lo que se pretende hacer con el conjunto de la definición de ambos términos, Seguridad e Informática. Hoy día la seguridad informática lo que pretende escudar es la información, siendo esta última la "agregación de datos que tiene un significado más allá de cada uno de éstos", y tendrá un sentido particular según cómo y quién la procese. Por ejemplo: 1, 9, 8, 0, son datos pero si lo vemos como 1980 será una fecha por lo que los datos se convierten en información.

El valor que puede tener la información es relativo y en muchos casos se le da una valoración subjetiva u objetiva debido a su alto grado de intangibilidad, algo que no suele ocurrir con las aplicaciones, los edificios, los equipos de cómputo o la documentación física.

La información puede ser pública o privada, publica como la cantidad de estudiantes de una facultad determinada en cualquier, o el porcentaje de alumnos de sexo masculino o femenino de la misma, pero la privada debe ser resguardada como es el caso de las notas finales de cada uno de los estudiantes por lo que se debe hacer lo posible para preservar dicha confidencialidad, reconociendo características como:

1. La información debe ser Crítica para garantizar la continuidad operativa.
2. Valiosa ya que es un activo con valor en sí misma y,
3. Sensitiva ya que solo debe ser conocida por las personas que tienen acceso a procesarla.

Teniendo en cuenta lo anterior podemos hablar de cualidades como:

- La **Integridad**, característica que hace que la información sea inalterada a menos que está sea modificada por personal autorizado y dicha modificación sea registrada para posteriores controles o auditorias.

- El **Aislamiento** y la **confidencialidad** se relacionan mucho ya que aislar la información se traduce en la capacidad de regular el acceso al sistema impidiendo que personas no autorizadas hagan uso del mismo.

- La **Disponibilidad** hace referencia a la posibilidad de estar siempre útil para aquellos que tienen acceso autorizado a ella, haciendo que la información se mantenga correctamente almacenada con el hardware y el software funcionando en perfecto estado y respetando los formatos para su recuperación.

- La **Autenticidad** permite decidir si la información es válida y utilizable en tiempo, forma y distribución. Esta cualidad permite también asegurar el origen de la información validando el emisor de la misma.

- **Protección a la réplica** es la característica de la información a asegurar que la información solo puede realizarse una vez, a menos que se especifique lo contrario.

- **No repudio** es la manera como se verifica que una entidad que recibió o envío la información diga posteriormente que no la envío o recibió.

- La **Consistencia** es la característica de poder asegurar que el sistema se comporta de la manera como se supone que debe hacerlo ante los usuarios que corresponda ya tengan privilegios o no.

- La **Privacidad** es la característica de la información a ser conocida solo por el personal autorizado para ello. La falta de confidencialidad hace que la información se vuelva obsoleta o provoque daños graves al propietario.

- El **Control** es la capacidad que se da sobre la información a aquellos usuarios autorizados de decidir cómo y cuándo se permite el acceso a la misma.

2.2.1.2 ¿QUÉ SE DEBE PROTEGER EN UN SISTEMA INFORMÁTICO?

Un sistema informático está compuesto por tres elementos, estos son: el hardware, el software y los datos. Teniendo en cuenta lo anterior se debe dejar claro que como sistema se deben proteger los tres, y los ataques que se pueden presentar al sistema se presentarán en igualdad de condiciones a cualquiera de los tres ya que si uno llegase a faltar el sistema se verá comprometido en su funcionamiento.

Los ataques que pueden adolecer estos tres elementos descritos anteriormente se pueden catalogar en dos grandes ramas, los activos y los pasivos. Los Activos hacen referencia a la modificación del flujo de datos que se trasmite o la incorporación de datos falsos en dicho flujo mientras que los Pasivos hacen referencia a aquellos que no modifican la información y su intención es la revisión del tráfico de datos que se encauza por el canal de comunicación con finalidad positiva o negativa dependiendo del escucha.

2.2.1.3 SEGURIDAD, ATAQUES Y VULNERABILIDADES

De acuerdo a (Brumley, D., Newsome, J., Song, D., Hao Wang & Jha, S., 2006) Los protocolos de comunicación se dan por el cambio de información entre dos o más personas. Un ejemplo donde se puede evidenciar claramente esta definición es la relación que se establece entre un cliente y un servidor que intercambian información a través de una red. Una vulnerabilidad compromete la seguridad de un sistema; generalmente se producen por la existencia de fallos de programación en el código de la aplicación o pueden deberse a faltas en la configuración del servicio que se está prestando. De acuerdo a lo anteriormente mencionado la seguridad en los sistemas informáticos no solo son responsabilidad de los dispositivos adquiridos también a los desarrolladores como los administradores de software, siendo estos los encargados de optimizar las aplicaciones para el desempeño de las mismas con el propósito de poder detectar de manera oportuna las vulnerabilidades con miras a corregirlo y evitar ser objeto de ataques potenciales.

Los intrusos se clasifican con base en la magnitud del daño que causan al sistema atacado de acuerdo a lo mencionado por (Huerta Antonio, 2002):

- Los IC (Intrusos curiosos), se refiere a los usuarios que no tienen conocimiento previo de informática pero sin embargo intentan a acceder a sistemas a los cuales no tienen acceso y no causan grandes perjuicios al sistema solo se encargan de hacer procesos de espionaje.
- Los Crackers, se refiere a los usuarios que tienen cierto conocimiento de informática, dichos usuarios usan los sistema de manera ilegal para ejecutar programas maliciosos con el propósito de robar información para su beneficio.
- Los IR (Intrusos Remunerados) se refiere a los usuarios catalogados los de más peligro ya que cuentan con una gran experiencia en temas relacionados con seguridad y un amplio dominio del sistema, el cual utilizan para iniciar acciones maliciosas con fines delictivos.

2.2.1.4 ATAQUES A SISTEMAS INFORMÁTICOS

Los ataques informáticos son intrusiones ilegales a la seguridad de un sistema, siendo una intrusión la materialización de una amenaza. Autores como (Heady, R., Luger, G., Maccabe, A., & Servilla, M., 1990 **)** definen intrusión como cualquier conjunto de acciones que tratan de comprometer la integridad, confidencialidad o disponibilidad de un recurso. Por otro lado (Powell, D. & Stroud, R., 2001) emite una de las definiciones más populares de intrusión es: fallo operacional maligno, inducido externamente, aunque es bien sabido que muchas de las intrusiones proceden del interior del sistema de información.

La gran parte de los accesos mal intencionados son realizadas a través de los puertos de las computadoras destinos. Estos ataques o intrusiones son realizados con programas que se encargan de escanear los puertos abiertos de la computadora. Esta técnica de exploración tiene como finalidad hallar los servicios que están disponibles o ofrecidos por una red o servidor, para realizar conexiones o intentos de conexión a diferentes puertos (TCP o UDP) en la computadora de la

víctima con el propósito de obtener respuesta de alguno o algunos de ellos e inferir qué aplicación o servicio está activo en dicho puerto.

Los puertos de una computadora pueden encontrarse en varios estados: abierto, cerrado o bloqueado. El estado más vulnerable a un ataque es cuando el puerto se encuentra abierto, esto significa que una aplicación servidor está escuchando por ese puerto las peticiones de los clientes que se conecten. Un puerto abierto puede brindar información sobre las vulnerabilidades de seguridad del sistema. Por esta razón, una de las primeras actividades que un atacante es realizar un proceso de exploración de puertos por lo que se aconseja tener todos los puerto.

Existe una gran cantidad de tipos de escaneo (http://www.insecure.org/nm.ap.org) que pueden basarse en los protocolos TCP o UDP, diferenciándose básicamente por las banderas de protocolo utilizadas tales como SYN, ACK o RST aplicables sólo a TCP, pero los dos tipos de escaneo más comunes son el TCP Connect y el TCP SYN como se muestra en la Figura No 2. El primero utiliza el proceso de conexión convencional del protocolo TCP conocido como triple handshaking o saludo triple, llamado así por los tres mensajes que se intercambian al inicio de una nueva conexión (SYN, SYN_ACK y ACK). El escaneo TCP SYN o semi-abierto no establece una conexión por cada puerto, es más rápido y difícil de detectar.

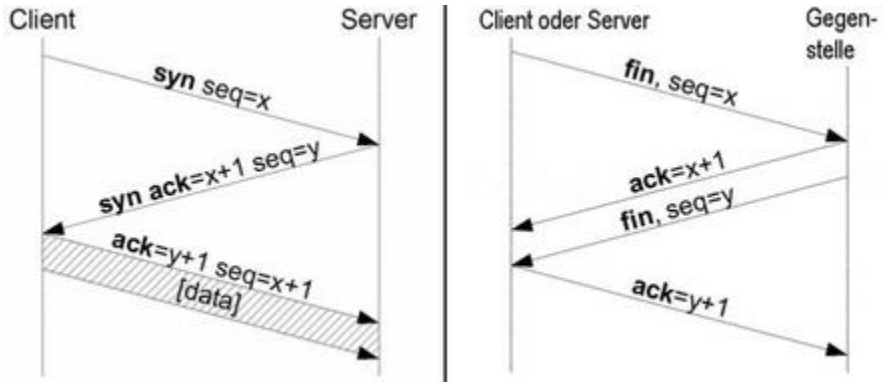

Grafico No 1: Tipos de escaneo más comunes: TCP Connect y TCP SYN

(http://www.internet-sicherheit.de/en/research/recent-projects/internet-early-warning-systems/internet-analysis-system/recent-results/)

A continuación se mencionan algunos de los ataques informáticos más usados:

- **Spoofing:** Este tipo de ataque se caracteriza por la elaboración de tramas TCP/IP utilizando una dirección IP falseada. El objeto de dicho ataque es que el atacante simule la identidad de otra máquina en una red de datos con el propósito de adquirir acceso a recursos de un tercer sistema con el que se ha podido llegar a tener confianza basándose en el nombre o la dirección IP de la maquina suplantada.(Guo, F., Chen, J., & Chiueh, T., 2006)
- **Negación de servicio**: Este ataque es también conocido por sus siglas en inglés como DOS y en su gran mayoría están dirigidos contra un recurso informático, ya sea una máquina o red. En este ataque el o los atacantes tratan de limitar de una manera parcial o total a usuarios legítimos el acceso a servicios prestados por un recurso informático. Esta característica hace que este ataque se convierta en uno de los ataques más sencillos y contundentes contra todo tipo de servicios convirtiéndose en un serio problema, ya que un delincuente informático puede interrumpir constantemente un servicio sin necesidad de grandes conocimientos o recursos, utilizando programas sencillos o con la ayuda de una gran masa de usuarios ingenuos al ataque. (Kumar, S., 1995)

Se puede lograr un ataque aún más agresivo si se combinan dos o más técnicas de ataques como es el caso reportado por Computer Wire donde se utilizó la técnica de spoofing de direcciones IP (Guo, F., Chen, J., & Chiueh, T., 2006) y un ataque de negación de servicio.

- La **interceptación:** conocida en el medio de la informática y en el ámbito de la seguridad como Passive Wiretapping, es un ataque que hace alusión al procedimiento en el cual un agente es capaz de tomar información ya sea encriptado o no encriptado la cual no se encontraba disponible para él. Un intruso puede capturar una cantidad de información privilegiada y claves para ser utilizadas más adelante haciendo que este ataque sea considerado de pasivo a activo y uno de los más peligrosos ya que no es detectable sino hasta que se activa.

La interceptación puede ser implementada utilizando software llamados Sniffers, los cuales se encargan de hacer un espionaje a la red o tomar las tramas que circulan por ella de una o todas las computadoras conectada a ella.

- Los **Ataques a aplicaciones** se deben a la potencialidad de datos que pueden ser adquiridos en las redes de datos por la estructura física y lógica de las mismas. Como una red se compone de equipos de cómputo aparte de servidores de correo, web, ftp entre otros, los delincuentes informáticos tratan en lo posible de atacar la red en sí y apoderarse del tráfico que es introducido en ellas, más aun si pueden acceder a alguno de los servidores que en ella radican. Entre los ataques más comunes que podemos encontrar a los elementos de una red citamos los ataques al correo electrónico, a los servidores Webs, aquellos ataques que se realizan mediante conexión remota mediante el protocolo SSH (Ylonen, T., 1996), diccionarios de datos (Feldmeier, D., & Philip, K.) o programas maliciosos como los troyanos (Wu, N., Qian, Y., & Chen, G., 2006) y virus informáticos (Harrald, J., Schmitt, S., & Shrestha, S., 2004).

2.2.1.5 MECANISMOS DE PREVENCIÓN

Debido al incremento de los ataques informáticos ocurridos en actualmente se han desarrollado mecanismos para prevenir o tratar en lo posible que dichos ataques no produzcan el daño deseado por los atacantes o en su defecto no produzcan daño alguno. Tales mecanismos previenen la ocurrencia de violaciones a la seguridad; por ejemplo, el uso de cifrado en la transmisión de datos se puede considerar un mecanismo de este tipo, ya que evita que un posible atacante escuche las conexiones hacia o desde un sistema en la red.

Entre los mecanismos más habituales de prevención en las redes de datos de hoy día podemos encontrar la **autenticación e identificación** ya que estos hacen posible identificar entidades del sistema de una forma única, y después de ser identificadas, autenticarlas. Se les cataloga como los mecanismos de primera línea de todo sistema informático. (Olovsson, T., 1992).

Entre las practicas que se pueden utilizar para la autenticación de usuarios se pueden reconocer: la comúnmente conocida por contraseña que promueve que únicamente el usuario tiene y es su deber y responsabilidad mantener seguro el su sistema gracias al par Usuario-Contraseña que se le ha asignado; la segunda practica implica la posesión de un objeto físico que el usuario legitimo posea, como una tarjeta inteligente o credencial que ofrece funciones para un almacenamiento seguro de información y también para su procesamiento (Everett, D., 1992).

En una segunda enfoque encontramos a los **mecanismos de control de acceso** que se han establecido para controlar todos los tipos de acceso sobre un objeto en particular de cualquier entidad del sistema. Otro de los mecanismos utilizados con el ánimo de separar de la manera más segura posible una maquina o subred del resto es el **cortafuegos**, éste se encarga de proteger los servicios y protocolos que desde el exterior puedan suponer una amenaza a la seguridad.

Para garantizar que las comunicaciones de datos por las redes privadas o públicas sean confiables, se utiliza hoy día la Criptografía (Huerta Antonio, 2002), cifrados de clave pública, privada, firmas digitales, etc. Aunque cada vez se utilizan más los protocolos seguros como SSH o Kerberos **(**Kohl, J., Neuman, B., & Ts'o, T., 1994**)** como es el caso de sistemas Unix en red. Es un común denominador que también se presenten en gran parte de las redes actuales mucho tráfico sin cifrar haciendo que los ataques encaminados a robar contraseñas o suplantar la identidad de máquinas de la red se produzcan con más frecuencia.

2.2.1.6 MECANISMOS DE DETECCIÓN

Los mecanismos de detección son aquellos utilizados para detectar violaciones de la seguridad o cualquier tipo de intento que pueda comprometer el óptimo desempeño del sistema. Dentro de los mecanismos de detección se encuentra una herramienta que será explorará en más adelante con más detalle ya que hace parte del núcleo base de este proyecto de investigación, hablamos de los Sistemas de Detección de Intrusos. Este tipo de sistemas son los encargados de supervisar y registrar toda actividad de un sistema para su análisis en busca de alguna actividad maliciosa y dar así la respuesta más apropiada a dicha actividad. Entre los Sistemas de Detección de disponibles

podemos nombrar: OSSEC (Daniel, B., 2006), Tripwire (Chet, H., & Duren, M., 1998), Snort (Martin, R., 2005), RealSecure (Dain, O., & Cunningham, R., 2001), Prelude (Girardin, L., 1999), entre otros.

2.2.1.7 MECANISMOS DE RECUPERACIÓN

Existen mecanismos que ayudan a recuperar aquellas anomalías presentadas después de un ataque o violación de un sistema. La idea que se piensa conseguir con estos mecanismos es poder regresar al sistema informático a su funcionamiento normal. Entre los elementos que podemos mencionar de estos mecanismos son: los antivirus, hardware adicional o mecanismos de detección que incluyan software para la recuperación de sistemas a su estado inicial.

Teniendo en cuenta los tres mecanismos mencionados anteriormente se puede concluir que los sistemas informáticos hoy día están propensos a múltiples ataques y vulnerabilidades posibles ya que la racha de delincuentes informáticos va en aumento y la libertad de información acompañando de políticas de seguridad no planificadas ni llevadas a la práctica de la manera más eficaz y eficiente posible permiten a estos delincuentes atentar contra cualquier sistema que no cumpla con los requerimientos mínimos de protección aunque como se mencionó en apartados anteriores ningún sistema es 100% seguro, lo más que podemos decir de él es que es confiable de que se comporte como se espera hasta que algún ataque no contemplado por los fabricantes o el administrador del sistema luego de su afinamiento logre fragmentar la barrera de seguridad que estos le proveen desde su grado de inteligencia. En consecuencia de esto se expone a continuación el tema de los Sistemas de Detección de Intrusos IDS.

2.2.1.8 SISTEMAS DE DETECCIÓN DE INTRUSOS IDS

En la búsqueda de mejorar la complejidad de la auditoría y la habilidad para la vigilancia de sistemas informáticos James P. Anderson en los años de 1980 empieza como un trabajo de consultoría realizado para el gobierno de los Estados Unidos, los principios de los IDS (Sistemas de Detección de Intrusos) introduciendo el término “amenaza” en la seguridad informática definiéndolo como la potencial posibilidad de un intento deliberado de acceso a información, manipulación de la misma, o hacer que un sistema sea inutilizable. (Anderson, J., 1980)

Con el pasar del tiempo se empezaron a acuñar términos referentes a la seguridad informática como es el caso del término “intrusión”. Finalmente es el NIST (National Institute of Standards and Technology) quien define la detección de intrusos como el proceso de monitorización de eventos que suceden en un sistema informático o red y análisis de dichos eventos en busca de signos de intrusiones.

Los SDI supervisan y registran los eventos que ocurren en una computadora o en una red de computadoras, para analizarlos y correlacionarlos en la búsqueda de señales de intrusión y dar así una respuesta que permita corregir esta situación (Siraj, A., Vaughn, R. & Bridges, S., 2004), además de la ocurrencia de malas prácticas, como en el caso de los usuarios autorizados que intentan sobrepasar sus límites de restricción de acceso a la información (Wu, S., & Banzhaf, W., 2010), con el ánimo de poder dar con los responsables del ataque y tomar acciones conducentes a mejorar la vulnerabilidad y castigar si se puede a él o a los responsables de dicho ataque.

Además, un IDS dispone de los medios para manejar alertas cuando se detectan signos de intrusión que permitan tomar las medidas correspondientes en el menor tiempo posible. Es por ello que los IDS como se les conoce en el argot informático han ganado terreno en la mayoría de organizaciones que busca darle un poco más de seguridad a sus sistemas informáticos. Los IDS independientemente que sistema vigilen o como es su forma de trabajo, generalmente cumplen las siguientes características:

- Minimizar el consumo de recursos.
- Permitir aplicar una configuración según las políticas de seguridad que dicte la organización.
- Adaptabilidad a los cambios vertiginosos que sufren los sistemas y los usuarios. Además de incluir un proceso rápido y sencillo de actualización.
- Ejecución continúa de forma transparente, y con un mínimo de supervisión.
- Poder tolerar fallos y recuperarse de ellos aun si existiesen fallos de red.
- En caso de verse comprometido alguno de sus componentes intentar recuperar dicho componente, y en caso contrario administrar un tipo de alerta.

2.2.1.9 EFICIENCIA DE LOS IDS

Para medir la eficiencia de un IDS a la hora de detectar intrusiones en un sistema determinado se usan las siguientes características:

La **Precisión**, hace referencia a la capacidad que tiene un sistema de detección de intrusos tara identificar ataques y distinguirlos del tráfico normal de la red. Para medir la precisión de un IDS se utilizan dos criterios: el porcentaje de falsos positivos, es decir el número de veces que se detecta un ataque que no existe, y el porcentaje de falsos negativos, es decir el número de ataques no detectados.

El **Rendimiento** hace referencia al número de eventos que es capaz de analizar un sistema. Lo ideal en este caso es que analice el tráfico de la red en tiempo real aunque esto es altamente difícil de hacer si se quiere analizar a fondo el contenido de los paquetes. El rendimiento de un sistema depende de la capacidad de procesamiento hardware de la que se disponga. Cada tipo de IDS debe alcanzar un equilibrio entre la cantidad de paquetes que debe analizar y la profundidad de este análisis.

La **Completitud** de un IDS se cumple cuando es capaz de detectar todos los tipos de ataques. Por lo general, los sistemas detectan solo algunos tipos y es necesario combinar varias técnicas para conseguir la completitud. La completitud de un sistema debe equilibrarse con la precisión, es decir, poder detectar la mayor parte de los ataques sin que el número de falsas alarmas crezca demasiado.

La **Tolerancia a fallos** da al IDS la propiedad de resistir a los ataques. Un IDS debe ser seguro y robusto para evitar que un ataque lo inutilice dejando todo el sistema vulnerable. Además de resistir a los ataques, esta propiedad determina la capacidad del IDS a resistir un fallo del sistema, por ejemplo un corte de luz o la destrucción del terminal donde se ejecuta. Es importante que un IDS guarde los patrones que utiliza para la detección y los “logs” que ha ido recogiendo con su ejecución.

El **Tiempo de respuesta** es el tiempo que tarda en reaccionar ante un ataque ya sea lanzando una alarma o poniendo medidas para detener o retardar la intrusión. Obviamente, cuanto menor sea el

tiempo de respuesta más efectivo será el sistema, aunque debe asegurarse de que existe realmente un ataque antes de actuar.

2.2.1.10 CLASIFICACIÓN DE LOS IDS

Desde las primeras investigaciones realizadas en los años de 1980 con (Anderson, J., 2010) hasta la aparición de sistemas expertos capaces de detectar las desviaciones a partir del comportamiento de diferentes sujetos en el Instituto de Investigación SRI International como lo menciona (Lunt, T., 1990) y (Lunt, T., & Jagannathan, R., 1988) donde se han tratado de buscar mejores prácticas que hagan uso de diferentes técnicas y algoritmos para analizar el comportamiento de los sistemas informáticos en miras de darles mayor protección contra ataques de los ya mencionados delincuentes informáticos.

La categorización de los Sistemas de Detección de Intrusos (IDS) ha sido tratada por varios autores, entre ellos podemos mencionar Hervé Debar (Debar, H., Dacier, M., & Wespi, A., 1999), Stefan Axelsson (Axelsson, S., 2000), así como (Wu, S. , Banzhaf, W., 2010) quienes los clasificaron según la fuente de datos, el análisis y la estrategia de respuesta.

Los IDS por **fuentes de información** hacen referencia a la fuente u origen de los datos que se utilizan en el análisis para determinar si una intrusión se ha llevado a cabo. Aquellos que utilizan el **análisis** hacen alusión al método de detección utilizado como estrategia. Y por último los que utilizan **mecanismos de respuesta** son los que una vez se ha determinado si ha sucedido alguna intrusión, pueden o bien responder de forma activa ante la misma, o bien registrar la detección y no realizar acción alguna.

(Huerta Antonio, 2002) clasifico a los Sistemas de Detección de Intrusos IDS en función de a que sistemas vigilan o bien en función de cómo lo hacen (IDS basados en Red, IDS basados en Maquina)

"IDS *basados en red*: *monitoriza los paquetes que circulan por nuestra red en busca de elementos que denoten un ataque contra alguno de los sistemas ubicados en ella; el IDS puede situarse en*

cualquiera de los hosts o en un elemento que analice todo el tráfico (como un HUB o un enrutador). Este donde este, monitorizara diversas máquinas y no una sola: esta es la principal diferencia con los sistemas de detección de intrusos basados en host.

IDS basados en maquina: *Mientras que los sistemas de detección de intrusos basados en red operan bajo todo un dominio de colisión, los basados en maquina realizan su función protegiendo un único sistema; de una forma similar guardando las distancias, por supuesto a cómo actúa un escudo antivirus residente en MS-DOS, el IDS es un proceso que trabaja en background (o que despierta periódicamente) buscando patrones que puedan denotar un intento de intrusión y alertando o tomando las medidas oportunas en caso de que uno de estos intentos sea detectado.*

Tradicionalmente, los modelos de detección basados en maquina han consistido por una parte en la utilización de herramientas automáticas de análisis de logs generados por diferentes aplicaciones o por el propio kernel del sistema operativo, prestando siempre especial atención a los registros relativos a demonios de red, como un servidor web o el propio inetd, y por otra quizás no tan habitual como la anterior en el uso de verificadores de integridad de determinados ficheros vitales para el sistema, como el de contraseñas ".

2.2.2 TECNICAS DE SELECCIÓN DE CARACTERISTICAS

2.2.2.1 ¿A QUE SE REFIERE SELECCIÓN DE CARACTERISTICAS?

La selección de características se refiere a un concepto utilizado en minería de datos con el objetivo de reducir el tamaño de los datos de entrada para facilitar el procesamiento y análisis de dicha información. La selección de características no solo tiene en cuenta la disminución de cardinalidad, es decir, mantener un límite parcial o predefinido en la cantidad de atributos tenidos en cuenta al crear un modelo, sino también la elección de atributos, lo que permite descartar de forma adecuada los atributos en función de la utilidad para la realización de un buen proceso de análisis.

La capacidad de emplear la selección de características en primordial para realizar un análisis eficaz, debido a que los datos contienen información que no es necesaria para la generación del modelo. Por ejemplo, un conjunto de datos puede contener 41 columnas que describen las características de conexiones de redes de computadores, pero si los datos de algunas de ellas están muy dispersos, no obtendrá muchas ventajas al agregarlas al modelo. Si mantiene las columnas innecesarias durante la generación del modelo, se necesitarán más cantidad de procesamiento y memoria durante el proceso de entrenamiento, así como más espacio de almacenamiento para el modelo completado.

Aunque la capacidad de cómputo no represente un problema, se deben quitar las columnas innecesarias porque pueden degradar la calidad de los patrones detectados por las razones siguientes:

- *"Algunas columnas son ruidosas o redundantes. Este ruido dificulta la detección de patrones significativos a partir de los datos.*
- *Para detectar patrones de calidad, la mayoría de los algoritmos de minería de datos requieren un conjunto de datos de entrenamiento mucho más grande en un conjunto de datos multidimensional. Sin embargo, en algunas aplicaciones de minería de datos se dispone de muy pocos datos de entrenamiento."(*http://msdn.microsoft.com/es-es/library/ms175382.aspx*).*

Un típico proceso de Selección de Características se ejecuta en 4 pasos (Oporto, S., Aquino, I., Chavez, J., & Perez, C.):

- **Generación del subconjunto**, hace referencia al proceso de búsqueda que obtiene como resultado un subconjunto de características disponibles para ser evaluadas. Además se menciona que la búsqueda completa avala el hallazgo del subconjunto óptimo, sin necesidad de realizar una búsqueda de todos los posibles subconjuntos, que es una búsqueda exhaustiva (Goldberg, D., 1989), mientras que la búsqueda secuencial genera subconjuntos de forma directa, iniciando con un subconjunto vacío, para luego adicionar características importantes de manera gradual (Liu, H., & Motorola, H., 1998) y La búsqueda aleatoria genera subconjuntos de manera aleatoria, luego aumenta o disminuye características también aleatoriamente para generar el siguiente subconjunto que será evaluado.

- **Evaluación del subconjunto**, tiene como propósito verificar el nivel de la optimalidad del subconjunto generado anterior para su posterior implementación de un problema de aprendizaje, que en este caso es de Clasificación. El criterio de evaluación es independiente del algoritmo de aprendizaje (ejemplo: Redes Neuronales, Maquinas de Vector Soporte

- **Criterio de paro**, determina cuando un proceso de selección de características debería detenerse. Generalmente el proceso de Selección de características se termina bajo tres criterios: 1. cuando se llaga a un valor o umbral establecido; 2. Cuando se terminó de realizar la búsqueda completa; 3. Cuando se encontró un subconjunto de características óptimo.

- **Validación de resultados**, es la evaluación del Modelo de Selección de Características con datos reales para verificar si el desempeño obtenido refleja el resultados deseados de los experimentos.

A continuación se mencionan algunas técnicas que permiten realizar Selección de características:

- Bayes naive
- Árboles de decisión
- Red neuronal
- Regresión logística
- Agrupación en clústeres
- Regresión lineal

2.2.2.2 FEATURE SELECTION TOOLBOX (FEAST)

FEAST proporciona la implementación de algoritmos de selección de características basándose en información común, y una implementación de RELIEF. Todas las funciones esperan entradas discretas (con excepción de RELIEF, que no depende de la MIToolbox) y devuelven los índices de función seleccionada. Todo el código FEAST está disponible bajo la Licencia BSD.

FEAST contiene implementación de métodos como**:** mim, mrmr, mifs, cmim, jmi, disr, cife, icap, condred, cmi, relief, fcbf, betagamma (http://www.cs.man.ac.uk/~gbrown/fstoolbox/).

- **MIM**: Mutual Information Maximisation Lewis (1992)
- **MIFS**: Mutual Information Feature Selection Battiti (1994)
- **JMI**: Joint Mutual Information Yang and Moody (1999)
- **FCBF** : Fast Correlation Based Filter Yu and Liu (2004)
- **CMIM** : Conditional Mutual Info Maximisation Fleuret (2004)
- **MRMR**: Max-Relevance Min-Redundancy Peng et al. (2005)
- **ICAP**: Interaction Capping Jakulin (2005)
- **CIFE**: Conditional Infomax Feature Extraction Lin and Tang (2006)
- **DISR**: Double Input Symmetrical Relevance Meyer and Bontempi (2006)

2.2.3 TECNICAS DE REDUCCION DE DIMENSION Y METRICAS DE APRENDIZAJE

La Toolbox de Matlab para la Reducción de dimensionalidad contiene implementaciones de 34 técnicas de reducción de dimensionalidad y métricas de aprendizaje. Un gran número de implementaciones se desarrolló a partir de cero, mientras que otras implementaciones son versiones mejoradas del software que ya estaba disponible en la Web. Las implementaciones en la Toolbox son conservadores en el uso de la memoria.

Matlab Toolbox for Dimensionality Reduction (http://homepage.tudelft.nl/19j49/Matlab_Toolbox_for_Dimensionality_Reduction.html), contiene la implementación de las siguientes técnicas:

No	TECNICAS
1	Principal Component Analysis (PCA)
2	Probabilistic PCA
3	Factor Analysis (FA)
4	Classical multidimensional scaling (MDS)
5	Sammon mapping
6	Linear Discriminant Analysis (LDA)
7	Isomap
8	Landmark Isomap
9	Local Linear Embedding (LLE)
10	Laplacian Eigenmaps
11	Hessian LLE
12	Local Tangent Space Alignment (LTSA)
13	Conformal Eigenmaps **(34xtensión of LLE)**
14	Maximum Variance Unfolding (34xtensión of LLE)
15	Landmark MVU (LandmarkMVU)
16	Fast Maximum Variance Unfolding (FastMVU)
17	Kernel PCA
18	Generalized Discriminant Analysis (GDA)
19	Diffusion maps
20	Neighborhood Preserving Embedding (NPE)
21	Locality Preserving Projection (LPP)
22	Linear Local Tangent Space Alignment (LLTSA)
23	Stochastic Proximity Embedding (SPE)
24	Deep autoencoders **(using denoising autoencoder pretraining)**
25	Local Linear Coordination (LLC)
26	Manifold charting
27	Coordinated Factor Analysis (CFA)
28	Gaussian Process Latent Variable Model (GPLVM)
29	Stochastic Neighbor Embedding (SNE)
30	Symmetric SNE

31	t-Distributed Stochastic Neighbor Embedding (t-SNE)
32	Neighborhood Components Analysis (NCA)
33	Maximally Collapsing Metric Learning (MCML)
34	Large-Margin Nearest Neighbor (LMNN)

Tabla No 1. Técnicas de Reducción de Dimensión y Métricas de Aprendizaje

Para el desarrollo de esta investigación se tomó en cuenta como técnica de reducción de dimensión Principal Component Analysis (**PCA**), KERNEL PCA (**KPCA**) y Isometric Feature Mapping **ISOMAP**, además se tomó como técnica de métrica de aprendizaje la implementación de máquinas de soporte vectorial (**SVM**).

PRINCIPAL COMPONENT ANALYSIS (PCA)

Es una manera de la identificación de patrones en los datos, expresados los datos de tal manera para resaltar sus similitudes y diferencias. Dado que los patrones en los datos pueden ser difíciles de encontrar a causa de la alta dimensión de dichos datos, donde la representación gráfica no está disponible, el PCA es una herramienta poderosa para el análisis de datos. La otra ventaja principal del PCA es que una vez que haya encontrado estos patrones en el de datos, estos son comprimidos, es decir. Mediante la reducción del número de dimensiones, sin mucha pérdida de información. Esta técnica se utiliza en la compresión de imágenes.

También podemos mencionar que PCA es un procedimiento de reducción variable. Útil cuando se tiene obtenido datos sobre un número de variables (posiblemente un gran número de variables), y creer que hay una cierta redundancia en esas variables. En este caso, la redundancia significa que algunos de los variables están correlacionados entre sí, posiblemente debido a que están midiendo el mismo construir. A causa de esta redundancia, usted cree que debería ser posible reducir el variables observadas en un número menor de componentes principales (variables artificiales) que se cuenta para la mayor parte de la varianza en las variables observadas.

KERNEL PRINCIPAL COMPONENT ANALYSIS (KPCA)

El análisis de componentes principales Kernel (KPCA) (Scholk, B., Smola, A., & Muller, K., 1998)(Scholk, B., Smola, A., & Muller, K., 1999) es una técnica no lineal para la extracción de características, estrechamente relacionados con los métodos aplicados en Maquinas de Soporte Vectorial. Ha demostrado ser útil para varias aplicaciones, tales como eliminación de ruido (Mika, S., Scholk, B., Smola, A., Muller, K., Scholz, M., & Ratsch, G., 1999) y como un pre-procesamiento intervenir en los problemas de regresión (Rosipal, R., Girolami, M., & Trejo, L., 2000).

De acuerdo a lo mencionado por (Fauvel, M., Chanussot, J., & Benediktsson, J.) el modelo matemático dado para KPCA es:

$$\phi(\boldsymbol{x})\,{}^{\boldsymbol{k}}_{\boldsymbol{kpc}} = \sum_{\boldsymbol{i=1}}^{\boldsymbol{N}} \boldsymbol{\alpha_i^k}\big(\phi(\boldsymbol{x_i}).\phi(\boldsymbol{x})\big)$$

ISOMETRIC FEATURE MAPPING (ISOMAP)

Desarrollado por Tanenbaum, de Silva, y Langford, es una mejora del MDS clásico. Es un método no lineal para la reducción de dimensión. Busca el mapa que conserva la, geometría no lineal mundial de los datos mediante la preservación de las geodésicas (curva más corta a lo largo del colector que conecta dos puntos) En primer lugar se aproxima a las distancias entre puntos geodésicos, a continuación, ejecuta MDS (Multidimensional Scaling) para encontrar la proyección que conserva estas distancias.

MAQUINAS DE SOPORTE VECTORIAL SVM

Es una técnica de clasificación de última generación que ha tomado mucha relevancia en años recientes (Burges, C., Schölkopf, B., & Smola, A., 1999) (Burges, C.,1998). SVM se basa en la idea de minimización de riego estructural (Vapnik, V., 1995). En muchas aplicaciones, las SVM han mostrado tener resultados adecuados, superando las máquinas de aprendizaje como las redes neuronales (Burges, C., 1998) y han sido tomadas como herramientas eficaces para resolver problemas de clasificación.

Investigadores como (Betancourt, G., 2005). menciona "*una Máquina de Soporte Vectorial aprende la superficie decisión de dos clases distintas de los puntos de entrada. Como un clasificador de una sola clase, la descripción dada por los datos de los vectores de soporte es capaz de formar una frontera de decisión alrededor del dominio de los datos de aprendizaje con muy poco o ningún conocimiento de los datos fuera de esta frontera. Los datos son mapeados por medio de un kernel Gaussiano u otro tipo de kernel a un espacio de características en un espacio dimensional más alto, donde se busca la máxima separación entre clases. Esta función de frontera, cuando es traída de regreso al espacio de entrada, puede separar los datos en todas las clases distintas, cada una formando un agrupamiento*".

Otros autores como Jose Vargas definen a Máquinas de Soporte Vectorial como "*un método de clasificación supervisada. SVM emplea un algoritmo de optimización para determinar la frontera óptima entre dos grupos; se puede generalizar para múltiples grupos. El caso más simple es el linealmente separable, donde existe una distancia positiva entre ambos grupos y es posible elegir un hiperplano de separación maximizando la distancia de éste a cada grupo. Para decidir a qué grupo pertenece una nueva observación, se considera el signo de la función que define al hiperplano de separación*" (Vargas, J., Conde, B., Paccapelo, V., Zingaretti, L.).

2.2.4 DESCRIPCION DATASET NSL-KDD

NSL-KDD es un conjunto de datos para resolver algunos de los problemas inherentes de la KDD'99. Aunque, esta nueva versión de los datos KDD establecidos todavía sufre de algunos de los problemas y puede no ser un representante perfecta de las redes reales existentes, debido a la falta de conjuntos de datos públicas para IDS basados en la red, que todavía se puede aplicar como un conjunto de datos de referencia eficaces establecidos para ayudar a los investigadores a comparar los diferentes métodos de detección de intrusos. Por otra parte, el número de registros de NSL-KDD y equipos de prueba son razonables. Esta ventaja hace que sea asequible para ejecutar los experimentos sobre el conjunto completo sin la necesidad de seleccionar al azar una pequeña porción. En consecuencia, los resultados de evaluación de los diferentes trabajos de investigación serán consistentes y comparables.

Desde 1999, KDD'99, ha sido el más utilizado conjunto de datos para la evaluación de los métodos de detección de anomalías. Este conjunto de datos se prepara por (Stolfo, S., Fan, W., Lee, W., Prodromidis, A., & Chan, P., 2000), y está construido sobre la base de los datos capturados en DARPA'98 evaluación IDS programa. DARPA'98 es un conjunto de datos producto de 7 semanas de tráfico de la red, que se puede procesar en alrededor de 5 millones de conexión registros, cada uno con cerca de 100 bytes. Las dos semanas de datos de prueba tienen alrededor de 2 millones de registros de conexión. KDD formación de datos se compone de aproximadamente 4.900.000 sola vectores de conexión cada uno de los cuales contienen 41 características y está etiquetado como normal o un ataque, con exactamente un tipo de ataque específico. Los ataques simulados caen en una de las siguientes cuatro categorías (Tribak, H., Febrero 2012):

- **Denegación de Servicio (DoS):** el propósito de estos ataques es detener el buen funcionamiento de una red de computadores, equipo que esté conectado a la red , detener un proceso activo dentro de la red de computadores, denegar el uso de los recursos o servicios a usuarios autorizados. Hay dos tipos de ataques DOS; por un lado ataques de sistema operativo, los cuales tienen como finalidad explotar los fallos en determinados sistemas operativos que pueden evitarse aplicando los respectivos parches; y ataques de red, que explotan limitaciones de los protocolos e infraestructuras de red. Entre los tipos de ataque más comunes clasificados como DOS encontramos: "mailbomb", "Neptune", "smurf", "teardrop", "apache2" , "back" .
- **Indagación o exploración (probing):** Este tipo de ataques monitorean las redes tratando de identificar direcciones IP válidas y recoger información acerca de ellas (servicios que ofrecen, sistemas operativos que usan). A menudo, esta información proporciona una lista de vulnerabilidades permitidas que podrían ser utilizadas para llevar a cabo ataques a los servicios y a las máquinas seleccionadas.
- **R2L (Remote to Local):** cuando un atacante que no dispone de cuenta alguna en una máquina, logra acceder (tanto como usuario o como root) a dicha máquina. En la mayoría de los ataques R2L, el atacante ingresa al sistema informático a través de Internet. Algunos ataques explotan el desbordamiento de búfer causado por el software de servidor de red

“imap, named, sendmail". Los ataques de " ftp_write, xsnoop y guest” tratan de explotar la debilidad o la mala configuración de las políticas de seguridad del sistema.

- **U2R (User to Root):** Este tipo de ataque se da cuando un atacante que dispone de una cuenta en un sistema informático es capaz de elevar sus privilegios explotando vulnerabilidades en los mismos, un agujero en el sistema operativo o en un programa instalado en el sistema.

A continuación se listan las 41 características presentes en cada registro de conexión del conjunto de datos NSL-KDD (Sabnani, S., Enero 2008):

No.	FEATURE NAME
1	duration
2	protocol
3	service
4	flag
5	Src bytes
6	Dst bytes
7	Land
8	wrong fragment
9	Urgent
10	Hot
11	num failed logins
12	logged in
13	num compromised
14	root Shell
15	su attempted
16	num root
17	num file creations
18	num shells
19	num access files
20	num outbound cmds
21	is host login
22	is guest login
23	Count
24	srv coun

25	serror rate
26	srv serror rate
27	rerror rate
28	srv rerror rate
29	same srv rate
30	diff srv rate
31	srv diff host rate
32	dst host count
33	dst host srv count
34	dst host same srv rate
35	dst host diff srv rate
36	dst host same src port rate
37	dst host srv diff host rate
38	dst host serror rate
39	dst host srv serror rate
40	dst host rerror rate
41	num outbound cmds
42	Class

Tabla No 2. Features DataSet NSL-KDD99

2.3 ESTADO DEL ARTE

El diseño incorrecto de una red de información puede impactar con consecuencias indeseables sobre el funcionamiento de los sistemas que la utilizan. Un diseño entendido especialmente como la forma de conectar los nodos de la red -topología- aunque también se podrían incluir otros aspectos como por ejemplo el tipo de enlace entre cada par de nodos. La topología está íntimamente ligada a las facilidades, posibilidades y restricciones de los servicios que corren sobre la red. Es por ello que resulta de sumo interés en el área contar con las herramientas y técnicas necesarias para poder llevar a cabo el diseño de una red en la forma más conveniente posible (Corbalan, L., 2006).

La seguridad de las redes de datos se ha transformado en un serio problema en los últimos años. El crecimiento vertiginoso que ha presentado la Internet ha permitido mostrar las fallas de seguridad en las implementaciones de los protocolos de red subyacentes. Situación que resulta comprensible, ya que muchos de estos protocolos originalmente fueron pensados para unir de 10 a 50 computadoras de las universidades de los Estados Unidos. Las falencias en la seguridad de protocolos como ARP, TCP, TELNET, SMTP, FTP han sido la causa de ataques contra la confidencialidad, la disponibilidad y la autenticidad de los datos transportados. Si bien estos problemas han sido corregidos a lo largo de los años, continuamente se van descubriendo nuevas maneras de realizar estos ataques. El ingeniero en seguridad de redes debe estar alerta para detectar estos ataques, informándose de las nuevas vulnerabilidades descubiertas o tipos de ataques perpetrados. Sin embargo una gran cantidad de estos ataques tienen lugar antes que se conozcan siquiera las vulnerabilidades o fallas que los provoca. Para afrontar este problema es que en los últimos años han surgido propuestas para la aplicación de técnicas de inteligencia artificial en el ámbito de la seguridad en redes (Catania, C., & Garcia, C., 2008).

La aplicación del conocimiento adquirido a través del estudio de las diferentes técnicas de inteligencia artificial (**Algoritmos Genéticos, Arboles de Decisión, Redes Neuronales Artificiales RNA**) han logrado poner en practica la efectividad de las mismas para la detección de intrusos en redes de computadores, muestra de ello es la implementación de algoritmos genéticos estudiada con anterioridad (M. Crosbie & G. Spafford, 1995) (Gong, R., Zulkernine, M., & Abolmaesmumi, P.,

2005)(Li, W.,2003)(Sinclair, C., Lyn, P., & Matzer, S., 1999). Gong, R., et al.(2005) y Li, W.(2003) exploran la utilización de algoritmos genéticos para la generación de reglas de clasificación en el tráfico de red, para lo cual buscan los patrones más comunes en las instancias de tráfico que presenten anomalías. Con este fin se necesita un conjunto de instancias de traficó que previamente haya sido analizado por un experto en seguridad de redes, en el cual se encuentre evidenciado el resultado de las anomalías.

Con el desarrollo de aplicaciones de Internet, las intrusiones en la red son cada vez más considerables. La detección de intrusiones es una parte muy importante para la seguridad de la red (Lippmann, R., & Cunningham, R., 2000) (Wuu, L., Hung, C., & Chen, S., 2007). Por lo tanto, es necesario estudiar la detección de intrusiones a través de métodos que aseguren una capacidad de respuesta temprana. En la actualidad, hay muchos métodos de detección de intrusos disponibles, como la red neuronal artificial que incluye elementos de procesos interconectados y transforman un conjunto de entradas a un conjunto de resultados deseados.

Una forma de mejorar la detección de intrusiones en la red, se basa en la utilización de técnicas de machine learning como Maquinas de Soporte Vectorial para la detección intrusiones de red. Las Máquinas soporte Vectorial (SVM) es un método de aprendizaje máquina que cuenta con un óptimo global de soluciones para problemas de clasificación no-lineales. Por lo tanto, la detección de Intrusiones en la red basada SVM se propone para reconocer los tipos de intrusión dentro de los dispositivos conectados a la red (Xiaoqing, G.)

Las técnicas de Machine Learning permiten la detección de anomalías y posibles nuevos ataques por parte de intrusos. La verificación de las anomalías a través de métodos de detección de actividades normales de un ordenador permite identificar comportamientos sospechosos tales como intrusiones.

Los Sistemas de Detección de Intrusiones (IDS) son capaces de detectar nuevos ataques, la mayoría de ellos sufren de un error debido a una deficiencia en su capacidad de discriminación.

Desde estos enfoques de detección sólo puede distinguir anomalías de los ataques, por ello se pierden muchos ataques que no son significativamente diferentes de los comportamientos normales.(Kuang, L.)

Ha habido un número de investigaciones sobre la viabilidad de aprendizaje de máquina para la detección de intrusiones (Hu, W., & Liao, Y., & Vemuri, V., 2003) (Sinclair, C., Pierce, L., & Matzner, S., 1999), la mayoría de los cuales han sido inherentemente alguna forma de detección de anomalías.

Según M. Canderle, G. Aguirre, F. Piccoli, Proteger a las redes y los usuarios de intrusos resulta una tarea prioritaria. Las tecnologías para la detección de Intrusos en una red son diseñadas para monitorear todas las actividades en la red y determinar las violaciones a la seguridad. Establecer que actividad se corresponde con una violación a la seguridad de una red, depende de la organización a la que pertenece la red, **no existe un detector de intrusos universal**, mientras los principios, objetivos y métodos de seguridad son estándares, la determinación de seguridad es diferente para cada organización (Canderle, M., Aguirre, G., & Piccoli, F.)

La utilización de conjunto de datos "**DATA-SET**" compuestos por capturas de ataques de tráfico normal provenientes de un escenario real es algo muy importante por ello la utilización de conjuntos de datos como **NSL-KDD**, **KDD-CUP 99**, **DARPA 98**, es motivo de estudio permanente para el desarrollo de sistemas de detección de intrusos (Herrera, D., Carvajal, H., 2010) (Kayacık, G., Zincir, N., Heywood, M.). La utilización de conjuntos de datos "**DATA-SET**" implica el análisis de los datos que estos contienen de una forma discriminada, uno de los principales problemas presentes en el análisis de los datos hace referencia a la buena preparación de los mismos, entendiendo la preparación de los datos como el proceso en el cual se realiza la selección de características de una forma adecuada.

Dentro del contexto de búsqueda de mejoras en las redes la detección de intrusos a través de técnicas que permitan un análisis de detallado de los datos que son transportados a través de una red de computadores tendrían un gran impacto en cuanto a la seguridad e integridad de la

información por ello cabe preguntarse **¿la aplicación de técnicas de selección de características, métricas de aprendizaje y reducción de dimensión contribuirían a la detección de intrusos en las redes?**

Capítulo 3: Metodología

El desarrollo de este capítulo tiene como propósito exponer las fases que fueron necesarias para la ejecución de la temática de investigación, también se realiza una descripción de las actividades utilizadas como soporte para dar cumplimiento a cada una de las fases estipuladas para la ejecución de la investigación.

Para el desarrollo de esta investigación se plantearon dos etapas que permitirán realizar un seguimiento detallado de la forma en que se desarrolló la temática de investigación **APLICACIÓN DE SELECCIÓN DE CARACTERISTICAS, METRICAS DE APRENDIZAJE Y REDUCCION DE DIMENSION EN SISTEMAS DE DETECCION DE INTRUSOS,** a continuación se realiza una breve descripción de la metodología a seguir:

Fase Numero 1 (Análisis De Antecedentes): Esta etapa tiene como principal propósito realizar un amplio estudio acerca de los antecedentes que dieron origen a la utilización de técnicas de Selección De Características, Reducción De Dimensión Y Métricas De Aprendizaje en los sistemas de detección de intrusos, para hacer posible el levantamiento de dicha información se realizó una amplia búsqueda en fuentes de información confiables (Bases de datos Especializadas). A continuación se describen las actividades que permiten la culminación de la Fase:

- Identificar las diferentes técnicas existentes para la selección de características
- Identificar las diferentes técnicas existentes para la reducción de dimensión y métricas de Aprendizaje

Fase Numero 2 (Diseño, Generación De Sistema Prototipo Para La Detección De Ataques En Redes De Computadores): Esta etapa tiene como objeto el diseño y generación de un prototipo que permita la detección de ataques en una red de computadores. Para hacer posible el cumplimiento de esta Fase fue necesario tener conocimiento previo de la herramienta MATLAB y la

implementación de diferentes componentes en dicha herramienta. A continuación se describen las actividades que permiten la culminación de la Fase Numero 2:

- Construir y entrenar el prototipo que analice información proveniente del Data-Set NSL-KDD, implementando las mejores técnicas de selección de características, reducción de dimensión y métricas de aprendizaje.
- Validar el prototipo creado respecto a los resultados obtenidos con las técnicas SOM y GHSOM bajo el mismo conjunto de datos (DataSet NSL-KDD)

Capítulo 4: Implementación de Técnica para Selección de Características, Reducción de Dimensión y Métricas de Aprendizaje

El desarrollo de este capítulo tiene como propósito exponer los procedimientos para la implementación de las diferentes Técnicas de Selección de Características, Reducción de Dimensión y Métricas de aprendizaje. Se realiza la descripción de los diferentes criterios a tener en cuenta para la implementación y buen funcionamiento de cada una de las técnicas seleccionadas. Además se realiza una breve descripción de los recursos de hardware y software utilizados.

4.1 RECURSOS DE HARDWARE

Para el desarrollo de la investigación fue necesario tener una capacidad respuesta adecuada por parte de los equipos de cómputo, debido a que las pruebas realizadas requieren gran utilización de recursos por parte del sistema, en consecuencia se utilizó el laboratorio de redes de computadores de la **CORPORACIÓN UNIVERSIDAD DE LA COSTA, CUC** el cual está dotado por 20 equipos que tienen las siguientes características:

Tipo de CPU:	4x , 3400 MHz (12 x 283)
Nombre del motherboard :	Dell OptiPlex 990
Memoria del sistema :	8148 MB
Placa de video:	AMD RADEON HD 6350 (512 MB)
Disco rígido:	ST9250410AS (250 GB, 7200 RPM, SATA-II)
Procesador:	Intel(R) Core(TM) i7-2600 CPU @ 3.40GH

4.2 RECURSOS DE SOFTWARE

Para el desarrollo de la investigación fue necesario contar con un sistema operativo seguro que además soportara una plataforma de desarrollo con gran capacidad para realizar operaciones matemáticas, en consecuencia se utilizaron los siguientes recursos de software:

- Sistema Operativo: Ubuntu 12.04 LTS x 64
- Plataforma de Desarrollo: Matlab R2011b x 64
- Librería Feature Selection Toolbox

4.3 IMPLEMENTACIÓN DE FEAST

Para el desarrollo de esta investigación se han tenido en cuenta los diferentes tipos de ataques y problemas de seguridad presentes en una red de computadores, utilizando como técnica de selección de característica la toolbox de Matlab denominada Features Selection Toolbox "FEAST" descrita en el capítulo 2. Para la implementación del FEAST previamente debe existir claridad acerca de los datos o conjuntos de datos que están presentes en el DataSet NSL-KDD el cual tiene un conjunto de conexiones y cada una de dichas conexiones tiene 41 características. Este espacio de características se compone de vectores que contienen el tráfico de conexión. Entre las características tomadas en cuenta se mencionan la duración de la conexión, el tipo de protocolo, o el número de intentos de user-to-root.

Luego de tener claridad de los datos contenidos en el DataSet se debe tener en cuenta que el conjunto de datos contiene un test para el entrenamiento y un test para realizar la clasificación, esto con el objeto de llevar a cabo las pruebas a través de la validación cruzada seleccionando cada uno de los test dentro del contexto a desarrollar. Una vez se tienen el conjunto de datos para el entrenamiento y para la clasificación se identifican los diferentes métodos del Feast que se van a someter a estudio (**mim, mrmr, mifs, cmim, jmi, disr, cife, icap, condred, cmi, relief, fcbf, betagamma**) los cuales serán probados con un numero especifico de características (5, 10, 15, 20, 25, 30,35) los cuales permiten validar el nivel de precisión de cada método del FEAST , con la finalidad de identificar los tipos de ataques presentes en la información contenida en el DataSet NSL-KDD. A continuación se presenta el código necesario para la implementación del FEAST.

```
labNOR=(labels5==0);                                  (1)
F_labNOR=double(labNOR);                              (2)
feat_NOR = feast('icap',25,D,F_labNOR')';             (3)

labDOS=(labels5==1);                                  (1)
F_labDOS=double(labDOS);                              (2)
feat_DOS = feast('icap',25,D,F_labDOS')';             (3)

labPRO=(labels5==2);                                  (1)
F_labPRO=double(labPRO);                              (2)
feat_PRO = feast('icap',25,D,F_labPRO')';             (3)
```

Para la implementación del FEAST se necesitan obtener previamente un conjunto de etiquetas de los diferentes tipos de ataques que pueden presentarse dentro del DataSet NSL-KDD **(Normales, Dos, Probe, R2L, U2R)** los cuales se encuentran dentro de la variable **labels5** para luego almacenarlos en una variable tal como se puede apreciar en **(1)**. Luego de tener claridad acerca del contenido presente en la variable **labels5** se procede a crear una variable nueva la cual permita tener la cantidad por tipos de ataques contenidos en el DataSet **(2).** Posteriormente se procede a la creación de una variable, la cual almacena las mejores características de cada tipo de ataque de acuerdo al método del FEAST seleccionado, Numero de características definidas, matriz de datos en este caso de ejemplo **D** y la variable que contiene la cantidad por tipos de ataques presentes en el conjunto de datos **(3)**. Seguidamente ejecutado cada método del FEAST se procede analizar los datos resultantes de cada simulación para seleccionar el método que mejores resultados de precisión arroje.

4.4 Implementación de FEAST con ISOMAP

El desarrollo de esta investigación somete a prueba la técnica de reducción de dimensión y métrica de aprendizaje **ISOMAP (**Isometric Feature Mapping**)** la cual es implementada luego de haber seleccionado el mejor algoritmo de los implementados por el método del FEAST. A continuación se describe el procedimiento para la ejecución del algoritmo de la técnica Isomap para el cual se manejó la siguiente función:

[mappedX, mapping] = landmark_isomap(X, no_dims, k, percentage);

La función de ejecución del algoritmo landmark Isomap es aplicada en el conjunto de datos correspondientes al DataSet NSL-KDD, la función tiene como propósito reducir la dimensionalidad del DataSet (Numero de Dimensiones). Para ello recibe como parámetros número de vecinos utilizados en la computación no_dims, el valor establecido para k por defecto es 12, la variable porcentaje se encuentra entre 0 y 1, si el gráfico del vecino está construido y este no se encuentra completamente conectado, solo estará conectado el componente más grande. Los índices de los componentes son retornados en [mappedX mapping_NOR]. Esta función es aplicada a cada uno de los tipos de ataques que se pueden presentar dentro de una red de computadores.

A continuación se puede apreciar el código utilizado para probar el método de reducción de dimensión y métrica de aprendizaje seleccionado.

```
% ISOMAP
[mappedX mapping_NOR] =
landmark_isomap(D(:,feat_NOR),nfeat,25,0.8);
mapping_NOR.name='LandmarkIsomap';
D_NOR=out_of_sample(D(:,feat_NOR),
mapping_NOR);

[mappedX mapping_DOS] =
landmark_isomap(D(:,feat_DOS),nfeat,25,0.8);
mapping_DOS.name='LandmarkIsomap';
D_DOS=out_of_sample(D(:,feat_DOS),
mapping_DOS);
```

4.5 Implementación de FEAST con Principal Component Analysis (PCA)

El desarrollo de esta investigación se somete a prueba la técnica de reducción de dimensión y métrica de aprendizaje **Principal Component Analysis (PCA)** la cual es implementada luego de haber seleccionado el mejor método del FEAST. A continuación se describe el procedimiento para la ejecución del algoritmo de la técnica PCA para el cual se manejó la siguiente función:

[COEFF,SCORE,latent] = princomp(X)

COEFF= princomp(x) realiza PCA en la datos de matriz X con dimensiones n x m, y retorna los coeficientes de los componentes principales, también conocido como cargas. Las filas de X corresponden a observaciones, columnas a variables. COEFF es una matriz n x n, cada columna contiene los coeficientes para un componente principal. Las columnas están en orden decreciente de acuerdo a la varianza de componentes.

PRINCOMP centra X restando las medias de las columna, pero no reescalar las columnas de X. Para realizar el análisis de componentes principales con las variables estandarizadas, es decir, sobre la base de correlaciones, se utiliza PRINCOMP (zscore (X)). Para llevar a cabo el análisis de componentes principales directamente en una matriz de covarianza o correlación, utilice pcacov.

[COEFF,SCORE] = PRINCOMP (X) devuelve SCORE (las puntuaciones de los componentes principales), es decir, la representación de X en el espacio de componentes principales. Filas de SCORE se corresponden con observaciones, las columnas a los componentes.

[COEFF,SCORE,latent] = PRINCOMP devuelve (X) latentes, un vector que contiene los valores propios de la matriz de covarianza de X.

Los SCORES son los datos formados por la transformación de datos originales en el espacio de las componentes principales. Los valores de la latente vector son la varianza de las columnas de SCORE. T2 de Hotelling es una medida de la multivariante distancia de cada observación desde el centro del conjunto de datos.

A continuación se puede apreciar el código utilizado para probar el método de reducción de dimensión y métrica de aprendizaje seleccionado.

```
% PCA
[ev_NOR score lat]=princomp(D(:,feat_NOR),'econ');
D_NOR=D(:,feat_NOR)*ev_NOR(:,1:nev);
[ev_DOS score lat]=princomp(D(:,feat_DOS),'econ');
D_DOS=D(:,feat_DOS)*ev_DOS(:,1:nev);
[ev_PRO score lat]=princomp(D(:,feat_PRO),'econ');
D_PRO=D(:,feat_PRO)*ev_PRO(:,1:nev);
[ev_R2L score lat]=princomp(D(:,feat_R2L),'econ');
D_R2L=D(:,feat_R2L)*ev_R2L(:,1:nev);
[ev_U2R score lat]=princomp(D(:,feat_U2R),'econ');
D_U2R=D(:,feat_U2R)*ev_U2R(:,1:nev);
```

4.6 Implementación de FEAST con Kernel Principal Component Analysis (KPCA)

El desarrollo de esta investigación somete a prueba la técnica de reducción de dimensión y métrica de aprendizaje **Kernel Principal Component Analysis (KPCA)** la cual es implementada luego de haber seleccionado el mejor método del FEAST. A continuación se describe el procedimiento para la ejecución del algoritmo de la técnica KPCA para el cual se manejó la siguiente función:

[mappedX, mapping] = kernel_pca(X, no_dims)
[mappedX, mapping] = kernel_pca(X, no_dims, kernel)
[mappedX, mapping] = kernel_pca(X, no_dims, kernel, param1)
[mappedX, mapping] = kernel_pca(X, no_dims, kernel, param1, param2)

La función ejecuta Kernel PCA en un conjunto de puntos de datos X. La variable no_dims establece el número de dimensiones de los puntos de característica en el espacio de características incrustado (no_dims >= 1, default = 2). Para no_dims, también puede especificar un número entre 0 y 1, la determinación la cantidad de varianza que desea conservar en la etapa de PCA. El valor de kernel determina el kernel utilizado. Los valores posibles son 'linear', 'gauss', 'poly', 'subsets', or 'princ_angles' (default = 'gauss'). La función devuelve las ubicaciones de los datos de entrenamiento incrustados en mapeado. Por otra parte, devuelve información sobre la mapping en la mapping.

A continuación se puede apreciar el código utilizado para probar el método de reducción de dimensión y métrica de aprendizaje seleccionado.

```
%KPCA
tec='KernelPCA';
param='poly'
[mappedX, mapping_NOR] = compute_mapping(D(:,feat_NOR), tec,
nfeat,param,0,3);
mapping_NOR.name=tec;
D_NOR=out_of_sample(D(:,feat_NOR), mapping_NOR);
[mappedX, mapping_DOS] = compute_mapping(D(:,feat_DOS), tec,
nfeat,param,0,3);
mapping_DOS.name=tec;
D_DOS=out_of_sample(D(:,feat_DOS), mapping_DOS);
[mappedX, mapping_PRO] = compute_mapping(D(:,feat_PRO), tec,
nfeat,param,0,3);
mapping_PRO.name=tec;
D_PRO=out_of_sample(D(:,feat_PRO), mapping_PRO);
[mappedX, mapping_R2L] = compute_mapping(D(:,feat_R2L), tec,
nfeat,param,0,3);
mapping_R2L.name=tec;
D_R2L=out_of_sample(D(:,feat_R2L), mapping_R2L);
[mappedX, mapping_U2R] = compute_mapping(D(:,feat_U2R), tec,
nfeat,param,0,3);
mapping_U2R.name=tec;
D_U2R=out_of_sample(D(:,feat_U2R), mapping_U2R);
```

Capítulo 5: Diseño, Prueba y Validación de Resultados

En este capítulo se exponen los resultados obtenidos luego de someter a prueba las diferentes técnicas del métodos FEAST, una vez seleccionadas las técnicas que arrojen los mejores resultados se procede a describir los resultados obtenidos de las técnicas de reducción de dimensión y métricas de aprendizaje que en este caso son (ISOMAP, PCA, KPCA)

5.1 PRUEBA Y VALIDACION FEAST

La aplicación de las diferentes técnicas del método FEAST se realiza con el propósito de garantizar que el prototipo, producto del desarrollo de la temática de investigación tenga resultados óptimos. Para la implementación o puesta en marcha de las pruebas se sometió cada técnica a un proceso riguroso tal como se puede observar en el capítulo 4. Posteriormente a la ejecución del código generado se obtiene el siguiente resultado.

	5	10	15	20	25	30	35	NORMAL
CIFE	76,90%	56,92%	78,48%	79,90%	86,75%	85,23%	85,37%	
CMI	77,03%	76,76%	76,76%	76,76%	76,76%	76,76%	76,76%	
CONDRED	79,92%	80,57%	81,48%	77,54%	76,85%	81,96%	82,64%	
JMI	79,92%	85,42%	85,70%	86,60%	85,07%	81,72%	82,41%	
MRMR	85,42%	86,64%	86,71%	86,69%	86,86%	86,06%	86,96%	
CMIN	78,83%	85,39%	85,47%	84,81%	87,73%	85,48%	85,46%	
DISR	86,76%	86,82%	86,82%	87,51%	87,50%	86,12%	87,30%	
ICAP	78,83%	78,46%	78,25%	78,39%	78,39%	78,39%	78,39%	
MIFS	78,72%	78,71%	78,71%	78,71%	78,71%	78,71%	78,69%	
RELIEF	86,20%	86,20%	86,20%	86,20%	86,20%	86,20%	86,20%	

Tabla No 3. Precisión de técnicas del Método FEAST para la detección Conexiones tipo Normales.

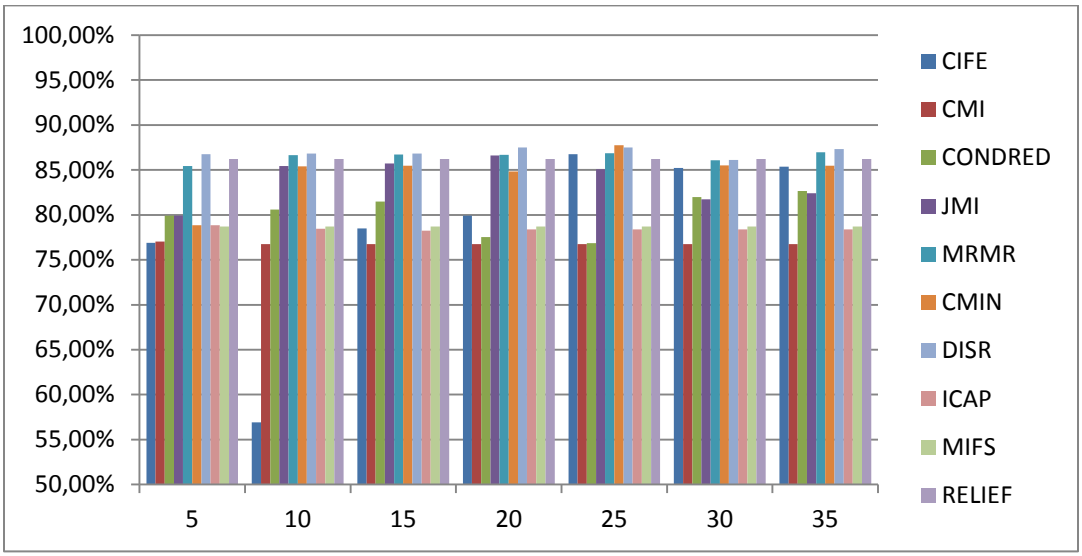

Gráfico No 2. Precisión de técnicas del Método FEAST para la detección de Conexiones tipo Normal

Se puede apreciar en tabla No 3 y Grafico No 2, el resultado obtenido al someter a prueba las diferentes técnicas del método FEAST, donde podemos observar la utilización de un numero especifico de características, que en este caso corresponde a "5, 10, 15, 20, 25, 30, 35" respectivamente. El número de características es seleccionado para poder tomar cual técnica tiene mayor precisión y así identificar qué conexiones de red son tipo NORMAL y en qué rango de características se encuentra el mayor porcentaje de precisión.

	5	10	15	20	25	30	35	DOS
CIFE	85,04%	86,65%	86,62%	86,83%	85,90%	85,87%	85,87%	
CMI	83,13%	83,13%	83,13%	83,13%	83,13%	83,13%	83,13%	
CONDRED	87,06%	66,92%	87,34%	86,39%	86,28%	87,74%	87,68%	
JMI	81,72%	83,81%	87,02%	86,89%	85,94%	86,43%	87,42%	
MRMR	81,50%	83,06%	83,98%	84,27%	84,20%	86,60%	86,52%	
CMIN	84,31%	84,52%	87,98%	87,69%	87,57%	87,15%	87,26%	
DISR	79,72%	83,06%	83,57%	81,15%	80,99%	81,84%	81,73%	
ICAP	83,23%	84,54%	86,43%	86,43%	86,43%	86,43%	86,43%	
MIFS	78,80%	78,81%	78,81%	78,81%	78,81%	78,81%	78,81%	
RELIEF	85,71%	85,71%	85,71%	85,71%	85,71%	85,71%	85,71%	

Tabla No 4. Precisión de técnicas del Método FEAST para la detección de ataques tipo DOS.

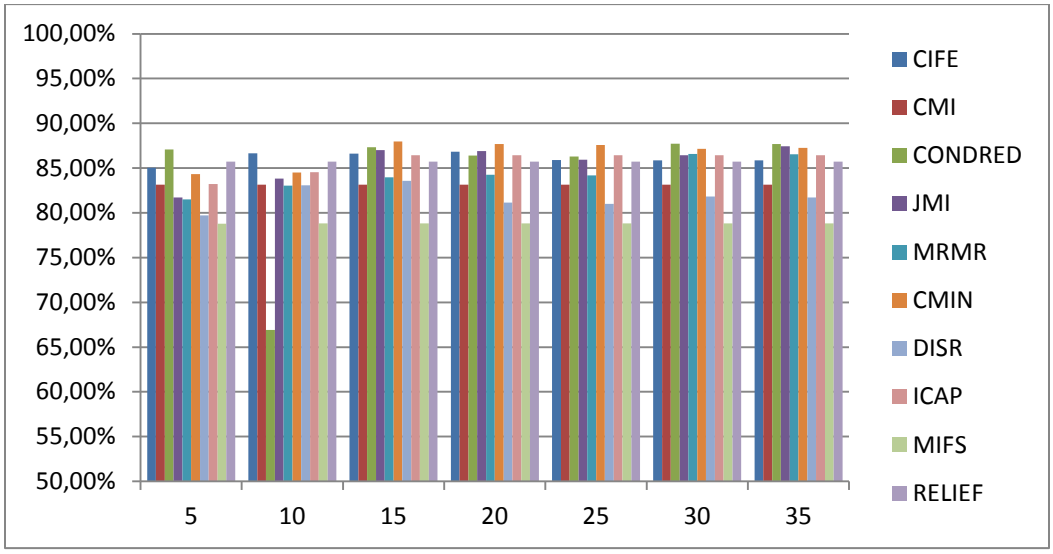

Gráfico No 3. Precisión de técnicas del Método FEAST para la detección de ataques tipo DOS

Se puede apreciar en tabla No 4 y Grafico No 3, el resultado obtenido al someter a prueba las diferentes técnicas del método FEAST, donde podemos observar la utilización de un numero especifico de características, que en este caso corresponde a "5, 10, 15, 20, 25, 30, 35" respectivamente. El número de características es seleccionado para poder tomar cual técnica tiene mayor precisión y así identificar qué conexiones de red son tipo DOS y en qué rango de características se encuentra el mayor porcentaje de precisión.

	5	10	15	20	25	30	35	PROBE
CIFE	59,68%	81,88%	91,47%	92,85%	92,71%	92,91%	93,04%	
CMI	81,79%	74,89%	74,89%	74,89%	74,89%	74,89%	74,89%	
CONDRED	53,76%	61,15%	70,54%	75,77%	82,39%	92,31%	91,77%	
JMI	91,84%	81,72%	87,85%	90,95%	90,19%	89,76%	89,74%	
MRMR	90,44%	91,90%	92,66%	92,56%	92,43%	90,12%	90,55%	
CMIN	92,46%	91,89%	94,46%	94,66%	93,10%	92,13%	89,81%	
DISR	90,44%	90,74%	92,56%	91,91%	90,26%	92,27%	91,86%	
ICAP	86,60%	83,03%	84,32%	84,32%	84,32%	84,32%	84,32%	
MIFS	92,69%	92,69%	92,69%	92,69%	92,69%	92,69%	92,69%	
RELIEF	88,42%	88,42%	88,42%	88,42%	88,42%	88,42%	88,42%	

Tabla No 5. Precisión de técnicas del Método FEAST para la detección de ataques tipo PROBE.

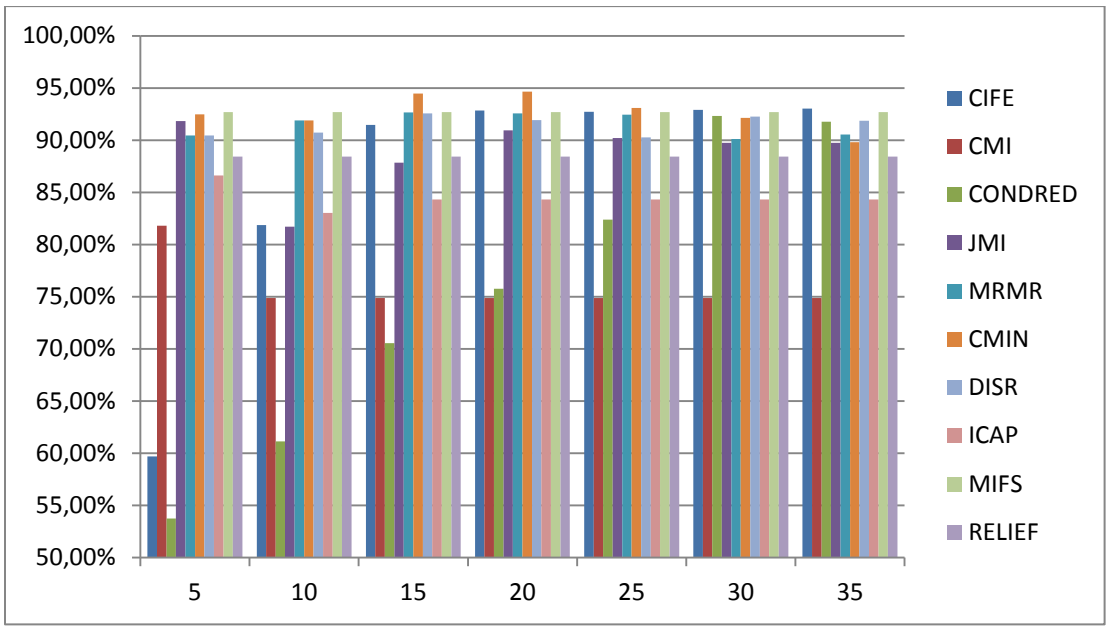

Gráfico No 4. Precisión de técnicas del Método FEAST para la detección de ataques tipo PROBE

Se puede apreciar en tabla No 5 y Grafico No 4, el resultado obtenido al someter a prueba las diferentes técnicas del método FEAST, donde podemos observar la utilización de un numero especifico de características, que en este caso corresponde a "5, 10, 15, 20, 25, 30, 35" respectivamente. El número de características es seleccionado para poder tomar cual técnica tiene mayor precisión y así identificar qué conexiones de red son tipo PROBE y en qué rango de características se encuentra el mayor porcentaje de precisión.

	5	10	15	20	25	30	35	U2R
CIFE	93,20%	93,18%	93,63%	93,19%	93,00%	93,48%	92,09%	
CMI	93,19%	91,50%	91,50%	91,50%	91,50%	91,50%	91,50%	
CONDRED	67,65%	93,19%	93,49%	93,49%	93,59%	93,65%	93,47%	
JMI	93,21%	93,19%	93,53%	93,53%	93,54%	93,57%	93,66%	
MRMR	93,21%	93,26%	93,16%	91,69%	91,47%	91,90%	91,91%	
CMIN	93,23%	93,22%	93,49%	93,50%	93,62%	93,70%	93,80%	
DISR	93,22%	91,64%	91,73%	91,79%	91,89%	91,90%	91,71%	
ICAP	93,23%	93,18%	90,21%	89,97%	89,97%	89,97%	89,97%	
MIFS	92,66%	92,66%	92,66%	92,66%	92,66%	92,66%	92,66%	
RELIEF	93,11%	93,11%	93,11%	93,11%	93,11%	93,11%	93,11%	

Tabla No 6. Precisión de técnicas del Método FEAST para la detección de ataques tipo U2R.

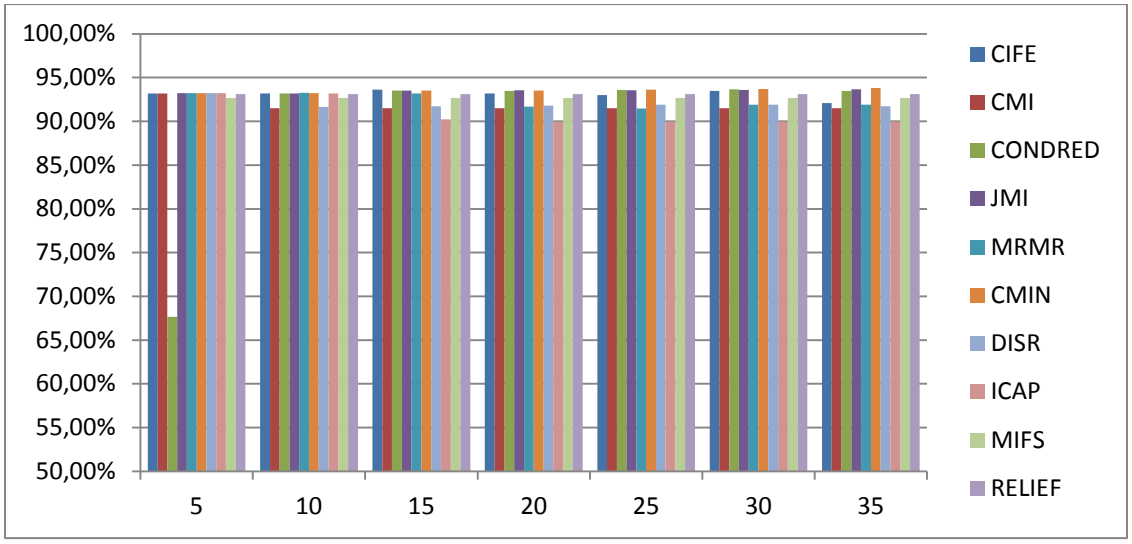

Gráfico No 5. Precisión de técnicas del Método FEAST para la detección de ataques tipo U2R

Se puede apreciar en tabla No 6 y Grafico No 5, el resultado obtenido al someter a prueba las diferentes técnicas del método FEAST, donde podemos observar la utilización de un numero especifico de características, que en este caso corresponde a "5, 10, 15, 20, 25, 30, 35" respectivamente. El número de características es seleccionado para poder tomar cual técnica tiene mayor precisión y así identificar qué conexiones de red son tipo U2R y en qué rango de características se encuentra el mayor porcentaje de precisión.

	5	10	15	20	25	30	35	R2L
CIFE	83,02%	83,64%	82,83%	81,52%	81,87%	82,31%	82,54%	
CMI	83,46%	83,39%	83,39%	83,39%	83,39%	83,39%	83,39%	
CONDRED	83,65%	83,13%	82,54%	85,01%	85,41%	84,74%	85,73%	
JMI	83,38%	82,78%	82,94%	86,06%	86,75%	86,11%	85,36%	
MRMR	83,38%	83,43%	84,40%	84,90%	87,53%	85,47%	85,65%	
CMIN	83,38%	83,21%	84,84%	84,03%	83,26%	82,43%	86,79%	
DISR	80,87%	87,11%	80,65%	82,77%	82,76%	84,79%	84,75%	
ICAP	83,46%	83,75%	83,61%	83,61%	83,61%	83,61%	83,61%	
MIFS	83,54%	83,54%	83,54%	83,54%	83,54%	83,54%	83,54%	
RELIEF	90,07%	90,07%	90,07%	90,07%	90,07%	90,07%	90,07%	

Tabla No 7. Precisión de técnicas del Método FEAST para la detección de ataques tipo R2L.

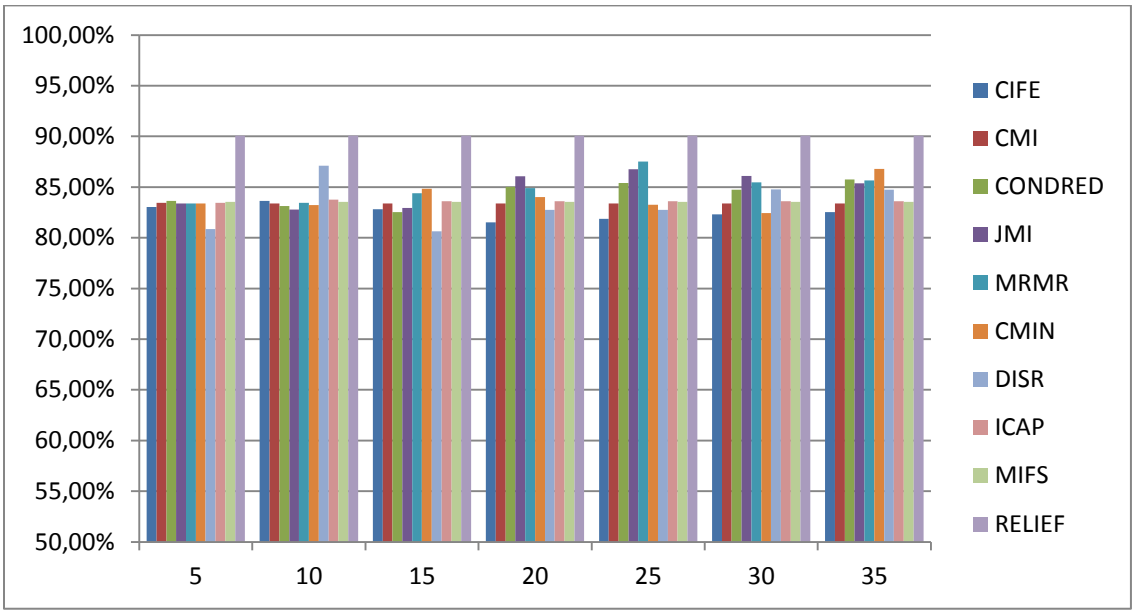

Gráfico No 6. Precisión de técnicas del Método FEAST para la detección de ataques tipo R2L

Se puede apreciar en tabla No 7 y Grafico No 6, el resultado obtenido al someter a prueba las diferentes técnicas del método FEAST, donde podemos observar la utilización de un numero especifico de características, que en este caso corresponde a "5, 10, 15, 20, 25, 30, 35" respectivamente. El número de características es seleccionado para poder tomar cual técnica tiene mayor precisión y así identificar qué conexiones de red son tipo R2L y en qué rango de características se encuentra el mayor porcentaje de precisión.

Posteriormente de haber realizado las pruebas con cada técnica del método FEAST para medir la capacidad de identificación de tipos de ataques se seleccionan las 3 mejores técnicas que en este caso son "**JMI, MRMR, RELIEF**".

5.2 PRUEBA Y VALIDACION FEAST con ISOMAP

Una vez seleccionada las técnicas que arrojan mayores resultados del método FEAST se procede a realizar la implementación del METODO FEAST CON ISOMAP tal como se puede observar en el capítulo 4. Posteriormente a la ejecución del código generado se obtiene el siguiente resultado.

	5	10	15	20	25	30	35	NORMAL
JMI	39,32%	45,18%	70,39%	73,95%	66,79%	64,76%	61,72%	
MRMR	62,74%	76,70%	75,91%	79,05%	64,42%	67,47%	56,84%	
RELIEF	65,02%	65,37%	75,23%	78,15%	64,42%	65,17%	65,25%	

Tabla No 8. Precisión de conjunto de técnicas FEAST-ISOMAP para la detección de ataques tipo NORMAL.

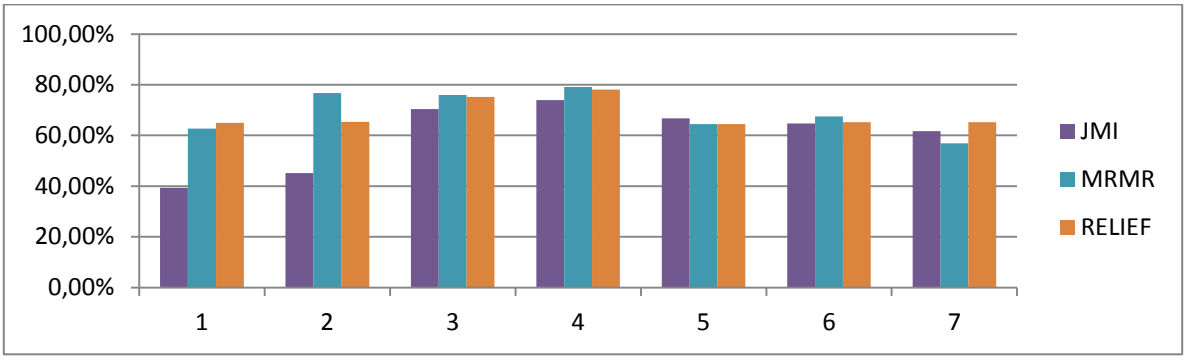

Gráfico No 7. Precisión de conjunto de técnicas FEAST-ISOMAP para la detección de Conexiones tipo NORMAL

Se puede apreciar en tabla No 8 y Grafico No 7, el resultado obtenido al someter a prueba las diferentes técnicas del método FEAST-ISOMPA, donde podemos observar la utilización de un numero especifico de características, que en este caso corresponde a "5, 10, 15, 20, 25, 30, 35" respectivamente. El número de características es seleccionado para poder tomar cual técnica tiene mayor precisión y así identificar qué conexiones de red son tipo NORMAL y en qué rango de características se encuentra el mayor porcentaje de precisión.

	5	10	15	20	25	30	35	DOS
JMI	82,47%	71,07%	67,26%	77,63%	73,11%	37,09%	32,35%	
MRMR	70,06%	75,64%	77,96%	83,11%	79,64%	79,40%	63,95%	
RELIEF	39,75%	34,64%	79,74%	80,38%	79,64%	39,43%	34,62%	

Tabla No 9. Precisión de conjunto de técnicas FEAST-ISOMAP para la detección de ataques tipo DOS.

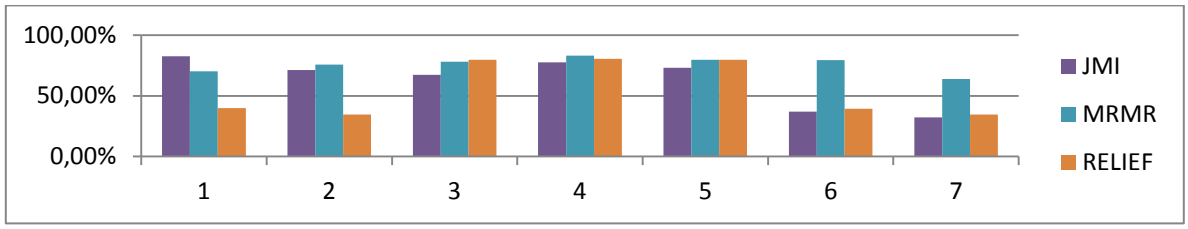

Gráfico No 8. Precisión de conjunto de técnicas FEAST-ISOMAP para la detección de ataques tipo DOS

Se puede apreciar en tabla No 9 y Grafico No 8, el resultado obtenido al someter a prueba las diferentes técnicas del método FEAST-ISOMPA, donde podemos observar la utilización de un numero especifico de características, que en este caso corresponde a "5, 10, 15, 20, 25, 30, 35" respectivamente. El número de características es seleccionado para poder tomar cual técnica tiene mayor precisión y así identificar qué conexiones de red son tipo DOS y en qué rango de características se encuentra el mayor porcentaje de precisión.

	5	10	15	20	25	30	35	PROBE
JMI	53,21%	49,84%	82,24%	70,54%	83,38%	82,77%	82,42%	
MRMR	91,75%	87,43%	79,36%	59,56%	78,10%	78,47%	78,38%	
RELIEF	69,07%	69,57%	67,83%	69,84%	78,10%	70,66%	70,08%	

Tabla No 10. Precisión de conjunto de técnicas FEAST-ISOMAP para la detección de ataques tipo PROBE.

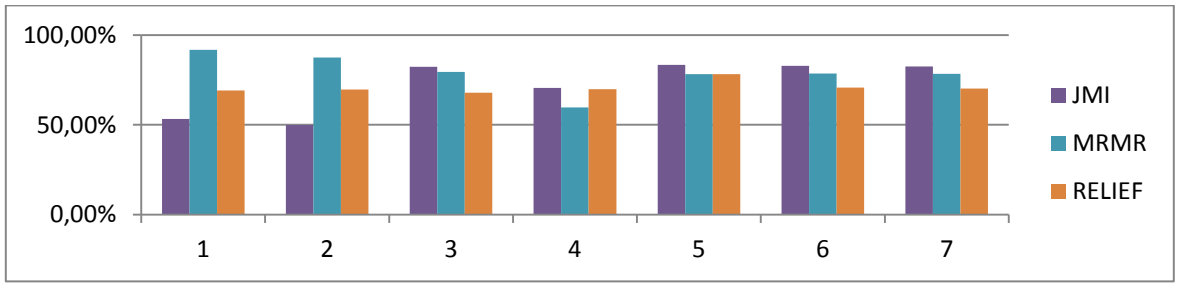

Gráfico No 9. Precisión de conjunto de técnicas FEAST-ISOMAP para la detección de ataques tipo PROBE

Se puede apreciar en tabla No 10 y Grafico No 9, el resultado obtenido al someter a prueba las diferentes técnicas del método FEAST-ISOMPA, donde podemos observar la utilización de un numero especifico de características, que en este caso corresponde a "5, 10, 15, 20, 25, 30, 35" respectivamente. El número de características es seleccionado para poder tomar cual técnica tiene

mayor precisión y así identificar qué conexiones de red son tipo PROBE y en qué rango de características se encuentra el mayor porcentaje de precisión.

	5	10	15	20	25	30	35	U2R
JMI	57,07%	96,88%	34,67%	66,21%	68,88%	66,73%	32,81%	
MRMR	1,41%	99,70%	36,14%	45,61%	59,00%	94,34%	94,33%	
RELIEF	66,35%	66,22%	32,62%	32,80%	59,00%	66,17%	66,19%	

Tabla No 11. Precisión de conjunto de técnicas FEAST-ISOMAP para la detección de ataques tipo U2R.

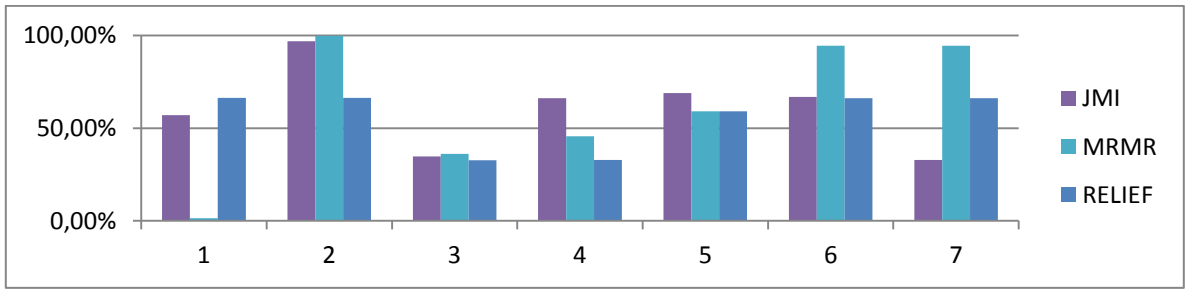

Gráfico No 10. Precisión de conjunto de técnicas FEAST-ISOMAP para la detección de ataques tipo U2R.

Se puede apreciar en tabla No 11 y Grafico No 10, el resultado obtenido al someter a prueba las diferentes técnicas del método FEAST-ISOMPA, donde podemos observar la utilización de un numero especifico de características, que en este caso corresponde a "5, 10, 15, 20, 25, 30, 35" respectivamente. El número de características es seleccionado para poder tomar cual técnica tiene mayor precisión y así identificar qué conexiones de red son tipo U2R y en qué rango de características se encuentra el mayor porcentaje de precisión.

	5	10	15	20	25	30	35	R2L
JMI	86,87%	0,00%	65,71%	41,68%	35,14%	60,34%	55,39%	
MRMR	86,87%	33,16%	86,85%	86,05%	67,57%	64,52%	69,69%	
RELIEF	60,63%	60,54%	48,90%	48,87%	67,57%	60,49%	60,53%	

Tabla No 12. Precisión de conjunto de técnicas FEAST-ISOMAP para la detección de ataques tipo R2L.

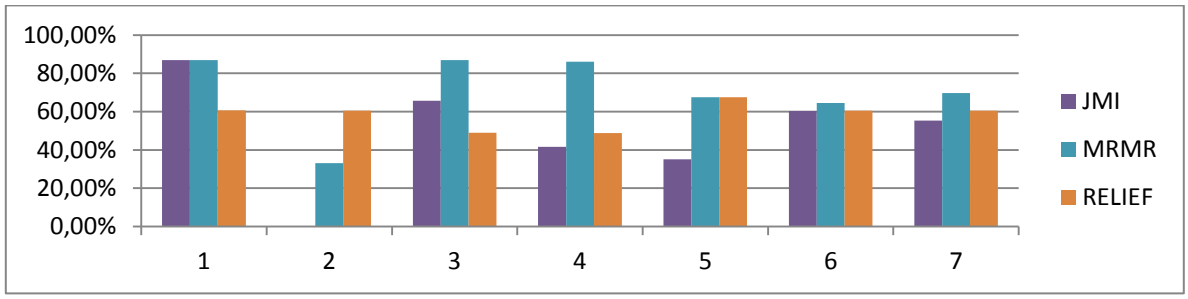

Gráfico No 11. Precisión de conjunto de técnicas FEAST-ISOMAP para la detección de ataques tipo R2L.

Se puede apreciar en tabla No 12 y Grafico No 11, el resultado obtenido al someter a prueba las diferentes técnicas del método FEAST-ISOMPA, donde podemos observar la utilización de un numero especifico de características, que en este caso corresponde a "5, 10, 15, 20, 25, 30, 35" respectivamente. El número de características es seleccionado para poder tomar cual técnica tiene mayor precisión y así identificar qué conexiones de red son tipo R2L y en qué rango de características se encuentra el mayor porcentaje de precisión.

5.3 PRUEBA Y VALIDACION FEAST con PCA

Una vez seleccionada las técnicas que arrojan mayores resultados del método FEAST se procede a realizar la implementación del METODO FEAST - PCA tal como se puede observar en el capítulo 4. Posteriormente a la ejecución del código generado se obtiene el siguiente resultado.

	5	10	15	20	25	30	35	NORMAL
JMI	78,49%	77,02%	74,54%	74,78%	74,26%	75,14%	75,44%	
MRMR	85,42%	68,91%	76,61%	78,94%	77,67%	77,29%	77,29%	
RELIEF	76,40%	76,40%	76,40%	76,40%	76,41%	76,41%	76,41%	

Tabla No 13. Precisión de conjunto de técnicas FEAST-PCA para la detección de ataques tipo NORMAL

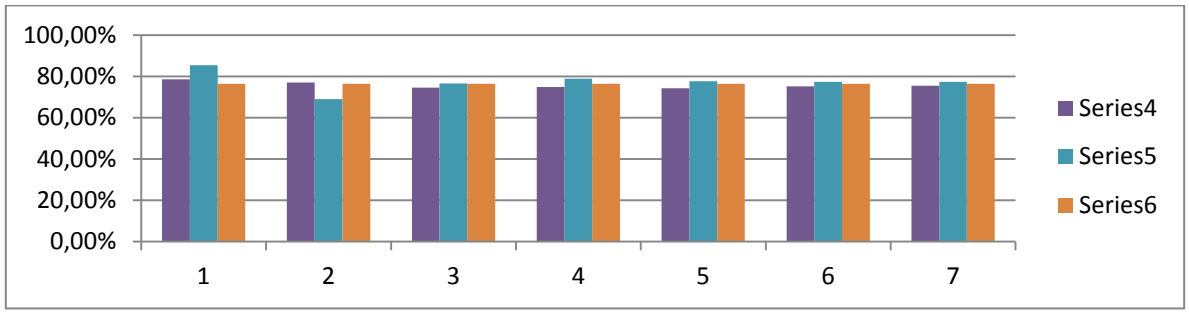

Gráfico No 12. Precisión de conjunto de técnicas FEAST-PCA para la detección de ataques tipo NORMAL.

Se puede apreciar en tabla No 13 y Grafico No 12, el resultado obtenido al someter a prueba las diferentes técnicas del método FEAST- PCA, donde podemos observar la utilización de un numero especifico de características, que en este caso corresponde a "5, 10, 15, 20, 25, 30, 35" respectivamente. El número de características es seleccionado para poder tomar cual técnica tiene mayor precisión y así identificar qué conexiones de red son tipo NORMAL y en qué rango de características se encuentra el mayor porcentaje de precisión.

	5	10	15	20	25	30	35	DOS
JMI	80,19%	83,48%	84,86%	87,70%	84,82%	85,43%	84,78%	
MRMR	80,77%	84,25%	84,02%	84,31%	84,30%	83,64%	83,64%	
RELIEF	85,87%	85,87%	85,87%	85,87%	85,87%	85,87%	85,87%	

Tabla No 14. Precisión de conjunto de técnicas FEAST-PCA para la detección de ataques tipo DOS

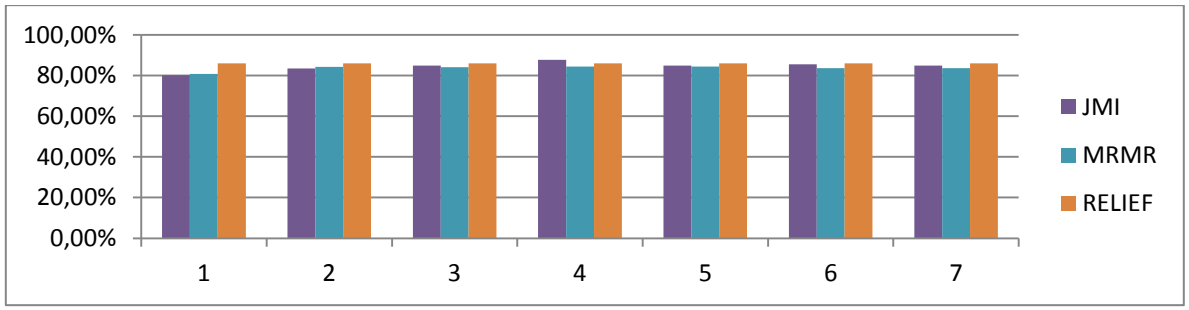

Gráfico No 13. Precisión de conjunto de técnicas FEAST-PCA para la detección de ataques tipo DOS.

Se puede apreciar en tabla No 14 y Grafico No 13, el resultado obtenido al someter a prueba las diferentes técnicas del método FEAST- PCA, donde podemos observar la utilización de un numero

especifico de características, que en este caso corresponde a “5, 10, 15, 20, 25, 30, 35” respectivamente. El número de características es seleccionado para poder tomar cual técnica tiene mayor precisión y así identificar qué conexiones de red son tipo DOS y en qué rango de características se encuentra el mayor porcentaje de precisión.

	5	10	15	20	25	30	35	PROBE
JMI	92,56%	85,68%	92,38%	95,55%	94,06%	81,09%	82,66%	
MRMR	90,41%	79,86%	94,83%	94,72%	94,74%	87,66%	87,66%	
RELIEF	82,60%	82,60%	82,60%	82,60%	82,60%	82,60%	82,60%	

Tabla No 15. Precisión de conjunto de técnicas FEAST-PCA para la detección de ataques tipo PROBE

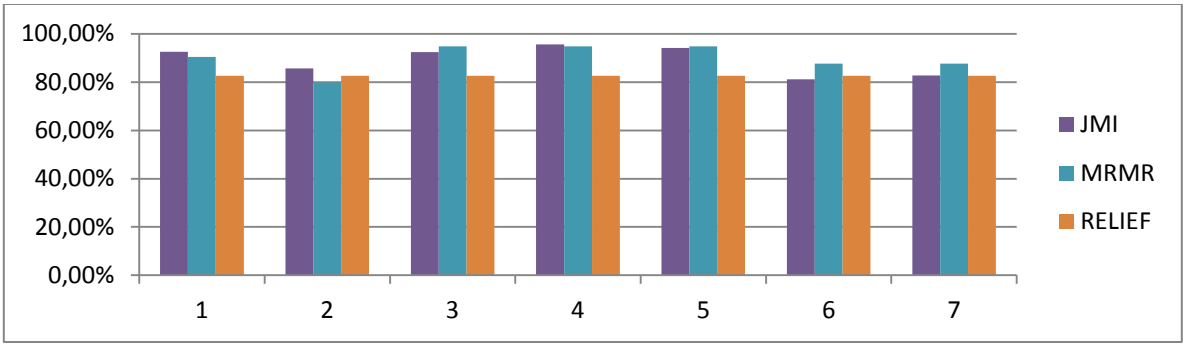

Gráfico No 14. Precisión de conjunto de técnicas FEAST-PCA para la detección de ataques tipo PROBE.

Se puede apreciar en tabla No 15 y Grafico No 14, el resultado obtenido al someter a prueba las diferentes técnicas del método FEAST- PCA, donde podemos observar la utilización de un numero especifico de características, que en este caso corresponde a “5, 10, 15, 20, 25, 30, 35” respectivamente. El número de características es seleccionado para poder tomar cual técnica tiene mayor precisión y así identificar qué conexiones de red son tipo PROBE y en qué rango de características se encuentra el mayor porcentaje de precisión.

	5	10	15	20	25	30	35	U2R
JMI	91,87%	85,80%	72,92%	74,60%	73,90%	68,74%	68,59%	
MRMR	91,87%	95,89%	95,34%	93,80%	86,21%	63,55%	63,55%	
RELIEF	56,40%	56,40%	56,40%	56,40%	56,45%	56,45%	56,45%	

Tabla No 16. Precisión de conjunto de técnicas FEAST-PCA para la detección de ataques tipo U2R

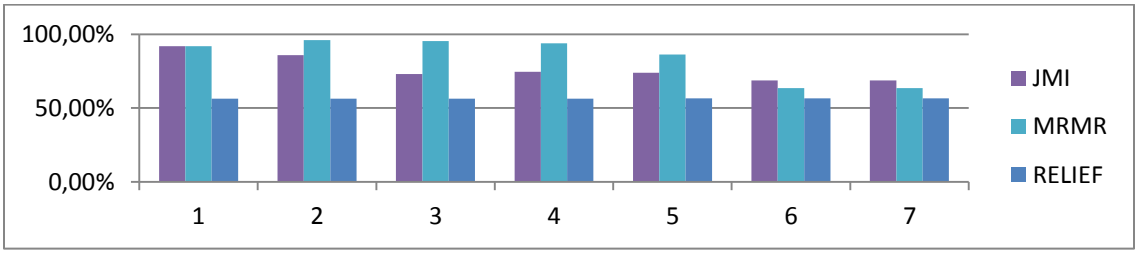

Gráfico No 15. Precisión de conjunto de técnicas FEAST-PCA para la detección de ataques tipo U2R.

Se puede apreciar en tabla No 16 y Grafico No 15, el resultado obtenido al someter a prueba las diferentes técnicas del método FEAST- PCA, donde podemos observar la utilización de un numero especifico de características, que en este caso corresponde a "5, 10, 15, 20, 25, 30, 35" respectivamente. El número de características es seleccionado para poder tomar cual técnica tiene mayor precisión y así identificar qué conexiones de red son tipo U2R y en qué rango de características se encuentra el mayor porcentaje de precisión.

	5	10	15	20	25	30	35	R2L
JMI	83,25%	82,90%	84,99%	85,30%	84,93%	81,00%	80,90%	
MRMR	83,25%	84,67%	78,35%	75,53%	85,93%	85,42%	85,42%	
RELIEF	78,40%	78,40%	87,19%	87,19%	87,19%	87,19%	87,19%	

Tabla No 17. Precisión de conjunto de técnicas FEAST-PCA para la detección de ataques tipo R2L

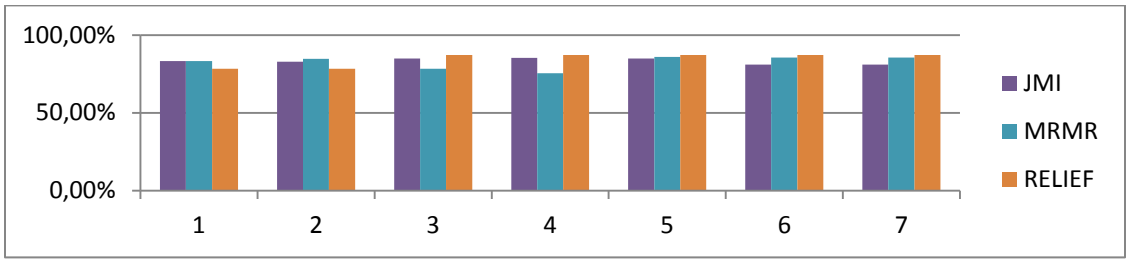

Gráfico No 16. Precisión de conjunto de técnicas FEAST-PCA para la detección de ataques tipo R2L.

Se puede apreciar en tabla No 17 y Grafico No 16, el resultado obtenido al someter a prueba las diferentes técnicas del método FEAST- PCA, donde podemos observar la utilización de un numero especifico de características, que en este caso corresponde a "5, 10, 15, 20, 25, 30, 35" respectivamente. El número de características es seleccionado para poder tomar cual técnica tiene mayor precisión y así identificar qué conexiones de red son tipo R2L y en qué rango de características se encuentra el mayor porcentaje de precisión.

5.4 PRUEBA Y VALIDACION FEAST con KPCA

Una vez seleccionada las técnicas que arrojan mayores resultados del método FEAST se procede a realizar la implementación del METODO FEAST CON KERNEL PCA tal como se puede observar en el capítulo 4. Posteriormente a la ejecución del código generado se obtiene el siguiente resultado.

	5	10	15	20	25	30	35	NORMAL
JMI	53,66%	58,61%	50,70%	61,36%	66,87%	66,22%	61,78%	
MRMR	56,92%	58,08%	44,23%	46,53%	46,34%	45,21%	57,95%	
RELIEF	43,12%	43,12%	43,12%	43,12%	43,12%	43,12%	43,12%	

Tabla No 18. Precisión de conjunto de técnicas FEAST-KPCA para la detección de Conexiones tipo NORMAL

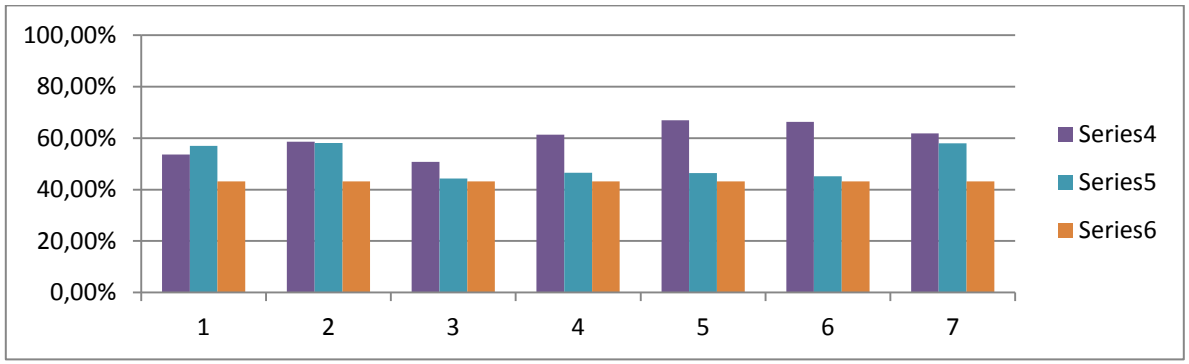

Gráfico No 17. Precisión de conjunto de técnicas FEAST-KPCA para la detección de Conexiones tipo NORMAL.

Se puede apreciar en tabla No 18 y Grafico No 17, el resultado obtenido al someter a prueba las diferentes técnicas del método FEAST- KPCA, donde podemos observar la utilización de un numero especifico de características, que en este caso corresponde a "5, 10, 15, 20, 25, 30, 35" respectivamente. El número de características es seleccionado para poder tomar cual técnica tiene mayor precisión y así identificar qué conexiones de red son tipo NORMAL y en qué rango de características se encuentra el mayor porcentaje de precisión.

	5	10	15	20	25	30	35	DOS
JMI	81,82%	85,39%	70,85%	73,58%	70,13%	70,07%	70,07%	
MRMR	82,27%	72,39%	72,80%	72,47%	72,47%	72,83%	72,80%	
RELIEF	33,37%	33,37%	33,37%	33,37%	33,37%	33,37%	33,37%	

Tabla No 19. Precisión de conjunto de técnicas FEAST-KPCA para la detección de ataques tipo DOS

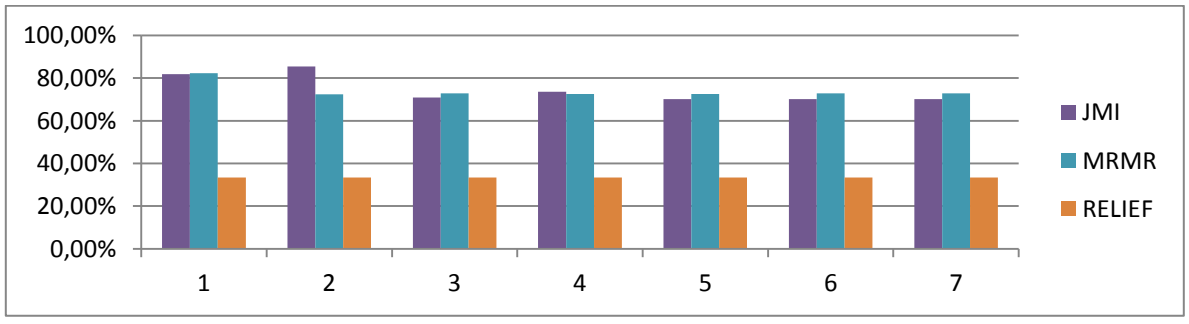

Gráfico No 18. Precisión de conjunto de técnicas FEAST-KPCA para la detección de ataques tipo DOS.

Se puede apreciar en tabla No 19 y Grafico No 18, el resultado obtenido al someter a prueba las diferentes técnicas del método FEAST- KPCA, donde podemos observar la utilización de un numero especifico de características, que en este caso corresponde a “5, 10, 15, 20, 25, 30, 35” respectivamente. El número de características es seleccionado para poder tomar cual técnica tiene mayor precisión y así identificar qué conexiones de red son tipo DOS y en qué rango de características se encuentra el mayor porcentaje de precisión.

	5	10	15	20	25	30	35	PROBE
JMI	90,57%	91,84%	92,36%	92,53%	91,69%	92,08%	91,94%	
MRMR	93,31%	93,73%	92,12%	92,06%	93,35%	92,18%	92,09%	
RELIEF	13,19%	13,19%	13,19%	13,19%	13,19%	13,19%	13,19%	

Tabla No 20. Precisión de conjunto de técnicas FEAST-KPCA para la detección de ataques tipo PROBE

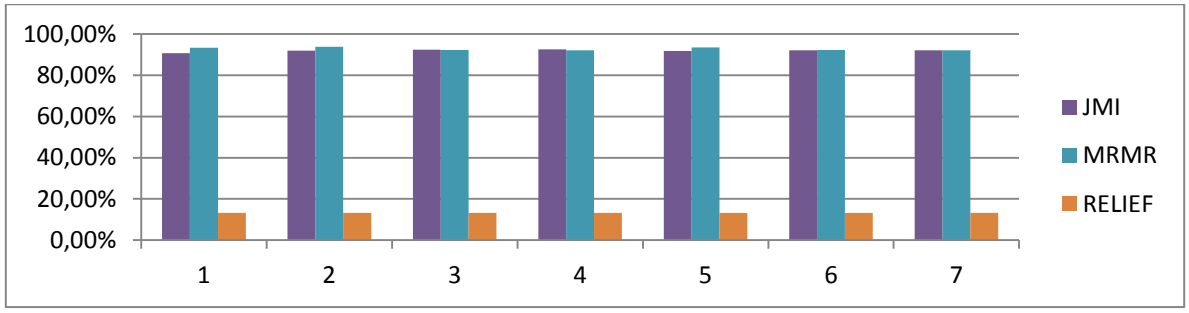

Gráfico No 19. Precisión de conjunto de técnicas FEAST-KPCA para la detección de ataques tipo PROBE

Se puede apreciar en tabla No 20 y Grafico No 19, el resultado obtenido al someter a prueba las diferentes técnicas del método FEAST- KPCA, donde podemos observar la utilización de un numero especifico de características, que en este caso corresponde a “5, 10, 15, 20, 25, 30, 35” respectivamente. El número de características es seleccionado para poder tomar cual técnica tiene mayor precisión y así identificar qué conexiones de red son tipo PROBE y en qué rango de características se encuentra el mayor porcentaje de precisión.

	5	10	15	20	25	30	35	U2R
JMI	96,55%	97,92%	99,62%	13,36%	10,46%	10,00%	9,79%	
MRMR	99,60%	99,62%	99,61%	99,61%	99,61%	99,61%	99,61%	
RELIEF	99,70%	99,70%	99,70%	99,70%	99,70%	99,70%	99,70%	

Tabla No 21. Precisión de conjunto de técnicas FEAST-KPCA para la detección de ataques tipo U2R

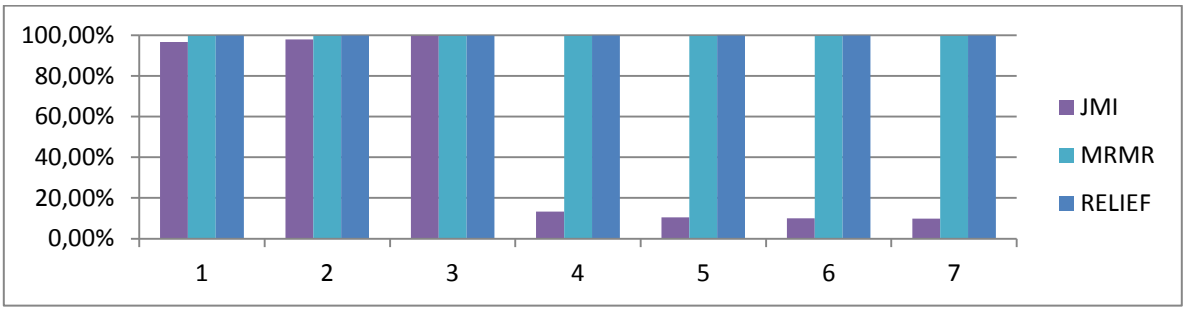

Gráfico No 20. Precisión de conjunto de técnicas FEAST-KPCA para la detección de ataques tipo U2R

Se puede apreciar en tabla No 21 y Grafico No 20, el resultado obtenido al someter a prueba las diferentes técnicas del método FEAST- KPCA, donde podemos observar la utilización de un numero especifico de características, que en este caso corresponde a "5, 10, 15, 20, 25, 30, 35" respectivamente. El número de características es seleccionado para poder tomar cual técnica tiene mayor precisión y así identificar qué conexiones de red son tipo U2R y en qué rango de características se encuentra el mayor porcentaje de precisión.

	5	10	15	20	25	30	35	R2L
JMI	87,19%	86,04%	86,65%	87,16%	39,37%	34,20%	36,58%	
MRMR	87,19%	86,96%	87,19%	87,02%	87,15%	87,00%	87,00%	
RELIEF	87,13%	87,13%	87,13%	87,13%	87,13%	87,13%	87,13%	

Tabla No 22. Precisión de conjunto de técnicas FEAST-KPCA para la detección de ataques tipo R2L

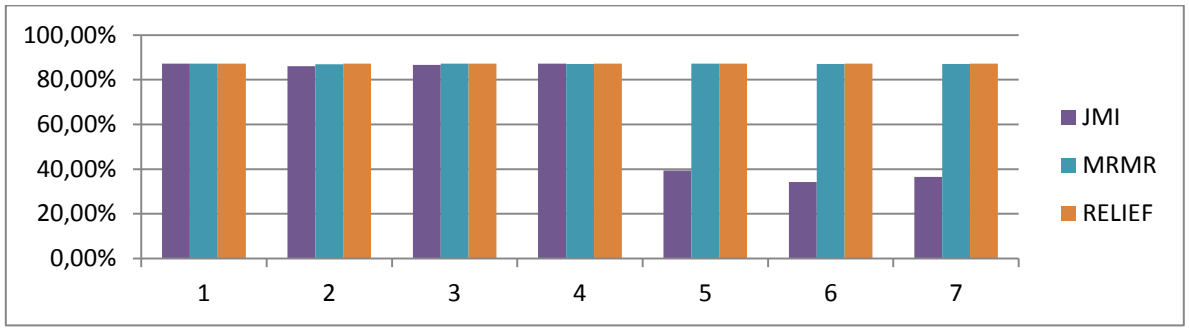

Gráfico No 21. Precisión de conjunto de técnicas FEAST-KPCA para la detección de ataques tipo R2L

Se puede apreciar en tabla No 22 y Grafico No 21, el resultado obtenido al someter a prueba las diferentes técnicas del método FEAST- KPCA, donde podemos observar la utilización de un numero especifico de características, que en este caso corresponde a "5, 10, 15, 20, 25, 30, 35" respectivamente. El número de características es seleccionado para poder tomar cual técnica tiene mayor precisión y así identificar qué conexiones de red son tipo R2L y en qué rango de características se encuentra el mayor porcentaje de precisión.

5.5 DISEÑO DE UN PROTOTIPO PARA LA DETECCIÓN DE INTRUSOS

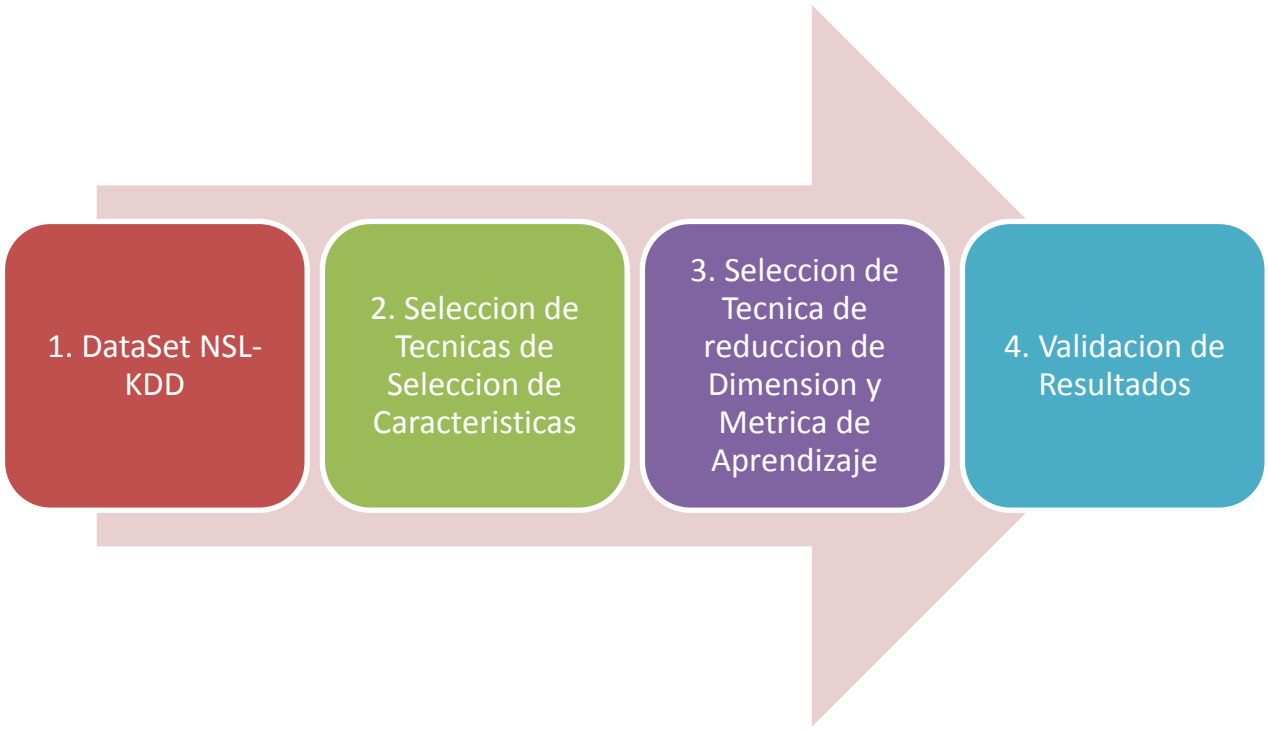

Grafico No 22. Diseño de Prototipo para la Detección de Intrusos

El diseño del prototipo para la detección de intrusos, es construido teniendo en cuenta la información contenida en el conjunto de datos (DataSet NSL-KDD), también es importante seleccionar una técnica adecuada para la Selección de Características, Reducción de Dimensión y Métricas de Aprendizaje, esto con motivos de obtener un prototipo capaz de identificar de forma adecuada los tipos de ataques dentro de una red. A continuación se describen los componentes tenidos en cuenta para la construcción del prototipo:

DataSet NSL-KDD 	El conjunto de datos es un elemento indispensable para el desarrollo del prototipo que permita la detección de intrusos. Para utilizar el DataSet se procedió primero a cargar los datos,	Carga y Preparación de 20% de datos: Ver Anexo 1 Carga y Preparación de 100% de datos: Ver Anexo 2

	posteriormente se procede a preparar los datos de tal manera que pueden ser procesados o analizados en primera instancia por las técnicas de selección de características elegidas.	
Selección de Características 	Las técnicas de selección de características son un elemento muy importante para el correcto diseño del prototipo realizado, para la selección de dichas técnicas se sometieron a rigurosas pruebas tal como se puede observar en los resultados obtenidos en el capítulo 5.	Código para la Implementación de Selección de Características: Ver Anexo 4.
Selección Técnicas de Reducción de Dimensión y Métricas de Aprendizaje	Las técnicas de Reducción de Dimensión son un elemento a tener en cuenta para el desarrollo del prototipo, para la selección de dichas técnicas se sometieron a prueba ISOMAP, PCA y KPCA tal como se puede observar en el capítulo 5.	Código para la Implementación de FEAST- ISOMAP Ver Anexo 5. Código para la Implementación de FEAST- PCA Ver Anexo 6. Código para la Implementación de FEAST- KPCA Ver Anexo 7.

Finalmente el prototipo generado tiene como técnica de selección de características MRMR, además utiliza como técnica de Reducción de Dimensión y Métrica de Aprendizaje la técnica PCA. Dichas técnicas seleccionadas arrojan resultados prometedores para la identificación de diferentes tipos de ataques en redes de computadores utilizando 5 características.

Para la generación del prototipo fue necesario superar las siguientes dificultades:

- **Capacidad de computo:** para la elección de la mejor técnica de selección de característica se colocaron a prueba 10 métodos de la toolbox de matlab FEAST y cada método fue probado con “5,10,15,20,25,30,35” características lo cual dio como necesidad tener una

capacidad de computo adecuada que permitiera llevar a cabo las pruebas de cada uno de los métodos utilizados, para dicha necesidad se utilizó el laboratorio de redes de computadores de la **Corporación Universitaria De La Costa, CUC**, que contiene 20 equipos de cómputo de última generación con las respectivas conexiones de red, a través de dispositivos como routers, switch y un servidor.

- **Elección de plataforma de desarrollo y sistema operativo:** el desarrollo prototipo requirió utilización de gran capacidad de cómputo y de una plataforma robusta con gran capacidad para realizar operaciones matemáticas, por ello se utilizó la herramienta MATLAB y el sistema operativo Linux en su distribución UBUNTU 12.04.
- **Elección de conjunto de datos:** realizar una buena selección del conjunto de datos es una tarea compleja, debido a que existen diferentes conjuntos de datos para situaciones específicas, la elección del conjunto de datos fue realizada acorde a la cantidad de registros contenidos, diferentes tipos de ataques y la cantidad de documentos a disposición para el entrenamiento y clasificación lo que género como resultado la elección del conjunto de datos NSL-KDD destacándose de otros conjuntos de datos como KDD´99 y DARPA´98.
- **Definición de rango de características:** la cantidad de características a manejar es un criterio importante para el desarrollo del prototipo, acorde a esto la dificultad presentada para la elección del rango de características se debe a que no existe un patrón que permita definir la cantidad de características que se van a utilizar, por ello fue necesario basarse en la literatura existente donde se han realizado estudios similares los cuales conducen a realizar las pruebas en rangos que incrementen de 5 en 5 tal como se realizó en la presente investigación.
- **Elección de técnicas de selección de características, reducción de dimensión y métricas de aprendizaje:** realizar el proceso de escogencia de las mejores técnicas que permitan garantizar buenos resultados por parte del prototipo generado es un proceso importante lo cual demanda gran cantidad de análisis teniendo en cuenta que se sometieron a prueba 10 métodos del FEAST y 3 técnicas de reducción de dimensión y métricas de aprendizaje. De acuerdo a lo anteriormente mencionado fue necesario en

primera instancia someter a prueba los 10 métodos del FEAST de los cuales se eligieron los 3 que mejores resultados arrojaron, posteriormente a esto para la elección de la mejor técnica de reducción de dimensión y métrica de aprendizaje fue necesario someter a prueba las mejores métodos del FEAST con las técnicas de ISOMAP, PCA y KPCA, lo que permitió desarrollar un prototipo basándose en las técnicas que mejores resultados arrojaron luego de someterlas a prueba con un mismo conjunto de datos.

Capítulo 6: Conclusiones y Trabajos Futuros

6.1 CONCLUSIONES

Los sistemas de detección de intrusos son de gran beneficio para garantizar la seguridad de los diferentes tipos de redes. La utilización de Técnicas de selección de características, Reducción de Dimensión y Métricas de Aprendizaje son de gran aporte para la construcción de **IDS** (Sistemas de Detección de Intrusos) que permitan identificar tipos de ataques de forma eficiente.

MRMR es un método perteneciente a la toolbox de matlab **FEAST**, utilizado para la selección de características el cual obtiene porcentajes de precisión considerados como **confiables** para la identificación de tipos de ataques. Los resultados logrados por el método en mención son: 85,42% para conexiones tipo NORMAL, 81,50% para conexiones tipo DOS, 90,44% para conexiones tipo PROBE, 93,21% para conexiones tipo U2R, 83,38% para conexiones tipo R2L.

Es indispensable tener en cuenta las técnicas de reducción de dimensión y métricas de aprendizaje para la construcción de un **IDS,** en el desarrollo de la investigación se someten a pruebas técnicas como ISOMAP, PCA, KPCA, las cuales son implementadas con el mejor método de selección de características **MRMR** para lograr obtener un sistema con los mejores resultados de precisión para la identificación de tipos de ataques en una red de computadores.

Los métodos **MRMR** como selección de característica y **PCA** como técnica de reducción de dimensión y métrica de aprendizaje arrojan resultados **prometedores,** para la identificación de los distintos tipos de ataques teniendo en cuenta que son utilizadas solo 5 de las 41 características posibles que contiene el conjunto de datos. Los resultados obtenidos son: 85,42% (NORMALES), 80,77% (DOS), 90,41% (PROBE), 91,87%(U2R), 83,25%(R2L),

De acuerdo a los resultados expuestos anteriormente, el desarrollo de esta investigación utiliza los métodos de **MRMR** y **PCA** para la construcción de un prototipo que permita identificar los tipos de ataques en una red de computadores, utilizando solo 5 características de las 41 que contiene el conjunto de datos NSL-KDD, obteniendo un prototipo con resultados prometedores en cotejo con técnicas como SOM y GHSOM (De la Hoz, E., 2011) utilizadas bajo el mismos ambiente de pruebas donde se obtiene resultados de 99.749046% para la identificación de patrones de trafico normal y 99.423503% para identificar los patrones de ataques anormales utilizando 41 características.

6.2 TRABAJOS FUTUROS

Como trabajo futuro, se considera la implementación en tiempo real del IDS (Sistema de Detección de Intruso), permitiendo hacer una aplicación eficaz y eficiente en la detección de conductas normales y anormales con la precisión del tiempo real así como mejoras a la implementación como lo son el bloqueo de IP en tiempo real con miras a mejorar la calidad de la prevención de intrusiones en la red.

De igual forma se piensa trabajar en la implementación de las técnicas de Selección de Características, Técnicas de reducción y métricas de aprendizaje bajo la herramienta Octave, debido a que es una aplicación de código abierto que permitiría la distribución adecuada del prototipo realizado. También se pretende como trabajo futuro la implementación del prototipo generado a partir de esta investigación utilizando Programación **MPI (**Message Passing Interface**)**.

REFERENCIAS BIBLIOGRÁFICAS

- Brumley, D.; Newsome, J.; Song, D.; Hao Wang; Jha, S., "Towards automatic generation of vulnerability-based signatures," Security and Privacy, 2006 IEEE Symposium on , vol., no., pp.15 pp.,16, 21-24 May 2006
- Huerta, A., Seguridad en Unix y redes.http://es.tldp.org/Manuales-LuCAS/SEGUNIX/unixsec-2.1.pdf, 2002.
- Heady, R., Luger, G., Maccabe, A., Servilla, M., The Architecture of a Network Level Intrusion Detection System. Technical report, Department of Computer Science, University of New Mexico, August 1990
- Powell, D., Stroud, R., Conceptual Model and Architecture, Deliverable D2, Project MAFTIA IST-1999-11583, IBM Zurich Research Laboratory Research Report RZ 3377, Nov. 2001.
- Fyodor, Network Mapping Tool [Online]. Disponible: http://www.insecure.org/nmap.org
- Institute for Internet Security [Online]. Disponible: http://www.internet-sicherheit.de/en/research/recent-projects/internet-early-warning-systems/internet-analysis-system/recent-results/
- Guo, F., Chen, J., Chiueh, T., "Spoof Detection for Preventing DoS Attacks against DNS Servers," Distributed Computing Systems, 2006. ICDCS 2006. 26th IEEE International Conference on , vol., no., pp.37,37, 2006
- Kumar, S., Classification and Detection of Computer Intrusions. Tesis de Doctorado, Purdue University, citeseer.ist.psu.edu/kumar95classification.html, 1995.
- ComputerWire: DDoS Really, Really Tested UltraDNS. Informe técnico, http://www.theregister.co.uk/2002/12/14/ddos_attack_really_really_tested/ attack really really tested, December 2002.
- Ylonen, T., SSH - Secure Login Connections over the Internet. En Proceedings of the 6th Security Symposium) (USENIX Association: Berkeley, CA), citeseer. ist.psu.edu/ylonen96ssh.html, 1996.
- Feldmeier, D., Philip, K., UNIX Password Security - Ten Years Later.

- Wu, N., Qian, Y., Chen, G., "A Novel Approach to Trojan Horse Detection by Process Tracing," Networking, Sensing and Control, 2006. ICNSC '06. Proceedings of the 2006 IEEE International Conference on , vol., no., pp.721,726, 0-0 0
- Harrald, J., Schmitt, S., Shrestha, S., "The effect of computer virus occurrence and virus threat level on antivirus companies' financial performance," Engineering Management Conference, 2004. Proceedings. 2004 IEEE International , vol.2, no., pp.780,784 Vol.2, 18-21 Oct. 2004
- Olovsson, T., 1992, A Structured Approach to Computer Security, Department of computer engineering, Chalmers University of Technology.
- Everett, D., Identity Verification and Biometrics, capítulo 10. Butterworth-Heinemann, 1992.
- Kohl, J., Neuman, B., Ts'o, T., 1994, The Evolution of the Kerberos Authentication Services. En IEEE Computer Society Press, 1994.
- Daniel, B., OSSEC. Disponible: www.ossec.net, 2006.
- Chet, H., Duren, M., September 1998, Detecting Subtle System Changes Using Digital Signatures. En Information Technology Conference, IEEE. Laboratory at Purdue University.
- Roesch, M., Lightweight Intrusion Detection for Networks. www.snort.org, 2005.
- Dain, O., Cunningham, R., September 2001, Fusing Heterogeneous Alert Streams into Scenarios. Massachusetts Institute of Technology.
- Girardin, L., USA 1999, An Eye on Network Intruder-administrator Shootouts. En Proceedings of the Workshop on Intrusion Detection and Network Monitoring.
- Anderson, J., Pennsylvania 1980, Computer Security Threat Monitoring and Surveillance.
- Siraj, A., Vaughn, R., Bridges, S., "Intrusion sensor data fusion in an intelligent intrusion detection system architecture," System Sciences, 2004. Proceedings of the 37th Annual Hawaii International Conference on , vol., no., pp.10 pp.,, 5-8 Jan. 2004
- Wu, S. , Banzhaf, W., The use of computational intelligence in intrusion detection systems: A review.2010
- Lunt, T., IDES: an intelligent system for detecting intruders. In Symposium: Computer Security, Threat and Countermeasures. Rome, Italy. 1990.

- Lunt, T., Jagannathan, R., A Prototype Real-Time Intrusion-Detection Expert System. Security and Privacy, IEEE Symposium on, 0, 59. doi: 10.1109/SECPRI.1988.8098. 1988.
- Debar, H., Dacier, M., Wespi, A., Octubre 1999, A revised taxonomy for intrusion-detection systems. IBM Research Technical Report.
- Axelsson, S., Sweden 2000, Intrusion Detection Systems: A Taxonomy and Survey. Technical Report 99-15, Dept. of Computer Engineering, Chalmers University of Technology.
- http://msdn.microsoft.com/es-es/library/ms175382.aspx
- Oporto, S., Aquino, I., Chavez, J., Perez, C., Comparacion de Cuatro Tecnicas de Selección de Caracteristicas Envolventes usando Neuronales, Arboles de Decisión, Maquinas de Vector de Soporte y Clasificador Bayesiano.
- Goldberg, D., Genetic algorithms in search, optimization, and machine learning. Addison-Wesley, 1989
- Liu, H., Motorola, H., Feature Selection for Knowledge Discovery and Data Mining. Boston: Kluwer Academy, (1998).
- http://www.cs.man.ac.uk/~gbrown/fstoolbox/
- http://homepage.tudelft.nl/19j49/Matlab_Toolbox_for_Dimensionality_Reduction.html
- Scholk, B., Smola, A., Muller, K., Nonlinear component analysis as a kernel eigenvalue problem. Neural Computation, 10:1299–1319, 1998.
- Scholk, B., Smola, A., Muller, K., Kernel principal component analysis. In B. Scholk ¨ opf, C. J. C. Burges, and A. J. Smola, editors, Advances in Kernel Methods - Support Vector Learning, pages 327–352. MIT Press, Cambridge, MA, 1999.
- Mika, S., Scholk, B., Smola, A., Muller, K., Scholz, M., Ratsch, G., Kernel PCA and de-noising in feature spaces. In M. S. Kearns, S. A. Solla, and D. A. Cohn, editors, Advancesin Neural Information Processing Systems, volume 11, pages 536– 542. MIT Press, Cambridge, MA, 1999.
- Rosipal, R., Girolami, M., Trejo, L., Kernel PCA for feature extraction and denoising in non-linear regression. Technical Report No. 4, Department of Computing and Information Systems, University of Paisley, 2000.

- Fauvel, M., Chanussot, J., Benediktsson, J., Kernel Principal Component Analysis For Feature Reduction In Hyperspectrale Images Analysis.
- Burges, C., Schölkopf and, B., Smola, A., Advances in kernel methods: Support vector machines. Cambridge, MA: MIT Press, 1999.
- Burges, C., A tutorial on support vector machines for pattern recognition. Data Mining and Knowledge Discovery, vol. 2, no. 2, 1998.
- Vapnik, V., The nature of statistical learning theory. New York: Springer-Verlag, 1995.
- Betancourt, G., 2005, Las Maquinas de Soporte Vectorial (SVMs), Universidad Tecnologica de Pereira.
- Vargas, J., Conde, B., Paccapelo, V., Zingaretti, L., Maquinas de Soporte Vectorial: Metodologia y Aplicación en R.
- Stolfo, S., Fan, W., Lee, W., Prodromidis, A., Chan, P., "Costbased modeling for fraud and intrusion detection: Results from the jam project," discex, vol. 02, p. 1130, 2000.
- Tribak, H., Febrero 2012, Análisis Estadístico de Distintas Técnicas de Inteligencia Artificial en Detección de Intrusos. Tesis Doctoral.
- Sabnani, S., Enero 2008, Computer Security: A machine learning Approach, Technical Report, University of London.
- Corbalan, L., 2006, Sistemas Inteligentes Aplicados a Redes de Datos, Universidad de la Plata.
- Catania, C., Garcia, C., 2008, Reconocimiento de Patrones en el Trafico de Red Basado en Algoritmos Geneticos, Revista Iberoamericana de Inteligencia Artificial, Vol 12, 65-75.
- Crosbie, M., Spafford, G., "Applying genetic programming to intrusion detection. " in AAAI Fall Symposium on Genetic Pro- gramming, 1995.
- Gong, R., Zulkernine, M., Abolmaesmumi, P., "A software implementation of a genetic algorithm based approach to network intrusion detection.," in Sixth Internatio- nal Conference on Software Engineering, Artificial Intelligence, Networking and Para- llel/Distributed

Computing and First ACIS International Workshop on Self-Assembling Wireless Networks (SNDP/SWAN'05), vol. 0, pp. 246–253, 2005.

- Li, W., "A genetic approach to network intrusion detection," tech. rep., SANS Institute, 2003.
- Sinclair, C., Lyn, P., Matzer, S., "An application of machine learning to network intrusion detection.," in 15th Annual Compu- ter Security Applications Conference, 1999.
- Lippmann, R., Cunningham, R., "Improving intrusion detection performance using keyword selection and neural networks", Computer Networks, 2000, vol.34,no.4,pp.597-603.
- Wuu, L., Hung, C., Chen, S., "Building intrusion pattern miner for Snort network intrusion detection system", Journal of Systems and Software, 2007, vol.80, no.10,pp.1699-1715.
- Xiaoqing, G., Beijing Vocational College of Electronic Science Beijing 100026,P.R. China
- Kuang, L., An Anomaly Intrusion Detection Method Using the CSI-KNN Algorithm School of Computing Queen's University Kingston, Canada.
- Hu, W., Liao, Y., Vemuri, V., Robust Support Vector Machines for Anomaly Detection in Computer Security. Proc. International Conference on Machine Learning and Applications, pages 23–24, 2003.
- Sinclair, C., Pierce, L., Matzner, S., An Application of Machine Learning to Network Intrusion Detection. Proc. Computer Security Applications Conference, page 371, 1999.
- Canderle, M., Aguirre, G., Piccoli, F., Un Sistema para la Detección de Intrusos basado en Agentes Autónomos, Departamento de Informática Universidad Nacional de San Luis.
- Herrera, D., Carvajal, H., IMPLEMENTACIÓN DE UNA RED NEURONAL PARA LA DETECCIÓN DE INTRUSIONES EN UNA RED TCP/IP, Revista Ingenierías USBMed, paginas 45-48, 2010.
- Kayacık, G., Zincir, N., Heywood, M., Selecting Features for Intrusion Detection: A Feature Relevance Analysis on KDD 99 Intrusion Detection Datasets , Dalhousie University, Faculty of Computer Science.
- De la Hoz, E., 2011, Aplicación de GHSOM (Growing Hierarchical Self-Organizing Maps) en Sistemas de Detección de Intrusos, Tesis de Maestria, Universidad de Granada, España.

ANEXOS

1. CODIGO PARA CARGA 20% DE DATOS

```
% GENERACION DE PMATRIZ
pmatrix(:,2)=[{'normal'},{'neptune'},{'teardrop'},{'apache2'},{'back'},{'buffer_overflow'},{'ftp_write'},{'guess_passwd'},{'httptunnel'},...{'imap'},{'ipsweep'},{'land'},{'loadmodule'},{'mailbomb'},{'mscan'},{'multihop'},{'named'},{'nmap'},{'perl'},...{'phf'},{'pod'},{'portsweep'},{'processtable'},{'ps'},{'rootkit'},{'saint'},{'satan'},{'sendmail'},{'smurf'},{'snmpgetattack'},...{'snmpguess'},{'spy'},{'sqlattack'},{'udpstorm'},{'warezclient'},{'warezmaster'},{'worm'},{'xlock'},{'xsnoop'},{'xterm'}]
;

% LISTA DE ATAQUE POR TIPOS
attack_DOS={'apache2','back','land','mailbomb','neptune','pod','processtable','smurf','teardrop','udpstorm'};
attack_PROBE={'ipsweep','mscan','nmap','portsweep','saint','satan'};
attack_U2R={'buffer_overflow','loadmodule','perl','ps','rootkit','sqlattack','xterm'};
attack_R2L={'ftp_write','guess_passwd','httptunnel','imap','multihop','named','phf','sendmail','snmpgetattack','snmpguess','spy','warezclient','warezmaster','worm','xlock','xsnoop'};

fprintf('CARGANDO DATOS DE ENTRENAMIENTO...\n');% SE CARGA EL 20%
load('KDD_20Percent.mat');

ind_normal=find(strcmp(pmatrix(:,42),'normal'));% SE CREA LA COLUMNA 42 EN LA CUAL SE DEFINE SI ES UNA CONEXION NORMAL
ind_ataque=find(~strcmp(pmatrix(:,42),'normal'));% SE CREA LA COLUMNA 42 EN LA CUAL SE DEFINE SI ES UNA CONEXION ANORMAL

% CREACION DE VECTORES QUE PERMITAN ALMACENAR LOS DIFERENTES TIPOS DE
% ATAQUES IDENTIFICADOS
ind_DOS=[];    ind_PROBE=[];      ind_U2R=[];    ind_R2L=[];

% RECORRIDO DE LOS VECTORES QUE CONTIENEN LOS DIFERENTES TIPOS DE ATAQUES
for i=1:size(attack_DOS,2)
   ind_DOS=[ind_DOS;find(strcmp(pmatrix(:,42),attack_DOS(i)))];
end;
for i=1:size(attack_PROBE,2)
   ind_PROBE=[ind_PROBE;find(strcmp(pmatrix(:,42),attack_PROBE(i)))];
end;
for i=1:size(attack_U2R,2)
   ind_U2R=[ind_U2R;find(strcmp(pmatrix(:,42),attack_U2R(i)))];
end;
for i=1:size(attack_R2L,2)
   ind_R2L=[ind_R2L;find(strcmp(pmatrix(:,42),attack_R2L(i)))];
end;

%GENERACION DE MATRICES POR CADA TIPO DE ATAQUE PARA SU RESPECTIVO
%ENTRENAMIENTO
Matriz_NORMALES= tcp_con_feat(ind_normal,:);
Matriz_ATAQUES= tcp_con_feat(ind_ataque,:);
Matriz_DOS= tcp_con_feat(ind_DOS,:);
Matriz_PROBE= tcp_con_feat(ind_PROBE,:);
Matriz_U2R= tcp_con_feat(ind_U2R,:);
Matriz_R2L= tcp_con_feat(ind_R2L,:);
Matriz= C_TCP;
```

```
fprintf('GENERACION DE MATRICES EXITOSAS...\n');
ParseoEtiquetasEntrenamiento;
fprintf('PARSEO DE ETIQUETAS EXITOSO...\n');

%ALMACENAMIENTO DE LA INFORMACION CONTENIDA EN LAS MATRICES DE
%ENTRENAMIENTO
save MatrizEntrenamiento tcp_con_feat Matriz pmatrix Matriz_NORMALES Matriz_ATAQUES Matriz_DOS Matriz_PROBE Matriz_U2R Matriz_R2L
clear all;
```

2. CODIGO PARA CARGA 100% DE DATOS

```
% GENERACION DE PMATRIZ
pmatrix(:,2)=[{'normal'},{'neptune'},{'teardrop'},{'apache2'},{'back'},{'buffer_overflow'},{'ftp_write'},{'guess_passwd'},{'httptunnel'},...{'imap'},{'ipsweep'},{'land'},{'loadmodule'},{'mailbomb'},{'mscan'},{'multihop'},{'named'},{'nmap'},{'perl'},...{'phf'},{'pod'},{'portsweep'},{'processtable'},{'ps'},{'rootkit'},{'saint'},{'satan'},{'sendmail'},{'smurf'},{'snmpgetattack'},...{'snmpguess'},{'spy'},{'sqlattack'},{'udpstorm'},{'warezclient'},{'warezmaster'},{'worm'},{'xlock'},{'xsnoop'},{'xterm'}]
;

% LISTA DE ATAQUE POR TIPOS
attack_DOS={'apache2','back','land','mailbomb','neptune','pod','processtable','smurf','teardrop','udpstorm'};
attack_PROBE={'ipsweep','mscan','nmap','portsweep','saint','satan'};
attack_U2R={'buffer_overflow','loadmodule','perl','ps','rootkit','sqlattack','xterm'};
attack_R2L={'ftp_write','guess_passwd','httptunnel','imap','multihop','named','phf','sendmail','snmpgetattack','snmpguess','spy','warezclient','warezmaster','worm','xlock','xsnoop'};

fprintf('CARGANDO DATOS DE ENTRENAMIENTO...\n');
load('KDD_100Percent.mat');

ind_normal=find(strcmp(pmatrix(:,42),'normal'));% SE CREA LA COLUMNA 42 EN LA CUAL SE DEFINE SI ES UNA CONEXION NORMAL
ind_ataque=find(~strcmp(pmatrix(:,42),'normal'));% SE CREA LA COLUMNA 42 EN LA CUAL SE DEFINE SI ES UNA CONEXION ANORMAL

% CREACION DE VECTORES QUE PERMITAN ALMACENAR LOS DIFERENTES TIPOS DE
% ATAQUES IDENTIFICADOS
ind_DOS=[];    ind_PROBE=[];      ind_U2R=[];    ind_R2L=[];

% RECORRIDO DE LOS VECTORES QUE CONTIENEN LOS DIFERENTES TIPOS DE ATAQUES
for i=1:size(attack_DOS,2)
   ind_DOS=[ind_DOS;find(strcmp(pmatrix(:,42),attack_DOS(i)))];
end;
for i=1:size(attack_PROBE,2)
   ind_PROBE=[ind_PROBE;find(strcmp(pmatrix(:,42),attack_PROBE(i)))];
end;
for i=1:size(attack_U2R,2)
   ind_U2R=[ind_U2R;find(strcmp(pmatrix(:,42),attack_U2R(i)))];
end;
for i=1:size(attack_R2L,2)
   ind_R2L=[ind_R2L;find(strcmp(pmatrix(:,42),attack_R2L(i)))];
```

```
end;

%GENERACION DE MATRICES POR CADA TIPO DE ATAQUE PARA SU RESPECTIVO
%ENTRENAMIENTO
Matriz_NORMALES= tcp_con_feat(ind_normal,:);
Matriz_ATAQUES= tcp_con_feat(ind_ataque,:);
Matriz_DOS= tcp_con_feat(ind_DOS,:);
Matriz_PROBE= tcp_con_feat(ind_PROBE,:);
Matriz_U2R= tcp_con_feat(ind_U2R,:);
Matriz_R2L= tcp_con_feat(ind_R2L,:);
Matriz= C_TCP;
fprintf('GENERACION DE MATRICES EXITOSAS...\n');
ParseoEtiquetasEntrenamiento;
fprintf('PARSEO DE ETIQUETAS EXITOSO...\n');

%ALMACENAMIENTO DE LA INFORMACION CONTENIDA EN LAS MATRICES DE
%ENTRENAMIENTO
save MatrizEntrenamiento tcp_con_feat Matriz pmatrix Matriz_NORMALES Matriz_ATAQUES Matriz_DOS Matriz_PROBE Matriz_U2R Matriz_R2L
clear all;
```

3. CODIGO PARA LA GENERACION DE ETIQUETAS DE TIPOS DE ATAQUES

```
load MatricesDeEntrenamiento.mat

[x y] = size(MatrizCompletaDeEntrenamiento);
n=1;
d=1;
p=1;
r=1;
u=1;

for i=1:y %for que recorre la Matriz 'labels' o matriz de entrada

  LabesTodosLosTipos(i) = MatrizCompletaDeEntrenamiento(i).ID;

  if (strcmp(MatrizCompletaDeEntrenamiento(i).label,'normal'))
   Labes2Tipos(i) = 0;
  else
   Labes2Tipos(i) = 1;
  end

  if (strcmp(MatrizCompletaDeEntrenamiento(i).TipoDeAtaque,'normal'))
   Labes5Tipos(i) = 0;
   LabelsNormalesPos(n) = i;
   n=n+1;
  else
     if (strcmp(MatrizCompletaDeEntrenamiento(i).TipoDeAtaque,'DOS'))
```

```
    Labes5Tipos(i) = 1;
    LabelsDOSPos(d) = i;
    d=d+1;
    else
      if (strcmp(MatrizCompletaDeEntrenamiento(i).TipoDeAtaque,'PROBE'))
      Labes5Tipos(i) = 2;
      LabelsPROBEPos(p) = i;
      p=p+1;
      else
        if (strcmp(MatrizCompletaDeEntrenamiento(i).TipoDeAtaque,'R2L'))
        Labes5Tipos(i) = 3;
        LabelsR2LPos(r) = i;
        r=r+1;
        else
          if (strcmp(MatrizCompletaDeEntrenamiento(i).TipoDeAtaque,'U2R'))
          Labes5Tipos(i) = 4;
          LabelsU2RPos(u) = i;
          u=u+1;
          end
        end
      end
    end
  end

end;
 clear i;
 clear x;
 clear y;
 clear j;
 clear n;
 clear d;
 clear p;
 clear r;
 clear u;
```

4. CODIGO IMPLEMENTACION METODO FEAST

```
fprintf('Aplicando FEAST...\n');

%%%%% Creación De Etiqueta Que Maneja Los Diferentes Tipos De Ataques
labels5=labels5t;
%%%%%

%%%%% Implementación del método ICAP Perteneciente a la técnica de selección
 %%%%% de característica FEAST

%%%%% labNOR=(labels5==0) contiene los valores de características tipo Normal
%%%%% F_labNOR=double(labNOR) convierte los datos de labNOR a tipo doublé
```

```
%%%%% feat_NOR = feast('icap',25,D,F_labNOR')' feat_NOR recibe como parámetro el método
%%%%% feast con el numero de características que debe seleccionar, la matriz que contiene los
%%%%%registros de conexiones y por ultimo las etiquetas que tiene los diferentes tipos de
%%%%%ataques. El procedimiento se realiza igual para los ataques tipo DOS, PROBE, U2R, R2L

labNOR=(labels5==0);
F_labNOR=double(labNOR);
feat_NOR = feast('icap',25,D,F_labNOR')';

labDOS=(labels5==1);
F_labDOS=double(labDOS);
feat_DOS = feast('icap',25,D,F_labDOS')';

labPRO=(labels5==2);
F_labPRO=double(labPRO);
feat_PRO = feast('icap',25,D,F_labPRO')';

labR2L=(labels5==3);
F_labR2L=double(labR2L);
feat_R2L = feast('icap',25,D,F_labR2L')';

labU2R=(labels5==4);
F_labU2R=double(labU2R);
feat_U2R = feast('icap',25,D,F_labU2R')';

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
fprintf('Entrenando SVMs...\n');

% A continuación se implementa el código necesario para la entrenamiento de las SVM

options = optimset('maxiter',10000);
svm0 = svmtrain(D(:,feat_NOR),labNOR,'kernel_function','rbf','rbf_sigma',7,'showplot',true,'options',options);fprintf('SVM 1 OK...\n');
svm1 = svmtrain(D(:,feat_DOS),labDOS,'kernel_function','rbf','rbf_sigma',7,'showplot',true,'options',options);fprintf('SVM 2 OK...\n');
svm2 = svmtrain(D(:,feat_PRO),labPRO,'kernel_function','rbf','rbf_sigma',7,'showplot',true,'options',options);fprintf('SVM 3 OK...\n');
svm3 = svmtrain(D(:,feat_R2L),labR2L,'kernel_function','rbf','rbf_sigma',7,'showplot',true,'options',options);fprintf('SVM 4 OK...\n');
svm4 = svmtrain(D(:,feat_U2R),labU2R,'kernel_function','rbf','rbf_sigma',7,'showplot',true,'options',options);fprintf('SVM 5 OK...\n');

% A continuación se implementa el código necesario para la clasificación de los diferentes tipos de ataques
fprintf('Clasificando...\n');
labels5=labels5T;

%%%%% CLASIFICACION DE CONEXIONES TIPO NORMAL
fprintf('Buscando conexiones Normales...\n');
lt0=(labels5==0);
a=round(linspace(1,size(DT,1),20));
a(1)=1;
pred=zeros(size(DT,1),1);

DT_NOR=(DT(:,feat_NOR));
```

```
for i=2:20
   fprintf('%d : %d\n',a(i-1),a(i));
   pred(a(i-1):a(i)) = svmclassify(svm0,DT_NOR(a(i-1):a(i),:),'showplot',false);
end;
[acc_NOR sens_NOR spec_NOR ppv_NOR npv_NOR LRp_NOR LRn_NOR] = contingency_table(lt0,pred);


%%%%% CLASIFICACION DE CONEXIONES TIPO DOS
fprintf('Buscando conexiones DOS...\n');
lt1=(labels5==1);
a=round(linspace(1,size(DT,1),20));
a(1)=1;
pred=zeros(size(DT,1),1);

DT_DOS=(DT(:,feat_DOS));

for i=2:20
   fprintf('%d : %d\n',a(i-1),a(i));
   pred(a(i-1):a(i)) = svmclassify(svm1,DT_DOS(a(i-1):a(i),:),'showplot',false);
end;
[acc_DOS sens_DOS spec_DOS ppv_DOS npv_DOS LRp_DOS LRn_DOS] = contingency_table(lt1,pred);


%%%%% CLASIFICACION DE CONEXIONES TIPO PROBE
fprintf('Buscando conexiones PROBE...\n');
lt2=(labels5==2);
a=round(linspace(1,size(DT,1),20));
a(1)=1;
pred=zeros(size(DT,1),1);

DT_PRO=(DT(:,feat_PRO));

for i=2:20
   fprintf('%d : %d\n',a(i-1),a(i));
   pred(a(i-1):a(i)) = svmclassify(svm2,DT_PRO(a(i-1):a(i),:),'showplot',false);
end;
[acc_PRO sens_PRO spec_PRO ppv_PRO npv_PRO LRp_PRO LRn_PRO] = contingency_table(lt2,pred);

%%%%% CLASIFICACION DE CONEXIONES TIPO R2L
fprintf('Buscando conexiones R2L...\n');
lt3=(labels5==3);
a=round(linspace(1,size(DT,1),20));
a(1)=1;
pred=zeros(size(DT,1),1);

DT_R2L=(DT(:,feat_R2L));

for i=2:20
```

```
    fprintf('%d : %d\n',a(i-1),a(i));
    pred(a(i-1):a(i)) = svmclassify(svm3,DT_R2L(a(i-1):a(i),:),'showplot',false);
end;
[acc_R2L sens_R2L spec_R2L ppv_R2L npv_R2L LRp_R2L LRn_R2L] = contingency_table(lt3,pred);

%%%%% CLASIFICACION DE CONEXIONES TIPO U2R
fprintf('Buscando conexiones U2R...\n');
lt4=(labels5==4);
a=round(linspace(1,size(DT,1),20));
a(1)=1;
pred=zeros(size(DT,1),1);

DT_U2R=(DT(:,feat_U2R));

for i=2:20
    fprintf('%d : %d\n',a(i-1),a(i));
    pred(a(i-1):a(i)) = svmclassify(svm4,DT_U2R(a(i-1):a(i),:),'showplot',false);
end;
[acc_U2R sens_U2R spec_U2R ppv_U2R npv_U2R LRp_U2R LRn_U2R] = contingency_table(lt4,pred);

%ALMACENAMIENTO DE LOS RESULTADOS TENIENDO EN CUENTA PRECISIÓN, SENSIBILIDAD, ESPECIFICIDAD
ACC=[acc_NOR acc_DOS acc_PRO acc_R2L acc_U2R];
SENS=[sens_NOR sens_DOS sens_PRO sens_R2L sens_U2R];
SPEC=[spec_NOR spec_DOS spec_PRO spec_R2L spec_U2R];
```

5. CODIGO IMPLEMENTACION FEAST CON ISOMAP

```
fprintf('Aplicando FEAST...\n');

%%%%% Creación De Etiqueta Que Maneja Los Diferentes Tipos De Ataques
labels5=labels5t;
%%%%%

%%%%% Implementación del método CIFE Perteneciente a la técnica de selección
 %%%%% de característica FEAST

%%%%% labNOR=(labels5==0) contiene los valores de características tipo Normal
%%%%% F_labNOR=double(labNOR) convierte los datos de labNOR a tipo doublé
%%%%% feat_NOR = feast(cife,25,D,F_labNOR')' feat_NOR recibe como parámetro el método
%%%%% feast con el numero de características que debe seleccionar, la matriz que contiene los
%%%%%registros de conexiones y por ultimo las etiquetas que tiene los diferentes tipos de
%%%%%ataques. El procedimiento se realiza igual para los ataques tipo DOS, PROBE, U2R, R2L

labNOR=(labels5==0);
F_labNOR=double(labNOR);
feat_NOR = feast('cife',5,D,F_labNOR')';

labDOS=(labels5==1);
```

```
F_labDOS=double(labDOS);
feat_DOS = feast('cife',5,D,F_labDOS')';

labPRO=(labels5==2);
F_labPRO=double(labPRO);
feat_PRO = feast('cife',5,D,F_labPRO')';

labR2L=(labels5==3);
F_labR2L=double(labR2L);
feat_R2L = feast('cife',5,D,F_labR2L')';

labU2R=(labels5==4);
F_labU2R=double(labU2R);
feat_U2R = feast('cife',5,D,F_labU2R')';

% Implementación de técnica de reducción de dimensión y métrica de aprendizaje
 ISOMAP
[mappedX mapping_NOR] = landmark_isomap(D(:,feat_NOR),nfeat,25,0.8);
mapping_NOR.name='LandmarkIsomap';
D_NOR=out_of_sample(D(:,feat_NOR), mapping_NOR);
[mappedX mapping_DOS] = landmark_isomap(D(:,feat_DOS),nfeat,25,0.8);
mapping_DOS.name='LandmarkIsomap';
D_DOS=out_of_sample(D(:,feat_DOS), mapping_DOS);
[mappedX mapping_PRO] = landmark_isomap(D(:,feat_PRO),nfeat,25,0.8);
mapping_PRO.name='LandmarkIsomap';
D_PRO=out_of_sample(D(:,feat_PRO), mapping_PRO);
[mappedX mapping_R2L] = landmark_isomap(D(:,feat_R2L),nfeat,25,0.8);
mapping_R2L.name='LandmarkIsomap';
D_R2L=out_of_sample(D(:,feat_R2L), mapping_R2L);
[mappedX mapping_U2R] = landmark_isomap(D(:,feat_U2R),nfeat,25,0.8);
mapping_U2R.name='LandmarkIsomap';
D_U2R=out_of_sample(D(:,feat_U2R), mapping_U2R);

%Codigo Implementado para el Entrenamiento de los SVM

fprintf('Entrenando SVMs...\n');

options = optimset('maxiter',10000);
svm0 = svmtrain(D_NOR,labNOR,'kernel_function','rbf','rbf_sigma',7,'showplot',true,'options',options);fprintf('SVM 1 OK...\n');
svm1 = svmtrain(D_DOS,labDOS,'kernel_function','rbf','rbf_sigma',7,'showplot',true,'options',options);fprintf('SVM 2 OK...\n');
svm2 = svmtrain(D_PRO,labPRO,'kernel_function','rbf','rbf_sigma',7,'showplot',true,'options',options);fprintf('SVM 3 OK...\n');
svm3 = svmtrain(D_R2L,labR2L,'kernel_function','rbf','rbf_sigma',7,'showplot',true,'options',options);fprintf('SVM 4 OK...\n');
svm4 = svmtrain(D_U2R,labU2R,'kernel_function','rbf','rbf_sigma',7,'showplot',true,'options',options);fprintf('SVM 5 OK...\n');

%%%%% A continuación inicia el Proceso de Clasificación para los diferentes tipos de ataques
fprintf('Clasificando...\n');
labels5=labels5T;
```

```
%%%%% CLASIFICACION DE CONEXIONES TIPO NORMAL
fprintf('Buscando conexiones Normales...\n');
lt0=(labels5==0);
a=round(linspace(1,size(DT,1),20));
a(1)=1;
pred=zeros(size(DT,1),1);
DT_NOR=out_of_sample(DT(:,feat_NOR), mapping_NOR);
for i=2:20
   fprintf('%d : %d\n',a(i-1),a(i));
   pred(a(i-1):a(i)) = svmclassify(svm0,DT_NOR(a(i-1):a(i),:),'showplot',false);
end;
[acc_NOR sens_NOR spec_NOR ppv_NOR npv_NOR LRp_NOR LRn_NOR] = contingency_table(lt0,pred);

%%%%% CLASIFICACION DE CONEXIONES TIPO DOS
fprintf('Buscando conexiones DOS...\n');
lt1=(labels5==1);
a=round(linspace(1,size(DT,1),20));
a(1)=1;
pred=zeros(size(DT,1),1);
DT_DOS=out_of_sample(DT(:,feat_DOS), mapping_DOS);
for i=2:20
   fprintf('%d : %d\n',a(i-1),a(i));
   pred(a(i-1):a(i)) = svmclassify(svm1,DT_DOS(a(i-1):a(i),:),'showplot',false);
end;
[acc_DOS sens_DOS spec_DOS ppv_DOS npv_DOS LRp_DOS LRn_DOS] = contingency_table(lt1,pred);

%%%%% CLASIFICACION DE CONEXIONES TIPO PROBE
fprintf('Buscando conexiones PROBE...\n');
lt2=(labels5==2);
a=round(linspace(1,size(DT,1),20));
a(1)=1;
pred=zeros(size(DT,1),1);
DT_PRO=out_of_sample(DT(:,feat_PRO), mapping_PRO);
for i=2:20
   fprintf('%d : %d\n',a(i-1),a(i));
   pred(a(i-1):a(i)) = svmclassify(svm2,DT_PRO(a(i-1):a(i),:),'showplot',false);
end;
[acc_PRO sens_PRO spec_PRO ppv_PRO npv_PRO LRp_PRO LRn_PRO] = contingency_table(lt2,pred);

%%%%% CLASIFICACION DE CONEXIONES TIPO R2L
fprintf('Buscando conexiones R2L...\n');
lt3=(labels5==3);
a=round(linspace(1,size(DT,1),20));
a(1)=1;
pred=zeros(size(DT,1),1);
DT_R2L=out_of_sample(DT(:,feat_R2L), mapping_R2L);
for i=2:20
   fprintf('%d : %d\n',a(i-1),a(i));
   pred(a(i-1):a(i)) = svmclassify(svm3,DT_R2L(a(i-1):a(i),:),'showplot',false);
```

```
end;
[acc_R2L sens_R2L spec_R2L ppv_R2L npv_R2L LRp_R2L LRn_R2L] = contingency_table(lt3,pred);

%%%%% CLASIFICACION DE CONEXIONES TIPO U2R
fprintf('Buscando conexiones U2R...\n');
lt4=(labels5==4);
a=round(linspace(1,size(DT,1),20));
a(1)=1;
pred=zeros(size(DT,1),1);
DT_U2R=out_of_sample(DT(:,feat_U2R), mapping_U2R);
for i=2:20
   fprintf('%d : %d\n',a(i-1),a(i));
   pred(a(i-1):a(i)) = svmclassify(svm4,DT_U2R(a(i-1):a(i),:),'showplot',false);
end;
[acc_U2R sens_U2R spec_U2R ppv_U2R npv_U2R LRp_U2R LRn_U2R] = contingency_table(lt4,pred);

%ALMACENAMIENTO DE LOS RESULTADOS TENIENDO EN CUENTA PRECISIÓN, SENSIBILIDAD, ESPECIFICIDAD
ACC=[acc_NOR acc_DOS acc_PRO acc_R2L acc_U2R];
SENS=[sens_NOR sens_DOS sens_PRO sens_R2L sens_U2R];
SPEC=[spec_NOR spec_DOS spec_PRO spec_R2L spec_U2R];
```

6. CODIGO IMPLEMENTACION FEAST CON PCA

```
fdr_thr=0.05;

%%%%% Creación De Etiqueta Que Maneja Los Diferentes Tipos De Ataques
labels5=labels5t;

%%%%% Implementación del método JMI Perteneciente a la técnica de selección
 %%%%% de característica FEAST

%%%%% labNOR=(labels5==0) contiene los valores de características tipo Normal
%%%%% F_labNOR=double(labNOR) convierte los datos de labNOR a tipo doublé
%%%%% feat_NOR = feast(jmi,20,D,F_labNOR')' feat_NOR recibe como parámetro el método
%%%%% feast con el numero de características que debe seleccionar, la matriz que contiene los
%%%%%registros de conexiones y por ultimo las etiquetas que tiene los diferentes tipos de
%%%%%ataques. El procedimiento se realiza igual para los ataques tipo DOS, PROBE, U2R, R2L

labNOR=(labels5==0);
F_labNOR=double(labNOR);
feat_NOR = feast('jmi',20,D,F_labNOR')';

labDOS=(labels5==1);
F_labDOS=double(labDOS);
feat_DOS = feast('jmi',20,D,F_labDOS')';

labPRO=(labels5==2);
F_labPRO=double(labPRO);
```

```
feat_PRO = feast('jmi',20,D,F_labPRO')';

labR2L=(labels5==3);
F_labR2L=double(labR2L);
feat_R2L = feast('jmi',20,D,F_labR2L')';

labU2R=(labels5==4);
F_labU2R=double(labU2R);
feat_U2R = feast('jmi',20,D,F_labU2R')';

% Implementación de técnica de reducción de dimensión y métrica de aprendizaje  PCA
 [ev_NOR score lat]=princomp(D(:,feat_NOR),'econ');
D_NOR=D(:,feat_NOR)*ev_NOR(:,1:nev);
[ev_DOS score lat]=princomp(D(:,feat_DOS),'econ');
D_DOS=D(:,feat_DOS)*ev_DOS(:,1:nev);
[ev_PRO score lat]=princomp(D(:,feat_PRO),'econ');
D_PRO=D(:,feat_PRO)*ev_PRO(:,1:nev);
[ev_R2L score lat]=princomp(D(:,feat_R2L),'econ');
D_R2L=D(:,feat_R2L)*ev_R2L(:,1:nev);
[ev_U2R score lat]=princomp(D(:,feat_U2R),'econ');
D_U2R=D(:,feat_U2R)*ev_U2R(:,1:nev);

%Codigo Implementado para el Entrenamiento de los SVM
fprintf('Entrenando SVMs...\n');

options = optimset('maxiter',10000);
svm0 = svmtrain(D_NOR,labNOR,'kernel_function','rbf','rbf_sigma',5,'showplot',true,'options',options);fprintf('SVM 1 OK...\n');
svm1 = svmtrain(D_DOS,labDOS,'kernel_function','rbf','rbf_sigma',5,'showplot',true,'options',options);fprintf('SVM 2 OK...\n');
svm2 = svmtrain(D_PRO,labPRO,'kernel_function','rbf','rbf_sigma',5,'showplot',true,'options',options);fprintf('SVM 3 OK...\n');
svm3 = svmtrain(D_R2L,labR2L,'kernel_function','rbf','rbf_sigma',5,'showplot',true,'options',options);fprintf('SVM 4 OK...\n');
svm4 = svmtrain(D_U2R,labU2R,'kernel_function','rbf','rbf_sigma',5,'showplot',true,'options',options);fprintf('SVM 5 OK...\n');

 %%%%% A continuación inicia el Proceso de Clasificación para los diferentes tipos de ataques
fprintf('Clasificando...\n');
labels5=labels5T;

%%%%% CLASIFICACION DE CONEXIONES TIPO NORMAL
fprintf('Buscando conexiones Normales...\n');
lt0=(labels5==0);
a=round(linspace(1,size(DT,1),20));
a(1)=1;
pred=zeros(size(DT,1),1);
DT_NOR=DT(:,feat_NOR)*ev_NOR(:,1:nev);
for i=2:20
    fprintf('%d : %d\n',a(i-1),a(i));
    pred(a(i-1):a(i)) = svmclassify(svm0,DT_NOR(a(i-1):a(i),:),'showplot',false);
end;
[acc_NOR sens_NOR spec_NOR ppv_NOR npv_NOR LRp_NOR LRn_NOR] = contingency_table(lt0,pred);
%%%%% CLASIFICACION DE CONEXIONES TIPO DOS
```

```
fprintf('Buscando conexiones DOS...\n');
lt1=(labels5==1);
a=round(linspace(1,size(DT,1),20));
a(1)=1;
pred=zeros(size(DT,1),1);
DT_DOS=DT(:,feat_DOS)*ev_DOS(:,1:nev);
for i=2:20
   fprintf('%d : %d\n',a(i-1),a(i));
   pred(a(i-1):a(i)) = svmclassify(svm1,DT_DOS(a(i-1):a(i),:),'showplot',false);
end;
[acc_DOS sens_DOS spec_DOS ppv_DOS npv_DOS LRp_DOS LRn_DOS] = contingency_table(lt1,pred);

%%%%% CLASIFICACION DE CONEXIONES TIPO PROBE
fprintf('Buscando conexiones PROBE...\n');
lt2=(labels5==2);
a=round(linspace(1,size(DT,1),20));
a(1)=1;
pred=zeros(size(DT,1),1);
DT_PRO=DT(:,feat_PRO)*ev_PRO(:,1:nev);
for i=2:20
   fprintf('%d : %d\n',a(i-1),a(i));
   pred(a(i-1):a(i)) = svmclassify(svm2,DT_PRO(a(i-1):a(i),:),'showplot',false);
end;
[acc_PRO sens_PRO spec_PRO ppv_PRO npv_PRO LRp_PRO LRn_PRO] = contingency_table(lt2,pred);

%%%%% CLASIFICACION DE CONEXIONES TIPO R2L
fprintf('Buscando conexiones R2L...\n');
lt3=(labels5==3);
a=round(linspace(1,size(DT,1),20));
a(1)=1;
pred=zeros(size(DT,1),1);
DT_R2L=DT(:,feat_R2L)*ev_R2L(:,1:nev);
for i=2:20
   fprintf('%d : %d\n',a(i-1),a(i));
   pred(a(i-1):a(i)) = svmclassify(svm3,DT_R2L(a(i-1):a(i),:),'showplot',false);
end;
[acc_R2L sens_R2L spec_R2L ppv_R2L npv_R2L LRp_R2L LRn_R2L] = contingency_table(lt3,pred);

%%%%% CLASIFICACION DE CONEXIONES TIPO U2R
fprintf('Buscando conexiones U2R...\n');
lt4=(labels5==4);
a=round(linspace(1,size(DT,1),20));
a(1)=1;
pred=zeros(size(DT,1),1);
DT_U2R=DT(:,feat_U2R)*ev_U2R(:,1:nev);
for i=2:20
   fprintf('%d : %d\n',a(i-1),a(i));
   pred(a(i-1):a(i)) = svmclassify(svm4,DT_U2R(a(i-1):a(i),:),'showplot',false);
end;
```

```
[acc_U2R sens_U2R spec_U2R ppv_U2R npv_U2R LRp_U2R LRn_U2R] = contingency_table(lt4,pred);

%ALMACENAMIENTO DE LOS RESULTADOS TENIENDO EN CUENTA PRECISIÓN, SENSIBILIDAD, ESPECIFICIDAD
ACC=[acc_NOR acc_DOS acc_PRO acc_R2L acc_U2R];
SENS=[sens_NOR sens_DOS sens_PRO sens_R2L sens_U2R];
SPEC=[spec_NOR spec_DOS spec_PRO spec_R2L spec_U2R];
```

7. CODIGO IMPLEMENTACION FEAST CON KPCA

```
fdr_thr=0.05;

%%%%% Creación De Etiqueta Que Maneja Los Diferentes Tipos De Ataques
labels5=labels5t;

%%%%% Implementación del método JMI Perteneciente a la técnica de selección
 %%%%% de característica FEAST

%%%%% labNOR=(labels5==0) contiene los valores de características tipo Normal
%%%%% F_labNOR=double(labNOR) convierte los datos de labNOR a tipo doublé
%%%%% feat_NOR = feast(jmi,5,D,F_labNOR')' feat_NOR recibe como parámetro el método
%%%%% feast con el numero de características que debe seleccionar, la matriz que contiene los
%%%%%registros de conexiones y por ultimo las etiquetas que tiene los diferentes tipos de
%%%%%ataques. El procedimiento se realiza igual para los ataques tipo DOS, PROBE, U2R, R2L

labNOR=(labels5==0);
F_labNOR=double(labNOR);
feat_NOR = feast('jmi',5,D,F_labNOR')';

labDOS=(labels5==1);
F_labDOS=double(labDOS);
feat_DOS = feast('jmi',5,D,F_labDOS')';

labPRO=(labels5==2);
F_labPRO=double(labPRO);
feat_PRO = feast('jmi',5,D,F_labPRO')';

labR2L=(labels5==3);
F_labR2L=double(labR2L);
feat_R2L = feast('jmi',5,D,F_labR2L')';

labU2R=(labels5==4);
F_labU2R=double(labU2R);
feat_U2R = feast('jmi',5,D,F_labU2R')';

% Implementación de técnica de reducción de dimensión y métrica de aprendizaje  KPCA
tec='KernelPCA';
param='poly'
[mappedX, mapping_NOR] = compute_mapping(D(:,feat_NOR), tec, nfeat,param,0,3);
mapping_NOR.name=tec;
```

```
D_NOR=out_of_sample(D(:,feat_NOR), mapping_NOR);
[mappedX, mapping_DOS] = compute_mapping(D(:,feat_DOS), tec, nfeat,param,0,3);
mapping_DOS.name=tec;
D_DOS=out_of_sample(D(:,feat_DOS), mapping_DOS);
[mappedX, mapping_PRO] = compute_mapping(D(:,feat_PRO), tec, nfeat,param,0,3);
mapping_PRO.name=tec;
D_PRO=out_of_sample(D(:,feat_PRO), mapping_PRO);
[mappedX, mapping_R2L] = compute_mapping(D(:,feat_R2L), tec, nfeat,param,0,3);
mapping_R2L.name=tec;
D_R2L=out_of_sample(D(:,feat_R2L), mapping_R2L);
[mappedX, mapping_U2R] = compute_mapping(D(:,feat_U2R), tec, nfeat,param,0,3);
mapping_U2R.name=tec;
D_U2R=out_of_sample(D(:,feat_U2R), mapping_U2R);

 %Codigo Implementado para el Entrenamiento de los SVM
fprintf('Entrenando SVMs...\n');

options = optimset('maxiter',10000);
svm0 = svmtrain(D_NOR,labNOR,'kernel_function','rbf','rbf_sigma',5,'showplot',false,'options',options);fprintf('SVM 1 OK...\n');
svm1 = svmtrain(D_DOS,labDOS,'kernel_function','rbf','rbf_sigma',5,'showplot',false,'options',options);fprintf('SVM 2 OK...\n');
svm2 = svmtrain(D_PRO,labPRO,'kernel_function','rbf','rbf_sigma',5,'showplot',false,'options',options);fprintf('SVM 3 OK...\n');
svm3 = svmtrain(D_R2L,labR2L,'kernel_function','rbf','rbf_sigma',5,'showplot',false,'options',options);fprintf('SVM 4 OK...\n');
svm4 = svmtrain(D_U2R,labU2R,'kernel_function','rbf','rbf_sigma',5,'showplot',false,'options',options);fprintf('SVM 5 OK...\n');

%%%%% A continuación inicia el Proceso de Clasificación para los diferentes tipos de ataques
fprintf('Clasificando...\n');
labels5=labels5T;

%%%%% CLASIFICACION DE CONEXIONES TIPO NORMAL
fprintf('Buscando conexiones Normales...\n');
lt0=(labels5==0);
a=round(linspace(1,size(DT,1),20));
a(1)=1;
pred=zeros(size(DT,1),1);
DT_NOR=out_of_sample(DT(:,feat_NOR), mapping_NOR);
for i=2:20
   fprintf('%d : %d\n',a(i-1),a(i));
   pred(a(i-1):a(i)) = svmclassify(svm0,DT_NOR(a(i-1):a(i),:),'showplot',false);
end;
[acc_NOR sens_NOR spec_NOR ppv_NOR npv_NOR LRp_NOR LRn_NOR] = contingency_table(lt0,pred);

%%%%% CLASIFICACION DE CONEXIONES TIPO DOS
fprintf('Buscando conexiones DOS...\n');
lt1=(labels5==1);
a=round(linspace(1,size(DT,1),20));
a(1)=1;
pred=zeros(size(DT,1),1);
DT_DOS=out_of_sample(DT(:,feat_DOS), mapping_DOS);
for i=2:20
```

```
    fprintf('%d : %d\n',a(i-1),a(i));
    pred(a(i-1):a(i)) = svmclassify(svm1,DT_DOS(a(i-1):a(i),:),'showplot',false);
end;
[acc_DOS sens_DOS spec_DOS ppv_DOS npv_DOS LRp_DOS LRn_DOS] = contingency_table(lt1,pred);

%%%%% CLASIFICACION DE CONEXIONES TIPO PROBE
fprintf('Buscando conexiones PROBE...\n');
lt2=(labels5==2);
a=round(linspace(1,size(DT,1),20));
a(1)=1;
pred=zeros(size(DT,1),1);
DT_PRO=out_of_sample(DT(:,feat_PRO), mapping_PRO);
for i=2:20
    fprintf('%d : %d\n',a(i-1),a(i));
    pred(a(i-1):a(i)) = svmclassify(svm2,DT_PRO(a(i-1):a(i),:),'showplot',false);
end;
[acc_PRO sens_PRO spec_PRO ppv_PRO npv_PRO LRp_PRO LRn_PRO] = contingency_table(lt2,pred);

%%%%% CLASIFICACION DE CONEXIONES TIPO R2L
fprintf('Buscando conexiones R2L...\n');
lt3=(labels5==3);
a=round(linspace(1,size(DT,1),20));
a(1)=1;
pred=zeros(size(DT,1),1);
DT_R2L=out_of_sample(DT(:,feat_R2L), mapping_R2L);
for i=2:20
    fprintf('%d : %d\n',a(i-1),a(i));
    pred(a(i-1):a(i)) = svmclassify(svm3,DT_R2L(a(i-1):a(i),:),'showplot',false);
end;
[acc_R2L sens_R2L spec_R2L ppv_R2L npv_R2L LRp_R2L LRn_R2L] = contingency_table(lt3,pred);

%%%%% CLASIFICACION DE CONEXIONES TIPO U2R
fprintf('Buscando conexiones U2R...\n');
lt4=(labels5==4);
a=round(linspace(1,size(DT,1),20));
a(1)=1;
pred=zeros(size(DT,1),1);
DT_U2R=out_of_sample(DT(:,feat_U2R), mapping_U2R);
for i=2:20
    fprintf('%d : %d\n',a(i-1),a(i));
    pred(a(i-1):a(i)) = svmclassify(svm4,DT_U2R(a(i-1):a(i),:),'showplot',false);
end;
[acc_U2R sens_U2R spec_U2R ppv_U2R npv_U2R LRp_U2R LRn_U2R] = contingency_table(lt4,pred);

%ALMACENAMIENTO DE LOS RESULTADOS TENIENDO EN CUENTA PRECISIÓN, SENSIBILIDAD, ESPECIFICIDAD
ACC=[acc_NOR acc_DOS acc_PRO acc_R2L acc_U2R];
SENS=[sens_NOR sens_DOS sens_PRO sens_R2L sens_U2R];
SPEC=[spec_NOR spec_DOS spec_PRO spec_R2L spec_U2R];
```

8. CUADRO COMPARATIVO CONJUNTOS DE DATOS SELECCIONADOS

CONJUNTOS DE DATOS	DESCRIPCION	TIEMPO CREACION	REGISTROS	AÑO CREACION	FUENTE
DARPA´98	Es un conjunto de datos producto de 7 semanas de tráfico de la red, que se puede procesar en alrededor de 5 millones de conexión registros, cada uno con cerca de 100 bytes.	7 Semanas de Trafico de Red	5,000,000 millones	1998	http://www.ll.mit.edu/mission/communications/cyber/CSTcorpora/ideval/data/
KDD´99	El conjunto de datos KDD´99 utilizado para la evaluación comparativa problemas de detección de intrusos se utiliza en nuestro experimento. la conjunto de datos era una colección de datos en bruto simulados volcado sobre TCP en un período de nueve semanas con una red de área local. La formación los datos se procesaron a aproximadamente cinco millones de registros de conexiones de siete semanas de tráfico de la red y dos semanas de pruebas datos resultantes de alrededor de dos millones de registros de conexión. la datos de entrenamiento se compone de 22 ataques diferentes de los 39 presentar en los datos de prueba. Los tipos de ataque conocidos son los presentes en el conjunto de datos de entrenamiento mientras que los nuevos ataques son el ataques adicionales en los conjuntos de datos de prueba que no están disponibles en el los conjuntos de datos de entrenamiento.	7 Semanas de Trafico de Red y 2 Semanas de Pruebas de Datos	5,000,000 millones de registros durante 7 semanas y 2,000,000 de registros en 2 semanas de pruebas	1999	http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html
NSLKDD	NSL-KDD es un conjunto de datos para resolver algunos de los problemas inherentes de la KDD'99. Aunque, esta nueva versión de los datos KDD establecidos todavía sufre de algunos de los problemas y puede no ser un representante perfecta de las redes reales existentes, debido a la falta de conjuntos de datos públicas para IDS basados en la red. El conjunto de datos permite a los investigadores comparar los diferentes métodos de detección de intrusos. Por otra parte, el número de registros en el tren NSL-KDD y equipos de prueba son razonables. Esta ventaja hace que sea asequible para ejecutar los experimentos sobre el conjunto completo sin la necesidad de seleccionar al azar una pequeña porción. En consecuencia, los resultados de evaluación de los diferentes trabajos de investigación serán consistentes y comparables.	7 Semanas de Trafico de Red y 2 Semanas de Pruebas de Datos	5,000,000 millones de registros durante 7 semanas y 2,000,000 de registros en 2 semanas de pruebas	1999	http://nsl.cs.unb.ca/NSL-KDD/

9. CUADRO COMPARATIVO TECNICAS UTILIZADAS PARA DISEÑO DE PROTOTIPO

METODOS DEL FEAST UTILIZADOS	**ETAPA 1** SELECCIÓN DE CARACTERISTICAS	**ETAPA 2** REDUCCION DE DIMENSION Y METRICAS DE APRENDIZAJE			**ETAPA 3** DISEÑO DE PROTOTIPO **(RESULTADO CON PCA)**					**ETAPA 4** VALIDACION CON **SOM Y GHSOM**	
		ISOMAP	**PCA**	**KPCA**	**NORMAL**	**DOS**	**PROBE**	**U2R**	**R2L**	**NORMAL**	**ATAQUE**
CIFE	X										
CMI	X										
CONDRED	X										
JMI	X	X	X	X							
MRMR	X	X	X	X	85,42%	80,77%	90,41%	91,87%	83,25%	99,74%	99,42
CMIN	X										
DISR	X										
ICAP	X										
MIFS	X										
RELIEF	X	X	X	X							

En este cuadro podemos apreciar las diversas etapas realizadas para el desarrollo del prototipo que permite identificar los diferentes tipos de ataques. En cada etapa fueron utilizadas una serie de técnicas específicas con la finalidad de seleccionar la que mejores resultados de precisión arroje. En la etapa No 1 se sometieron a prueba todos los métodos del FEAST por ello se encuentran marcados con el carácter "x", posteriormente en la segunda etapa se puede apreciar que solo se implementaron los 3 mejores métodos del FEAST con las técnicas de ISOMAP, PCA y KPCA los cuales podemos observar marcados en la etapa No 2 con el carácter "x", luego en la etapa No 3 observamos los resultados arrojados por las técnicas seleccionadas para el diseño del prototipo que en este caso corresponden al método **MRMR** como método de selección de características y **PCA** como técnica de reducción de dimensión y métrica de aprendizaje. En la etapa de Validación descrita como etapa No 4 podemos observar los resultados de las técnicas SOM y GHSOM para la identificación de ataques con el mismo conjunto de datos. Cabe destacar que el prototipo producto del desarrollo de esta investigación trabaja solo con 5 de la 41 características que contiene el conjunto de datos NSL-KDD mientras que las técnicas del SOM y GHSOM arrojan resultados utilizando las 41 características con el mismo conjunto de datos.