UNIVERSIDAD SEÑOR DE SIPÁN

FACULTAD DE INGENIERÍA, ARQUITECTURA Y URBANISMO

ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMAS

TESIS

EVALUACIÓN DE MÁQUINAS DE SOPORTE VECTORIAL Y RED NEURONAL ARTIFICIAL PARA DETERMINAR LA EFICIENCIA EN LA DETECCIÓN DE INTRUSOS DE UNA RED INFORMÁTICA.

PARA OPTAR EL TÍTULO PROFESIONAL DE INGENIERO DE SISTEMAS

Autor:

Bach. Santisteban Ayasta Leonardo Euler

ORCID: https://orcid.org/0000-0002-4791-2408

Asesor:

Mg. Mejía Cabrera Heber Iván

ORCID: https://orcid.org/0000-0002-0007-0928

Línea de Investigación

Infraestructura, Tecnología y Medio Ambiente

Pimentel – Perú 2020

EVALUACIÓN DE MÁQUINAS DE SOPORTE VECTORIAL Y RED NEURONAL ARTIFICIAL PARA DETERMINAR LA EFICIENCIA EN LA DETECCIÓN DE INTRUSOS DE UNA RED INFORMÁTICA.

APROBACIÓN DEL JURADO

_____

Bach. Santisteban Ayasta Leonardo Euler

Autor

_____

Mg. Mejía Cabrera Heber Iván

Asesor

_____ _____

Dr. Vásquez Leyva Oliver

Presidente de Jurado

Mg. Tuesta Monteza Victor Alexci

Secretario de Jurado

_____

Mg. Atalaya Urrutia Carlos William

Vocal de Jurado

DEDICATORIA

A Dios.

Por haberme permitido llegar hasta este punto y haberme dado salud para lograr mis objetivos, además de su infinita bondad y amor.

A mi madre Magdalena.

Por haberme apoyado en todo momento, por sus consejos, sus valores, por la motivación constante que me ha permitido ser una persona de bien, pero más que nada, por su amor.

A mi familia

No fue fácil el camino para llegar hasta donde estoy, pero gracias a su apoyo, a su amor incondicional, a su enorme amabilidad y acompañamiento, lo difícil se hizo más fácil y llevar a feliz término este proyecto se hizo una realidad. Les agradezco, y hago eco de mi enorme aprecio hacia ustedes, mi hermosa familia.

AGRADECIMIENTO

Agradezco a Dios por bendecirnos la vida, por guiarnos a lo largo de nuestra existencia, ser el apoyo y fortaleza en aquellos momentos de dificultad y de debilidad.

Gracias a mi madre: Magdalena, por ser los principales promotores de nuestros sueños, por confiar y creer en nuestras expectativas, por los consejos, valores y principios que nos han inculcado.

Agradezco a nuestros docentes de la Escuela de Ingeniería de Sistemas la Universidad Señor de Sipán, por haber compartido sus conocimientos a lo largo de la preparación de nuestra profesión.

RESUMEN

La presente investigación evaluará los algoritmos Perceptrón Multicapa y Máquinas de Soporte Vectorial, el objetivo de la investigación es evaluar la eficiencia de las técnicas en la detección de intrusos.

Muchas empresas según los reportes de antivirus sufren de ataque diariamente ocasionando, pérdidas millonarias en las pequeñas y medianas empresas, pues a pesar de contar con mecanismos de seguridad, Firewall, antivirus, Sistemas de detección de intrusos (IDS) no son suficiente, pues uno de los problemas que enfrentan, es como reducir la tasa de falsas alarmas y reducir la tasa de falsos positivos es por ello el motivo de esta investigación, el cual consta de 4 fases.

La fase inicial de esta investigación se origina con la preparación del conjunto de datos, se extrajo patrones de ataques, elaborando primero un estudio de los últimos ciberataques a Nivel de Latinoamérica y del mundo, los patrones recolectados son: Denegación de Servicio Distribuido (DDOS), Phishing, Backdoor, y de las diferentes familias de Ransomware, formando así un nuevo conjunto de datos denominado Dataset_Attack_20018. En la fase de selección de técnicas de aprendizaje automático, se evaluaron la eficiencia de los algoritmos, para lograr este objetivo se realizó un estudio comparativo de las técnicas, logrando obtener un top de las 6 mejores técnicas, el cual se realizó todo un proceso, siendo evaluados por las métricas de desempeño. La fase de implementación consistió en evaluar el top realizado de las 6 mejores técnicas y elegir las técnicas Perceptrón Multicapa y Máquinas de Soporte Vectorial para que sean evaluadas por el conjunto de datos mencionados líneas arriba. En la última fase de evaluación, las técnicas Perceptrón Multicapa y Máquinas de Soporte Vectorial, se empleó la matriz de confusión para evaluar con las métricas de desempeño, llegándose a la conclusión que el Perceptrón Multicapa en esta investigación fue el más eficiente tanto en exactitud del 99%, precisión del 98 % sensibilidad del 67%, y especificidad del 98%.

La herramienta utilizada para la implementación es la librería scikit-learn 0.19.2.

PALABRAS CLAVES: Perceptrón Multicapa, detección, evaluación, Algoritmos, Maquinas de soporte Vectorial.

ABASTRACT

The present investigation will evaluate the Multilayer Perceptron algorithms and Vector Support Machines, the objective of the investigation is to evaluate the efficiency of the techniques in intrusion detection.

According to antivirus reports, many companies suffer from attacks on a daily basis causing millions in losses in small and medium-sized companies, because despite having security mechanisms, Firewall, antivirus, Intrusion Detection Systems (IDS) are not enough, because one One of the problems they face, is how to reduce the rate of false alarms and reduce the rate of false positives is therefore the reason for this research, which consists of 4 phases. The initial phase of this research originates with the preparation of the data set, patterns of attacks were extracted, first preparing a study of the latest cyberattacks in Latin America and the world, the patterns collected are: Distributed Denial of Service (DDOS) , Phishing, Backdoor, and the different Ransomware families, thus forming a new data set called Dataset_Attack_20018. In the selection phase of machine learning techniques, the efficiency of the algorithms were evaluated, to achieve this objective a comparative study of the techniques was carried out, obtaining a top of the 6 best techniques, which was carried out a whole process, being evaluated by performance metrics. The implementation phase consisted of evaluating the top made of the 6 best techniques and choosing the Multilayer Perceptron techniques and Vector Support Machines to be evaluated by the data set mentioned above. In the last evaluation phase, the Multilayer Perceptron and Vector Support Machines techniques, the confusion matrix was used to evaluate with the performance metrics, reaching the conclusion that the Multilayer Perceptron in this investigation was the most efficient both in terms of accuracy of the 99%, precision 98%, sensitivity 67%, and specificity 98%.

The tool used for the implementation is the scikit-learn 0.19.2 library.

KEY WORDS: Multilayer Perceptron, detection, evaluation, Algorithms, Vector support machines.

ÍNDICE

DEDICATORIA ..... **iii**

AGRADECIMIENTO ..... **iv**

RESUMEN ..... **v**

ABASTRACT ..... **vi**

I. INTRODUCCIÓN **18**

1.1. Realidad Problemática. **18**

1.2. Antecedentes de Estudio **27**

1.2.1. Antecedentes de la Investigación. **27**

1.2.2. Estado del Arte. **31**

1.3. Teorías relacionadas al tema **47**

1.3.1. Seguridad informática: **47**

1.3.2. Principios de seguridad de los sistemas de información: **48**

1.3.2.1. Confidencialidad: **48**

1.3.2.2. Integridad: **48**

1.3.2.3. Disponibilidad: **49**

1.3.3 Redes de Computadoras. **49**

1.3.3.1. El modelo de referencia ISO OSI **49**

1.3.4. Sistema De Detección De Intrusiones (IDS) **51**

1.3.5. Métricas de Desempeño **51**

1.3.6. Inteligencia Artificial **52**

1.3.7. Aprendizaje Automático (Machine Learning) **53**

1.3.7.2. Desventajas del aprendizaje automático **53**

1.3.8. Tipos De Aprendizaje Automático **54**

1.3.8.1 Aprendizaje Supervisado **54**

1.3.8.2. Aprendizaje No Supervisado **55**

1.3.8.3. Aprendizaje Semi-Supervisado: **55**

1.3.9. Pasos Para Realizar Una Tarea De Aprendizaje Automático 55

1.3.10. Minería De Datos (Data Mining).....56

1.3.11. Ataques informáticos.....57

1.3.11.1. Ransomware.....57

1.3.11.3. Backdoor: 58

1.3.11.4. Exploit:.....59

1.3.11.6. Smurf.....60

1.3.11.7. Http- Flood.....61

1.3.11.8. UDP- Flood.....62

1.3.11.9. SIDDOS: 63

1.3.12. Redes Neuronales Artificiales (RNA). 63

1.3.13. Arquitecturas de Redes Neuronales Artificiales 64

1. 3.14. Esquema del Perceptrón. 65

1.3.15. Arquitecturas de Redes Neuronales Artificiales. 66

1.3.15.1. Arquitectura Feedforward de una Sola Capa. 66

1.3.15.2. Arquitecturas de avance de múltiples capas. 67

1.3.15.3. Arquitectura recurrente o de retroalimentación.....68

1.3.15.4. Perceptrón Multicapa (Multilayer Perceptrón).....69

1.3.16. Máquinas de Soporte Vectorial (SVM). 70

1.3.17. Random Forest.....71

1.3.18. Naive Bayes.....71

1.3.19. AdaBoost.....71

1.3.20. C 4.5. 72

1.3.21. Herramienta para evaluar un modelo de clasificación 72

1.3.21.1 Matriz de confusión.....72

1.3.22. Técnicas de Validación y Evaluación 73

1.3.22.1. Validación cruzada:.....73

1.3.23. Herramientas utilizadas 73

1.3.23.1. Scikit-learn.....73

1.3.23.2. Pandas 73

1.3.23.3. Numpy.....73

1.3.24. Técnicas de Normalización 73

1.4. Formulación del Problema 74

1.5. Justificación e importancia de la investigación 74

1.6. Hipótesis.....75

1.7. Objetivos de la Investigación. 75

1.7.1. Objetivo General: 75

1.7.2. Objetivos específicos 75

II. MATERIAL Y MÉTODO.....76

2.1. Tipo y Diseño de Investigación.....76

2.1.1. Tipo de la Investigación.....76

2.1.2. Diseño de la Investigación 76

2.2. Población y Muestra 76

2.2.1. Población.....76

2.2.2. Muestra 76

2.3. Variables y Operacionalización.....77

2.3.1. Variable Independiente 77

2.3.3. Operacionalización 77

2.4. Técnicas e instrumentos de recolección de dato, validez y confiabilidad……78

2.4.1. Abordaje metodológico 78

2.4.2. Técnicas de recolección de datos 78

2.4.2.1 La Observación. 78

2.4.2.2 Análisis Documental. 79

2.5. Procedimientos de análisis de datos. 79

2.5.1. Plan de análisis estadísticos de datos. 79

2.6. Procedimiento de análisis de datos 79

2.7. Criterios éticos. 79

III. RESULTADOS 81

3.1. Resultados en tablas y Figuras. 81

3.2. Discusión de resultados. 89

3.3. Aporte Práctico. 90

3.3.1. Método propuesto para el desarrollo del proyecto de investigación. 90

3.3.1.1. Fase de Preparación del conjunto de datos. 92

3.3.1.1.1. Limpieza y transformación de los datos obtenidos. 96

3.3.1.1.2. Min Max Normalización del conjunto de datos Ejemplo con el Dataset_Attack_2018 112

3.3.1.1.3. Conjuntos de Ataques Cibernéticos denominado Dataset_ Attack_2018 115

3.3.2. Fase Selección de las técnicas de mayor eficiencia en la detección de intrusos. 115

3.3.3. Fase Implementación de las técnicas de Máquinas de Soporte Vectorial y Perceptrón Multicapa. 117

3.3.3.1. Implementación del algoritmo Máquinas de Soporte Vectorial 117

3.3.3.2. Implementación del Algoritmo Perceptrón Multicapa. 130

3.3.4. Fase de Evaluación de las técnicas de Aprendizaje Automático. 136

3.3.4.1. Evaluación del Algoritmo Maquinas de Soporte Vectorial. 137

3.3.4.2. Evaluación del Algoritmo Perceptrón Multicapa. 141

IV. CONCLUSIONES Y RECOMENDACIONES. 146

4.1. Conclusiones. 146

4.2. Recomendaciones. 147

REFERENCIAS 148

Anexos. 165

Anexo 1: Resolución de Aprobación del Proyecto de Investigación. 165

Anexo 2: Matriz de consistencia del informe de investigación. 166

Anexo 3: Instrumentos de Recolección De Datos. 168

Anexo 3.1: Constancia de validación por Juicio De Expertos. 168

Anexo 3.2 : Ficha de observación resumen de la técnica Maquinas de Soporte Vectorial. 169

Anexo 3.3: Ficha de observación resumen de la técnica Redes Neuronales Artificiales. 170

Anexo 3.4: Ficha de observación resumen del análisis del ataque cibernético Backdoor implementado con la técnica Máquinas de Soporte Vectorial. ...171

Anexo 3.5: Ficha de observación resumen del análisis del ataque cibernético Exploits implementado con la técnica Maquinas de Soporte Vectorial. 172

Anexo 3.6: Ficha de observación resumen del análisis del ataque cibernético HTTP-Flood implementado con la técnica Maquinas de Soporte Vectorial. 173

Anexo 3.7: Ficha de observación resumen del análisis del ataque cibernético Phishing implementado con la técnica Maquinas de Soporte Vectorial. 174

Anexo 3.8: Ficha de observación resumen del análisis del ataque cibernético Ransomware implementado con la técnica Maquinas de Soporte Vectorial……..175

Anexo 3.9: Ficha de observación resumen del análisis del ataque cibernético SIDDOS implementado con la técnica Maquinas de Soporte Vectorial. 176

Anexo 3.10: Ficha de observación resumen del análisis del ataque cibernético Smurf implementado con la técnica Maquinas de Soporte Vectorial.....177

Anexo 3.11: Ficha de observación resumen del análisis del ataque cibernético UDP-Flood implementado con la técnica Maquinas de Soporte Vectorial...178

Anexo 3.12: Ficha de observación resumen del análisis del ataque cibernético Backdoor implementado con la técnica Redes Neuronales Artificiales. 179

Anexo 3.13: Ficha de observación resumen del análisis del ataque cibernético Exploits implementado con la técnica Redes Neuronales Artificiales. 180

Anexo 3.14: Ficha de observación resumen del análisis del ataque cibernético HTTP-Flood implementado con la técnica Redes Neuronales Artificiales…181

Anexo 3.15: Ficha de observación resumen del análisis del ataque cibernético Phishing implementado con la técnica Redes Neuronales Artificiales. 182

Anexo 3.16: Ficha de observación resumen del análisis del ataque cibernético Ransomware implementado con la técnica Redes Neuronales Artificiales. 183

Anexo 3.17: Ficha de observación resumen del análisis del ataque cibernético SIDDOS implementado con la técnica Redes Neuronales Artificiales. 184

Anexo 3.18: Ficha de observación resumen del análisis del ataque cibernético Smurf implementado con la técnica Redes Neuronales Artificiales. 185

Anexo 3.19: Ficha de observación resumen del análisis del ataque cibernético UDP-Flood implementado con la técnica Redes Neuronales Artificiales. ...186

Anexo 4: Proceso de la Selección de las técnicas de mayor eficiencia en la detección de intrusos. 187

Anexo 5: Listado de las características del conjunto de datos UNSW-NB15……189

Anexo 6: Listado de las características del conjunto de datos Dataset_ddos……193

Anexo 7: Listado de las características del conjunto de datos Dataset_Phishing 195

Anexo 08: Listado de las características del conjunto de datos CICAndMal2017 198

Anexo 9: Lista de características de Dataset_Attack_2018 204

Anexo 10: Matriz de Confusión Multiclase de Máquinas de Soporte Vectorial usando PCYM.....214

Anexo 11: Matriz de Confusión Multiclase del Perceptrón Multicapa usando PCYM 215

Anexo 12: Código de implementación del algoritmo Máquinas de Soporte Vectorial.....216

Anexo 13: Código de Implementación del Algoritmo Perceptrón Multicapa……217

ÍNDICE DE FIGURAS

Figura 1: Los 20 principales países atacados por el Ataque Malware. 19
Figura 2: Los 10 principales países atacados por el Ataque Malware. 20
***Figura 3:* Comparativa anual de infecciones por Malware en Latinoamérica (2009-216). 20**
Figura 4: Los veinte primeros países afectados por el ciberataque Ransomware.....21
Figura 5: Cantidad de variantes diferentes de Ransomware vistas en países. 22
Figura 6: Ranking de países latinoamericanos afectados por Phishing.....23
Figura 7: Incidentes de Phishing por países 2016.....23
Figura 8: Distribución de los Ataques DDoS por País, III y IV Trimestres de 2018.....24
Figura 9: Detecciones de Filecoder en países de LATAM durante 2017.....25
Figura 10: Tipos de Ataques DDOS 2018. 26
Figura 11: Tráfico en tiempo real del ataque DDoS.....26
Figura 12: Los tres principios de sistemas de información seguridad.....48
Figura 13: El Modelo Referencial OSI. 50
Figura 14: Tipos de técnicas de aprendizaje automático.....54
Figura 15: Anatomía de un ataque de Ransomware.....57
Figura 16: Ataque Phishing.....58
Figura 17: Ataque dirigido típico en una red corporativa. 59
Figura 18: Ataque Exploit. 59
Figura 19: Ataque DDOS.....60
Figura 20: Ataque Smurf.....61
Figura 21: Ataque Http Get.....61
Figura 22: Http Post Attack. 62
Figura 23: Ataques de Inundación UDP. 62
Figura 24: DDOS mediante Inyección SQL (SiDDOS).....63
Figura 25: Ejemplo de una Red Neuronal totalmente Conectada.....64
Figura 26: Esquema del Perceptrón. 65
Figura 27: Algoritmo de aprendizaje del Perceptrón.....66
Figura 28: Ejemplo de una red de avance de una sola capa.....67

Figura 29: Ejemplo de una red feedforward con múltiples. 68
Figura 30: Ejemplo de una red recurrente. 68
Figura 31: Diagrama del Perceptrón Multicapa. 69
Figura 32: Diagrama del Perceptrón Multicapa. 71
Figura 33: Matriz de confusión para un modelo de Clasificación. 72
Figura 34: Comparación de Algoritmos basados en Exactitud, Precisión, Sensibilidad y Especificidad del Ataque Backdoor. 81
Figura 35: Comparación de Algoritmos basados en Exactitud, Precisión, Sensibilidad y Especificidad del Ataque Exploit. 82
Figura 36: Comparación de Algoritmos basados en Exactitud, Precisión, Sensibilidad, Especificidad del Ataque HTTP-Flood. 83
Figura 37: Comparación de Algoritmos basados en Exactitud, Precisión, Sensibilidad y Especificidad del Ataque Phishing. 84
Figura 38: Comparación de Algoritmos basados en Exactitud, Precisión, Sensibilidad y Especificidad del Ataque Ransomware. 85
Figura 39:Comparación de Algoritmos basados en Exactitud, Precisión, Sensibilidad y Especificidad del Ataque SIDDOS. 86
Figura 40: Comparación de Algoritmos basados en Exactitud, Precisión, Sensibilidad y Especificidad del Ataque Smurf. 87
Figura 41: Comparación de Algoritmos basados en Exactitud, Precisión, Sensibilidad y Especificidad del Ataque UDP-Flood. 88
Figura 42:Tiempo de Entrenamiento de los algoritmos. 88
Figura 43: Método Propuesto para el desarrollo de la Implementación. 91
Figura 44: Método Propuesto para el desarrollo de la Implementación. 98
Figura 45: Eliminación de la columna Flow_Id, Node_From y Node_Name.102
Figura 46: Asignación de valores para estado. 104
Figura 47:División del Conjunto de Datos Entrenamiento y Pruebas del Dataset_Attack_2018. 114
Figura 48: Diagrama de dispersión empleando el algoritmo SVM para el Dataset_attack_2018. 118
Figura 49:Posibles hiperplanos que separan la Clase 1 y Clase 2 (Exploits y Backdoor). 119
Figura 50:Vectores de soporte VS1, VS2 y VS3. 125

Figura 51: Representación del método de Regla.....127
Figura 52: Incremento de Ecuación de Kramer. 127
Figura 53: Primer Incremento aplicando la Ecuación de Kramer. 127
Figura 54: Segundo Incremento aplicando la Ecuación de Kramer. 128
Figura 55: Tercer Incremento de Ecuación de Kramer.....128
Figura 56: Predicción de la Clase 1 (Exploit) y de la Clase (Backdoor). 129
Figura 57: Algoritmo de Máquina de Soporte Vectorial Multiclase. 130
Figura 58: Esquema de Red de Trabajo del Algoritmo Perceptrón Multicap…….....135
Figura 59: Exactitud del Algoritmo Máquinas de Soporte Vectorial.....139
Figura 60: Precisión del Algoritmo Máquinas de Soporte Vectorial. 140
Figura 61:Precisión del Algoritmo Máquinas de Soporte. Vectorial.....140
Figura 62: Especificidad del Algoritmo Máquinas de Soporte Vectorial.....141
Figura 63: Exactitud del Algoritmo Perceptrón Multicapa. 143
Figura 64: Precisión del Algoritmo Perceptrón Multicapa. 144
Figura 65:Sensibilidad del Algoritmo Perceptrón Multicapa. 144
Figura 66: Sensibilidad del Algoritmo Perceptrón Multicapa. 145

ÍNDICE DE TABLAS

Tabla 1: Métricas de rendimiento de los clasificadores usando NSL-KDD....36
Tabla 2: Rendimiento de los clasificadores. 37
Tabla 3: Lista del rendimiento de los algoritmos.....38
Tabla 4: Comparación en el porcentaje de precisión y tipos de muestra de los dataset. 39
Tabla 5: Comparación de diferentes métodos de detección de intrusión utilizando el Conjunto de datos UNSW-NB15. 40
Tabla 6: Medidas de Desempeño. 41
Tabla 7: Valores de precisión y recuperación obtenidos por la clasificación del tráfico anómalo detectado por detectores Minkowski. 42
Tabla 8: Rendimiento de algoritmos con el dataset 1. 43
Tabla 9: Rendimiento de algoritmos con dataset 2. 44
Tabla 10: Rendimiento de algoritmos con dataset 3. 44
Tabla 11: Rendimiento de algoritmos con data 2. 45
Tabla 12: Comparando el enfoque propuesto con otros métodos. 45
Tabla 13: Rendimiento de los métodos de conjunto con cada clasificador. .46
Tabla 14: Comparación de exactitud. 47
Tabla 15. Notación Matemática Support Vector Machine.....70
Tabla 16: Operacionalización de Variable Dependiente. 77
Tabla 17: Operacionalización de Variable Independiente. 78
Tabla 18: Ataques Exploits y Backdoor extraídos del Conjunto de datos UNSW-NB15.....93
Tabla 19: Ataques Smurf, UDP Flood, SIDDOS, HTTP Flood extraídos del conjunto de datos DDOS. 93
Tabla 20: Ataques Phishing extraídos del conjunto de datos Phishing. 94
Tabla 21: Ataques Ransomware por Familias extraídos del conjunto de datos CICAndMal2017. 95
Tabla 22: Ataques del Nuevo Conjunto de Datos. 96
Tabla 23: Análisis y comparación de características de los conjuntos de datos. 97
Tabla 24.Eliminación de las columnas innecesarias Flow_ID. 99

Tabla 25: Eliminación de las columnas NODE_NAME_FROM y NODE_NAME_TO. 101
Tabla 26: Columna Source_IP antes de la Conversión a dato. 103
Tabla 27: Columna Destination_IP antes de la conversión a dato numérico…… 105
Tabla 28:Columna Time_Stamp antes de la conversión a dato numérico. ..107
Tabla 29: Conversión de la columna Sate. 109
Tabla 30: Conversión de la columna Sate categórico a Numérico. 111
Tabla 31: Conversión de la columna Service. 111
Tabla 32: Ejemplo del Conjunto de Datos antes de la Normalización. 112
Tabla 33: Comparación del Dataset antes y después de la Normalización. 114
Tabla 34: Ataques del Dataset Attack 2018 en formato CSV. 115
Tabla 35: Selección de las técnicas de clasificación con mayor sensibilidad…… 116
Tabla 36: Selección de las técnicas de clasificación de mayor exactitud. ..117
Tabla 37: SVM Características de demostración del Dataset_attack_2018.118
Tabla 38: Evaluación Punto de la Clase 1 (Exploit) con la clase 2. 120
Tabla 39: Resultado de la simplificación de la Clase 1 con la Clase A para obtener las distancias más cortas 120
Tabla 40: Resultado de la simplificación de la Clase 1 con la Clase B para obtener las distancias más cortas. 121
Tabla 41:Resultado de la simplificación de la Clase 1 con la Clase C para obtener las distancias más cortas. 121
Tabla 42: Resultado de la simplificación de la Clase 1 con la Clase D para obtener las distancias más cortas. 122
Tabla 43: Resultado de la simplificación de la Clase 1 con la Clase E para obtener las distancias más cortas. 122
Tabla 44: Resultado de la simplificación de la Clase 1 con la Clase F para obtener las distancias más cortas. 123
Tabla 45: Resultado de la simplificación de la Clase 1 con la Clase F para obtener las distancias más cortas 123
Tabla 46: Resultado de la simplificación de la Clase 1 con la Clase H para obtener las distancias más cortas. 124

Tabla 47: Resultado de la simplificación de la Clase 1 con la Clase I para obtener las distancias más cortas. 124
Tabla 48: MLP Características de demostración del Dataset_Attack_2018. 131
Tabla 49: Representación Entradas y Salidas deseadas del Dataset_Attack_2018. 131
Tabla 50: Resultados de la matriz de confusión o error del Algoritmo Máquinas de Soporte Vectorial. 138
Tabla 51: Resultados de la matriz de confusión o error del Algoritmo Máquinas de Soporte Vectorial. 139
Tabla 52: Resultados de la matriz de confusión o error del Algoritmo Perceptrón Multicapa.....142
Tabla 53: Resultado de las métricas de desempeño evaluando la eficiencia del Algoritmo Perceptrón Multicapa. 143

I. INTRODUCCIÓN

1.1. Realidad Problemática.

Hoy en día el tema de la seguridad no solamente se encuentran en las calles por el tema de los robos y la delincuencia, sino que también vivimos en el mundo globalizado de las Tecnologías de la Información (TIC), las empresas y personas se encuentran en mucha comunicación a través de las computadoras, laptops y modernos celulares tecnológicos, compartiendo e intercambiando información, realizando envíos de correos electrónicos y transfiriendo dinero a cuentas bancarias, es por eso que hablamos de un tema seguridad informática que ha obtenido gran connotación en el mundo. La seguridad de la información tiene como función primordial resguardar a los sistemas de información de las amenazas a los que están expuestos. (Gil & Gil, 2017).

Las redes informáticas se han convertido en una parte esencial en todo tipo de organizaciones, como instituciones financieras, médicas, industriales, de transporte, educativas, etc. Las empresas utilizan la red informática para optimizar sus procesos en la toma de decisiones, pues estos reúnen, procesan, almacenan y comparten grandes cantidades de información digital, utilizando para ello procedimientos que nos permiten incrementar su eficiencia. Pues conforme se va recolectando y compartiendo información, la protección de ello, se vuelve considerablemente importante tanto para la organización, como para su solidez económica. (Hamidian & Ospino, 2015).

Para (ISO 27001, 2017) las compañías y sus sistemas de información se encuentran propensos a innumerables cantidades de ataques informáticos, aprovechándose de cualquiera de sus vulnerabilidades existentes, sometiéndolos a fraudes y espionaje, perjudicando la información de la organización.

Según (Panda Security, 2020) afirma en su reporte realizado por la empresa Panda Security nos da a conocer un top 20 de países que presentaron el ciberataque Malware teniendo como primer país a Tailandia, con un 40% de este ataque cibernético y también vemos a Perú, ocupando el puesto número 18. Ver Figura Nº 01.

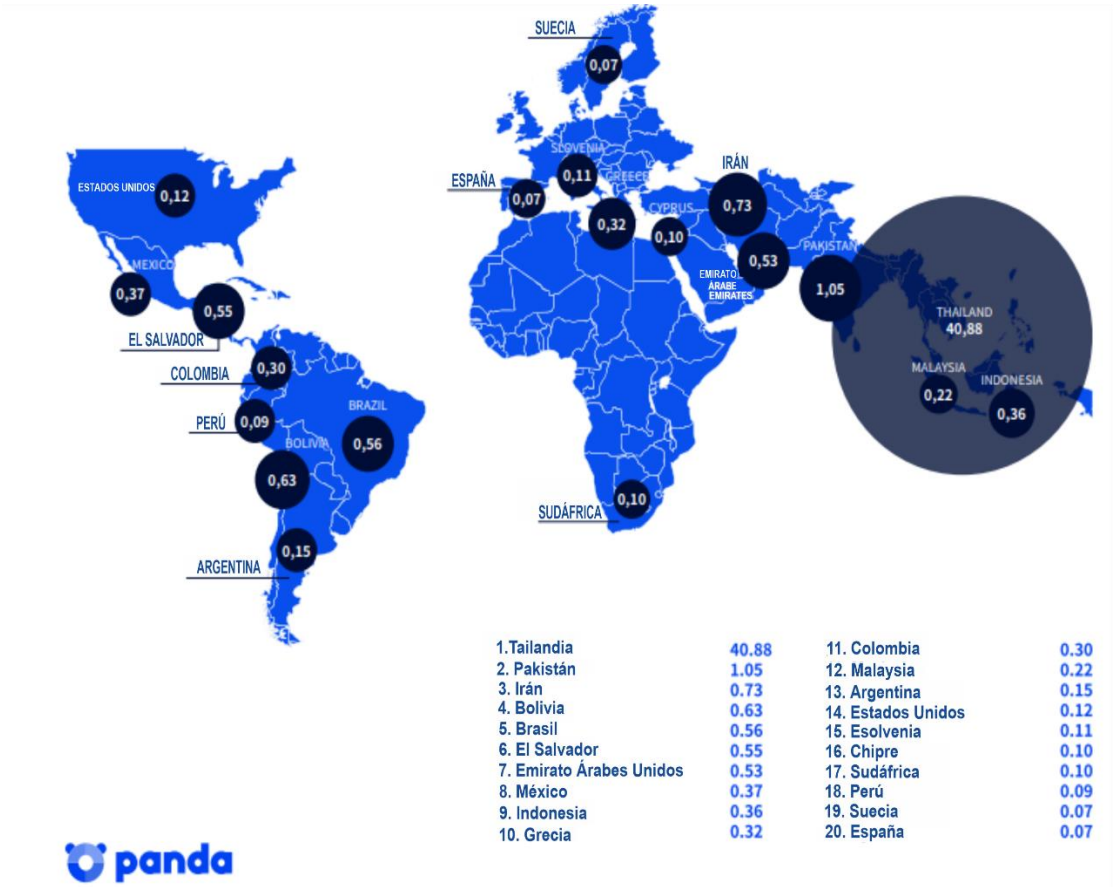

Figura 1: Los 20 principales países atacados por el Ataque Malware.
Fuente*:* (Panda Security, 2020, pág. 12)

Según (Karspesky, 2019) nos confirma en su informe del tercer trimestre, un top de los 10 principales países atacados por el ataque cibernético Malware, presentando como primer lugar al país de Irán más afectado por este ataque obtenido el 52.58%, el segundo país con mayor incidencias de ataque es Bangladesh ocupando el 2do lugar con un 30.94%, y el tercer lugar lo ocupó el país de la India, obtenido un porcentaje de ataque del 28.75% reportados por los usuarios. Ver Figura Nº 02.

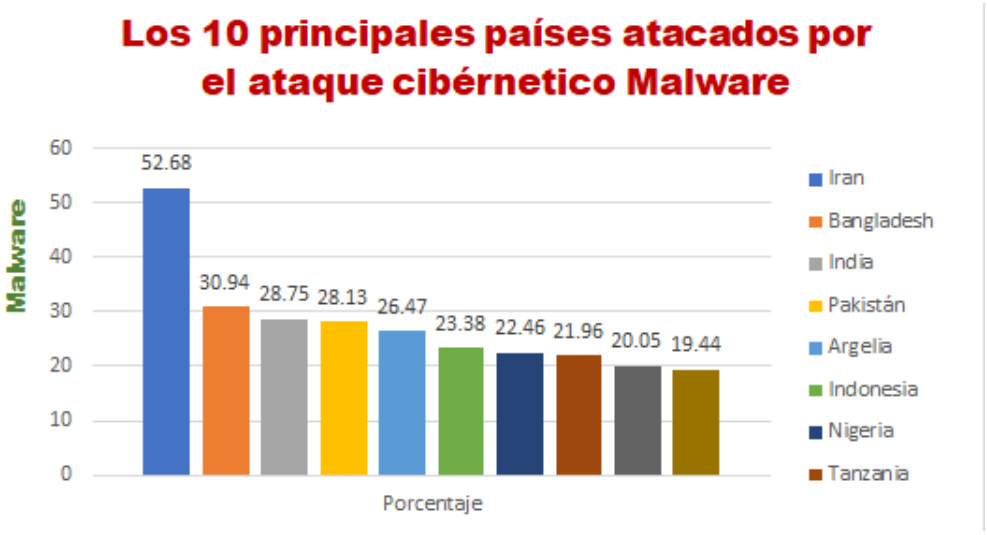

Figura 2: Los 10 principales países atacados por el Ataque Malware.
Fuente: (Karspesky, 2019)

Según (ESET, 2017) en su informe reporte anual, nos informa detalladamente como se ha venido desarollando los incidentes de seguridad relacionados sobre el software malicioso malware comprendido entre los años 2009 al 2016. En el año 2014 el 39% de las personas encuestadas afirmaron haber sufrido infecciones por malware, mientras que en 2015 el número aumentó al 40%; para 2016 el porcentaje ascendió a 49%. Ver Figura N° 03.

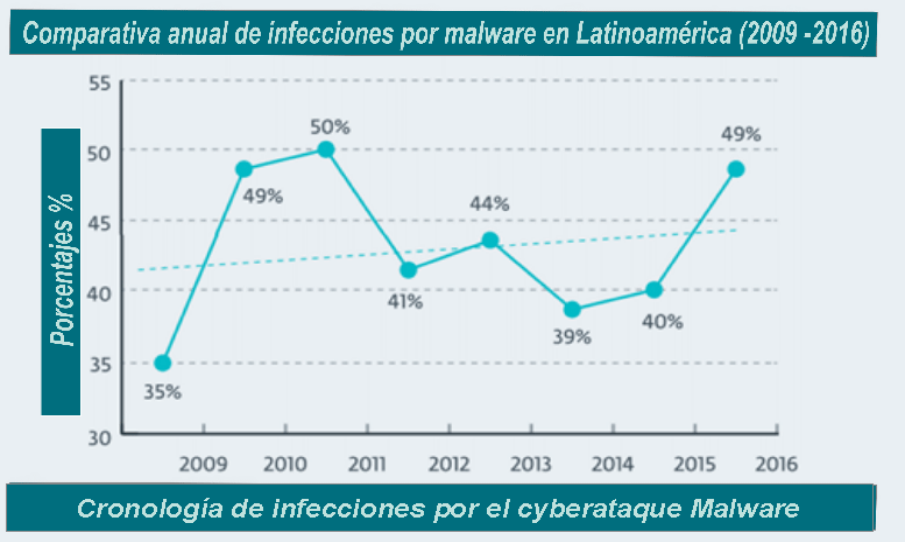

Figura 3: Comparativa anual de infecciones por Malware en Latinoamérica (2009-216).

Fuente: (ESET, 2017, pág. 9)

Según (Kaspersky, 2017) afirma que en junio del 2017, ransomware infectó mas de 200,000 ordenadores en 150 países, las telecomunicaciones fueron afectadas, grandes empresas a nivel mundial como Mondelez, Maersk, DLA Piper y Merck, la cual sufrieron un ciberataque de Ransomware por el que las oficinas de las empresas en España estaban totalmente paralizadas como se muestra en la Figura N° 04.

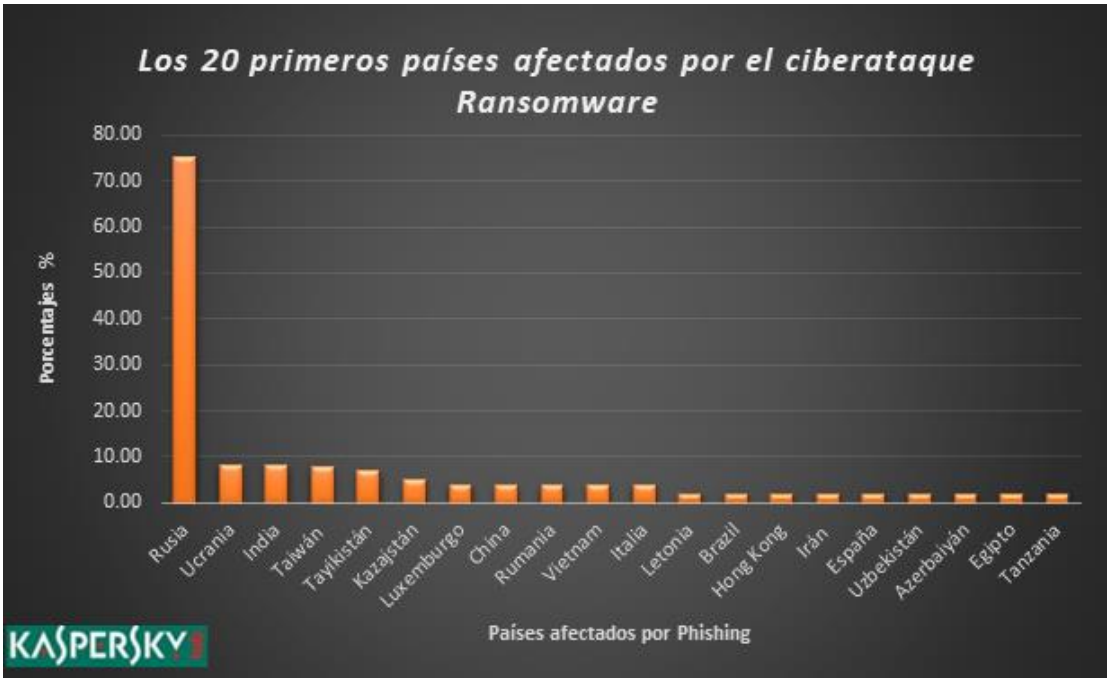

Figura 4: Los veinte primeros países afectados por el ciberataque Ransomware. Fuente: (Kaspersky, 2017)

Según (ESET, 2019) en su reporte digital presentado por la compañía de seguridad informática, nos da conocer las diferentes variantes que se presentó el ataque cibernético Ransomware en los años 2016 – 2018 y se puede observar que México ocupó el primer lugar con las variantes de este tipo de ataque, tambien podemos observar que el Segundo lugar lo ocupa el Perú, afectando a miles de usuarios ocasionados por este ciberataque. Ver Figura N° 05.

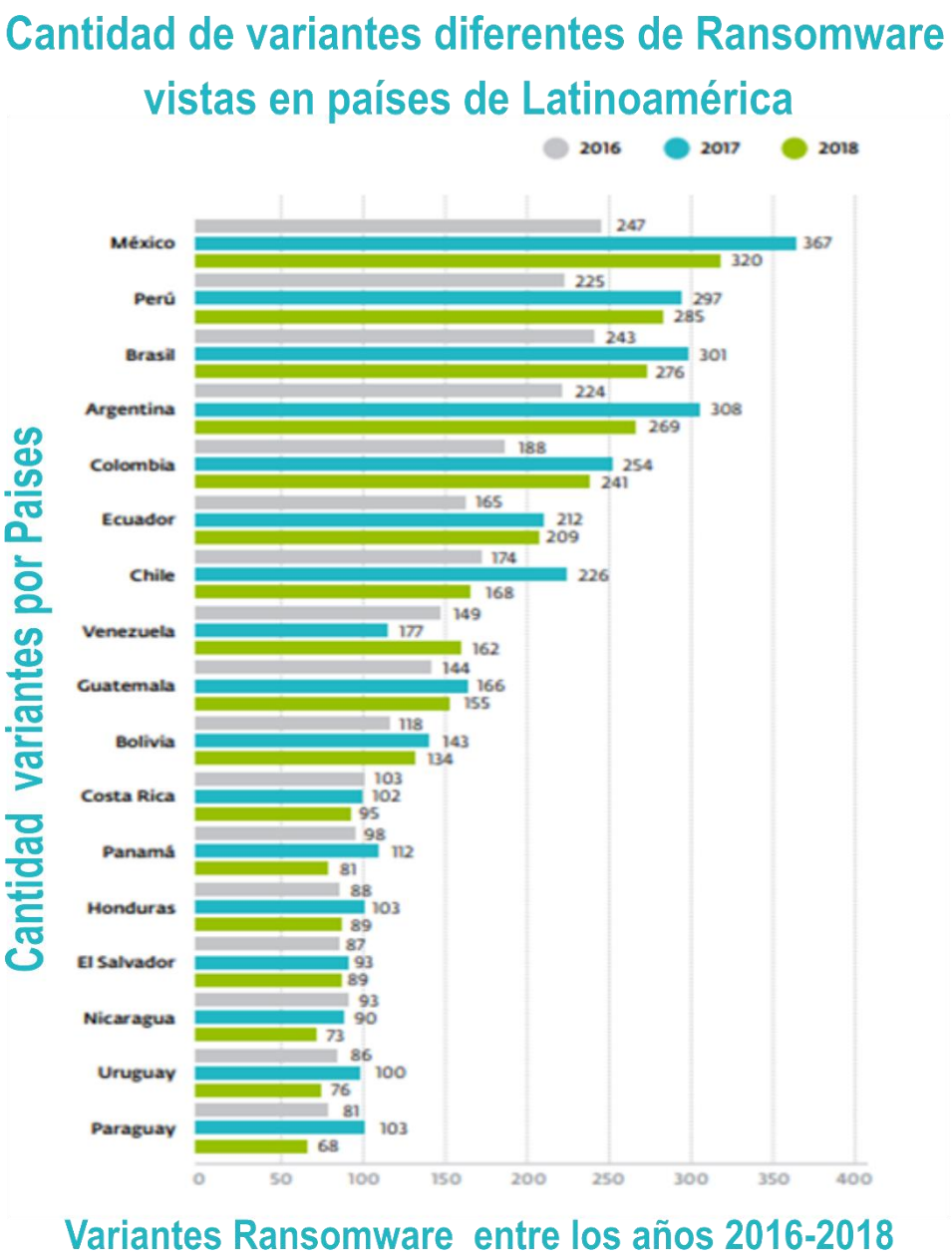

Figura 5: Cantidad de variantes diferentes de Ransomware vistas en países. Fuente: (ESET, 2019, pág. 7)

Según (Assolini, 2018) un analista senior de Seguridad de la compañía Kaspersky, afirma que se registraron más de 746 mil ataques de phishing en América Latina, además nos dio a conocer que Brasil estuvo dentro de los 20 países más atacado a nivel mundial por el ataque Phishing. En el presente año se bloqueó 40 millones de ataques Phishing en América Latina, siendo Brasil el país más afectado a continuación se presenta la Figura Nº 06.

Figura 6: Ranking de países latinoamericanos afectados por Phishing.

Fuente: (Assolini, 2018)

Según (ESET, 2017) afirma en su Reporte de Latinoamérica, nos da a conocer que innumerables compañías fueron vulneradas, por el ataque cibernético Phishing, las cuales se localizaban en los países de Ecuador, alcanzando un 20,9% de ataques, el Perú fue atacado con un 16,6% y México ocupando el tercer lugar con un 16,1%. El resultado obtenido se presenta en la Figura Nº 07.

Figura 7: Incidentes de Phishing por países 2016.

Fuente: (ESET, 2017, pág. 12)

Según (Karpesky, 2018) en su informe digital informa el ataque DDoS por países, presentado el tercer y cuarto trimestre, mostrando con mayor índice al País de China en el 3er trimestre con un total 77.67%, que superó el resultado del último trimestre en un 20 %, mientras el segundo lugar lo ocupó EE. UU con un porcentaje de 24.90 %, duplicando al cuarto Trimestre, mostrando el top de los 10 países más atacados por este ciberataque Figura Nº 08.

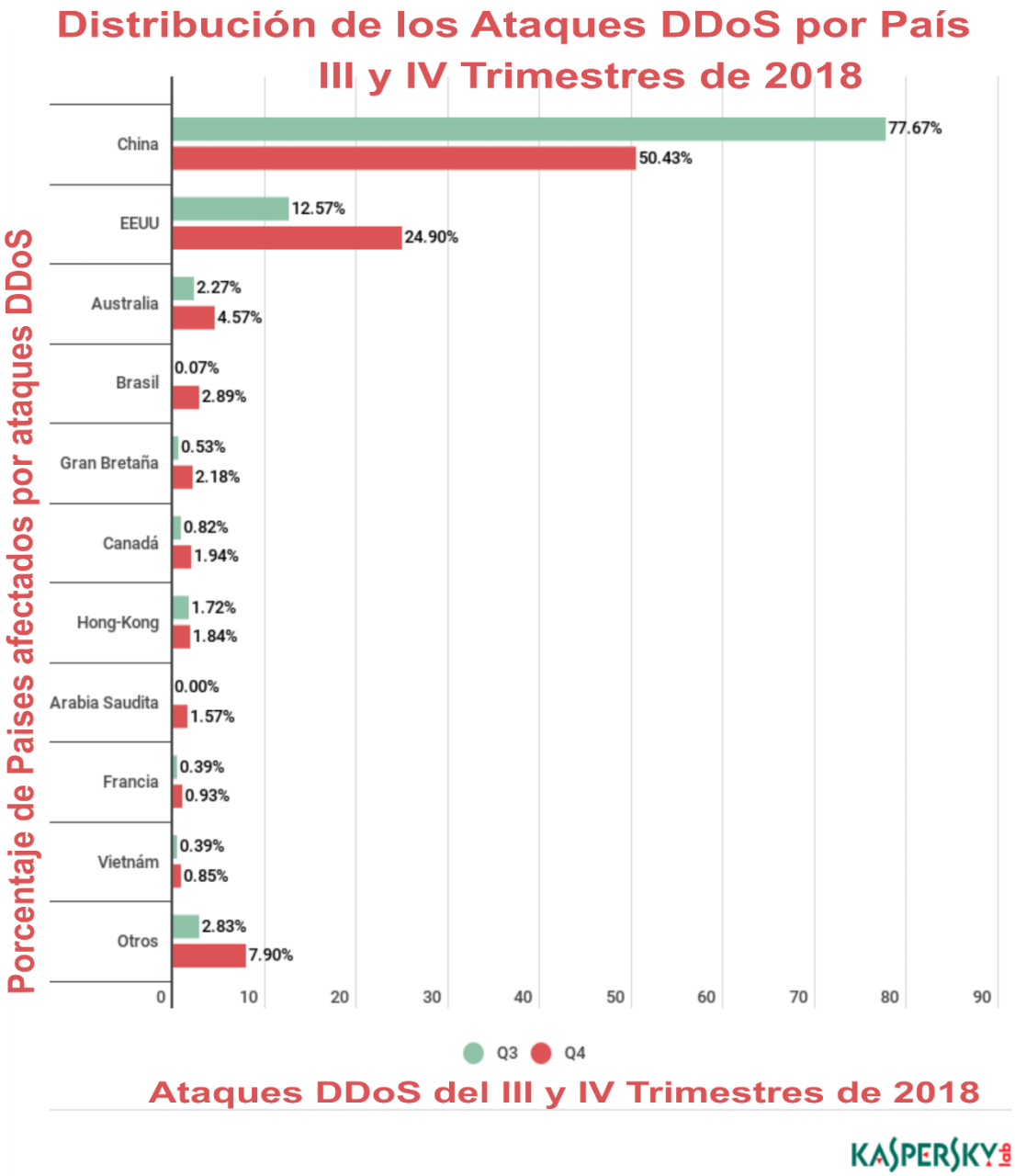

Figura 8: Distribución de los Ataques DDoS por País, III y IV Trimestres de 2018. Fuente: (Karpesky, 2018)

Según el informe (REPORT INTERNET CRIME, 2019) publicado por el Centro de Quejas de Delitos por Internet (IC3) del FBI, recibió 2,047 quejas identificadas por el ataque Ransomware, utilizados por los ciberdelincuentes generando pérdidas de más de $ 8.9 millones de dólares.

Según (ESET SECURITY, 2018) afirma en su reporte Latinoamérica del año 2018 se analizaron detecciones de códigos maliciosos provenientes de la Familia FileCode Ransomware, y se puedo observar que la mayor cantidad de detecciones durante el año 2017, se realizaron en Perú, con un 25%. El segundo puesto lo ocupó México, con el 20% de las detecciones de este ataque cibernético, en tercer lugar, Argentina alcanzó el 15%, Brasil (14%) y Colombia (10%). Ver Figura Nº 09.

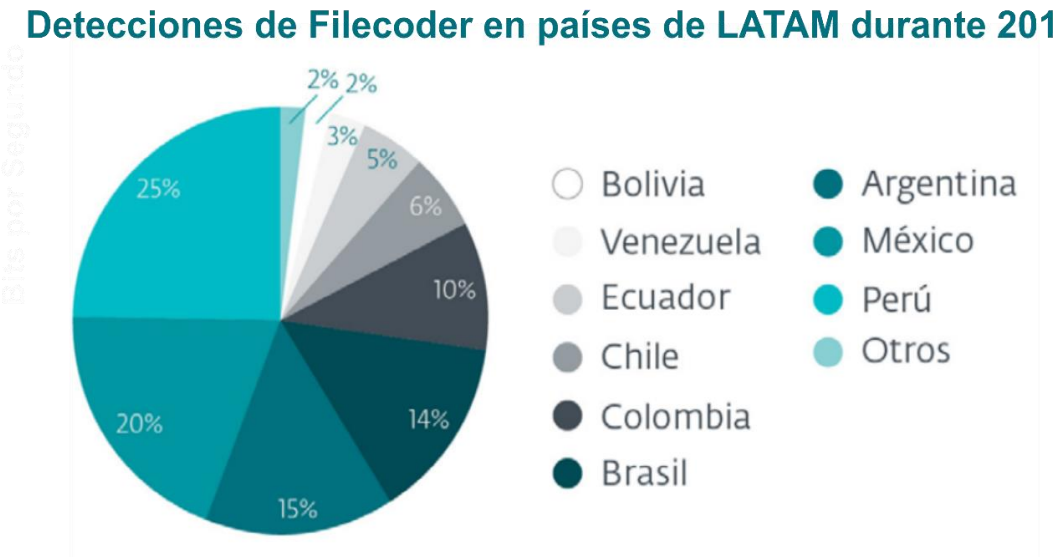

Figura 9: Detecciones de Filecoder en países de LATAM durante 2017.
Fuente: (ESET SECURITY, 2018, pág. 7)

VERSIGIN, (2018), afirmó en su Segundo Reporte Trimestral, la empresa Verisign observó el ataque de mayor intensidad alcanzando alrededor de 38 Gbps y 4.7 Mbps, y duró aproximadamente dos horas. Siendo el 56% de ataques DDOS inundados por UDP, y en segundo lugar obtuvo el ataque TCP, representado el 26% de ataques cibernéticos. Ver Figura Nº 10.

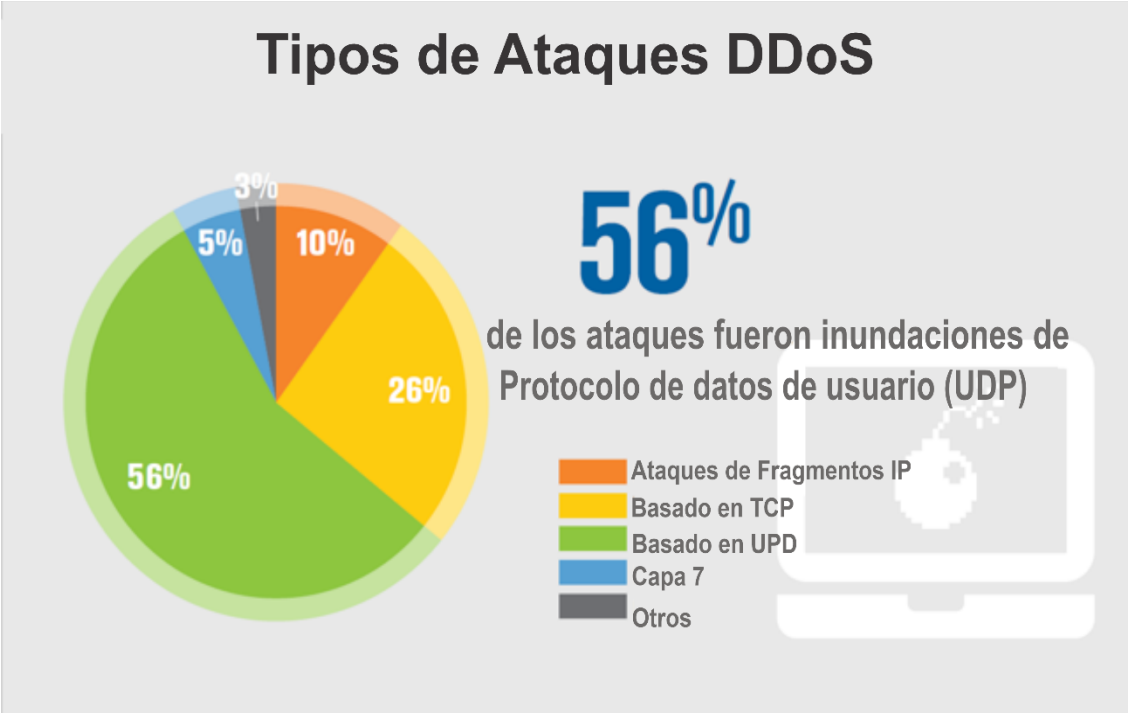

Figura 10: Tipos de Ataques DDOS 2018.

Fuente: (VERSIGIN, 2018)

WIRED, (2018), nos informa que la plataforma de desarrollo colaborativa de programación GitHub sobrevivió al ataque de Denegación de Servicio Distribuido más potente nunca antes registrado en la historia. El ciberataque DDoS a GitHub alcanzó un nivel de 1,35 Tbps (126,9 millones de paquetes por segundo) causando problemas en su conectividad. Ver Figura Nº 11.

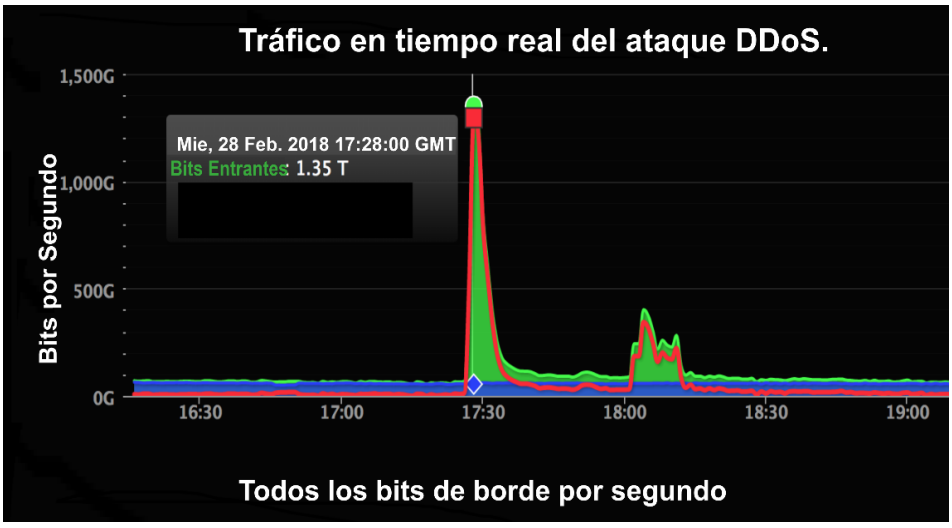

Figura 11: Tráfico en tiempo real del ataque DDoS.

Fuente: (WIRED, 2018)

Según (Gutiérrez, 2018) durante la primera mitad del año, se detectó una nueva familia Ransomware que apareció en el presente año por primera vez, posicionándose dentro del Top 5 de familias de Ransomware, registrándose esta familia Ransomware GandCrab en los países de: Perú con un 45,2%, México con un 38%, Ecuador con un 17,2%, Colombia con un 9,9% y Brasil obteniendo el 8,7% que registraron este ciberataque.

1.2. Antecedentes de Estudio.

1.2.1. Antecedentes de la Investigación.

Según (Tribak H. , 2012) desarrolló la investigación denominada: *Análisis Estadístico de Distintas Técnicas de Inteligencia Artificial en Detección de Intrusos,* desarrollado en la Universidad de Granada de España (Tesis Doctoral). En esta investigación el autor realiza un análisis comparativo de las diferentes técnicas de Machine Learning y establece que el algoritmo es mejor en determinadas condiciones, aplicadas en la detección de intrusos, para entrenar y evaluar los algoritmos se utilizó el conjunto de datos, Knowledge Discovery in Databases **(**-KDD), empleando para ello la selección de atributos, reduciendo su dimensionalidad y eliminando el ruido, mejorando el desempeño del algoritmo, también se utilizó la discretización que es empleado en Inteligencia Artificial, permitiendo cambiar los atributos nominales a categóricos La discretización de tipo no supervisada ha tenido mejores prestaciones, esto es debido a que el sistema ha contado con más datos para su entrenamiento que en la de Fayyad & Irani.

Esquivel, (2012), realizó la investigación denominada: *Sistema de detección de intrusos sobre la red basado en redes neuronales,* desarrollado en el Instituto Tecnológico De Costa Rica (Tesis de Post-Grado), el investigador nos da a conocer cómo funciona la seguridad en los IDS basado en anomalías, el cual permite reconocer si es un ataque, indistintamente ya sea un nuevo ataque o un ataque ya conocido. Pero para poder lograrlo este propósito, los sistemas utilizan la inteligencia artificial o de minería de datos, esta investigación utilizó redes neuronales de Perceptrón multicapa, y el aporte que realizó el autor, fue la implementación de un modelo de red neuronal perceptrón multicapa, el cual fue utilizado como un artilugio para detectar

intrusiones en tiempo real, empleando para ello patrones fuertemente adaptados en mensajes que se usan para comunicarse y permitiendo la predicción con otros mecanismos de detección sin necesidad de poner en funcionamiento la totalidad del método, se empleó el dataset Darpa para entrenar dicha técnica. De los resultados obtenidos, la tasa de falsos positivos mostró un 0,7% con tráfico normal, los verdaderos positivos alcanzó un 8,33% con ataques conocidos y los falsos positivos obtuvieron el 100% de ataques desconocidos.

Pérez & Deyban, (2017), desarrollaron la investigación: *Diseño, Implementación y Evaluación de Técnicas Híbridas de Aprendizaje Automático en la Detección de Intrusos en Redes de Computadoras*, desarrollado por la Universidad Central de Venezuela (Tesis de Grado). En este trabajo nos afirma que con el incremento de la red de computadoras a nivel mundial se han ido implementando nuevas tecnologías como son la interconexión digital de objetos frecuentes con internet y los servicios en la nube, que demandan novedades en metodologías de seguridad que brinden la confianza a los cliente un servicio confiable y fiable, para ello primeramente, el autor llevo a cabo una investigación meticulosa de las últimas tendencias de Aprendizaje Automático y seguridad en redes de computadoras, el investigador empleó el dataset NSL-KDD para estudiar, diseñar e implementar patrones híbridos de Aprendizaje Automático, el cual combinó algoritmos de machine learning como Support Vector Machine con redes neuronales acoplado con el algoritmo no-supervisado K-Medias, obteniendo muy buenos resultados y mejorando así la tasa de aciertos en la detección de intrusiones.

Luna, (2015), realizó la investigación denominada: *Sistema Detector de Intrusiones Ocupando una Red Neuronal Artificial,* desarrollado en la Universidad Autónoma del Estado de México (Tesis de Post-Grado). En esta investigación el autor se enfocó en desarrollar e implementar un modelo de red neuronal artificial, que fue acoplada en un IDS, logrando obtener resultados sumamente favorables, como la obtención de disminución de tasa de falsos positivos, gracias al debido entrenamiento que se realizó, proporcionándole a la red neuronal un aprendizaje de alto nivel, permitiendo

así predecir los ataques y disminuyendo la complejidad del algoritmo neuronal. Para la validación de su propuesta planteada se diseñó e implementó las pruebas en un IDS, fundamentado con redes neuronales artificiales, es por ello que de los resultados obtenidos se logró alcanzar diferentes patrones de comportamiento para cada cliente, obteniéndose su identidad de manera correcta, finalmente se verificó el resultado entre estos comportamientos y se examinó que se reconocieron intrusiones de manera correcta.

Mendoza, (2013), desarolló la investigación: *Aplicación de Selección De Características, Métricas De Aprendizaje Y Reducción de Dimensión en Sistemas de Detección de intrusos*, desarrollado en la Universidad Tecnológica De Bolívar de Colombia (Tesis de Post-Grado), el investigador propone un prototipo que permite identificar los ataques realizados a una red, dicho prototipo se basa en los datos pertenecientes al conjunto de datos NSL-KDD, el cual es tratado a través de diferentes técnicas como selección de características y reducción de dimensión, siendo evaluados por las métricas de Aprendizaje, ya son de gran aporte para la construcción de los IDS permitiéndole identificar tipos de ataque de manera eficiente, la utilización de dichas técnicas tiene como objetivo lograr resultados favorables para la detección de ataques en redes de equipos de cómputo, 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. Se utilizó el método MRMR perteneciente al tolbox de Matlab Fast, el cual es utilizado para la selección de características el cual obtiene porcentajes de precisión considerados como confiables para la identificación de tipos de ataques. De 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 y el PCA como técnica de reducción de dimensión y métrica de aprendizaje arrojan resultados prometedores, para identificar los diferentes 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).

Novillo & Guaño, (2012), realizaron la investigación: *Implementación De un Sistema de Detección De Intrusos utilizando inteligencia artificial*, desarrollado en la Escuela Politécnica Nacional de Quito (Tesis de Grado), los autores de esta investigación diseñaron reglas de Inteligencia Artificial adaptándolo en un IDS, con el propósito de enriquecer, logrando así mejorar la detección de distintos tipos ataques, ya que los IDS presentaron un alto índice de generación de falsos positivos y negativos. De los resultados obtenidos no se tuvo pérdida de paquetes, ni en tráfico normal ni en ataques, por lo que se puede decir que el sistema es eficiente, también se puede decir que consumió mucha memoria mientras se ejecuta el software, lo que es beneficioso para el rendimiento del equipo en el que se instaló, se puedo comprobar que el tráfico normal del módulo inteligente desarrollado, no genera falsas alarmas como es el caso del módulo propio de Snort

Hoyos, (2015), desarrollo la investigación denominada: *Prototipo de detección de ataques distribuidos de denegación de servicios (ddos1) a partir de máquinas de aprendizaje*, desarrollado en la Universidad Autónoma de Manizales de Colombia (Tesis de Post- Grado). El autor de esta investigación implementó un prototipo computacional basado en modelos supervisados con técnicas de Máquinas de aprendizaje SVM, permitiéndole detectar tipos de comportamientos anómalos a través del entrenamiento y la respectiva evaluación del modelo generado. La metodología aplicada combino 2 modelos, un modelo para la gestión del conocimiento KDD con el objetivo de tener un proceso iterativo en la captura, filtrado, normalización, implementación y pruebas del artefacto computacional. En su primera etapa realizó todo el proceso de captura de tráfico de red, en su segunda etapa desarrolló un filtrado de cabeceras HTTP3, en su tercera etapa hizo el proceso de normalización de los datos teniendo como base las variables operacionales: Tasa de Falsos Positivos y Negativos, Tasa de Clasificación y envía la información al algoritmo Support Vector Machine para el respectivo entrenamiento y pruebas de detección , y como último paso, la respectiva

evaluación del prototipo en la clasificación de los registros anómalos o normales. Luego de esto lo integra con el software estadístico para minería de datos WEKA4, permitiéndole reconocer efectivamente estos comportamientos extraños, aumentando el tiempo de disponibilidad de los servicios. Luego de esto el objetivo es evaluar, validar y comparar la técnica del prototipo basado en un modelo supervisado SVM, contra un modelo tradicional basado en reglas como SNORT.

1.2.2. Estado del Arte.

Sultana & Jabbar, (2016), realizaron la investigación denominada: *Intelligent network intrusion detection system using data mining techniques*, mediante el cual proponían un método inteligente de detección de intrusos en la red, utilizando para ello el algoritmo AODE para la detección de diferentes tipos de ataques, primeramente emplean el algoritmo AODE, luego cargan el conjunto de datos NSL KDD con ataques DDOS, Sonda U2R, R2L, después aplican la técnica de preprocesamiento: discretización, el siguiente paso agrupan los conjuntos de datos en cuatro tipos, después particionan cada grupo en conjuntos de entrenamiento y prueba, después de ello dataset se entrega al algoritmo AODE para el entrenamiento para clasificar los tipos de ataque y luego obtienen el registro de rendimiento (la precisión, DR, FAR). Se empleó también la técnica de validación cruzada para poder clasificarlos. De los resultados obtenidos mostraron que el modelo propuesto basado en el algoritmo AODE es eficiente con bajo de tasa de falsas alarmas (FAR) y tasa de detección alta.

Ahmad, Kusrini & Sudarmawan, (2017), desarrollaron la investigación, *Classification of Intrusion Detection System (IDS) Based on Computer Network*, los autores proponen un nuevo método aplicando la técnica de clasificación de aprendizaje automático se realizaron pruebas de clasificación de Sistemas de detección de intrusiones, utilizando el algoritmo naivebayes con una combinación de otros métodos para mejorar el rendimiento del método, también se aplicaron la normalización de los datos, empleando para ello el conjunto de datos NSL-KDD. De los resultados alcanzados demostraron que, al utilizar el algoritmo de agrupación k-means para la

discretización continua de variables y la selección de características optimizó el rendimiento del algoritmo naivebayes en la clasificación de los tipos de intrusión.

Wang, Xu, Lee & Lee, (2017), realizaron la investigación denominada: *Network Intrusion Detection Using Equality Constrained-Optimization-Based Extreme Learning Machine*, la perspectiva de estos autores, es la implementación máquinas de aprendizaje extremo, el cual constaba en la implementación de redes neuronales artificiales con una sola capa, pues estas no necesitaban de un entrenamiento repetitivo, su aprendizaje es acelerado y veloz, eran importante para implementar Sistema de detección de intrusos en una Red (NIDS),los resultados demostraron que el prototipo propuesto en esta investigación se logró alcanzar muy buenas tasas de detección, logrando así una mayor precisión y rapidez en detectar intrusos en la red. para su evaluación emplearon los datasets KDD-Cup 99y UNSW-NB15 en esta investigación.

Yihunie, Abdelfattah & Regmi, (2019), desarrollaron la investigación llamada: *Applying Machine Learning to Anomaly-Based Intrusion Detection Systems*, teniendo como misión principal hallar un técnica de Machine Learning que detecte tráfico de anomalías, para cumplir con su objetivo evaluaron 5 técnicas de aprendizaje automático, las técnicas son: El Gradiente Estocástico Decente, Random Forests, Regresión Logística, Máquina de Soporte Vectorial y el Modelo Secuencial, siendo evaluados y probados para la obtener su eficiencia de los algoritmos. De los resultados obtenidos se calcularon y comparan distintas medidas de rendimiento, donde algoritmo Random Forest mostró resultados significativamente altos alcanzando una precisión del 0.9992 %, concluyéndose que el algoritmo Random Forest superó significativamente a los otros cuatro algoritmos, sin aplicar el proceso de normalización al conjunto de datos.

Mylavarapu, Johnson & Kumar (2015), realizaron la investigación*: Real-time Hybrid Intrusion Detection System using Apache Storm*, los autores propusieron un método de detección híbrido basado en redes neuronales artificiales. La red neuronal CC4 conjunto con el Perceptrón Multicapa, reduce

la tasa de falsos positivo, el cual consta de 2 redes neuronales. La red neuronal CC4, el cual ejerce como una detección basada en anomalías para ataques desconocidos y la red Neuronal Perceptrón Multicapa interviene como una detección basada en el mal uso para ataques conocidos, el método híbrido obtuvo en la métrica de precisión el 89% en la predicción de ataques, también alcanzó una baja tasa falsos positivos del 4.32%, demostrando eficiencia de este modelo, utilizando para lograr este objetivo el conjunto de datos ISCX- 201.

Depren, Topallar, Anarim, & Kemal (2005), desarollaron la investigación denominada: *An intelligent intrusion detection system (IDS) for anomaly and misuse detection in computer networks*, los autores propusieron una nueva arquitectura del Sistema de Detección de Intrusos (IDS) que emplea enfoques en detección de anomalías y de uso indebido con el fin de solucionar y proteger las redes informáticas, protegiénodse de los famosos ataques de denegación de servicio (DoS), mejorando así la divulgación no autorizada de la información, y evitando la modificación o destrucción de los datos, para lograr esta solución se empleó la técnica J.48 de arbol de decisión, para así poder clasificar los diferentes tipos de ataques, ya que el objetivo del algoritmo de detección de intrusos se convierte en entrenar el sistema con datos normales y modelar el tráfico de red normal a partir del conjunto de datos normales ,tambien emplearon una estructura de mapa autoorganizado (SOM) para modelar el comportamiento normal, ya que El SOM se basa en el aprendizaje no supervisado para mapear relaciones estadísticas no lineales entre alta dimensión de datos de entrada en celosía o cuadrícula bidimensional que también se denomina espacio de salida, los Mapas autoorganizados colocan de manera eficiente patrones similares a ubicaciones adyacentes en el espacio de salida y brindan opciones de proyección y visualización para datos de alta dimensión,tambien se empleó la tecnica k-means, pues como resultado, para cada valor de byte de origen se calculan las probabilidades de pertenecer a cada centro de clúster, usando el enfoque propuesto anteriormente, se ha superado el problema de normalización definido anteriormente, llegando a la conclusion que de los resultados de simulación de anomalía y módulos de detección de uso indebido

se logró obtener una alta tasa en detección de intrusos con un 98.96% y en falsos positivos se alcanzó 1.01% y también para el siguiente módulo, se alcanzó una tasa de clasificación del 99.61% y a la vez lograndose la mas baja tasa de falsos positivos de 0,20% para el módulo de detección de uso indebido.

Mustapha, Salah & Mohamed (2018), realizaron la investigación denominada: *Performance evaluation of intrusion detection based on machine learning using Apache Spark*, informándonos que los métodos desarrollados de detección de intrusiones de red tienen como objetivo identificar ataques o actividades maliciosas en un entorno de red, y que ya se han propuesto varios métodos para encontrar una solución efectiva y eficiente para detectar y prevenir la intrusión en la red,el problema que enfrenta la investigación es la baja precisión y el tiempo de predicción en un IDS ,las cuales emplean clasificadores múltiples con diferentes paradigmas de aprendizaje para evaluar diferentes modelos clasificadores, se utilizo 4 algoritmos de aprendizaje automático conocidos Naïve Bayes, Decision Tree, and Random Forest, Se trabajo con el conjunto de datos UNSW-NB15. Para el análisis utilizan la herramienta de procesamiento de big data llamada Apache Spark, la cual la utilizan para la detección de intrusiones en el tráfico de la red y de los resultados obtenidos se dedujo clasificador Random Forest funciona mejor que todos los clasificadores restantes en términos de sensibilidad. Esto se debe al hecho de que obtiene un 93.53% de sensibilidad, seguido por el Árbol de decisiones con un 92.52%. Naïve Bayes y SVM tienen casi la misma sensibilidad con valores 92.16% y 92.13%,de la especificidad en Random Forest y Decision Tree son casi iguales con 97.75% y 97.10% respectivamente. Sin embargo, la especificidad para el esquema basado en SVM es aproximadamente 91.15%. Naïve Bayes es el menos calificado entre todos los clasificadores en términos de especificidad. y en tiempo de prediccion Random Forest tomó el menor tiempo aproximadamente 0.08 segundos y por lo tanto es el esquema de detección más rápido de todos.

Manjula & Muniyal, (2016), desarollaron la investigación denominada: *Performance Evaluation of Supervised Machine Learning Algorithms for*

Intrusion Detection", afirmando que el Modelo de detección de intrusión, denominado como modelo predictivo permite predecir el tráfico de datos de red y en la cual evaluaron los agoritmos Regresión Logística, Gaussian Naive Bayes, Máquinas de Soporte Vectorial y Bosque aleatorio, pues se utilizó el cojunto de datos NSL-KDD,la cual disponer de las 42 características, pues lo que primero es realiza el pre-proceso del cojunto de los datos, la cual los divide en datos de prueba y entrenamiento, luego construye los modelos Regresión logística, Gaise Naive Bayes, Máquina de vectores de soporte y clasificadores Random Forest, ya que essos los modelos se usan para predecir las etiquetas de los datos de prueba y la Metodología que utiliza en la etapa de pre-procesamiento hacen que todos los datos categóricos que están en forma textual se convierten a forma numérica,logrando asi llegar a la conclusión de que el clasificador Random Bosque Clasi (RFC) supera a los otros otros clasificadores para los parámetros y conjuntos de datos considerados. Pues tiene la precisión del 99%, también no dice que los resultados experimentales muestran que Random Bosque Clasi (RFC) realiza los otros métodos en la identificación de si el tráfico de datos es normal o un ataque.

Nadiammai & Hemalatha, (2012), realizaron la ivestigación: *Perspective Analysis of Machine Learning Algorithms for Detecting Network Intrusions*, proponiendo la idea de aplicar técnicas de minería de datos a la base de datos de detección de intrusos para así poder evaluar la efectividad de los clasificadores basados en reglas como: Ridor, NNge, DTNB, JRip, Regla Conjunta, One R, Zero R, Tabla de decisiones, RBF, Percepción de múltiples capas y algoritmos SMO en la cual utilizan (Fold-Cross Validation) o validación cruzada también, ya que este es un modelo que nos permite acceder a los resultados de un análisis estadístico-general para generalizar un conjunto de datos independiente. Tambien se ha utilizado en este conjunto de datos KDD CUP 99 para evaluar los métodos de detección de anomalías, evaluando el redimiento de los clasificadores o algoritmos, ya sea por Precisión, Sensibilidad, Especificidad, error absoluto medio (MAE) y el error cuadrático medio (RMSE) y en los resultados se obtuvieron fue que el clasificador SMO obtuvo el porcentaje de 97.78 % de precisión, 96.83% en sensibilidad y en

especificidad el 97.82% más alto en comparación con los otros clasificadores basados en reglas y llegaron a la conclusion que el algoritmo de clasificación SMO funciona bien en términos de precisión, especificidad y sensibilidad entre estos clasificadores, un algoritmo de optimización mínimo secuencial (SMO) supera a todos los demás algoritmos utilizados en precision, sensibilidad y especificidad.

En la investigación denominada: *A Comparative Study of Classification Techniques for Intrusion Detection*, realizada la investigación por los autores (Chauhan, Kumar, Pundir, & Pilli, 2013). La finalidad primordial de esta investigación científica es encontrar y descubrir la mejor técnica de clasificación disponible, aplicada a una NIDS (Sistema de Detección de Intrusos de red). El estudio que se hizo fue análisis del Conjunto de datos NSL-KDD (Network Socket Layer- Knowledge Discovery and Data Mining) utilizando las métricas de Exactitud Sensibilidad y especificidad para así poder evaluar el rendimiento de los algoritmos de clasificación dando como resultado que los clasificadores que Random Forest ha superado con respecto a la exactitud, especificidad y sensibilidad a continuación la siguiente Tabla Nº 01.

Tabla 1:

Métricas de rendimiento de los clasificadores usando NSL-KDD.

Classsifiers	**Accuracy**	**Sensibility**	**Specificity**	**Time (Sec)**
BayesNet	96.562	99.115	93.638	2.2.
Logistic	97.269	97.88	96.568	14.21
SGD	97.443	98.535	96.193	28.17
IBK	99.44	99.583	99.615	0.04
Jrip	99.583	99.695	99.454	28.67
PART	99.603	99.687	99.506	14.49
J48	99.559	99.553	99.565	6.73

Classsifiers	Accuracy	Sensibility	Specificity	Time (Sec)
Random Forest	99.746	99.881	99.591	6.77
Random Tree	99.507	99.561	99.446	0.6
REP Tree	99.543	99.628	99.446	1.88

Nota: Tomado de (Chauhan et al., 2013, pág. 42).

En la siguiente investigación denominada: *Comparative Analysis of Classification Algorithms Performance for Statistical based Intrusion Detection System*, realizado por los autores (Muzammil, Qazi, & Ali, 2013). En este estudio científico, se evaluó los cinco clasificadores (Naive Byaes, C4.5, Decision Table, Zero R, Oner R), midiéndolos mediante la métrica de desempeño Exactitud para el IDS, basado en estadísticas, con el objetivo primordial de superar los inconvenientes del IDS basados en firmas, ya que no tiene la capacidad de detectar exploits cuyas firmas no existen en la base de datos. Se Concluyó finalmente en esta investigación que el mejor clasificador es el C 4.5 dando un porcentaje muy alto exactitud, ver a continuación la Tabla Nº 02.

Tabla 2:

Rendimiento de los clasificadores.

Classifier	Accuracy %	Modelo Building Time (segundos)	Size of the Tree
Naive Bayes	81.3813	3.69	
C4.5	99.7352	64.23	271
Decision Table	99.5598	89.76	
ZeroR	84.8837	0,1	
pOneR	99.5024	2.44	

Nota: Tomado de (Muzammil, Qazi, & Ali, 2013, págs. 4-6).

Así mismo en la siguiente investigación denominada: *Comparative Analysis of Machine Learning Algorithms along with Classifiers for Network Intrusion Detection*, realizado por los investigadores (Choudhury & Bhowal, 2015). En este estudio comparativo, se evaluó mediante métricas de desempeño el rendimiento de varios clasificadores utilizando la herramienta WEKA y se detalla a continuación de acuerdo con ciertas medidas de rendimiento en el siguiente Tabla N° 05. La simulación de estos modelos de clasificación se ha realizado utilizando una validación cruzada de 10 veces. Llegándose a la conclusión que **RANDOM FOREST Y BAYESNET** son los apropiados para la detección de intrusos de una red. Ver tabla Nº 03.

Tabla 3*:*

Lista del rendimiento de los algoritmos.

Clasificador	**Sensitivity (%)**	**Specificity (%)**	**Precision**	**Accuracy (%)**	**Training Time (seconds)**
BayesNet	85.8	96.18	0.962	90.66	0.02
Logistic	87.34	82.2	0.848	84.9651	0.71
IBk	88.68	93.06	0.936	90.7323	0.01
JRip	84.47	91.02	0.915	87.54	0.38
PART	85.32	93.63	0.938	89.2167	0.23
J48	86.48	93.29	0.936	89.6727	0.18
Random Forest	88.68	88.04	0.951	91.5236	0.49
Random Tree	87.74	94.43	0.947	90.8798	0.01
REPTree	79.78	93.52	0.933	86.2124	0.56

Nota: Tomado de (Choudhury & Bhowal, 2015, pág. 94).

Así mismo en la siguiente investigación denominada: *Generación de un vector característico para la detección de intrusos en redes computacionales,* realizado por los autores (Alcántara, López, & Rueda, 2017).En este estudio científico, se propone la generación de un vector característico a partir de información real y también evaluaron los algoritmos mediante métricas de precisión para comprobar la eficacia en la detección de intrusos, utilizando los conjuntos de datos KDD99, KDDCUP99, KDD99 y a continuación se presenta la siguiente Tabla Nº 04. Concluyéndose que el algoritmo **J48** es el que tuvo un alto porcentaje en precisión.

Tabla 4:

Comparación en el porcentaje de precisión y tipos de muestra de los dataset.

Precisión				
Muestra	**Red Neuronal**	**Alg. J48**	**Naive Bayes**	**Random Forest**
KDD99	98.52%	99.02%	98.14%	NA
KDDCUP99	77.41%	81.05%	76.56%	80.67%
KDD99 y Gure KDD	NA	NA	76.56%	80.67%

Nota: Tomado de (Alcántara, López, & Rueda, 2017, pág. 30).

En la siguiente investigación científica denominada: *Performance evaluation of intrusion detection based on machine learning using Apache Spark*, realizado por los autores (Belouch, El Hadaja, & Idhammadb, 2018), evaluando el rendimiento 4 algoritmos de clasificación Maquinas de Soporte Vectorial, Naive Bayes , Árbol de Decisión y Bosques Aleatorios en la cual utilizaron la herramienta Apache Spark utilizando las métricas de Exactitud, Sensibilidad y Especificidad, dando los investigadores como resultado al clasificador **RANDOM FOREST**, en la cual dio el mejor rendimiento y se utilizó la base de datos UNSW-NB15 que a continuación se presenta en la siguiente Tabla Nº 05.

Tabla 5:

Comparación de diferentes métodos de detección de intrusión utilizando el Conjunto de datos UNSW-NB15.

Methods	Accuracy	Sensitivity	Especificity	Training Time	Prediction Time
SVM	92.28	92.13	91.15	38.91	0.20
Naive Bayes	74.19	92.16	67.82	2.25	0.18
Decision Tree	95.82	95.52	97.10	4.80	0.13
Random Forest	97.49	93.53	97.75	5.69	0.08

Nota: Tomado de (Belouch, El Hadaja, & Idhammadb, 2018, pág. 4).

En esta investigación denominada: *Performance Evaluation of Supervised Machine Learning Algorithms for Intrusion Detection*, realizada por los investigadores (Belavagi & Muniyal, 2016). En este artículo científico se realiza los modelos predictivos en la detección de intrusiones, evaluando los algoritmos, la clasificación y los modelos predictivos en la detección de intrusiones se construyen, concluyendo que el clasificador de bosque aleatorio tiene la más alta precisión, supera a los otros métodos. Mientras que Support Vector Machine (SVM) tiene la más baja precisión, Regresión logística, estos algoritmos fueron utilizaron el conjunto de datos NSL-KDD a continuación se presenta la siguiente tabla Nº 06.

Tabla 6:

Medidas de Desempeño.

-	Precisión	Recal	F1- Score	Accuracy
LR	0.83	0.85	0.82	0.84
GNB	0.79	0.81	0.78	0.79
SVM	0.76	0.79	0.77	0.75
RFC	0.99	0.99	0.99	0.99

Nota: Tomado de (Belavagi & Muniyal, 2016, pág. 122).

En esta investigación denominada *Comparison of classification techniques applied for network intrusion detection and classification*, realizada por los autores (Aziz, Hanafi, & Hassanien, 2017).En este documento se presentó un método de detección de intrusos híbrido y jerárquico inspirado en conceptos de inmunidad, en lo cual muchos clasificadores fueron probados y comparados con el fin de encontrar el mejor clasificador para dar más información de los ataques conocidos, llegándose a la conclusión que cada algoritmo implementado es bueno en cierto ataque; por lo tanto, el clasificador **MLP (Multi-layer perceptron)** obtuvo buenos resultados en un alto porcentaje de precisión. Ver tabla Nº 07.

Tabla 7:

Valores de precisión y recuperación obtenidos por la clasificación del tráfico anómalo detectado por detectores Minkowski.

Classifier	**TP**	**FP**	**FN**	**Precision**	**Recall**	**F-Score**
Using all of the train data rec						
NB	7027	1264	2534	84.75%	73.50%	78.73%
BFTree	6744	121	4454	98.24%	60.23%	74.67%
J48	6580	156	4493	97.68%	59.42%	73.90%
MLP	6842	102	4322	98.53%	61.29%	75.57%
NBTreed	6627	105	4309	98.44%	60.60%	75.02%
RFT	6690	113	4663	98.34%	58.93%	73.69%
Using 20% of NB	the train data 6271	records 1297	2677	82.86%	70.08%	75.94%
BFTree	6874	136	3954	98.06%	63.48%	77.07%
J48	6822	105	4351	98.48%	61.06%	75.38%
MLP	6691	123	4260	98.19%	61.10%	75.33%
NBTree	6506	114	4392	98.28%	59.70%	74.28%
RFT	6608	100	4730	98.51%	58.28%	73.24%

Nota: Tomado de (Aziz, Hanafi, & Hassanien, 2017, pág. 115).

En la siguiente investigación denominada*: Performance comparison of intrusion detection systems and application of machine learning to Snort system*, realizado por los autores (Syed & Bijú, 2016). En este artículo científico se evaluó el rendimiento de dos sistemas de detección de intrusión de código abierto (IDS), Snort y Suricata para detectar con precisión el tráfico malicioso en la computadora, observándose que Suricata procesaba una mayor velocidad de tráfico de red y que es Snort tenía una menor tasa de caída de paquetes, pero consumía recursos en los equipos luego se realizó un estudio empírico con los algoritmos evaluados con diferentes dataset,, los algoritmos empleados son SVM, Decision Tre, Fuzzy Logic, Bayentes y Naive Bayes para así poder reducir las alarmas de falsos positivos. Concluyendo finalmente que el **SVM** mostró una mejor tasa de detección alta promedio de 96% y una tasa baja de falsos positivos promedio de 3%. Ver tablas Nº 08,09 y 10.

Tabla 8:

Rendimiento de algoritmos con el dataset 1.

Machine learning Algorithms	**DR**	**FPR**	**DA**
Support Vector Machines	96.8%	0.7%	95.6%
Decision Trees	79.2%	2.9%	82%
Fuzzy Logic	94.5%	0.2%	92.3%
BayesNet	65%	3.5%	73%
NaiveBayes	62%	3%	70%

Nota: Tomado de (Shah & Issac, 2016, pág. 16).

Tabla 9:

Rendimiento de algoritmos con dataset 2.

Machine learning Algorithms	**DR**	**FPR**	**DA**
Support Vector Machines	97%	0.5%	94.2%
Decision Trees	81.1%	1.9%	85%
Fuzzy Logic	92%	1.6%	94%
BayesNet	63%	5.1%	71.2%
NaiveBayes	65%	6%	71%

Nota: Tomado de (Shah & Issac, 2016, pág. 16).

Tabla 10:

Rendimiento de algoritmos con dataset 3.

Machine learning Algorithms	DR	FPR	DA
Support Vector Machines	97.3%	3.1%	95.4%
Decision Trees	78%	10%	81.2%
Fuzzy Logic	95%	4%	94%
BayesNet	69%	8%	74%
NaiveBayes	70%	7.6%	79%

Nota: Tomado de (Shah & Issac, 2016, pág. 16).

Así mismo en la siguiente investigación denominada*: Comparison of data mining algorithms and an analysis of relevant features for detecting cyber-attacks*, realizado por el investigador (Petersen, 2015). Se evaluó el rendimiento de estos algoritmos en comparación con la precisión, la tasa de error y el costo,utilizando el dataset NSL-KDS y concluyendo que el algoritmo de clasificación más preciso según los resultados del investigador es el **K-NN.** Ver tabla Nº 11.

Tabla 11:

Rendimiento de algoritmos con data 2.

Algorithm	Name in Weka	Accuracy	Error rate	Cost
k-NN	Lazy.IBk	77.0892%	22.9108%	0.6612
C4.5	Trees.J48	75.2573%	24.7437%	0.6738
CART	Trees.SimpleCART	74.5697%	25.4303%	0.6647
Naïve Bayes	Bayes.NaiveBayes	71.203%	28.797%	0.6808

Nota: Tomado de (Petersen, 2015, pág. 34).

Así mismo en la siguiente investigación denominada: *Anomaly network-based intrusion detection system using a reliable hybrid artificial bee colony and AdaBoost algorithms*, realizado por los autores (Mazini, Shirazi, & Mahdavi, 2018), en el año 2018.Los autores de esta investigación plantearon un sistema híbrido confiable para un IDS basado en la red (A-NIDS), basado en anomalías que emplea el algoritmo de colonia artificial (ABC) y AdaBoost, obteniendo así mejores resultados en alta tasa de detección (DR) , y una tasa baja de falsos positivos (FPR). El algoritmo ABC se usa para seleccionar las características y AdaBoost se usa para evaluación de características utilizando el conjunto de datod NSL-KDD a continuación se presenta la siguiente tabla N°12.

Tabla 12:

Comparando el enfoque propuesto con otros métodos.

Classification Algorithms	DR%	FPR	AC%	Feature Selection Method
K-NN+K-means	91.86	0.78	93.29	All Feature
DT [2]	91.500	3.372	92.500	CAT
SVM [17]	97.14	0.83	N/A	HG - GA
AdaBoost [This Paper]	99.61	0.01	98.90	ABC

Nota: Tomado de (Mazini, Shirazi, & Mahdavi, 2018, pág. 22).

Así mismo en la siguiente investigación denominada: *Improve the automatic classification accuracy for Arabic tweets using ensemble methods*, realizada por los investigadores (Abdelaal, Elmahdy, Halawa, & Youness, 2018). El objetivo de esta investigación es de clasificar y conceder automáticamente los Tweets Árabes a categorías predeterminadas en función de sus características lingüísticas, se emplearon tres algoritmos diferentes: para mejorar la precisión de los clasificadores NB, J48 y SMO, el objetivo principal de estos algoritmos es transformar un algoritmo de aprendizaje débil en un algoritmo de aprendizaje más robusto fuerte, reduciendo la tasa de falsos positivos. Ver tabla Nº 13.

Tabla 13:

Rendimiento de los métodos de conjunto con cada clasificador.

Porcentaje	Accuracy %			
Classifier	Individual	Begging	Boosting	Stacking
J48	83.6	84.6	86.0	86.8
JNB	87.0	88.6	87.0	87.6
SMO	86.4	88.0	88.6	88.4

Nota: Tomado de (Abdelaal, Elmahdy, Halawa, & Youness, 2018, pág. 8).

Así mismo también se evaluó la siguiente investigación denominada: *Building A Fast Intrusion Detection System For High-SpeedNetworks - Probe and DoS Attacks Detection*, realizado por los autores (Tchakoucht & Ezziyyani, 2018). Los autores de la investigación propusieron un método de detección para intrusiones, detectando ataques probe y DoS en redes de alta velocidad. Se utilizó el dataset KDD'99, para mejora la precisión y la eficiencia. Ver tabla Nº 14.

Tabla 14:

Comparación de exactitud.

Tipo de Ataque	Método	Todas las características				Características más importantes		
		DR (%)	FPR (%)	F-Score	ROC Area (%)	Dr (%)	FPR (%)	FScore
Probe	Reep Tree	99.3	3.8	0.682	98.5	99.8	2.7	0.748
	C4.5	87.3	0.5	0.873	94.3	87.3	0.5	0.873
	RF	95.9	0.4	0.931	99.8	90.0	0.4	0.896
	Reep Tree	89.3	1.1	0.828	94.2	87.5	0.5	0.872
	NB	99.5	8.0	0.913	96.1	99.6	5.1	0.942
DOS	C4.5	99.6	2.6	0.969	98.5	99.64	0.3	0.995
	RF	9.6	0.4	0.993	99.8	99.5	0.4	0.993
	REPTree	99.6	2.6	0.969	98.6	99.6	2.5	0.970

Nota: Tomado de (Tchakoucht & Ezziyyani , 2018, pág. 7).

1.3. Teorías relacionadas al tema

1.3.1. Seguridad informática:

Según (Alegre Ramos & Cervigón Hurtado, 2011). La seguridad de tecnología de la información se encuentra conformada por métodos, e instrumentos de medios informáticos responsables de asegurar la integridad, confidencialidad y disponibilidad, evitando vulnerabilidades, amenazas y ataques salvaguardando la información, reduciendo las amenazas. (p. 2).

Según (Gómes Beites, 2011). En su libro lo define como la prevención que evita, la ejecución de operaciones ilegales sobre un sistema o red Informática, cuya consecuencia puede dañar la información, comprometiendo su confidencialidad, autenticidad o integridad.

La seguridad Informática es una de las ramas más importantes en la seguridad de la información y son utilizados por empresas.

1.3.2. Principios de seguridad de los sistemas de información:

Según (Kim & Solomon, 2018, págs. 14-18), en su libro refiere que la información, que es segura satisface tres principios principales, o propiedades de información. Si puede garantizar estos tres principios, satisface los requisitos de seguridad información. Los tres principios son los siguientes:

Figura 12: Los tres principios de sistemas de información seguridad.
Fuente: (Kim & Solomon, 2018, pág. 15)

1.3.2.1. Confidencialidad:

(Kim & Solomon, 2018, págs. 14-18), se refiere a que sólo los usuarios autorizados pueden ver información. La confidencialidad significa proteger, prevenir o evitar divulgación de la información a personas inescrupulosas, a excepción de individuos que si se encuentran autorizadas. La información personal incluye lo siguiente:

a) Datos privados de individuos
b) Propiedad intelectual de las empresas
c) Seguridad nacional para países y gobiernos

1.3.2.2. Integridad:

Según (Kim & Solomon, 2018, págs. 14-18), este principio se ocupa de la autenticidad y la veracidad de los datos. Los datos se preservan íntegros, libre de alteraciones que no hayan sido manipulados o modificado algo en

ellos, si se pierde la integridad, estos datos no son válidos, no sirven para nada. Es por eso que la integridad es un principio de seguridad de los sistemas.

1.3.2.3. Disponibilidad:

Según (Kim & Solomon, 2018, págs. 14-18), en su libro refiere que la información es accesible por usuarios autorizados siempre que soliciten la información. La disponibilidad comúnmente es declarada como la capacidad de permanecer el tiempo necesario para que los clientes logren realizar el uso de los sistemas y que tengas acceso a sus datos al realizar una consulta.

1.3.3 Redes de Computadoras.

1.3.3.1. El modelo de referencia ISO OSI

Según (Kumar & Kalita, 2013) afirma que:

El modelo OSI fue presentado por la Organización Internacional para la estandarización (ISO) como un prototipo conceptual para la arquitectura de protocolos de computadora y como un marco para desarrollar estándares de protocolo. Este modelo comprende siete capas con funciones de red particulares.

1. **Capa de aplicación:** Esta capa permite a los usuarios finales interactuar con las aplicaciones como por ejemplo el envío de un correo.
2. **Capa de presentación:** Esta capa se responsabiliza de realizar la representación de los datos, como por ejemplo el cifrado de los datos, así como la descompresión de estos.
3. **Capa de sesión:** Proporciona la estructura de control para la comunicación entre aplicaciones. Establecimiento, gestión y la terminación de las conexiones (sesiones) entre las aplicaciones que cooperan son las principales responsabilidades de esta capa.
4. **Capa de transporte:** Esta capa admite la transferencia de datos confiable y transparente entre dos puntos finales. También es compatible con la recuperación de errores de extremo a extremo y el control de flujo.

5. **Capa de Red:** Esta capa proporciona independencia de las capas superiores de las tecnologías de transmisión y conmutación de datos utilizadas para conectar sistemas. Como por ejemplo aquí tenemos al direccionamiento, encapsulamiento y enrutamiento, además añade direcciones lógicas o de red

6. **Capa de enlace de datos:** La responsabilidad de transferir de manera confiable información a través del enlace físico se asigna a esta capa. Transfiere bloques (cuadros) con sincronización necesaria, error control y control de flujo.

7. **Capa física:** Esta capa es responsable de transmitir una secuencia de bits no estructurados a través del medio físico. Debe tratar problemas mecánicos, eléctricos, funcionales y de procedimiento para acceder al medio físico. (pág. 23).

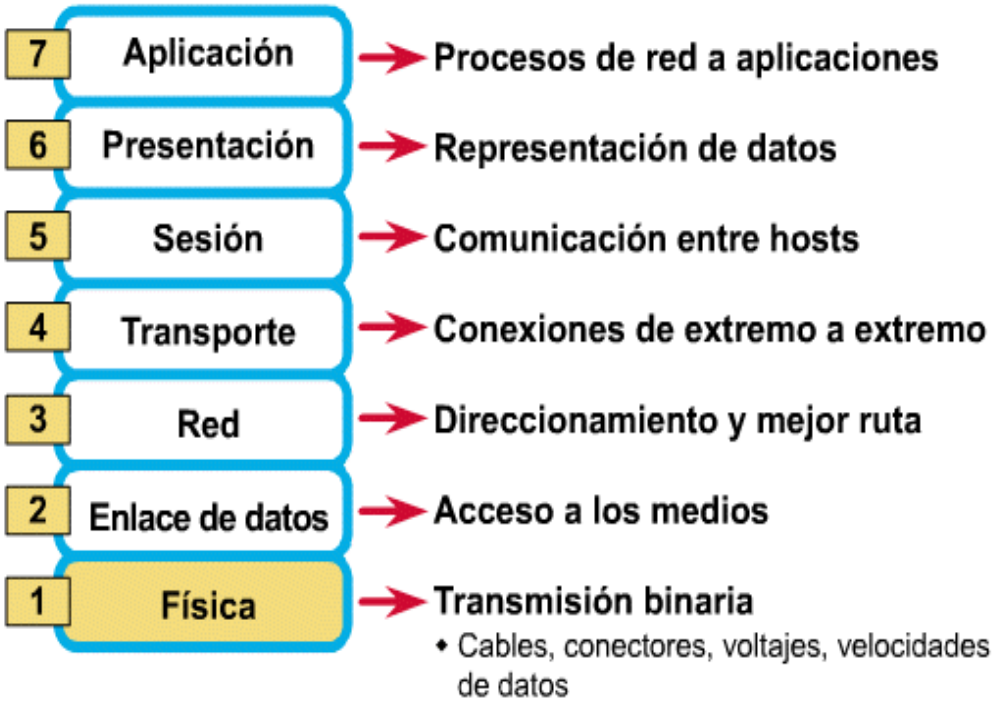

Figura 13: El Modelo Referencial OSI.

Fuente: (Kumar & Kalita, 2013, pág. 24)

1.3.4. Sistema De Detección De Intrusiones (IDS)

La definición del Sistema de Detección de Intrusos (IDS) es, según (Herrero & Corchado, 2014) afirma que: “Un conjunto de acciones que intentan comprometer cualquiera de los tres principios de seguridad (integridad, confidencialidad y disponibilidad) de cualquier recurso en un sistema informático mediante la explotación de vulnerabilidades”. (p. 8).

La detección de intrusos se convierte en un mecanismo de seguridad esencial para proteger sistemas y redes. Intenta detectar una actividad inadecuada o inconsistente en una red de computadoras, o en un host, mediante la exploración de ciertos tipos de datos a través del monitoreo (Khattab M, 2010).

1.3.5. Métricas de Desempeño

Según (De la Hoz, De la Hoz, Ortiz, & Ortega, 2012) en su investigación realizada informa que:

Para llevar a cabo la detección de intrusos y permita determinar las diferentes características o atributos que hacen referencia a los ataques, es necesario utilizar las siguientes métricas de desempeño, ya que esto nos permíteme determinar su eficiencia. (pag.93-94). A continuación, se nombrará las siguientes métricas.

Exactitud: Mide la tasa de las instancias de ataque correctamente clasificadas (los verdaderos positivos y verdaderos negativos). (Choudhury & Bhowal, 2015)

$$\text{Exactitud} = \frac{TP+TN}{TP+TN+FP+FN}$$

Precisión: Estima la posibilidad de que las técnicas de aprendizaje puedan pronosticar correctamente instancias positivas y negativas. (Chauhan, Kumar, Pundir, & Pilli, 2013).

$$\text{Precisión} = \frac{TP}{TP+FP}$$

Sensibilidad: Esta mide la proporción de la tasa de “verdaderos positivos” que son correctamente identificados como tales, dando la probabilidad de que el algoritmo pueda predecir instancias positivas correctamente. (Choudhury & Bhowal, 2015).

$$\text{Sensibilidad} = \frac{TP}{TP+FN}$$

Especificidad: Mide la proporción de “verdaderos negativos”, dando la probabilidad de que la técnica pueda predecir instancias negativas correctamente. (Mustapha, Salah, & Mohamed , 2018).

$$\text{Especificidad} = \frac{TN}{TN+FP}$$

TP (verdadero positivo) Se refiere a la cantidad de ataques clasificados correctamente.

TN (verdadero negativo) Es la cantidad de registros normales correctamente clasificados.

FP (falso positivo) Se refiere a la cantidad de ataques mal clasificado.

FN (falso negativo) Se refiere a la cantidad de registros normales mal clasificados.

Tiempo de entrenamiento: Según los autores (Karim & Kaysar, 2016), el tiempo de ejecución requiere terminar el pre procesamiento de datos o construcción el modelo y varía mucho de acuerdo a los algoritmos.

TI: Corresponde al tiempo Inicial del entrenamiento.

TF: Corresponde al tiempo Final del entrenamiento.

1.3.6. Inteligencia Artificial

La Inteligencia Artificial es definida como: “El arte, la ciencia y la ingeniería para fabricar agentes, máquinas y programas inteligentes” (Sarkar, Bali, & Sharma, 2017), La Inteligencia artificial abarca múltiples subcampos que incluyen Aprendizaje automático, procesamiento del lenguaje natural, extracción de datos, etc”.

1.3.7. Aprendizaje Automático (Machine Learning)

Según (Theobald, 2017), en su libro lo de fine como la disciplina de la ciencia de datos que aplica métodos estadísticos para mejorar el rendimiento en función de la experiencia previa o detectar nuevos patrones en grandes cantidades de datos.

Según (Zhenwei & Tsai, 2011, pág. 25)**,** en su libro lo define al aprendizaje automático: como el diseño, desarrollo de algoritmos y técnicas que permiten a los sistemas informáticos de forma autónoma adquirir e integrar conocimientos para mejorarlos continuamente hasta el final sus tareas de manera eficiente y efectiva. Algoritmos de aprendizaje automático demostrado ser de gran valor práctico en una variedad de dominios de aplicación.

1.3.7.1. Ventajas del aprendizaje automático

Las principales ventajas que ofrece Machine Learning o algoritmos de aprendizaje automático o las siguientes:

a) Se emplean para el manejo de datos multidimensionales y de múltiples variedades en entornos dinámicos.
b) Rápido análisis de predicción y procesamiento.
c) Permite la disminución del periodo o de tiempo y la utilización eficiente de los recursos.
d) Medios cambiantes: un sistema de aprendizaje automático permite el acoplamiento a los nuevos datos.

1.3.7.2. Desventajas del aprendizaje automático

Las principales desventajas que ofrece Machine Learning o algoritmos de aprendizaje automático o las siguientes:

a) Se emplean para el manejo de datos multidimensionales y de múltiples variedades en entornos dinámicos.
b) Rápido análisis de predicción y procesamiento.
c) Permite la disminución del periodo o de tiempo y la utilización eficiente de los recursos.
d) Medios cambiantes: un sistema de aprendizaje automático permite el acoplamiento a los nuevos datos.

Se han implementado muchos tipos diferentes de técnicas de aprendizaje automático para solucionar problemas en varios campos. “Estas técnicas de Machine Learning se clasifican en 3 tipos dependiendo del método de capacitación”. (Kim P. , 2017, pág. 12).

a) Aprendizaje Supervisado.

b) Aprendizaje No Supevisado.

c) Aprendizaje reforzado.

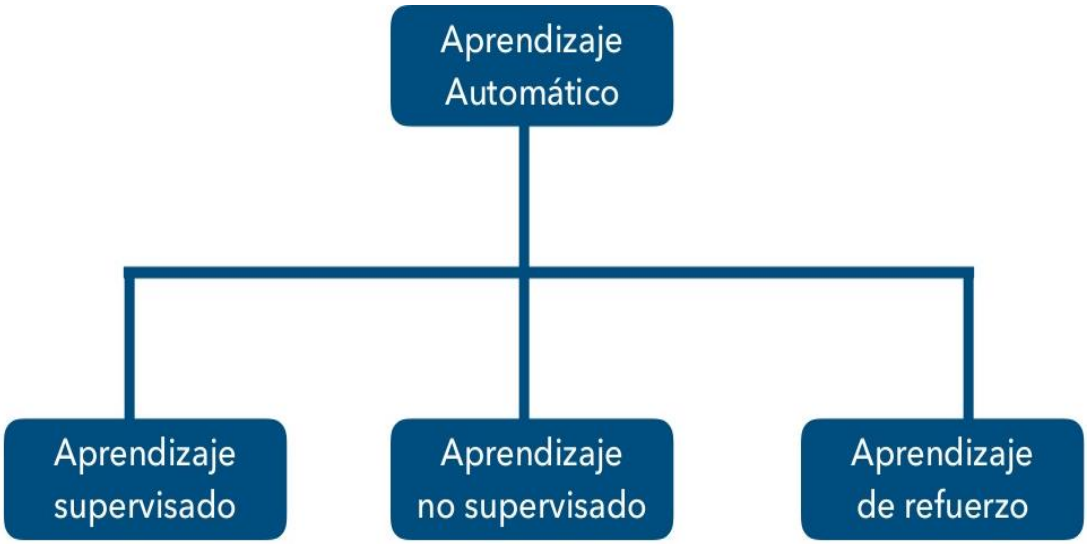

Figura 14: Tipos de técnicas de aprendizaje automático.

Fuente: (Kim P. , 2017, pág. 21)

1.3.8. Tipos De Aprendizaje Automático

Sullivan, (2017), en su libro denominado: *Machine Learning Beginners Guide Algorithms: Supervised & Unsupervised Learning, Decision Tree & Random Forest Introduction*, realiza una clasificación en nivel superficial, la cual lo divide en 3 partes, y son las siguientes:

1.3.8.1 Aprendizaje Supervisado

Este aprendizaje es considerado, la más avanzada y madura de todas las formas de aprendizaje, según (Sullivan, 2017) nos dice que:

Los datos de entrada se conocen como datos de entrenamiento. También se construye un modelo de predicción adecuado utilizando el proceso de capacitación en la cual es indispensable hacer predicciones, y estas predicciones se corrigen si están equivocadas. El proceso de

entrenamiento continúa reiterándose hasta lograr la perfección, o el nivel deseado de precisión se logra. Ejemplos: Logistic Regression y Back Propagation Neural Network.

1.3.8.2. Aprendizaje No Supervisado

En el Aprendizaje No Supervisado los datos que se introducen en el sistema, es decir, los datos de entrenamiento, no incluyen ningún resultado deseado. Por lo tanto, los datos que se van a alimentar no tienen salida, según (Sullivan, 2017) afirma que:

Los datos en esta categoría no tienen un resultado o resultado conocido ni mucho menos tiene una etiqueta. El modelo de predicción lo fabrica adivinando el número de estructuras presentes en los datos de entrada, a menudo para extraer reglas generales. Los problemas de ejemplo incluyen reducción de dimensionalidad, agrupamiento y aprendizaje de reglas de asociación. Aquí podemos encontrar los algoritmos de ejemplo incluyen varios algoritmos, como el algoritmo Apriori y k-Means.

1.3.8.3. Aprendizaje Semi-Supervisado:

En el aprendizaje Semi-Supervisado es una mezcla de ambos tipos de aprendizaje, donde los datos de entrenamiento contienen algunos, pero no todos, los resultados deseados, los datos de entrada no tienen forma sólida y es una mezcla mezclada de ejemplos etiquetados y no etiquetados. “Los datos de entrada no tienen una forma sólida y son una mezcla de ejemplos etiquetados y no etiquetados. Este modelo necesita aprender las estructuras que son necesarias no solo para organizar los datos sino también para hacer predicciones”. (Sullivan, 2017)**.**

1.3.9. Pasos Para Realizar Una Tarea De Aprendizaje Automático

Según (Kunal, 2015) en su libro nos dice que existen 5 pasos fundamentales empleados para el aprendizaje automático:

a) Recopilación de datos:

Aquí se recopilan los datos brutos de Excel, acceso de archivos de texto, etc., aquí se deberá seleccionar, averiguar y conseguir los datos para su preparación inicial y estudio representativo de los datos que serán empleados para el aprendizaje a futuro. (Kunal, 2015).

b) Preparación de los datos:

Esta fase se necesita dedicar más tiempo al preprocesamiento, limpieza y tratamientos, priorizando la calidad de los datos. El análisis exploratorio, es quizás uno de los métodos para estudiar los matices de los datos en detalle y de esta forma aumentar el contenido nutricional de los datos. (Kunal, 2015).

c) Entrenando un Modelo: En este paso se aplica la elección de las técnicas de aprendizaje apropiados al modelo. Los conjuntos de datos son divididos en 2 partes: entrenamiento y prueba; la Primera parte se usa para entrenar los datos al modelo y la segunda parte (datos de prueba) se usa como referencia. (Kunal, 2015).

d) Evaluación del Modelo: En la evaluación del modelo se aplica la métrica de desempeño para medir la exactitud, y es evaluado el modelo con los datos de prueba del dataset. Aquí la exactitud juega un rol importante, en la elección del algoritmo en función del resultado determinando el desempeño del modelo. (Kunal, 2015).

e) Mejora del rendimiento: Este paso puede implicar elegir un modelo diferente o introducir más variables para aumentar la eficiencia. Es por eso que se debe dedicar una gran cantidad de tiempo a la recopilación y preparación de datos. (Kunal, 2015).

1.3.10. Minería De Datos (Data Mining)

Sarkar, Bali, & Sharma, (2017), en su libro lo define como el campo de la minería de datos que involucra: procesos, metodologías, herramientas y técnicas para descubrir y extraer patrones, conocimientos, ideas e información valiosa a partir de conjuntos de datos no triviales.

1.3.11. Ataques informáticos

La finalidad de un ciberataque o también denominado ataque informático, es robar, alterar o eliminar información accediendo de forma inescrupulosa sin autorización, empleadas por ciberdelincuentes, que se aprovechan de las debilidades, imperfecciones que se encuentras en los protocolos, sistemas operativos, aplicaciones, causando daños en la seguridad de la información. (De la Hoz E. , 2015).

1.3.11.1. Ransomware

Es Ransomware es un software malicioso que corrompe a los equipos de cómputo, bloqueando dispositivos, encriptando información, tomando el control y obstaculizando a los usuarios acceder a su información personal, pues para poder tener acceso a sus archivos e información, tienen que realizar un pago de rescate para poder recuperar la información (INCIBED, pág. 2).

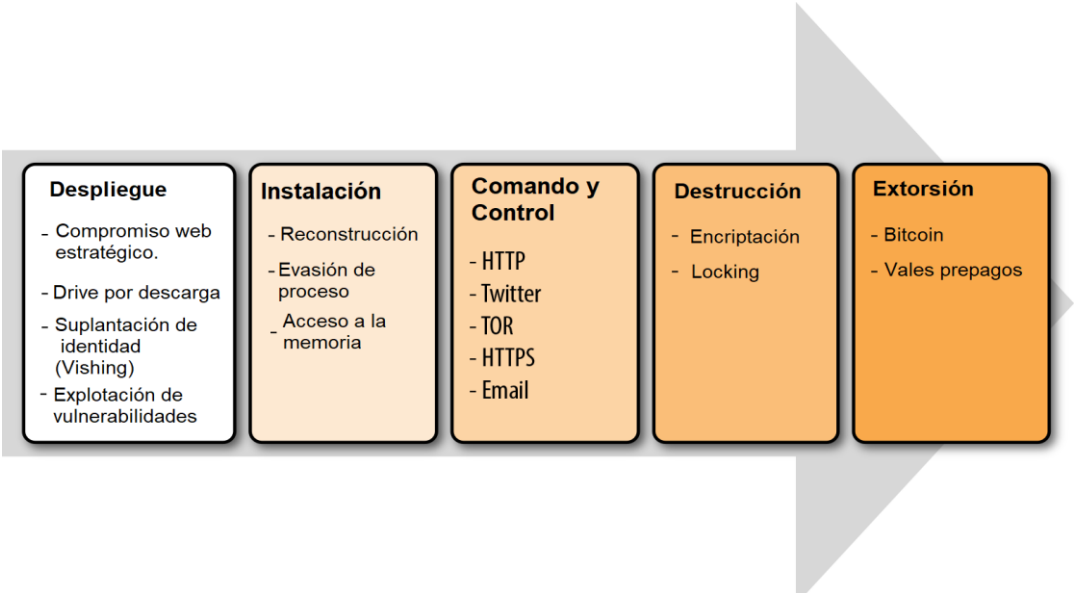

Figura 15: Anatomía de un ataque de Ransomware.

Fuente: (Liska & Gallo, 2016, pág. 13)

1.3.11.2. Phishing.

Es un método utilizado en la actualidad utilizado por los ciberdelincuentes que emplean un grupo de técnicas, el cual estafan y obtienen información confidencial de los usuarios, obteniendo así las contraseñas de sus cuentas bancarias, se hacen pasar por trabajadores de empresas de entidades bancarias, puestas estas personas inescrupulosas aplican la ingeniería social,

el cual realizan envió de correos electrónicos, publicación en redes sociales o mediante mensajes de textos, engañándolos con premios para que actualicen sus datos y coloquen su contraseña y de esa manera roben su información y realicen robos sistemáticos. (Rivero, 2018).

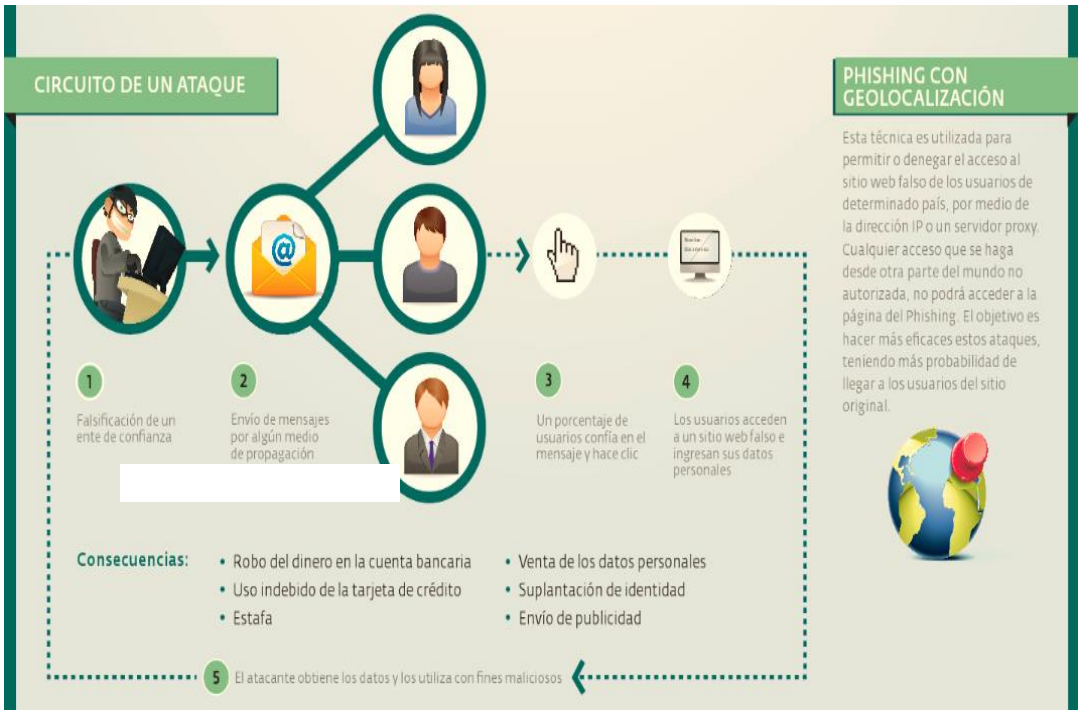

Figura 16: Ataque Phishing.

Fuente: (Rivero, 2018)

1.3.11.3. Backdoor:

Es un software malicioso que faculta a personas inescrupulosas acceder a los equipos de cómputo de forma remota, espiando al usuario o administrando sus archivos, estos se instalan en componentes de software y hardware. (TrendLabs, 2015).

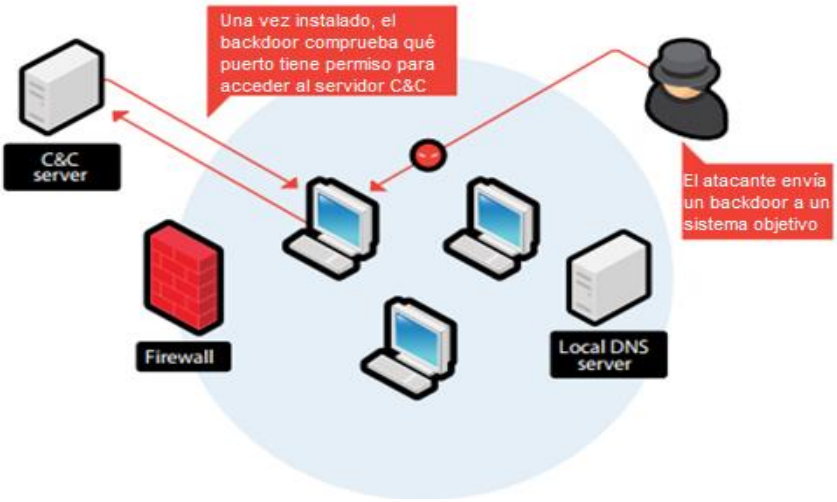

Figura 17: Ataque dirigido típico en una red corporativa.

Fuente: (TrendLabs, 2015)

1.3.11.4. Exploit:

Un exploit es un código malicioso que aprovecha la vulnerabilidad de software o una falla de la seguridad, estos permiten que un intruso acceda de forma remota a una red y obteniendo privilegios elevados, o se adentre en la red. Un exploit se puede utilizar como parte de un ataque de varios componentes. (Trend Micro, 2018).

Figura 18: Ataque Exploit.

Fuente: (Trend Micro, 2018)

1.3.11.5. DDOS:

Un ataque de Denegación de Servicio Distribuido (DDOS) se compone de varios sistemas infectados que se dirigen a un sistema específico con el objetivo de hacerlo inoperable. El DDOS operan mediante el envío de grandes volúmenes de tráfico a tu sitio web para sobrecargar tu hosting. Los sistemas infectados combinados se denominan botnets y atacan a la víctima inundando su sistema con una gran cantidad de tráfico, lo que provoca que se bloquee. (KeyCDN, 2018).

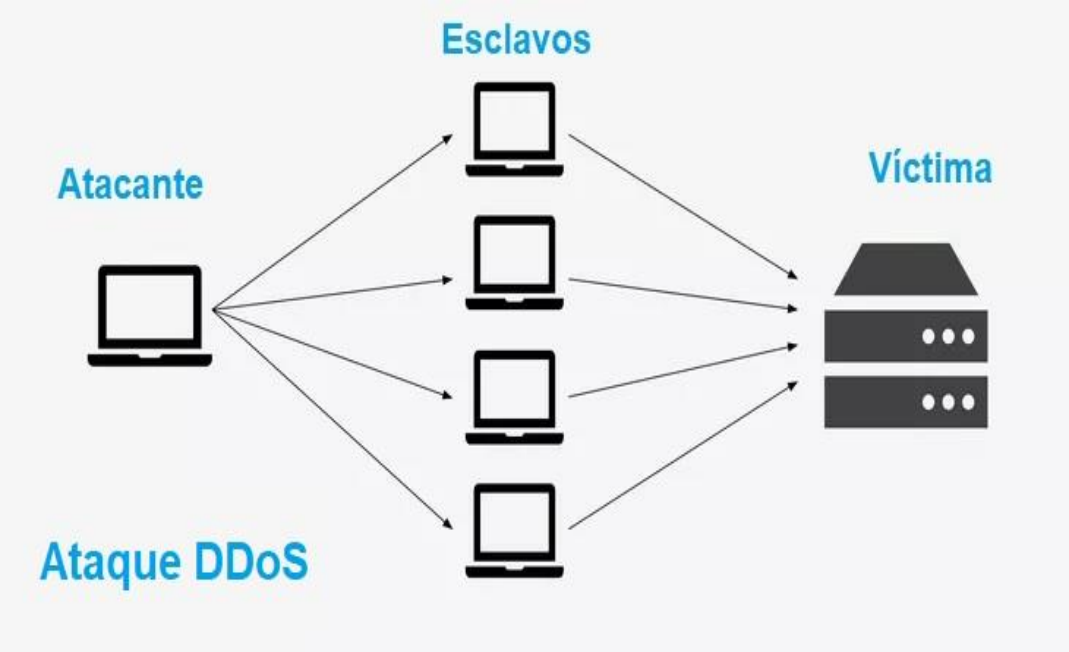

Figura 19: Ataque DDOS.
Fuente: (KeyCDN, 2018).

1.3.11.6. Smurf

El Smurf perteneciente al Ataque de Denegación de Servicio Distribuido (DDoS) en la capa de red, llamado así por el malware DDoS.Smurf que permite su ejecución. Los ataques Smurf son algo similares a las inundaciones de ping, ya que ambos se llevan a cabo enviando una serie de paquetes de solicitud ICMP. (Imperva, 2017)**.**

Figura 20: Ataque Smurf.

Fuente: (López, 2016)

1.3.11.7. Http- Flood

Según (Imperva, 2017) lo define como un tipo de ataque volumétrico de Denegación de Servicio Distribuido (DDoS) en el que el atacante explota solicitudes GET o POST aparentemente legítimas para atacar un servidor web o una aplicación, son ataques volumétricos, que con frecuencia usan una botnet “ejército de zombis”: un grupo de computadoras conectadas a Internet, cada una de las cuales ha sido maliciosa, generalmente con la ayuda de malware como los caballos de Troja.

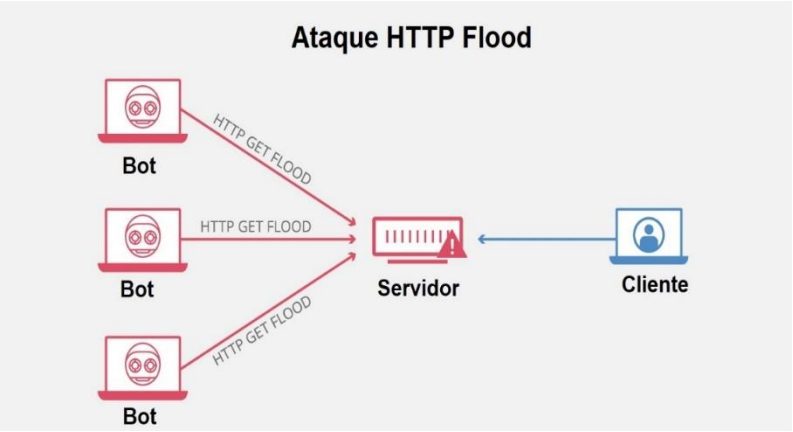

Figura 21: Ataque Http

Fuente: (Verisign, 2017)

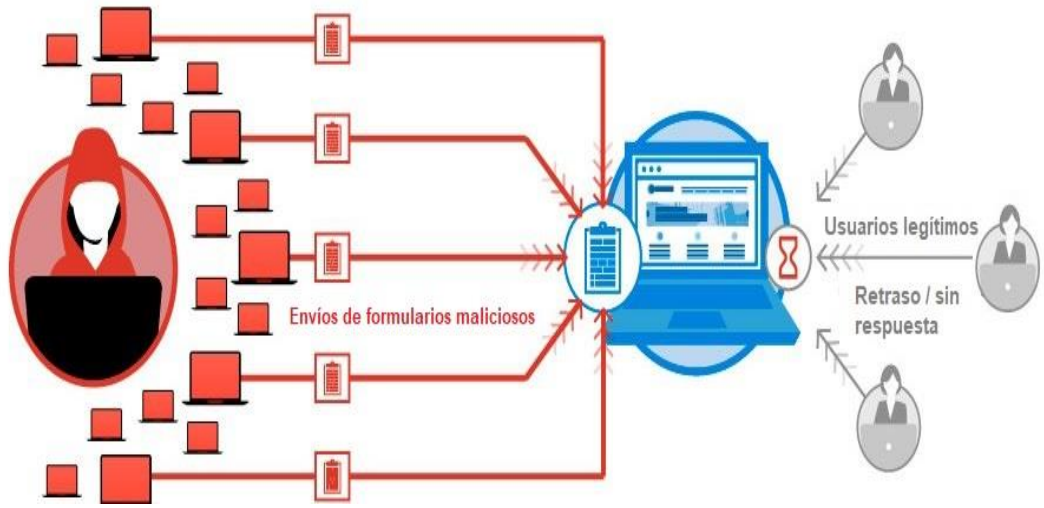

Figura 22: Http Post Attack.

Fuente: (Verisign, 2017)

1.3.11.8. UDP- Flood

Según (Imperva, 2017) define que este tipo de ataque envía una gran cantidad de paquetes UDP con direcciones de origen falsificadas a puertos aleatorios en un host seleccionado. El host verifica las aplicaciones asociadas con estos datagramas y, al no encontrar ninguna, responde con un paquete "Destino inalcanzable". El atacante envía más y más paquetes hasta que el host se siente abrumado y ya no puede responder a usuarios legítimos.

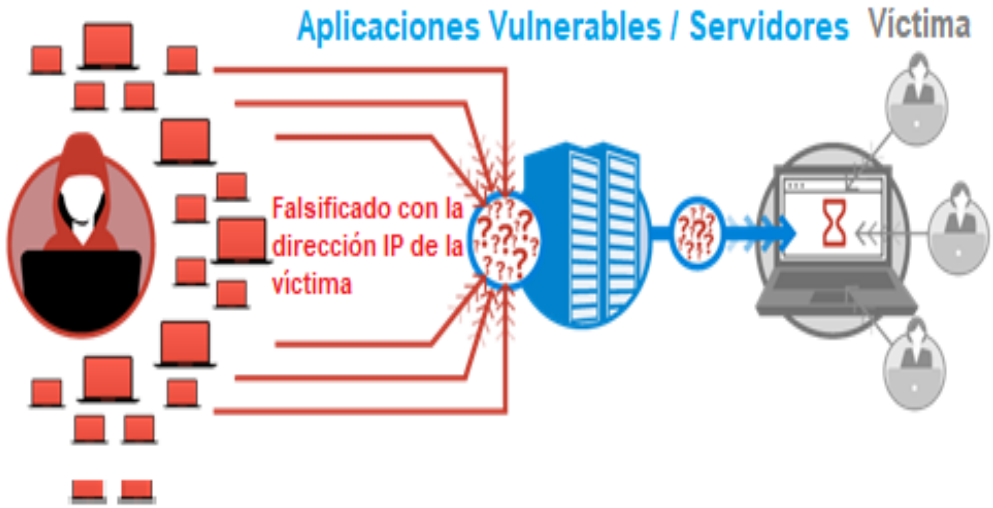

Figura 23: Ataques de Inundación UDP.

Fuente: (Verisign, 2017)

1.3.11.9. SIDDOS:

Este tipo de ataque es realizado en la capa de aplicación DDOS, se realiza cuando una aplicación es vulnerable a la inyección de SQL, pudiendo inyectar con éxito las consultas de SQL que se realizan en la base de datos consumiendo la potencia del servidor web, los recursos de la BD y del servidor de agotamiento, logrando que no acepte más conexiones al mismo tiempo dejando el servicio inaccesible para los clientes, modificando su información y de esta manera puede tomar la información del cliente. *(Kumar & Babu, 2018)*.

Figura 24: DDOS mediante Inyección SQL (SiDDOS).
Fuente: (Zenodermus, 2014)

1.3.12. Redes Neuronales Artificiales (RNA).

Una red neuronal artificial o denominadas redes computacionales intentan simular, el proceso de decisión en redes de células nerviosas (neuronas). Está simulación es realizada (neurona por neurona y elemento por elemento), permitiendo realizar el uso de operaciones computacionales para resolver problemas complejos, matemáticamente mal definidos, y problemas no lineales. (Graupe, 2013, pág. 1).

Una red neuronal artificial consiste en una colección de elementos de procesamiento que están altamente interconectados, el cual transforman un conjunto de entradas y luego lo procesan generando salidas deseadas que ayudan a resolver problemas de tareas complejas (Cannady J. , 1998, pág. 3)

Elementos Básicos que componen una Red Neuronal

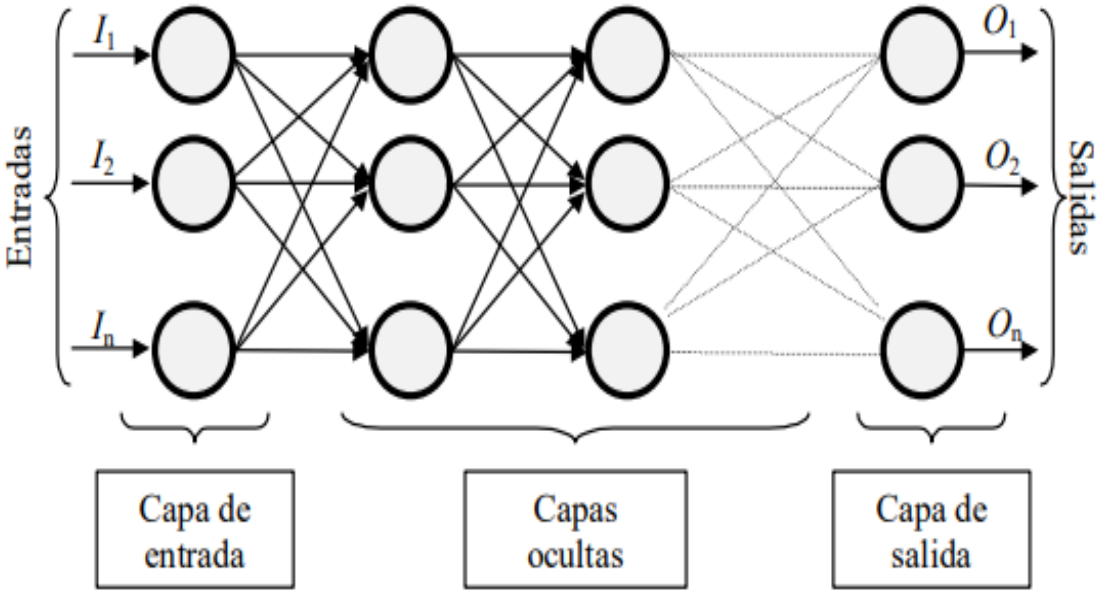

Figura 25: Ejemplo de una Red Neuronal totalmente Conectada.

Fuente: (Matich, 2001, pág. 12).

Los elementos básicos que conforman una Red Neuronal Perceptrón Multicapa son 3: la primera es la capa de entrada que recaudan la información, pasándolo después por la capa oculta, el cual puede estar conformada por una o varias capas, y luego obteniendo las salidas deseadas a través la capa de salida. (Matich, 2001, pág. 12).

1.3.13. Arquitecturas de Redes Neuronales Artificiales

Según (Nunes, Hernane, Andrade, Bartocci, & Reis, 2017) clasifican a la Arquitectura de la Red Neuronal en tres partes, denominadas capas, y son las siguientes:

a) Capa de entrada: Esta capa es la encargada de obtener y recaudar la información (datos), señales, características o medidas del ambiente externo. Estas entradas (muestras o patrones) normalmente se normalizan dentro de los valores límite producidos por las funciones de activación.

b) Capas ocultas, intermedias o invisibles Son responsables de extraer los patrones asociados con el proceso o sistema que se analiza. Estas capas realizan la mayor parte del procesamiento interno desde una red.

c) Capa de salida: Esta capa está compuesta de neuronas, y por lo tanto es responsable de producir y presentar las salidas de red finales, que resultan del procesamiento realizado por las neuronas en las capas anteriores.

1. 3.14. Esquema del Perceptrón.

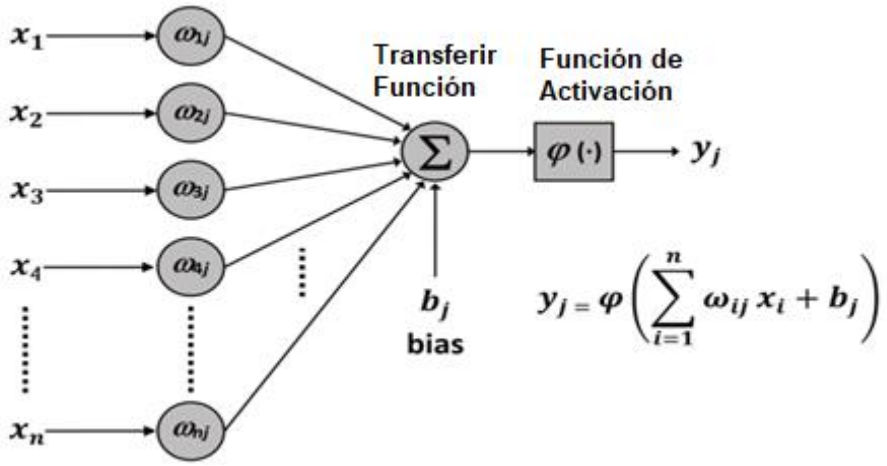

Figura 26: Esquema del Perceptrón.
Fuente: (Nunes, Hernane, Andrade, Bartocci, & Reis, 2017)

Donde:

a) **X1..Xn: Representan a las patrones de entrada de la red neuronal.**

b) **W1j…Wnj:** Compuesto por los Pesos otorgados de cada una de las entradas, que se acoplan de forma automático, conforme a la red neuronal va aprendiendo.

c) **Bj:** es el umbral de activación de la neurona.

d) **Φ:** Representada por la función de activación, que agrupa a cada valor añadido un único valor de salida

e) **yj:** Es la salida de la neurona

1. Asignar la salida deseada del perceptrón {-1,1} a cada punto del conjunto de entrenamiento.
2. Aumente cada conjunto de características para cada punto de datos con uno para formar $x=1,x_1,x_2,.....x_n$.
3. Aumente cada conjunto de características para cada punto de datos con uno para formar $(\text{eta } \epsilon(0,1))$ por lo general eta = 0.1.
4. Inicializar pesos de forma aleatoria $W= \{w_0,w_1, w_2,.....w_n\}$.
5. Ejecutar el algoritmo de ajuste de peso de perceptrón

```
error_flag=1
while error_flag=1
    error = 0
    for ii = 1 al número total de puntos de datos en el algoritmo de
entrenamiento
     tomar un vector de características (x) y su salida deseada {-1,1}
     salida= signum (wx^T)
      if salida no es igual a lo deseado
      w = w – eta x
error = error + salida - deseado
      end
   end
if error =0
  error_flag=0
   end
end
```

Figura 27: Algoritmo de aprendizaje del Perceptrón.
Fuente*: (Kevin, Priddy, Paull, & Keller, 2005, pág. 7)*

1.3.15. Arquitecturas de Redes Neuronales Artificiales.

Dentro de las arquitecturas de las redes Neuronales tenemos las siguientes:

1.3.15.1. Arquitectura Feedforward de una Sola Capa.

La arquitectura Feedforward tiene solo una capa de entrada y una única capa neuronal, que también es la capa de salida. Su información siempre fluye en una sola dirección siendo, unidireccional, que proviene desde la capa de entrada a la

capa de salida. Se emplean en problemas de clasificación de patrones y filtrado. Los pertenecientes a esta arquitectura tenemos al Perceptron y el ADALINE. (Nunes, Hernane, Andrade, Bartocci, & Reis, 2017).

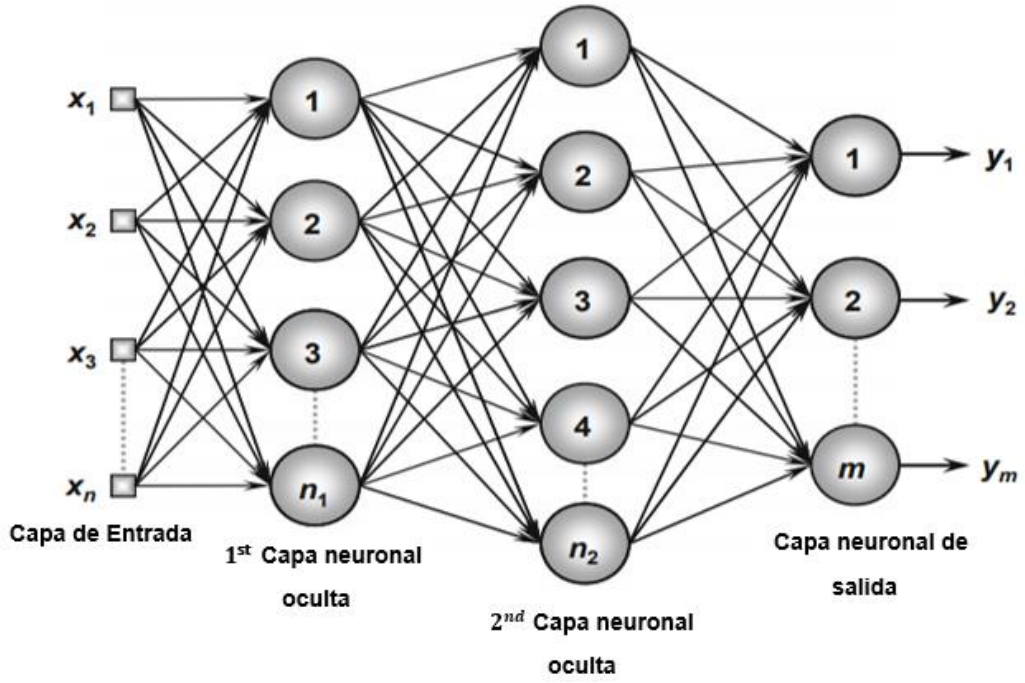

Figura 28: Ejemplo de una red de avance de una sola capa.
Fuente: (Nunes, Hernane, Andrade, Bartocci, & Reis, 2017, pág. 23)

1.3.15.2. Arquitecturas de avance de múltiples capas.

La Arquitectura de avance con múltiples capas están compuestas por 1 capa de entrada con n señales de muestra, dos capas ocultas de neuronas que consisten en n1 y n2 neuronas respectivamente, y, finalmente, una capa neural de salida, siendo empleados en diversos problemas, como los relacionados con la aproximación de funciones, la clasificación de patrones, la identificación de sistemas, el control de procesos, la optimización, la robótica. Dentro de esta arquitectura tenemos a él Perceptrón multicapa (MLP) y la Función de base radial (RBF) (Nunes, Hernane, Andrade, Bartocci, & Reis, 2017) .

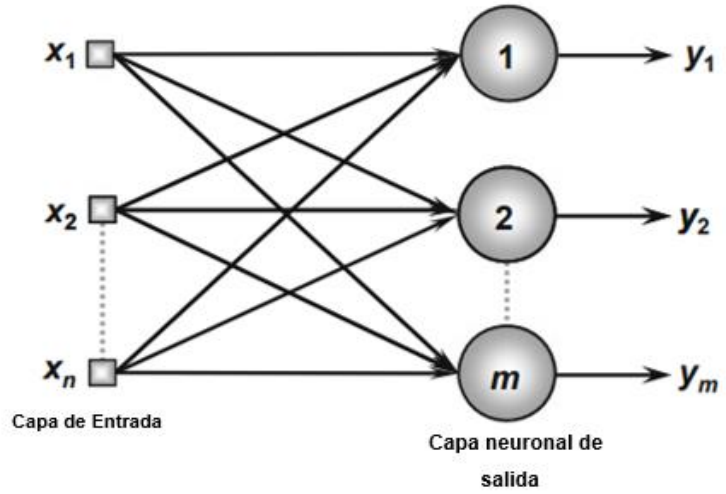

Figura 29: Ejemplo de una red feedforward con múltiples.

Fuente: (Nunes, Hernane, Andrade, Bartocci, & Reis, 2017, pág. 176)

1.3.15.3. Arquitectura recurrente o de retroalimentación.

En esta arquitectura recurrente, las salidas de las neuronas son empleadas como entradas de retroalimentación para otras neuronas. Aquí tenemos La función de retroalimentación, que pueden emplearse en sistemas con variantes de tiempo, como la pronosticación de series de tiempo, la identificación y optimización del sistema, el control de procesos, etc. Dentro de las redes de retroalimentación están el Hopfield y el Perceptron con retroalimentación entre neuronas de distintas capas (Nunes, Hernane, Andrade, Bartocci, & Reis, 2017).

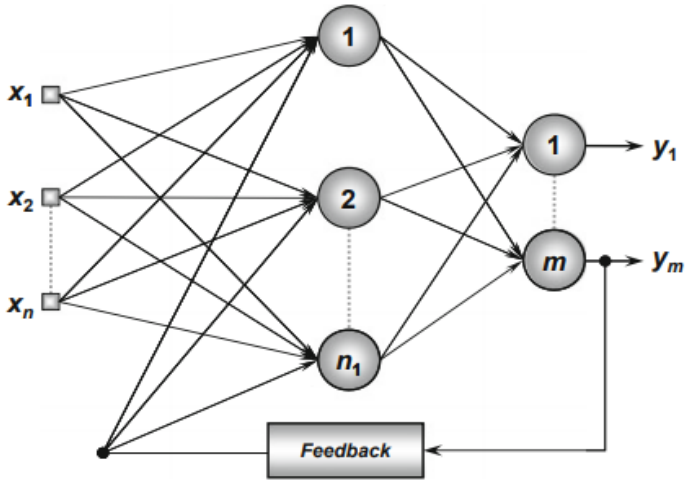

Figura 30: Ejemplo de una red recurrente.

Fuente: (Nunes, Hernane, Andrade, Bartocci, & Reis, 2017, pág. 24)

1.3.15.4. Perceptrón Multicapa (Multilayer Perceptrón)

La red neuronal multicapa es una extensión de la red neuronal monocapa, está compuesta por una capa de neuronas de entrada, una o varias capas intermedias y una capa de neuronas de salida. (Calvo, 2017).

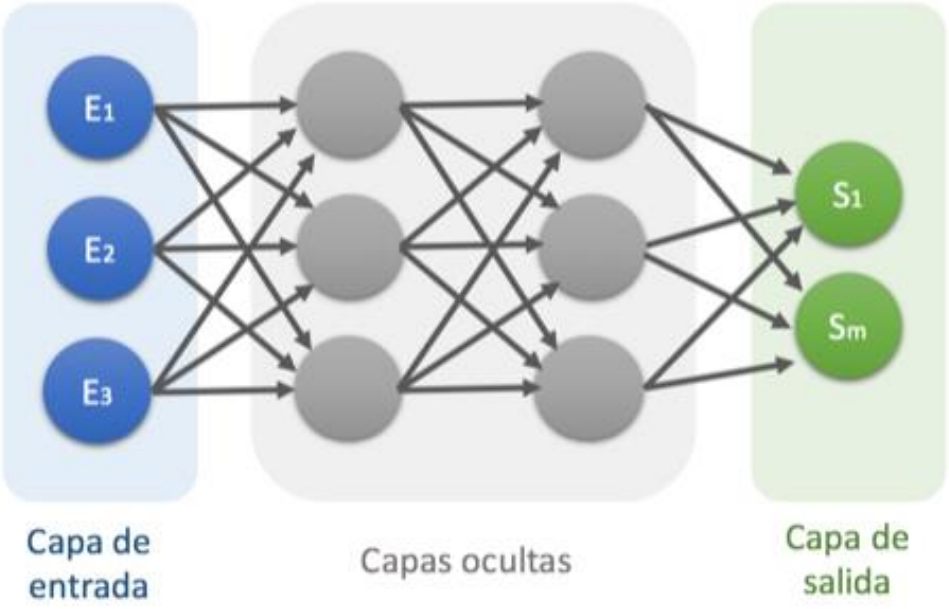

Figura 31: Diagrama del Perceptrón Multicapa.
Fuente: (Calvo, 2017)

Función Sigmoidal

La función sigmoide modifica los valores insertados en un rango de (0,1), donde los valores altos tienen de manera que se acerca a una curva 1 y los valores muy bajos desdoblan de manera asintótica a 0. (Caraballo, s.f.).

Notación Matemática

$$y = \frac{1}{1 + e^{-x}}$$

Donde: *Y* es la función sigmoide que transforma los valores introducidos a una escala (0,1).

Error cuadrático medio:

El error cuadrático medio se define utilizando las diferencias entre los elementos del vector de salida y el vector objetivo. (Kubat, 2017, pág. 107)

$$MSE = \frac{1}{m}\sum_{i=1}^{m}(t_i - y_i)^2$$

Donde: *t* es un vector de m predicciones y Y_i es el vector de los verdaderos valores

1.3.16. Máquinas de Soporte Vectorial (SVM).

Cristianini & Shawe-Taylor, (2000), el objetivo de la clasificación de vectores de soporte es emplear un método computacional eficiente de aprender hiperplanos 'buenos' de separación en un espacio de características de alta dimensión, donde por 'hiperplanos' buenos 'entenderemos los que optimizan los límites de generalización descritos. La visión popular de SVM es que encuentran un hiperplano “óptimo” como la solución al problema de aprendizaje. La formulación más simple de SVM es la lineal (Evgeniou & Pontil, 2001).

Tabla 15.

Notación Matemática Support Vector Machine

Notación Matemática Support Vector Machine		
Kernel	**Parameter**	**Fórmula**
Linear		$K(x, y) = x.y$
Poly	d,degree	$K(x, y) = (x.y + 1)^d$
Radial Basis Function (RBF)	sigma	$K(x, y) = e^{\frac{-\|x-y\|^2}{2\sigma^2}}$
Gausian	sigma	$K(x, y) = \frac{1}{2\pi^{N/2}.\sqrt{\sigma}}$

Nota: Elaboración Propia. Tomado de (Ray, 2017).

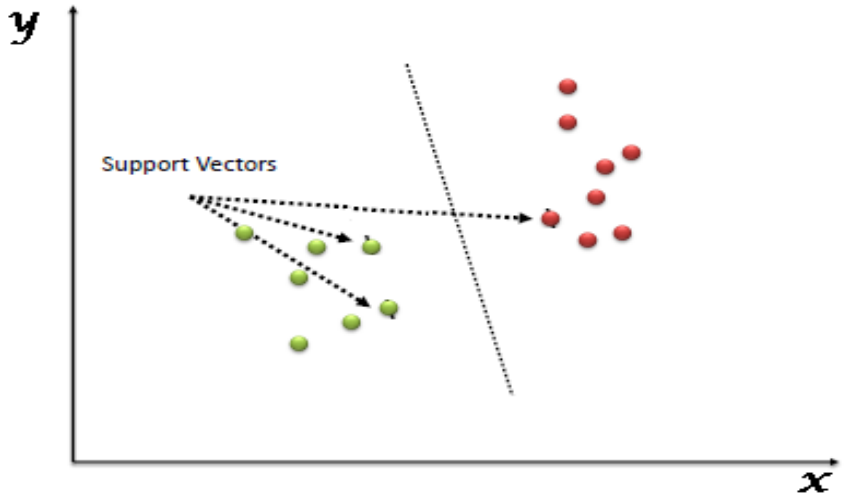

Figura 32: Diagrama del Perceptrón Multicapa.

Fuente: (Ray, 2017)

1.3.17. Random Forest

Los bosques aleatorios son muy populares y son un ganador para el científico de datos, ya que son divinos para un paquete con muchos problemas de clasificación. Tienen un funcionamiento más rápido para la mayoría de los problemas de clasificación. No necesita preocuparse por ajustar un conjunto de parámetros. Por el contrario, debe ocuparse de muchos parámetros y ajustes al manejar sus datos. (Kaysar, 2016).

1.3.18. Naive Bayes

Naive Bayes es un algoritmo intuitivo que utiliza la regla de Bayes para calcular las probabilidades de cada atributo que pertenece a cada clase para hacer una predicción Reduciendo el cálculo de probabilidades al suponer que los atributos son independientes, dada la etiqueta de todos los demás atributos. (Belouch, El Hadaja, & Idhammadb, 2018).

1.3.19. AdaBoost

Es un algoritmo eficiente del campo de aprendizaje conjunto. Se utiliza para aumentar el rendimiento de clasificación de un aprendiz débil. Lo realiza combinando una colección de funciones de clasificación débiles para formar un clasificador más fuerte. AdaBoost combina iterativamente los clasificadores débiles teniendo en cuenta una distribución de peso en las muestras de

entrenamiento, de manera que se atribuye más peso a las muestras mal clasificadas por las iteraciones anteriores. (Dezhen & Kai, 208).

1.3.20. C 4.5.

El algoritmo C4.5 es empleado en la minería de datos como un algoritmo de árbol de decisión, el cual se utiliza para generar arboles de decisión, basándose en una determinada muestra de datos, son empleados y utilizados para la clasificar conjunto de datos, siendo esto una de las razones por que es denominado un clasificador estadístico. (Santa, Veloza, & Montoya, 2013).

1.3.21. Herramienta para evaluar un modelo de clasificación

1.3.21.1 Matriz de confusión

La matriz de confusión es una de las formas más populares de evaluar un modelo de clasificación. Se puede crear una matriz de confusión para una clasificación binaria, así como un modelo de clasificación de múltiples clases. Esta comparación se repite para todo el conjunto de datos y los resultados de esta comparación se compilan en una matriz o formato tabular que mantiene para mantener un registro de las clasificaciones correctas y las clasificaciones erróneas, las métricas que se emplean en la mutilases son las mimas que se utilizan en la clasificación binaria. (Sarkar, Bali, & Sharma, 2018).

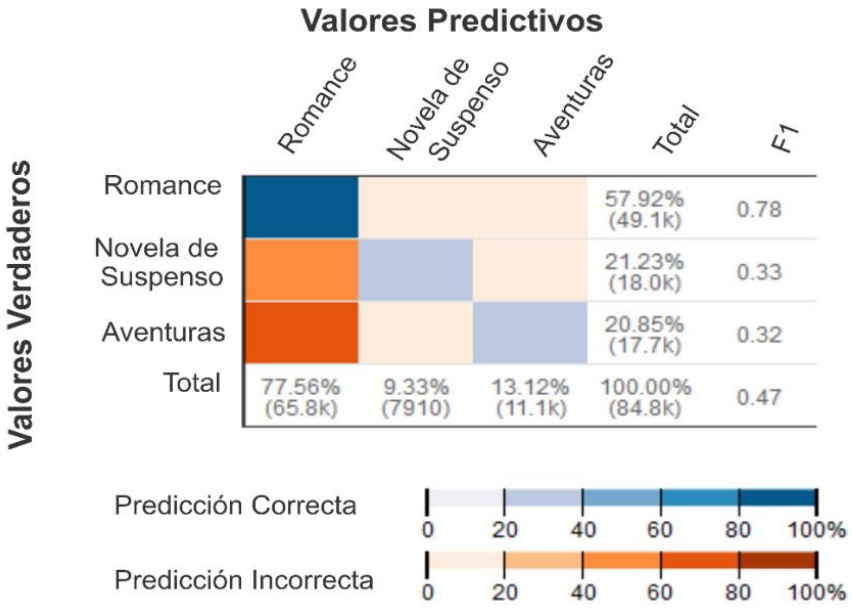

Figura 33: Matriz de confusión para un modelo de Clasificación.

Fuente: (Amazon Web Service, 2018)

1.3.22. Técnicas de Validación y Evaluación

1.3.22.1. Validación cruzada:

Es un método empleado que estima la precisión en términos de error de un modelo de aprendizaje automático dividiendo los datos en K subconjuntos mutuamente excluyentes o pliegues de x aproximadamente el mismo tamaño. Técnicamente ayuda a estimar cómo se desempeñará con precisión un modelo predictivo en la práctica cuando lo implemente como una aplicación. (Kaysar, 2016).

1.3.23. Herramientas utilizadas

1.3.23.1. Scikit-learn

Es un módulo de Pyhton para hacer machine learning este Implementa una amplia gama de algoritmos de aprendizaje automático que cubren las áreas principales del aprendizaje automático, como clasificación, agrupamiento, regresión, etc. (Sarkar, Bali, & Sharma, 2018).

1.3.23.2. Pandas

Pandas es una importante biblioteca de Python que permite manipulación de datos, disputas y análisis. Funciona como un conjunto de herramientas intuitivas y fáciles de usar para realizar operaciones en cualquier tipo de datos. (Sarkar, Bali, & Sharma, 2018).

1.3.23.3. Numpy

Es la biblioteca del lenguaje de Python, el cual aporta a una estructura de datos simple, la matriz multidimensional, funciones matemáticas de nivel y la recopilación de rutinas son importantes para procesar matrices, los investigadores lo emplean en la computación científica de datos. (Muller & Guido, 2015).

1.3.24. Técnicas de Normalización

Según los autores (Kumar, Verma, & Thoke, 2015) dentro de las técnicas de Normalización tenemos las siguientes:

a) Normalización Z – Score

Esta técnica es la más frecuente para normalizar las características a la media y la varianza unitaria es la normalización de la puntuación. (Kumar, Verma, & Thoke, 2015). Es una técnica lineal en la que, inicialmente, la media (x) y la desviación estándar (σ) de los valores de las características específicas se calculan utilizando con la siguiente fórmula:

$$\hat{x}_i = \frac{x_i - \bar{x}}{\sigma}$$

Donde: $\hat{x}_i$ es la puntuación y $\bar{x}$ es el valor de la media de nuestra población.

b) Normalización Mín. – Máx.:

Esta técnica permite realizar una transformación lineal en los datos originales. Min Max es una técnica que ayuda a normalizar los datos. Se escalan entre 0 y 1 y es dad por la siguiente fórmula.

$$\hat{x}_i = \frac{x_{i - min(X_i)}}{max(x_i) - min(x_i)} (max\ x_{new} - min\ x_{new}) + min\ x_{new}$$

Donde: $min(x_i)$ y $max(x_i)$, son los valores máximos de un atributo.

1.4. Formulación del Problema

¿Qué tecnica de aprendizaje automático es la mas eficiente en la detección de intrusos en una red informática?

1.5. Justificación e importancia de la investigación

La presente investigación está a alienada a la línea de investigación Infraestructura, Tecnología y Medio Ambiente de la escuela de Ingeniería de Sistemas de la Universidad señor de Sipán, pues esta investigación tiene un enfoque sistemático de suma importancia en la ciencias de la computación, ya que con el pasar del tiempo pues el aprendizaje automático se ha convertido en uno de los elementos más importantes de la inteligencia artificial actualmente, pues hoy en día esto permite automatizar tareas específicas, gracias al Aprendizaje Automático. En el campo de redes de computadoras, existen tendencias tales como computación en la nube o el Internet de las cosas que demandan nuevas medidas de seguridad. Muchas compañías del mundo tienen su información subida en la nube (Cloud

Computing), por eso es pertinente darle más importancia a la Seguridad a la Información de las empresas y proveer servicios de seguridad para así poder salvaguardar la información de los usuarios, evitando que su información pueda ser alterada o vulnerada por personas inescrupulosas. Pues con la Implementación de estas dos técnicas de Aprendizaje Automático, se evaluará la eficiencia en la detección de intrusos y aportará en la Seguridad de Información. Este trabajo podrá servir como base para trabajos futuras investigaciones científicas relacionadas con la Inteligencia Artificial y Seguridad en la Redes de Computadoras.

1.6. Hipótesis

La técnica Perceptrón Multicapa es la más eficiente en la detección de intrusos de una Red Informática.

1.7. Objetivos de la Investigación.

1.7.1. Objetivo General:

Evaluar las técnicas de Máquinas de Soporte Vectorial y Perceptrón Multicapa para determinar su eficiencia en la detección de Intrusos de una Red Informática.

1.7.2. Objetivos específicos

a) Preparar la base de datos para entrenamiento y pruebas.
b) Seleccionar las técnicas de mayor eficiencia en la Detección de Intrusos.
c) Implementar las Técnicas de Máquinas de Soporte Vectorial y Perceptrón Multicapa en la detección de Intrusos.
d) Evaluar las Técnicas de Perceptrón Multicapa y Máquinas de Soporte Vectorial.

II. MATERIAL Y MÉTODO

2.1. Tipo y Diseño de Investigación

2.1.1. Tipo de la Investigación

La presente investigación se encuentra fundamentado en la metodología cuantitativa, debido a que los indicadores se encuentran representado por cantidades numéricas como las métricas de desempeño, exactitud, precisión, sensibilidad y especificidad.

El diseño cuasi-experimental se emplea premeditadamente, teniendo una variable independiente para estudiar y examinar su causa-efecto y la relación con una o más variables dependientes. (Hernández, 2012).

2.1.2. Diseño de la Investigación

Según (Marroquín, 2012) define a la investigación aplicada como “La utilización de los conocimientos en la práctica, para aplicarlos, en la mayoría de los casos, en provecho de la sociedad” (p. 3)., ya que la investigación abarca el uso de estos conocimientos tecnológicos para aplicarlos en la práctica, y disponerlos en beneficio de las organizaciones.

2.2. Población y Muestra

2.2.1. Población

La población de la presente investigación está conformada por un top de 6 técnicas de aprendizaje automático, que han sido evaluados a través de las métricas de desempeño, para determinar la eficiencia de intrusos en una red informática.

2.2.2. Muestra

Por conveniencia, la presente investigación trabajará con dos técnicas de aprendizaje automático, Máquinas de Soporte Vectorial y Perceptrón Multicapa, ya que se realizó un estudio del top de las técnicas de aprendizaje automático.

2.3. Variables y Operacionalización

2.3.1. Variable Independiente

Técnicas de Aprendizaje Automático.

2.3.2. Variable Dependiente

Detección de Intrusos.

2.3.3. Operacionalización

Tabla 16:

Operacionalización de Variable Dependiente.

VARIABLE DEPENDIENTE	INDICADORES	FÓRMULA	TECNICAS E INSTRUMENTOS
Detección de Intrusos	*Exactitud (E)*	$= \frac{TP + TN}{TP + TN + FP + FN}$	*Técnica= Observación Ficha de Observación*
	Precisión (P)	$= \frac{TP}{TP + FP}$	
	Sensibilidad (S)	$= \frac{TP}{TP+FN}$	*Instrumento = Matriz de Confusión*
	Especificidad (Es)	$= \frac{TN}{TN+FP}$	

Nota: Elaboración Propia. Tomado de (Choudhury & Bhowal, 2015, pág. 92).

Tabla 17:

Operacionalización de Variable Independiente.

VARIABLE INDEPENDIENTE	INDICADOR	FÓRMULA	TÉCNICAS E INSTRUMENTOS
Técnicas de Aprendizaje Automático	*Tiempo de Entrenamiento*	$= TF - TI$	*Ficha de Obervación*

Nota: Elaboración Propia. Tomado de (Choudhury & Bhowal, 2015, pág. 5).

2.4. Técnicas e instrumentos de recolección de dato, validez y confiabilidad

2.4.1. Abordaje metodológico

La presente investigación empleo el análisis documental para comprender los conocimientos adecuados, la realización de esta investigación, este estudio se basó en la obtención de libros, artículos científicos, noticias, reportes de antivirus, visitas a páginas web y videos.

2.4.2. Técnicas de recolección de datos

El presente trabajo de investigación empleó técnicas de recolección de datos, que se utilizaron en la presente investigación son:

2.4.2.1 La Observación.

Considerada como una herramienta de la investigación de campo, en la que se realiza el registro visual y se hace una recolección de datos en una situación real, para el proceso de investigación, además esta lo realiza el investigador registrando datos e información, realizando apuntes en su ficha, para que luego compare y examine resultados con la hipótesis; los cuales se pueden observar y documentar tanto el investigador, los expertos y miembros del jurado.

2.4.2.2 Análisis Documental.

Este instrumento permite investigar, observar y analizar la información, siendo una de las técnicas de investigación más empleadas por los investigadores, permitiendo recoger los datos extraídos de bases de datos científicas, como informes, libros, revistas, papers, etc. (Tamayo & Silva, 2017).

2.5. Procedimientos de análisis de datos.

Para realizar el procedimiento de la recolección de datos se estudió y analizó diferentes conjuntos de datos, con el propósito de extraer de los conjuntos de datos, patrones de ataque más actuales en el Perú y Latinoamérica, luego de ellos los patrones de ataque serán normalizados utilizando la suite de Entrenar los algoritmos Maquinas de Soporte Vectorial y Perceptrón Multicapa.

2.5.1. Plan de análisis estadísticos de datos.

Para obtener la realización del análisis y así poder los resultados obtenidos de los algoritmos Maquinas de Soporte Vectorial y Perceptrón Multicapa, utilizando los indicadores de la Operacionalización de las variables, se utilizará la librería Scikit-Learn de Python.

2.6. Procedimiento de análisis de datos

a) Analizar los diferentes algoritmos, evaluándolos mediante las métricas de desempeño y formar un top de 6 algoritmos en una tabla de Excel.
b) Preparación del nuevo conjunto de datos dataset_attack_2018.
c) Evaluar los resultados en conjunto de los dos algoritmos seleccionados.

2.7. Criterios éticos.

a) Confidencialidad: La pertenencia de la información recolectada será protegida para que no sea publicada sin la facultad del investigador y protección de su identidad de las personas, las informaciones obtenidas en esta investigación solo pueden tener acceso a personas autorizadas a conocer dicha información.

b) Derechos del Autor: Toda cita la cual contengan conceptos, párrafos, artículos científicos en la investigación, serán debidamente referenciados al autor a quien le pertenezca con su respectivo Nombre de Autor, año en la cual fue publicado su investigación.

2.8. Criterios de Rigor Científico.

a) Validez:

Los resultados obtenidos en la implementación de los algoritmos serán correctamente supervisados, analizados y evaluados por los juicios de expertos para así poder lograr dar un resultado valioso verificando la problemática planteada.

b) Fiabilidad:

La fiabilidad en esta investigación es evaluada por el instrumento de medición que en este caso vendría hacer la matriz de confusión, el cual obtiene resultados a la hora de realizar las pruebas con los algoritmos.

III. RESULTADOS

3.1. Resultados en tablas y Figuras.

La evaluación de las técnicas se realizó utilizando uso de Anaconda que es una distribución de código libre basada en Python, utilizada en la ciencia de datos, y para el aprendizaje automático. Físicamente se implementó en una laptop con las siguientes características: Procesador Core I3: 2.0 GHZ de Sexta Generación, y una Memoria RAM DE 8 GB BUS 1666 y con un Sistema Operativo Windows 10 de 64 bits, obteniendo los siguientes resultados en la presente investigación.

La figura Nº 34, se muestra el análisis comparativo de los algoritmos Perceptrón Multicapa (MLP) y Maquinas de Soporte Vectorial (SVM), y se visualiza que Perceptrón Multicapa y SVM es más exacto en la predicción del ataque Backdoor, alcanzando el 100% en exactitud. Pero cuando se realiza la predicción, el algoritmo MLP alcanza el 83.3%, siendo más preciso en predecir el ataque Backdoor a diferencia del SVM que solo logra alcanzar el 76% términos de precisión. En la sensibilidad el perceptrón Multicapa alcanza el 0.01% de ataque correctamente identificados a diferencia de Maquinas de Soporte Vectorial, en especificidad los algoritmos MLP y SVM presentan la mayor tasa de detección de tráfico normal.

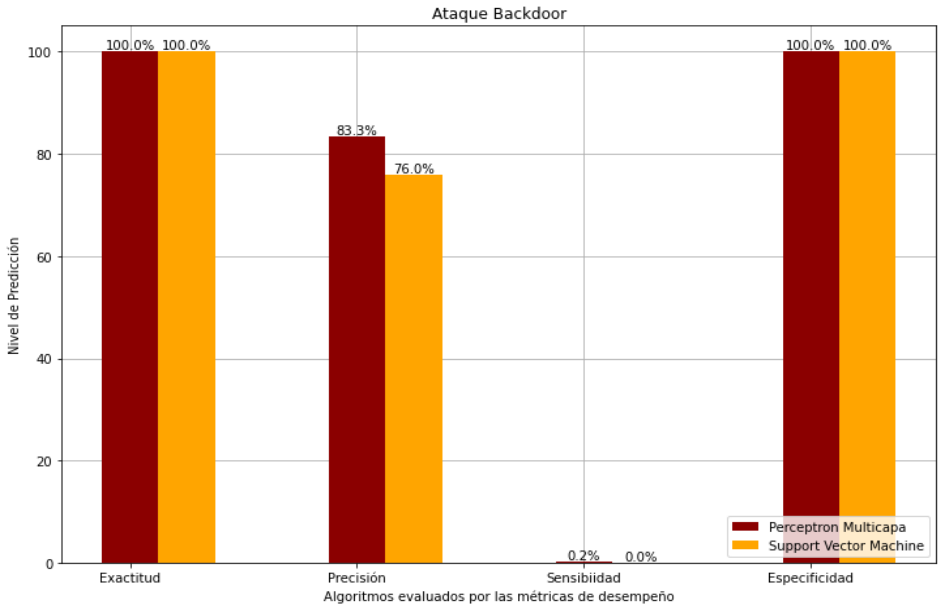

Figura 34: Comparación de Algoritmos basados en Exactitud, Precisión, Sensibilidad y Especificidad del Ataque Backdoor.

Fuente: Elaboración Propia

La figura Nº 35, se muestra el análisis comparativo de los algoritmos Perceptrón Multicapa (MLP) y Máquinas de Soporte Vectorial (SVM), y se visualiza que tanto SVM y MLP son más exacto en la predicción del ataque Exploits, alcanzando el 100% en exactitud. Pero cuando se realiza la predicción los dos algoritmos son más precisos en predecir ataques ataque Exploits, alcanzando el 95% términos de precisión. En la sensibilidad los 2 algoritmos alcanzan el 100% de ataques correctamente identificados y en la especificidad los algoritmos MLP y SVM presentan la mayor tasa de detección de tráfico normal logrando alcanzar el 100% en especificidad.

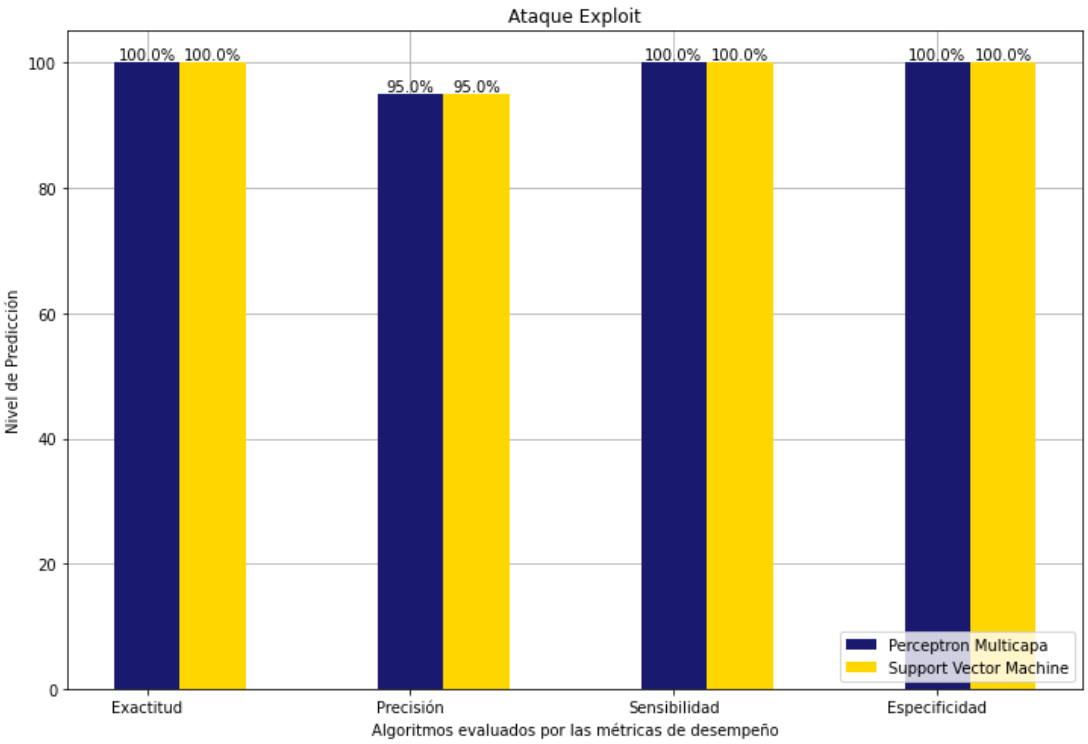

Figura 35: Comparación de Algoritmos basados en Exactitud, Precisión, Sensibilidad y Especificidad del Ataque Exploit.

Fuente: Elaboración Propia

La figura Nº 36, muestra el análisis comparativo de los algoritmos Perceptrón Multicapa (MLP) y Máquinas de Soporte Vectorial (SVM), y se visualiza que tanto SVM y MLP son más exacto en la predicción del ataque HTT- Flood, alcanzando el 100% en exactitud. Pero cuando se realiza la predicción el algoritmo MLP es más preciso en predecir ataques HTT-Flood obteniendo el 99%, a diferencia de SVM que llega al 98% en precisión. En la sensibilidad las 2 técnicas alcanzaron el 94% de ataques correctamente identificados,

mientras que en la especificidad los algoritmos MLP y SVM presentan la mayor tasa de detección de tráfico normal logrando alcanzar el 100% en especificidad.

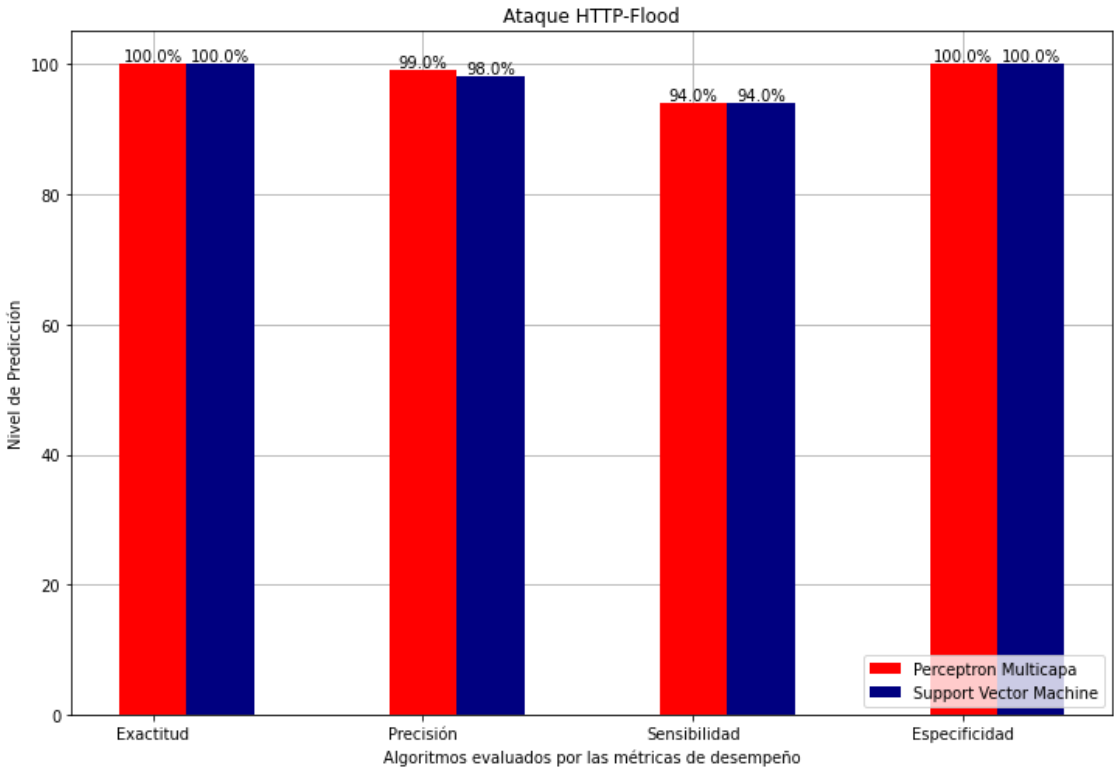

Figura 36: Comparación de Algoritmos basados en Exactitud, Precisión, Sensibilidad, Especificidad del Ataque HTTP-Flood.

Fuente: Elaboración Propia.

La figura Nº 37, muestra el análisis comparativo de los algoritmos Perceptrón Multicapa (MLP) y Máquinas de Soporte Vectorial (SVM), y se visualiza que tanto SVM y MLP son más exacto en la predicción del ataque Phishing alcanzando el 100% en exactitud. Pero cuando se realiza la predicción los dos algoritmos son preciso para predecir el ataque Phishing obteniendo el 100% en precisión. En la sensibilidad los algoritmos alcanzaron el 100% identificando ataques correctamente, en la especificidad los algoritmos MLP y SVM presentan la mayor tasa de detección de tráfico normal logrando alcanzar el 100% en especificidad.

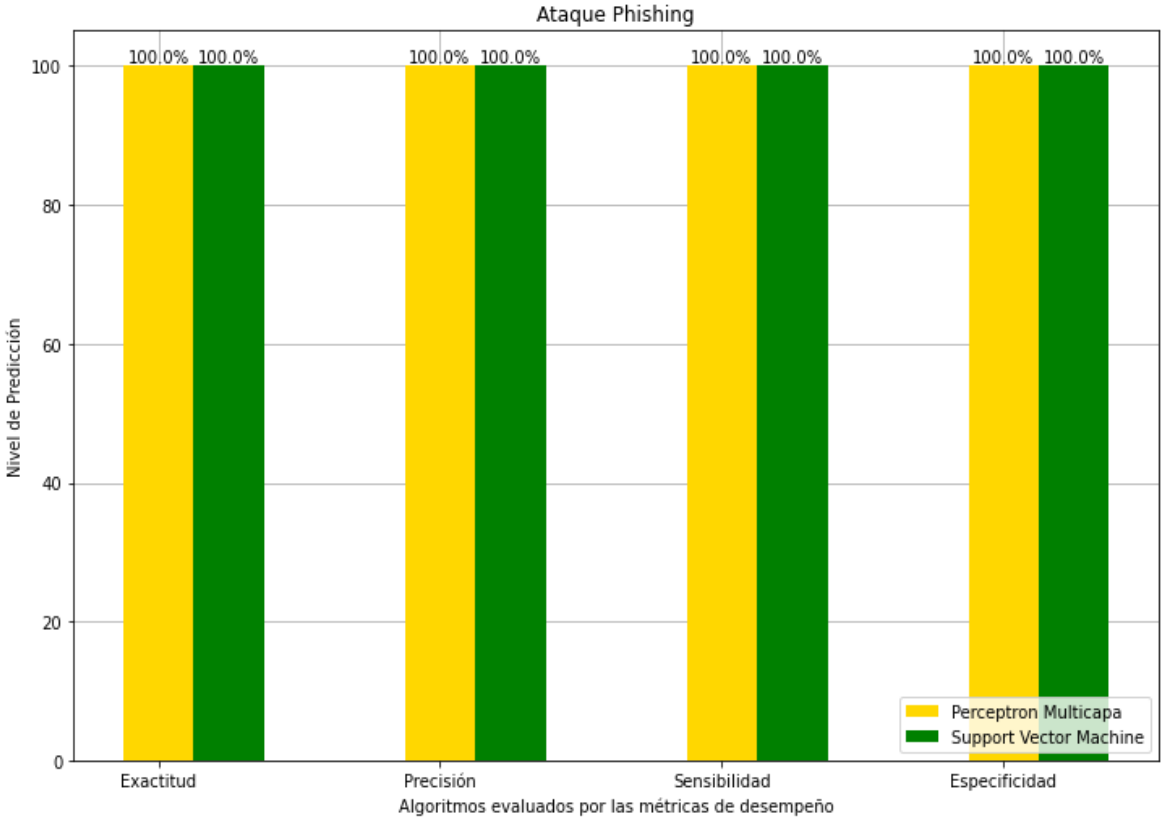

Figura 37: Comparación de Algoritmos basados en Exactitud, Precisión, Sensibilidad y Especificidad del Ataque Phishing.

Fuente: Elaboración Propia.

La figura Nº 38, muestra el análisis comparativo de los algoritmos Perceptrón Multicapa (MLP) y Máquinas de Soporte Vectorial (SVM), y se visualiza que tanto SVM y MLP son más exacto en la predicción del ataque Ransomware alcanzando el 100% en exactitud. Pero cuando se realiza la predicción los dos algoritmos son preciso para predecir el ataque Ransomware obteniendo el 100% en precisión. En la sensibilidad las 2 técnicas alcanzaron el 100% de ataques correctamente identificados, mientras que en la especificidad los algoritmos MLP y SVM presentan la mayor tasa de detección de tráfico normal logrando alcanzar el 100% en especificidad.

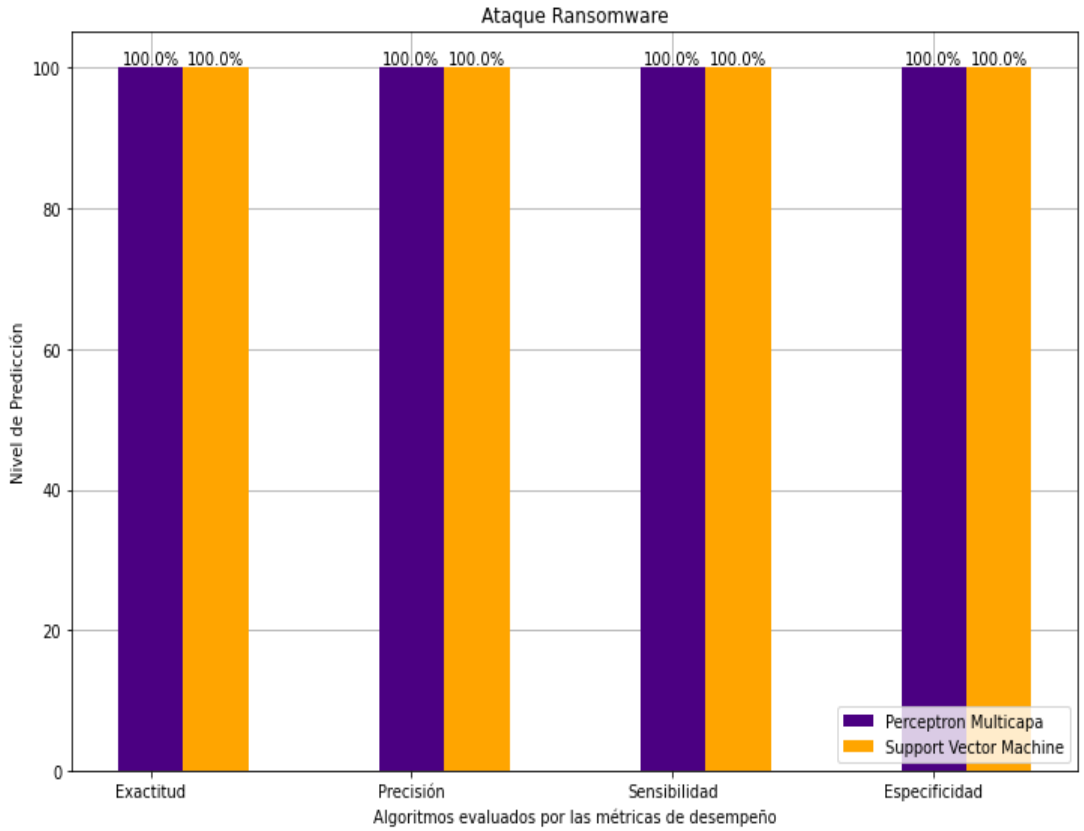

Figura 38: Comparación de Algoritmos basados en Exactitud, Precisión, Sensibilidad y Especificidad del Ataque Ransomware.

Fuente: Elaboración Propia.

La figura Nº 39, muestra el análisis comparativo de los algoritmos Perceptrón Multicapa (MLP) y Máquinas de Soporte Vectorial (SVM), y se visualiza que tanto SVM y MLP son más exacto en la predicción del ataque SIDDOS alcanzando el 100% en exactitud. Pero cuando se realiza la predicción el algoritmo MLP es más preciso para predecir el ataque SIDDOS obteniendo el 91%, a diferencia de SVM que solo alcanza el 90 % en precisión. En la sensibilidad los 2 algoritmos alcanzaron el 94% de ataques correctamente identificados y en la especificidad los algoritmos MLP y SVM presentan la mayor tasa de detección de tráfico normal logrando alcanzar el 100% en especificidad.

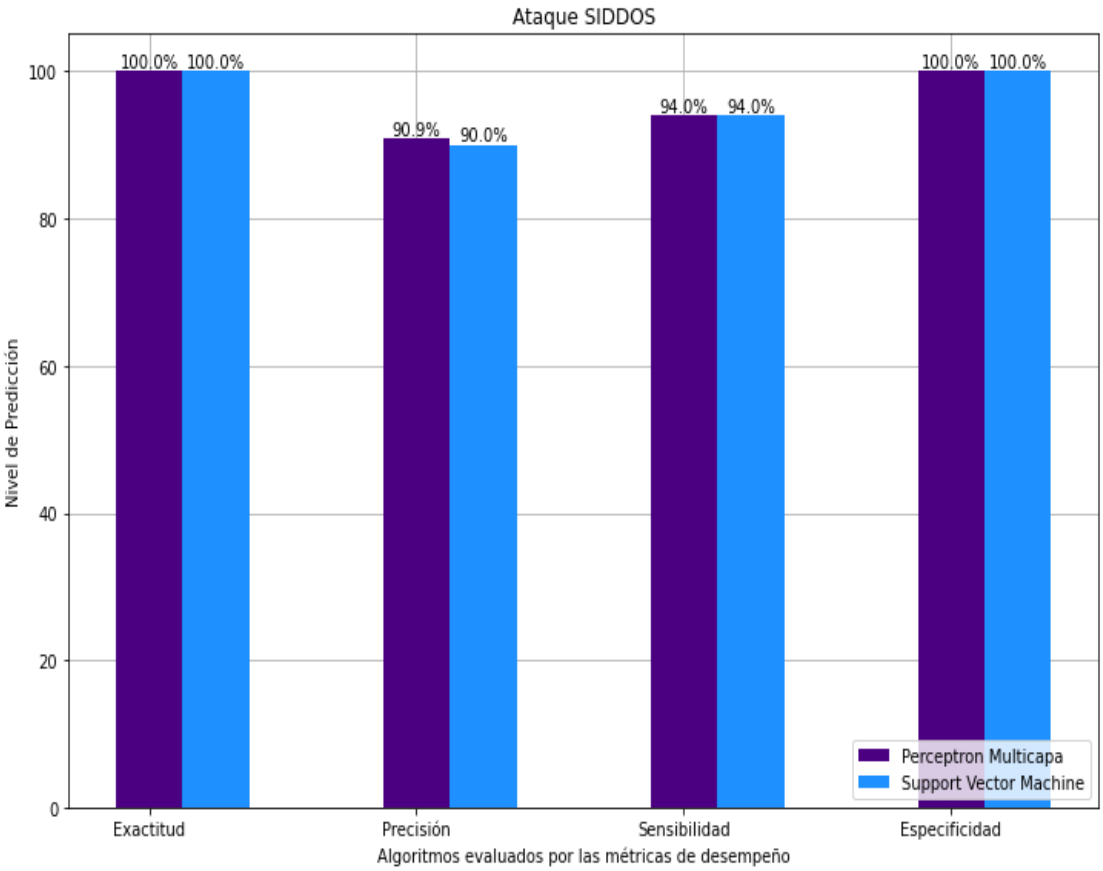

*Figura 39:*Comparación de Algoritmos basados en Exactitud, Precisión, Sensibilidad y Especificidad del Ataque SIDDOS.

Fuente: Elaboración Propia.

La figura Nº 40, muestra el análisis comparativo de los algoritmos Perceptrón Multicapa (MLP) y Máquinas de Soporte Vectorial (SVM), y se visualiza que tanto SVM y MLP son más exacto en la predicción del ataque Smurf, alcanzando el 99% en exactitud. Pero cuando se realiza la predicción el algoritmo MLP es más preciso para predecir el ataque Smurf obteniendo el 99%, a diferencia de SVM que solo alcanza el 97 % en precisión. En la sensibilidad el algoritmo perceptrón multicapa alcanzó el 35% y SVM obtuvo el 34% en identificar ataques correctamente y especificidad los algoritmos MLP y SVM presentan la mayor tasa de detección de tráfico normal logrando alcanzar el 100% en especificidad.

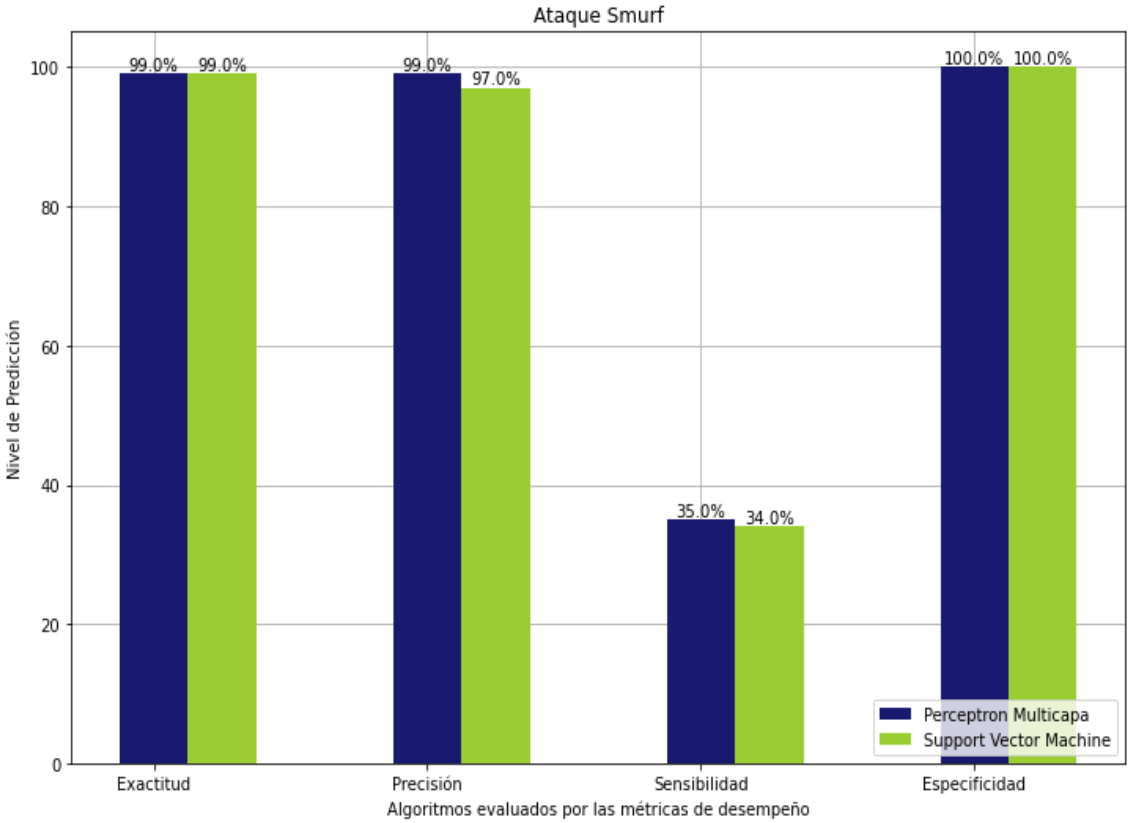

Figura 40: Comparación de Algoritmos basados en Exactitud, Precisión, Sensibilidad y Especificidad del Ataque Smurf.

Fuente: Elaboración Propia.

La figura Nº 41, muestra el análisis comparativo de los algoritmos Perceptrón Multicapa (MLP) y Máquinas de Soporte Vectorial (SVM), y se visualiza que tanto SVM y MLP son más exacto en la predicción del ataque UDP-Flood, alcanzando el 99% en exactitud. Pero cuando se realiza la predicción los dos algoritmos son preciso para predecir el ataque UDP-Flood, alcanzando el obteniendo el 96%. En la sensibilidad los 2 algoritmos alcanzaron el 100% en identificar ataques correctamente, y en la especificidad los algoritmos MLP y SVM presentan la mayor tasa de detección de tráfico normal logrando alcanzar el 100% en especificidad.

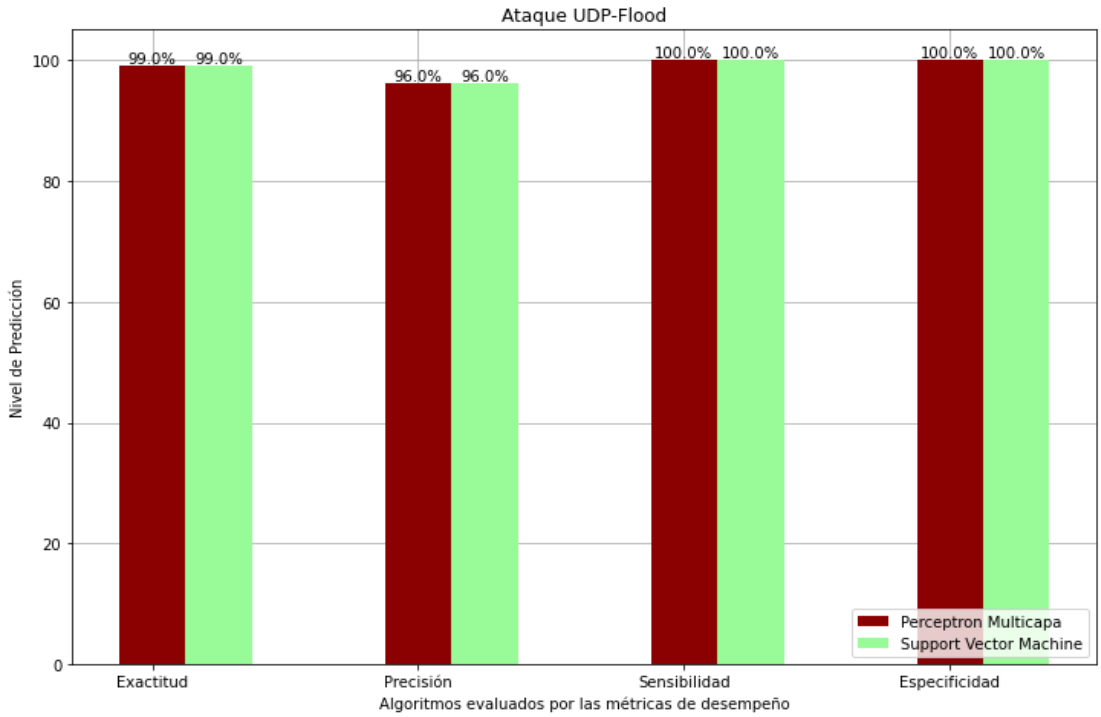

Figura 41: Comparación de Algoritmos basados en Exactitud, Precisión, Sensibilidad y Especificidad del Ataque UDP-Flood.

Fuente: Elaboración Propia.

La figura Nº 42, muestra el análisis comparativo del tiempo de entrenamiento de los algoritmos, Maquinas de soporte Vectorial alcanzó 114.83 milésimas de segundos menos tiempo en adaptar los datos al entrenamiento, a diferencia de la técnica Perceptrón Multicapa que tardo más tiempo en entrenarse alcanzado 141.52 milésima de segundos.

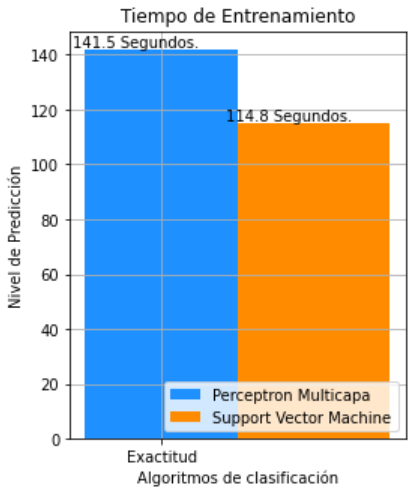

*Figura 42:*Tiempo de Entrenamiento de los algoritmos.

Fuente: Elaboración Propia.

3.2. Discusión de resultados.

Los algoritmos fueron evaluados, utilizando la matriz de confusión basada en las métricas de evaluación. El resultado de la matriz de confusión se muestra de los algoritmos SVM Y MLP, que se muestran en las Tablas Nº 35 y 36. A partir de estas matrices de confusión, se calculó la exactitud, precisión, sensibilidad y especificidad.

En la investigación de los autores Aziz, Hanafi, & Hassanien del año 2017, se usó el 20% de los datos, del conjunto de datos NSL-KDD, con un total de 25192 registros. A diferencia de la presente investigación, que empleó 186382 registros del dataset denominado Dataset_ Attack_2018, alcanzando una precisión del 99 % y los autores solo lograron alcanzar el 98.19%, incluso debido a la gran complejidad que presento el conjunto de datos.

El promedio ponderado para el algoritmo Perceptrón Multicapa en exactitud es el 99%, precisión del 98 %, sensibilidad del 67% y especificidad del 98%, obteniendo los mejores resultados, además el tiempo de entrenamiento este algoritmo fue de 141.52 segundos, mientras que Máquinas de Soporte Vectorial tuvo una exactitud del 98%, en precisión alcanzó solo el 96%, en sensibilidad logró el 67% y en especificidad, solo obtuvo el 97% resultados y su tiempo de entrenamiento fue de 114.83 segundos.

En los resultados de la Figura 69, se puede observar que Perceptrón Multicapa tuvo el mejor rendimiento para predecir el Ataque HTTP-Flood, obteniendo el 100% en exactitud, el 99% en Precisión, en sensibilidad alcanzó el 94% y el 100% en especificidad.

3.3. Aporte Práctico.

3.3.1. Método propuesto para el desarrollo del proyecto de investigación.

El método propuesto para esta investigación consta de las siguientes fases:

Fase es la preparación del Conjunto de datos o también denominado dataset, recolectándose patrones de ataques de los diferentes conjuntos de datos llamados: UNSW-NB15, DDOS_ATTACKS, PHISHING_WEBSITE y CICANDMAL2017 formando el nuevo conjunto de datos con 145 características y una etiqueta, luego se le aplicó la normalización, utilizando para ello lenguaje de programación Python que es utilizado para el análisis de datos, formando así el nuevo conjunto de datos con los ataques más actuales.

Fase de selección de algoritmos, la presente investigación realizó un estudio detallado, de la evaluación de la eficiencia en la detección de intrusos, evaluando diferentes algoritmos utilizando las métricas de desempeño, para ello se utilizó diferentes artículos científicos visitados en la paginas ieee, scielo, Springer, SienceResearch formando un top de los 6 mejores algoritmos para su evaluación, seleccionando en esta investigación los algoritmos Máquinas de Soporte Vectorial y Perceptrón Multicapa.

Fase de Implementación, la presente investigación seleccionó dos algoritmos con mayor eficiencia en la detección de intrusos, los algoritmos implementados en esta investigación son: Maquinas de soporte Vectorial y Perceptrón Multicapa.

Fase de Evaluación, se obtiene los resultados de la evaluación de los algoritmos de Maquinas de soporte vectorial y Perceptrón Multicapa, basándose en las métricas de eficiencia como son: la exactitud, sensibilidad, especificidad y precisión. Ver Figura N°43.

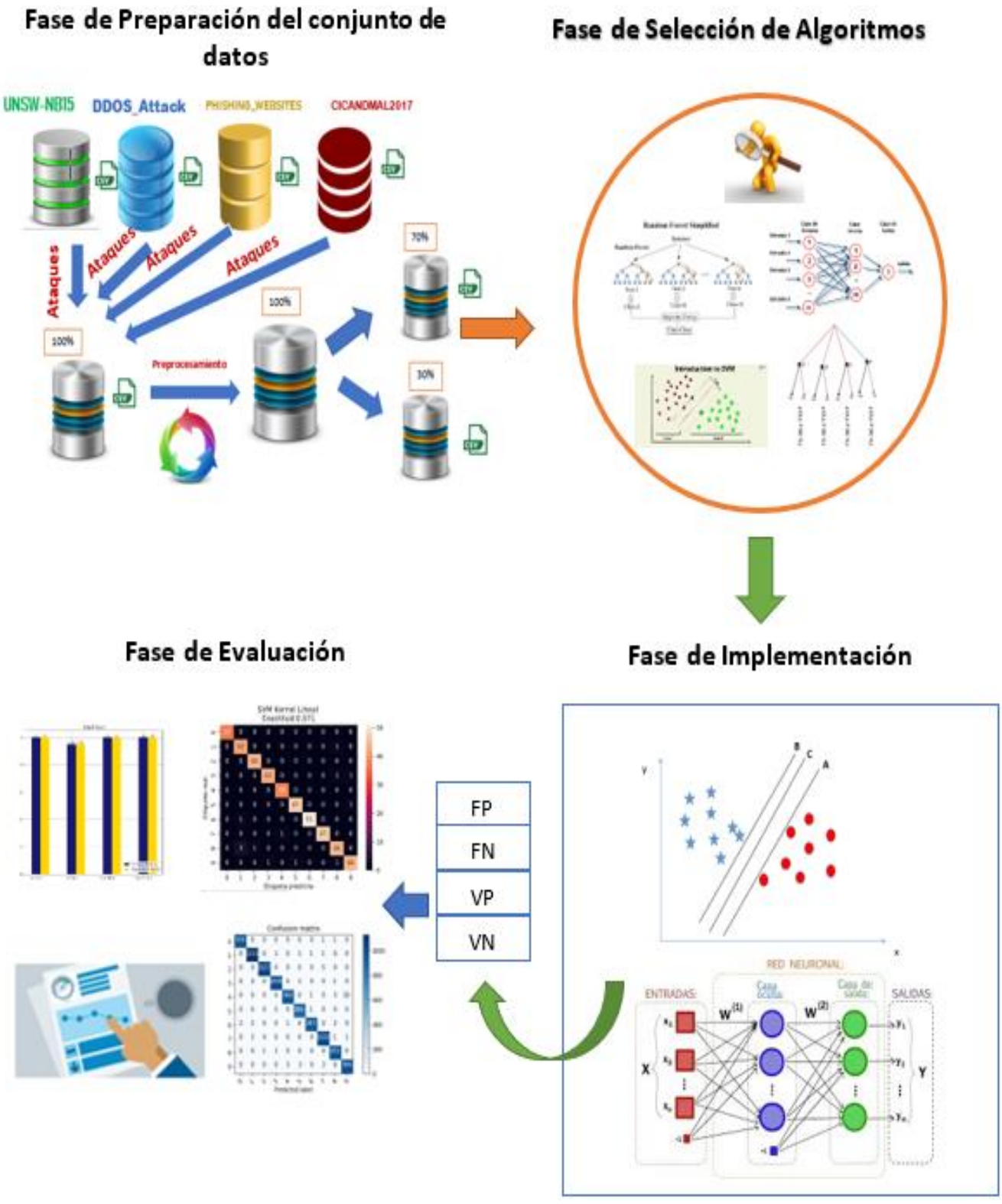

Figura 43: Método Propuesto para el desarrollo de la Implementación.

Fuente: Elaboración Propia.

3.3.1.1. Fase de Preparación del conjunto de datos.

Para la detección de ataques se utilizó 4 conjuntos de datos y son los siguientes el primero fue el UNSW-NB15, de donde se extrajo patrones de ataques de Exploits y Backdoor, el Segundo que se utilizó fue el Dataset_ddos, extrayendo los ataques Smurf, UDPFlood, SIDDOS, HTTP-Flood, el tercer dataset que se utilizó es el Dataset_Phishing extrayendo ataques de Pishing y el último conjunto seleccionado es el Dataset denominado CICAndMal2017, extrayendo los ataques de la familia Ransomware, datos que fueron pre procesados y normalizados para ser utilizados en el entrenamiento y en las pruebas aplicadas a las técnicas en estudio.

En esta fase de preparación de los datos, se utilizaron diferentes conjuntos de datos, utilizando los ataques más actuales y recientes según los reportes anuales y trimestrales por las empresas VERSIGIN, Eset, Karpesky, Report Internet Crime (IC3) durante el año 2018. El nuevo Dataset llamado Dataset_Attack_2018 es una recolección de ataques, de los diferentes conjuntos descritos anteriormente, que fueron obtenidos por investigadores que se mencionan a continuación.

Se utilizó la base de datos del proyecto llamado UNSW-NB15 de la Universidad de Nueva Gales del Sur, realizado por los autores Mustafá y Slay. El total de registros de este Dataset es de 2 000 540,044 que son divididos en 4 archivos de Valores Separados por comas (CSV) llamados UNSW-NB15_1.csv , UNSW-NB15_2.csv , UNSW-NB15_3.csv y UNSW-NB15_4.csv, para la realización de este conjunto utilizaron la herramienta IXIA PefefecStorn en el Cyber Range Lab del Centro Australiano para la seguridad(ACCS), encontrándose en este conjunto de datos ataques como son: Fuzzers, Analysis, Backdoors, DoS, Exploits, Generic, Reconnaissance, Shellcode, filtrando este conjunto de datos y extrayendo los ataques Backdoors y Exploits que a continuación se presenta el siguiente Tabla Nº 18 con la cantidad de registros extraídos. El conjunto de datos UNSW-NB15 cuenta con 49 características ver **Anexo N° 05.**

Tabla 18:

Ataques Exploits y Backdoor extraídos del Conjunto de datos UNSW-NB15.

N°	Nombre del Ataque	Cantidad de Registros
1	Exploit	44525
2	Backdoor	2329
	Total	**46,854**

Nota: Elaboración Propia. Tomado de (Moustafa & Slay, 2015, pág. 5).

Se utilizó el siguiente conjunto de datos denominado Dataset_ddos, perteneciente a la Universidad Princess Sumaya Universidad para la tecnología, desarrollada por los autores (Alkasassbeh, Al-Naymat, Hassanat, & Almseidein, 2016). Este conjunto de datos tiene 1 millón 048576 total de registros, encontrándose tráfico normal y 4 tipos de ataques dañinos como: Smurf, UDP-Flood, SIDDOS, HTTP-Flood, extrayendo solo los ataques que se presentan en la Tabla Nº 19.

Tabla 19:

Ataques Smurf, UDP Flood, SIDDOS, HTTP Flood extraídos del conjunto de datos DDOS.

N°	Nombre del Ataque	Cantidad de Registros
1	Smurf	12590
2	UDP Flood	201344
3	SIDDOS	6665
4	HTTP Flood	4110
	Total	**224709**

Nota: Elaboración Propia. Tomado de (Alkasassbeh, Al-Naymat, Hassanat, & Almseidein, 2016, pág. 5).

Así mismo del conjunto de datos denominado (Dataset_DDOS) cuenta con 28 características y una etiqueta, el conjunto de datos se obtuvo del repositorio (Researchgate,2016) ver **Anexo N° 06.**

También se utilizó el conjunto de datos (Phishing) perteneciente a la Universidad de Huddersfield (Mohammad, Thabtah, & McCluskey, 2012), que recolectaron 2500 URL de suplantación de identidad, del sitio web Phisttank, que es un sitio que examina phishing, obteniéndose de este conjunto de datos el ataque Phishing. El conjunto de datos tiene 1353 registros de Phishing extrayendo los ataques Phishing que continuación se presenta en la siguiente Tabla Nº 20 con la cantidad de registros extraídos.

Tabla 20:

Ataques Phishing extraídos del conjunto de datos Phishing.

N°	Nombre del Ataque	Cantidad de Registros
1	Phishing	702
2	Normal	548
3	Sospecha	103
	Total	**1353**

Nota: Elaboración Propia. Tomado de (Mohammad, Thabtah, & McCluskey, 2012).

Este conjunto de datos llamando Phishing cuenta con 10 características y una etiqueta ver **Anexo N° 07.**

El último conjunto de datos que se utilizó es el dataset llamado CICAndMal2017 perteneciente a la Universidad de New Brunswick (UNB) y el Instituto Canadiense de Ciberseguridad (CIC), el conjunto de datos se creó en base a las herramientas CICFlower-Meter-V3 y Biflow, que son herramientas de capturas de tráfico de red, desarrollados por los investigadores (Arash, Andi, Laya, & Ali, 2018). El conjunto de datos mencionado cuenta con 10, 854 muestras (4,354 malware y 6,500 benignas), divididas en 4 categorías: (Adware, Ransomware, Scareware y Malware), provenientes de 42 familias de Malware, de donde se extrajo el ataque Ransomware con sus respectivas familias que a continuación se detallará en la Tabla N° 21 con su cantidad de registros por familia.

Tabla 21*:*

Ataques Ransomware por Familias extraídos del conjunto de datos CICAndMal2017.

N°	Familia	Nombre de la Familia	Cantidad de Registros
1	Ransomware	Charger	39552
2	Ransomware	Jisut	25673
3	Ransomware	Koler	44556
4	Ransomware	Lockerpin	25307
5	Ransomware	Pletor	4715
6	Ransomware	PornDroid	46082
7	Ransomware	RansomBO	39859
8	Ransomware	Simplocker	36340
9	Ransomware	SVpeng	54161
10	Ransomware	WannaLocker	32701
		Total	**348946**

Nota: Elaboración Propia. Tomado de (Arash, Andi, Laya, & Ali, 2018, pág. 4).

El conjunto de datos mencionado contiene 84 Características y una etiqueta. Ver **Anexo N° 08.**

Para la creación del nuevo dataset se tomó como referencia los conjuntos de datos UNSW-NB15, Phishimg. Dataset_ddos y CICAndMal2017 formando un nuevo conjunto de datos con los ataques: Exploit, Backdoor, Phishing, Smurf, UDP Flood, SIDDOS, HTTP Flood, Charge, Jisut, Koler, Lockerpin, pletor, PornDroid, RansomBO, Simplocker, Svpeng, WannaLocker y Normal y cuentan con 146 características de tráfico de red y las etiquetas que identifican cada ataque de tráfico de red. Ver tabla N° 22.

*T*abla 22*:*

Ataques del Nuevo Conjunto de Datos.

N°	Tipo de Ataque	Etiqueta de Ataque	Cantidad de Registros
1	Expoit	Exploits	44525
2	Backdoor	Backdoor	2329
3	DDOS	Smurf	12590
4	DDOS	UDP-Flood	201344
5	DDOS	SIDDOS	6665
6	DDOS	HTTP Flood	4110
7	Ransomware	Ransomware	348493
8	Phising	Phishing	702
	Total		**621208**

Fuente: Elaboración Propia. Tomado de ([a]Moustafa & Slay, 2015, pág. 5), ([b]Alkasassbeh, Al-Naymat, Hassanat, & Almseidein, 2016, pág. 5), ([c]Mohammad, Thabtah, & McCluskey, 2012), ([d]Arash, Andi, Laya, & Ali, 2018, pág. 4).

3.3.1.1.1. Limpieza y transformación de los datos obtenidos.

En esta fase es importante la preparación de los datos, se analizó los diferentes conjuntos de datos, comparando con las coincidencias de características de los 4 conjuntos de datos seleccionados, con la finalidad de proceder a unir todos los registros, formando así el nuevo conjunto de datos dataset_Attack_2018, para ello se utilizó la librería de pandas en el lenguaje Python, para la visualización de los datos se utilizó EmEditor, en este proyecto también se realizó la transformación de los datos, de datos categórico a numéricos. Para la unión de todos los registros, esta investigación se basó en las coincidencias de las características que se presentan en la siguiente tabla Nº 23, luego se procedió a la simplificación de registros duplicados basándome en el autor (Benítez et all, 2014) en la reducción de la

dimensionalidad, eliminando redundancias y eliminaciones de ruido, realizando los siguientes pasos que a continuación detallaremos.

Tabla 23:

Análisis y comparación de características de los conjuntos de datos.

Ataques			
Backdoor y Exploit	Ransomware	DDoS	Phishing
Srcip	Source IP		
sport	Source Port	SRC_ADD	
Dstip	Destination IP	DES_ADD	
dsport	Destination Port		
proto	Protocol	PKT_TYPE	No hay coincidencia
dur	Flow Duration		
sbytes	Total Length of Fwd Packets		
dbytes	Total Length of Bwd Packets		
Spkts	Fwd Packets/s		
smeansz	Fwd Packet Length Mean		
dmeansz	Bwd Packet Length Mean		
Stime	Timestamp		
Dpkts	Bwd Packets/s		
	Average Packet Size	PKT_AVG_SIZE	

Nota: Elaboración Propia.

Primer Paso: Añadir ceros a los atributos vacíos.

Primeramente, se examina todos los atributos del dataset que se encuentran guardado en un archivo csv, dichos datos son representados en forma de una matriz multidimensional. Para lograr procedimientos de análisis y transformación de datos se empleó el módulo pandas y para visualización de

los datos el editor EmEditor, donde las columnas representan las variables o atributos y las filas corresponden a las observaciones realizadas.

La consecuencia de unificar de las muestras de los conjuntos de datos con base a las características es la formación de atributos vacíos y para resolver este problema se procedió a reemplazar los atributos vacios con ceros, ya que este valor no altera el aprendizaje de los algoritmos. Para la implementación del problema mencionado se construyó un script en python que consiste en cargar los datos con el método read_csv () de la librería pandas, la característica que tiene método es que analiza todos los atributos de las muestras para definir su tipo de dato y como resultado arrojó que la base de datos contiene atributos de tipo NAN (Not A Number), esto es producto de la unificación de las muestras. En la Figura Nº 44, se muestra el proceso para agregar ceros a los atributos vacíos, el funcionamiento de este script es recorrer cada muestra de la base de datos identificando los atributos de tipo NAN para luego reemplazarlo con el valor cero para finalmente devolver una base de datos sin atributos vacíos.

```
import numpy as np
import pandas as pd
def reemplazar_datos(fuente,remplazado,valor,nomArch):
    data = pd.read_csv(fuente)
    reemplazo = data.replace(remplazado,valor)

    reemplazo.to_csv(nomArch,index=False,index_label=False,sep=',')

reemplazar_datos('Attack.csv',np.nan,0,'temp_attck_2018.csv')
```

Figura 44: Método Propuesto para el desarrollo de la Implementación.
Fuente: Elaboración Propia

Segundo Paso: Eliminación de la columna Flow_Id, Node_From y Node_Name.

La eliminación de las columnas Flow_ID, conteniendo datos redundantes con las columnas Source_IP, Source_Port, Destination_IP, Destination_Port y Protocol. La eliminación de las columnas que son innecesarias para el aprendizaje ya que esto sería un problema para el aprendizaje de las técnicas según (Benítez, Escudero, & Kanaan, 2014) las columnas a eliminar son las siguientes Node_NAME_FROM y NODE _NAME_TO como se muestra en las

Tablas Nº 24 y 25 y se hizo con el software EmEditor para visualizar los datos y un script realizado en Python, utilizando el método drop ver Figura Nº 45.

Tabla 24:

Eliminación de las columnas innecesarias Flow_ID.

Flow ID	Source IP	Soruce Port	Destination IP	Destination Port	Protocol
216.58.219.234-10.42.0.211-443-54481-6	10.42.0.211	54481	216.58.219.234	443	6
216.58.219.234-10.42.0.211-443-36689-6	10.42.0.211	36689	216.58.219.234	443	6
172.217.10.46-10.42.0.211-443-44595-6	10.42.0.211	44595	172.217.10.46	443	6
216.58.225.225-10.42.0.211-443-46117-6	10.42.0.211	46117	216.58.225.225	443	6
172.217.11.4-10.42.0.211-443-50580-6	10.42.0.211	50580	172.217.11.4	443	6
216.58.219.234-10.42.0.211-443-48323-6	10.42.0.211	48323	216.58.225.234	443	6
216.58.219.234-10.42.0.211-443-44050-6	10.42.0.211	44050	216.58.225.234	443	6

Flow ID	Source IP	Soruce Port	Destination IP	Destination Port	Protocol
216.58.219.234-10.42.0.211-443-448356-6	10.42.0.211	448356	216.58.225.234	443	6
172.217.3.110-10.42.0.211-443-43774-6	10.42.0.211	43774	172.217.3.110	443	6
202.77.129.230-10.42.0.211-80-47517-6	10.42.0.211	47517	202.77.129.230	80	6
202.77.129.230-10.42.0.211-80-47517-6	10.42.0.211	47517	202.77.129.230	80	6
172.217.3.110-10.42.0.211-443-43599-6	10.42.0.211	43599	172.217.3.110	443	6

Nota: Elaboración Propia.

Tabla 25:

Eliminación de las columnas NODE_NAME_FROM y NODE_NAME_TO.

Node_Name From	Node Name To	PKT_In	PKT_Out	PKT_R	PKT_ Delay_ Node	PKT_ Rate
Switch1	Router 1	35.529786	35.529786	35.539909	0	328.240918
Router	Server 1	20.176725	20.176725	20.186848	0	328.205808
Router	Switch 2	7.049955	7.049955	7.059958	0	328.206042
Router	Switch 1	39.62797	39.62797	39.637973	0	328.064183
Router	Switch 1	16.039806	16.039806	16.04981	0	328.113525
Switch1	Client-1	21.885768	21.885768	21.895771	0	328.207902
Server1	Router	42.45032	42.45032	42.460323	0	328.460278
Router	Server 1	58.26832	58.26848	58.27856	0.00016	124.943625
Server1	Router	47.910078	47.910078	47.920081	0	328.26412
Switch1	Router	36.314926	36.31548	36.325603	0.000554	328.26404
Router	Server 1	32.180177	32.180177	32.191377	0.00108	1016.496869
Router	Switch 2	33.015317	33.015317	33.02532	0	328.522947
Switch1	Client-5	32.442133	32.442133	32.452136	0	328.204851
Router	Server 1	70.05264	32.442133	70.0632	0.0048	124.942027
Switch1	Client-5	49.037576	49.037576	49.0475579	0	328.204851
Switch1	Client-1	42.382176	42.382176	42.392179	0	328.297902
Switch1	Client-7	45.383957	45.383957	45.399573	0	328.167767
Router	Switch 1	2.524349	2.524349	2.534352	0	328.19309
Client-4	Switch 1	14.477384	14.477384	14.487507	0	328.217832

Nota: Elaboración Propia.

```
import pandas as pd

columnas = ['Flow_ID','NODE_NAME_FROM','NODE_NAME_TO']

def delete_columns(fuentes_datos,listcolumnDel,nombArch):
    obj_datos = pd.read_csv(fuentes_datos)
    obj_datos.drop(columnDel,axis = 1,inplace = True)
    obj_datos.to_csv(nombArch,index=False, index_label = False, sep = ',')

delete_columns('Attack_2018',columnas,'temp_attack_2018.csv')
```

Figura 45: Eliminación de la columna Flow_Id, Node_From y Node_Name.
Fuente: Elaboración Propia.

Tercer Paso: Conversión de la Columna Source_IP a Numérico, protocolos y estado

Se realizó la conversión de la columna Source_IP a Numérico, esta columna contenía una dirección IP y con la ayuda de la herramienta EmEditor, se extrajo la columna Source_IP del conjunto de datos Attack_2018.csv, y se guardó esa columna en un nuevo archivo de Excel llamando temporal.csv, luego se procedido a realizar un script en Python, llamando a los dos archivos, un archivo de origen (Attack_2018.csv) y otro de destino (temp_attack_2018.csv), ya con la función creada en Python, se realiza satisfactoriamente la conversión la columna Source_IP en un dato ya numérico en el conjunto de datos, protocolos y estado del dataset_2018_attack.csv. Ver Tabla Nº 26 y Figura Nº 46 antes de la conversión dato numérico y asignación de estados.

Tabla 26:

Columna Source_IP antes de la Conversión a dato.

Flow_ID	Source_IP	Soruce_Port	Destination_IP
216.58.219.234-10.42.0.211-443-54481-6	10.42.0.211	54481	216.58.219.234
216.58.219.234-10.42.0.211-443-36689-6	10.42.0.211	36689	216.58.219.234
172.217.10.46-10.42.0.211-443-44595-6	10.42.0.211	44595	172.217.10.46
216.58.225.225-10.42.0.211-443-46117-6	10.42.0.211	46117	216.58.225.225
172.217.11.4-10.42.0.211-443-50580-6	10.42.0.211	50580	172.217.11.4
216.58.219.234-10.42.0.211-443-48323-6	10.42.0.211	48323	216.58.225.234
216.58.219.234-10.42.0.211-443-44050-6	10.42.0.211	44050	216.58.225.234
216.58.219.234-10.42.0.211-443-448356-6	10.42.0.211	448356	216.58.225.234
172.217.3.110-10.42.0.211-443-43774-6	10.42.0.211	43774	172.217.3.110
202.77.129.230-10.42.0.211-80-47517-6	10.42.0.211	47517	202.77.129.230
202.77.129.230-10.42.0.211-80-47517-6	10.42.0.211	47517	202.77.129.230
172.217.3.110-10.42.0.211-443-43599-6	10.42.0.211	43599	172.217.3.110
202.77.129.150-10.42.0.211-80-42026-6	10.42.0.211	42026	202.77.129.150
202.77.129.150-10.42.0.211-80-42026-6	10.42.0.211	42026	202.77.129.150
202.77.129.230-10.42.0.211-80-47518-6	10.42.0.211	47518	202.77.129.230
202.77.129.130-10.42.0.211-80-47518-6	10.42.0.211	47518	202.77.129.230

Flow_ID	Source_IP	Soruce_Port	Destination_IP
202.77.129.185-10.42.0.211-80-34790-6	10.42.0.211	34790	202.77.129.185
202.77.129.185-10.42.0.211-80-34789-6	10.42.0.211	34789	202.77.129.185
202.77.129.185-10.42.0.211-80-42048-6	10.42.0.211	42048	202.77.129.185

Nota: Elaboración Propia.

```
def estados(x):
    opciones = {
        'CON':0,
        'FIN':1,
        'INT':2,
        'REQ':3,
        'ACC':4,
        'RST':5,
     }
    return opciones.get(x, "default")
data = pd.read_csv('Attack_2018.csv')
data['state'] = data['state'].apply(lambda x: ataques(x),0)
data.to_csv('temp_attack_2018.csv',index=False, index_label = False, sep = ',')
```

Figura 46: Asignación de valores para estado.

Fuente: Elaboración Propia.

Cuarto Paso: Conversión de la Columna Destination_IP.

Se realizó la conversión de la columna Destination_IP a Numérico, esta columna contenía una dirección IP y con la ayuda de la herramienta EmEditor, se extrajo la columna Destination_IP del conjunto de datos Dataset_2018_attack.csv, y se guardó esa columna en un nuevo archivo de Excel llamando temporal1.csv, luego se procedido a realizar un script en Python, llamando a los dos archivos, un archivo de origen (temporal1.csv) y otro de destino (Dataset_2018_attack.csv), ya con la función creada en Python, se realiza satisfactoriamente la conversión la columna Source_IP en un dato ya numérico en el conjunto de datos Dataset_2018_attack.csv. Ver Tabla Nº 27 antes de la conversión a dato numérico.

Tabla 27:

Columna Destination_IP antes de la conversión a dato numérico.

Flow_ID	Source IP	Soruce Port	Destination_IP	Destination Port
216.58.219.234-10.42.0.211-443-54481-6	10.42.0.211	54481	216.58.219.234	443
216.58.219.234-10.42.0.211-443-36689-6	10.42.0.211	36689	216.58.219.234	443
172.217.10.46-10.42.0.211-443-44595-6	10.42.0.211	44595	172.217.10.46	443
216.58.225.225-10.42.0.211-443-46117-6	10.42.0.211	46117	216.58.225.225	443
172.217.11.4-10.42.0.211-443-50580-6	10.42.0.211	50580	172.217.11.4	443
216.58.219.234-10.42.0.211-443-48323-6	10.42.0.211	48323	216.58.225.234	443
216.58.219.234-10.42.0.211-443-44050-6	10.42.0.211	44050	216.58.225.234	443
216.58.219.234-10.42.0.211-443-448356-6	10.42.0.211	448356	216.58.225.234	443
172.217.3.110-10.42.0.211-443-43774-6	10.42.0.211	43774	172.217.3.110	443
202.77.129.230-10.42.0.211-80-47517-6	10.42.0.211	47517	202.77.129.230	80

Flow_ID	Source IP	Soruce Port	Destination_IP	Destination Port
202.77.129.230-10.42.0.211-80-47517-6	10.42.0.211	47517	202.77.129.230	80
202.77.129.230-10.42.0.211-80-47517-6	10.42.0.211	47517	202.77.129.230	80
172.217.3.110-10.42.0.211-443-43599-6	10.42.0.211	43599	172.217.3.110	443
202.77.129.150-10.42.0.211-80-42026-6	10.42.0.211	42026	202.77.129.150	80

Nota: Elaboración Propia.

Quinto Paso: Conversión de la Columna TimeStamp

Se realizó la conversión de la columna Timestamp a Numérico, esta columna contenía una fecha y una hora y con la ayuda de la herramienta EmEditor, se extrajo la columna Timestamp del conjunto de datos Dataset_2018_attack.csv, y se guardó esa columna en un nuevo archivo de Excel llamando temporal2.csv, luego se procedido a realizar un script en Python, llamando a los dos archivos, un archivo de origen (temporal2.csv) y otro de destino (Dataset_2018_attack.csv), ya con la función creada en Python, se realiza satisfactoriamente la conversión la columna Timestamp en un dato ya numérico en el conjunto de datos Dataset_2018_attack.csv ver Tabla Nº 28 antes de la conversión.

Tabla 28:

Columna Time_Stamp antes de la conversión a dato numérico.

Source IP	Soruce Port	Destination IP	Destination Port	Protocol	TimeStamp
10.42.0.211	54481	216.58.219.234	443	6	30/08/2017 10:26
10.42.0.211	36689	216.58.219.234	443	6	30/08/2017 10:26
10.42.0.211	44595	172.217.10.46	443	6	30/08/2017 10:26
10.42.0.211	46117	216.58.225.225	443	6	30/08/2017 10:26
10.42.0.211	50580	172.217.11.4	443	6	30/08/2017 10:26
10.42.0.211	48323	216.58.225.234	443	6	30/08/2017 10:26
10.42.0.211	44050	216.58.225.234	443	6	30/08/2017 10:26
10.42.0.211	448356	216.58.225.234	443	6	30/08/2017 10:26
10.42.0.211	43774	172.217.3.110	443	6	30/08/2017 10:26
10.42.0.211	47517	202.77.129.230	80	6	30/08/2017 10:26
10.42.0.211	47517	202.77.129.230	80	6	30/08/2017 10:26
10.42.0.211	43599	172.217.3.110	443	6	30/08/2017 10:26
10.42.0.211	42026	202.77.129.150	80	6	30/08/2017 10:26
10.42.0.211	42026	202.77.129.150	80	6	30/08/2017 10:26
10.42.0.211	47518	202.77.129.230	80	6	30/08/2017 10:26

Source IP	Soruce Port	Destination IP	Destination Port	Protocol	TimeStamp
10.42.0.21	47518	202.77.129.230	80	6	30/08/2017
1					10:26
10.42.0.21	34790	202.77.129.185	80	6	30/08/2017
1					10:26
10.42.0.21	34789	202.77.129.185	80	6	30/08/2017
1					10:26
10.42.0.21	42048	202.77.129.185	80	6	30/08/2017
1					10:26

Nota: Elaboración propia.

Sexto Paso: Conversión de la Columna Protocol a Numérico

Se realizó la conversión de la columna Protocol a Numérico, esta columna contenía un dato categórico y con la ayuda de la herramienta EmEditor, se extrajo la columna Protocol del conjunto de datos Dataset_2018_attack.csv, y se guardó esa columna en un nuevo archivo de Excel llamando temporal3.csv, luego se procedido a realizar un script en Python, llamando a los dos archivos, un archivo de origen (temporal3.csv) y otro de destino (Dataset_2018_attack.csv), ya con la función creada en Python, se realiza satisfactoriamente la conversión la columna Protocol en el conjunto de datos Dataset_2018_attack.csv asignándole a los protocolos un valor numérico, basándome en la investigación de los autores (De la Hoz, De la Hoz, Ortiz, & Ortega, 2012) que realizan la conversión los protocolos que contenían un dato categórico a un dato numérico ver Tabla Nº 29.

Tabla 29:

Conversión de la columna Sate.

Proto**col**					
N°	Protocol	N°	Protocol	N°	Protocol
0	ip	**42**	il	**84**	pup
1	3pc	**43**	crtp	**85**	pvp
2	a/n	**44**	ipcomp	**86**	qnx
3	aes-sp3-d	**45**	ipcv	**87**	rdp
4	any	**46**	ipip	**88**	rsvp
5	argus	**47**	iplt	**89**	rvd
6	tcp	**48**	ipnip	**90**	sat-expak
7	ax.25	**49**	ippc	**91**	sat-mon
8	bbn-rcc	**50**	ipv6	**92**	sccopmce
9	bna	**51**	ipv6-frag	**93**	scps
10	br-sat-mon	**52**	ipv6-no	**94**	sctp
11	cbt	**53**	ipv6-opts	**95**	sdrp
12	cftp	**54**	ipv6-route	**96**	secure-vmtp
13	chaos	**55**	ipx-n-ip	**97**	sep
14	compaq-peer	**56**	irtp	**98**	skip
15	cphb	**57**	isis	**99**	sm
16	cpnx	**58**	iso-ip	**100**	smp
17	Messenger	**59**	iso-tp4	**101**	snp
18	crudp	**60**	kryptolan	**102**	sprite-rpc
19	dcn	**61**	l2tp	**103**	sps
20	ddp	**62**	larp	**104**	srp
21	ddx	**63**	leaf-1	**105**	st2
22	dgp	**64**	leaf-2	**106**	stp

Proto**col**					
N°	Protocol	N°	Protocol	N°	Protocol
23	egp	**65**	merit-inp	**107**	sun-nd
24	eigrp	**66**	mfe-nsp	**108**	swipe
25	emcon	**67**	mhrp	**109**	tcf
26	encap	**68**	micp	**110**	aris
27	etherip	**69**	mobile	**111**	tlsp
28	fc	**70**	mtp	**112**	tp++
29	fire	**71**	mux	**113**	trunk-1
30	ggp	**72**	narp	**114**	trunk-2
31	gmtp	**73**	netblt	**115**	ttp
32	gre	**74**	nsfnet-	**116**	udp
33	Hmp	**75**	nvp	**117**	unas
34	i-nlsp	**76**	ospf	**118**	uti
35	iatp	**77**	pgm	**119**	vines
36	ib	**78**	pim	**120**	visa
37	idpr	**79**	pipe	**121**	vmtp
122	vrrp	**126**	xnet	**130**	ack
123	wb-expak	**127**	xns-idp	**131**	cbr
124	wb-mon	**128**	xtp	**132**	ping
125	wsn	**129**	zero		

Nota: Elaboración Propia.

En el sexto paso se realizó la conversión de la columna State, ya que contenía un valor categórico y con la ayuda de un script realizado en Python se convirtió el dato categórico a un valor Numéricos asignándoles los valores enteros de (0,1.2 y 3) ver Tabla Nº 30.

Tabla 30:

Conversión de la columna Sate categórico a Numérico.

N°	State	N° de Registos
0	**CON**	63
1	**FIN**	323
2	**INT**	1898
3	**REQ**	45
4	**ACC**	1
5	**RST**	7

Nota: Elaboración Propia.

En el sétimo paso se realizó la conversión de la columna Service, ya que contenía un valor categórico y con la ayuda de un script realizado en Python se convirtió el dato categórico a un valor Numéricos asignándoles los valores enteros de (0,1.2,3,4,5,6,7,8,9.10.11 y12) ver Tabla Nº 31.

Tabla 31:

Conversión de la columna Service.

N°	Service	N° Registros
0	"-"	23045
1	ftp	2115
2	http	11481
3	Irc	12
4	Dhcp	64
5	ftp-data	1876
6	pop3	1435
7	Radius	9

N°	Service	N° Registros
8	Smtp	4162
9	Snmp	7
10	Ssh	14
11	Ssl	52
12	Dns	253

Nota: Elaboración Propia.

Una vez ya formado el nuevo conjunto de se procederá a realizar la normalización del Dataset_Attack_20018. Según los autores Basheer & Najmeer (2000) para redes neuronales se recomienda la técnica de normalización Mínimo Máximo, para las funciones de activación.

3.3.1.1.2. Min Max Normalización del conjunto de datos Ejemplo con el Dataset_Attack_2018

Min Max es una técnica que ayuda a normalizar los datos. Se escalan entre 0 y 1.

Proceso de la normalización realizada para esta investigación aplicando la normalización Minimo Max.

Tabla 32:

Ejemplo del Conjunto de Datos antes de la Normalización.

Source_IP
18853153
29218081
3729333344
3746728155

Nota: Elaboración Propia.

Min: El valor mínimo del atributo dado**.** En este ejemplo el min es 18853153

Max: El valor máximo del atributo. En este ejemplo el Max es 3746728155

V= Es el valor respectivo del atributo. En este ejemplo los valores son los siguientes: V1**=** 18853153, V2 = 29218081, V3= 3729333344, v4= 3746728155.

Nuevo Máximo: 1, **Nuevo Mínimo:** 0

Proceso del Valor 1: 18853153

$$MinMax=\frac{(18853153-18853153)}{(3746728155-\ 18853153\ )}*(1-0)+0$$

$$MinMax=\frac{(0)}{3727875002}*1$$

$$MinMax=0$$

Proceso del Valor 2: 29218081

$$MinMax=\frac{(\ 29218081-18853153)}{(3746728155-\ 18853153\ )}*(1-0)+0$$

$$MinMax=\frac{(10364928)}{3727875002}*1 = 0.00278039$$

Proceso del Valor 3: 3729333344

$$MinMax=\frac{(\ 3729333344-18853153)}{(3746728155-\ 18853153\ )}*(1-0)+0$$

$$MinMax=\frac{(3710480191)}{3727875002}*1 = 0.99533385$$

Proceso del Valor 4: 3746728155

$$MinMax=\frac{(3746728155-\ 18853153\ )}{(3746728155-\ 18853153\ )}*(1-0)+0$$

$$MinMax=\frac{(3727875002)}{3727875002}*1 \quad = \quad 1$$

Tabla 33:

Comparación del Dataset antes y después de la Normalización.

Source_IP	Source_IP
18853153	0
29218081	0.00278039
3729333344	0.99533385
3746728155	1.0

Nota. Elaboración Propia

Luego de haber culminado con las fases de procesamiento de datos con el nuevo conjunto de datos ya formado, basándome en libro denominado "Mastering Apache Spark",del autor (Frampton, 2015), se procedió a dividir al conjunto de datos que tiene 100% de la data, en 2 partes un conjunto de entrenamiento que tiene el 70 %, que sirve para construir el modelo y un conjunto de prueba que tiene el 30%, qué sirve para validar el modelo creado, quedando de la siguiente manera ver Figura Nº 47.

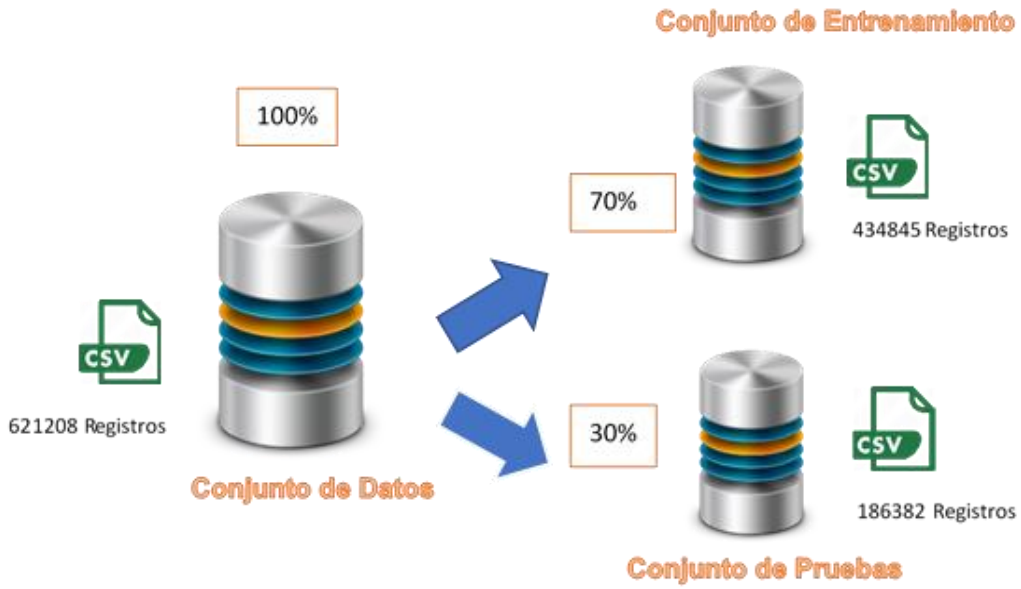

*Figura 47:*División del Conjunto de Datos Entrenamiento y Pruebas del Dataset_Attack_2018.

Fuente: Elaboración Propia

3.3.1.1.3. Conjuntos de Ataques Cibernéticos denominado Dataset_ Attack_2018

Luego de haber concluido con la fase de Normalización de los datos del dataset denominado Dataset_Attack_20018 en formato csv, tal y como se muestra en la Tabla Nº 34, se procederá con la implementación de los algoritmos Máquinas de Vectorial y Perceptrón Multicapa, utilizando para ello el conjunto de datos para la predicción de los algoritmos mencionados.

Tabla 34*:*

Ataques del Dataset Attack 2018 en formato CSV.

Source_IP	Source Port	Destination IP	Destination Port	Protocol	Time Stamp
0.77398349	0.00122074	0.03970338	0.76894547	0.0454545	0.9999184
0.77398349	0.00122074	0.03970338	0.76894547	0.0454545	0.9999184
0.77398349	0.00122074	0.03970338	0.76894547	0.0454545	0.9999184
0.70400257	0.61087375	0.03970338	0.89669857	0.0454545	0.9999184
0.70400257	0.61087375	0.03970338	0.89669857	0.0454545	0.9999184
0.70400257	0.61087375	0.03970338	0.89669857	0.0454545	0.9999184
0.13904492	0.00675985	0.03970338	0.78223972	0.0454545	0.9999184
0.77398349	0.00122074	0.03970338	0.76894547	0.0454545	0.9999184
0.04551297	0.65411847	0.24540455	0.13738419	0.0454545	0.0455129

Nota: Elaboración Propia. Tomado del Dataset_Attack_2018,

3.3.2. Fase Selección de las técnicas de mayor eficiencia en la detección de intrusos.

En esta investigación, se realizaron estudios de revistas científicas extraídas de la base de datos de la IEEE Xplore, Sciencedirect, Springer, en las que se encontraron técnicas de clasificación de mayor eficiencia en la detección de intrusos.

Para la obtención del top 6 ver **Anexo N° 04**, se realizó todo un proceso, seleccionando las diferentes técnicas de aprendizaje automático, siendo evaluados por las métricas de desempeño (Sensibilidad, exactitud), dándole de preferencia a la métrica Exactitud que mide la proporción de VP (verdaderos positivo), es decir ataque correctamente identificado como ataques" (De la Hoz, De la Hoz, Ortiz, & Ortega, 2012).

Se seleccionaron 2 algoritmos: Multilayer Percepetron y Suppor Vector Machine debido a estos autores (Wanga , Xub, Leec, & Leed, 2017) y a la eficiencia que ha tenido en el campo de la seguridad informática.

En la siguiente Tabla Nº 35 como resultado de los artículos estudiados se procedió a evaluar los algoritmos través de la métrica de Sensibilidad formando un top de los 6 mejores algoritmos y estos fue el resultado de acuerdo a su porcentaje en Sensibilidad.

Tabla 35:

Selección de las técnicas de clasificación con mayor sensibilidad.

Nº	Clasificador	Sensibilidad (%)
1	Multi Layer perceptrón	99.94
2	Random Forest	99,881
3	Naive Bayes	99,8
4	Árbol de decisiones C4.5 / J48	99,64
5	AdaBoost	99,61
6	Support Vector Machines	99.24

Nota: Elaboración Propia. Tomado de ([a]Wanga , Xub, Leec, & Leed, 2017, pág. 30-31), ([b]Chauhan, Kumar, Pundir, & Pilli, 2013, pág. 42), ([c]Tchakoucht & Ezziyyani , 2018, pág. 7), (Mazini, Shirazi, & Mahdavi, 2018, pág. 22).

En la siguiente Tabla Nº 36 como resultado de los artículos científicos estudiados se procedió a evaluar los algoritmos través de la métrica de Exactitud y estos fue el resultado de acuerdo a su porcentaje en Exactitud.

Tabla 36:

Selección de las técnicas de clasificación de mayor exactitud.

	Clasificador	Exactitud (%)
1	MultiLayer perceptron	99.90
2	Random Forest	99.746
3	Naive Bayes	97.12
4	Arbol de decisiones C4.5 / J48	99.7352
5	AdaBoost	98.9
6	Support Vector Machines	99.04

Nota: Elaboración propia. Tomado de ([a]Wanga , Xub, Leec, & Leed, 2017, pág. 30-31), ([b]Chauhan, Kumar, Pundir, & Pilli, 2013, pág. 42), ([c]Almansob & Lomte, 2017, pág. 3), ([d]Muzammil, Qazi, & Ali, 2013, pág. 4), ([e]Mazini, Shirazi, & Mahdavi, 2018, pág. 22).

3.3.3. Fase Implementación de las técnicas de Máquinas de Soporte Vectorial y Perceptrón Multicapa

3.3.3.1. Implementación del algoritmo Máquinas de Soporte Vectorial

Para lograr la realización de la implementación de la técnica Máquinas de Soporte Vectorial, se tomó algunos datos del Dataset_Attack_20018, tomando variables predictivas para explicar el proceso de la técnica mencionada tal y como se muestra en la Tabla Nº 37.

Tabla 37*:*

SVM Características de demostración del Dataset_attack_2018.

CLASE 1 (Exploit)		CLASE 2 (Backdoor)	
Protocol	Destination_Port	Protocol	Destination_Port
131	50379	7	5353
130	44320	6	38507
131	46320	6	80
131	24312	130	7
140	14325	17	53
131	24	120	15
131	46886	17	34447
130	28342	6	17364
80	43809	17	53302

Nota: Elaboración Propia.

Como se puede observar en la Figura Nº 50, en el diagrama de Dispersión de Support Vector Machine, la Clase 1 y la clase 2 representan a los ataques cibernéticos Exploit y Backdoor.

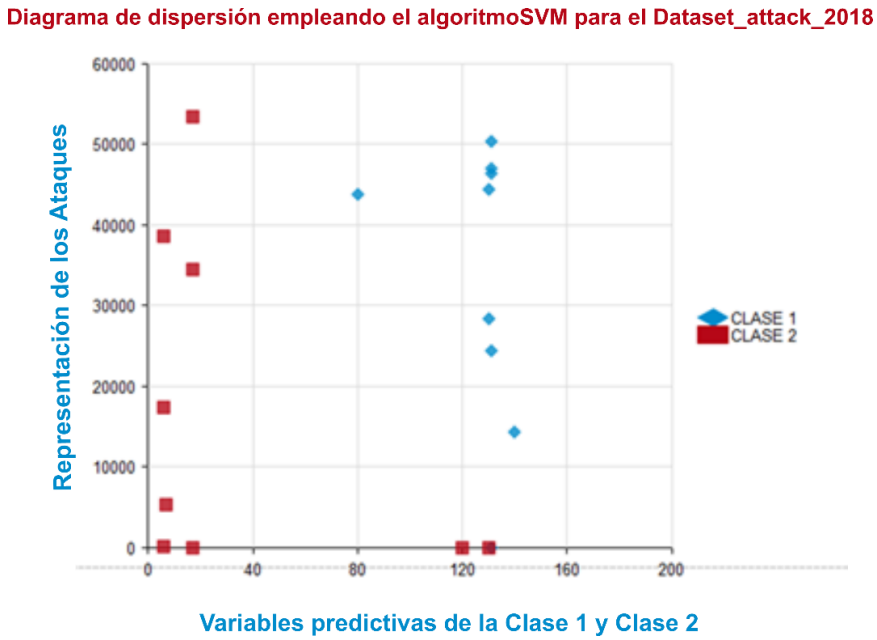

Figura 48: Diagrama de dispersión empleando el algoritmo SVM para el Dataset attack 2018.

Fuente: Elaboración Propia.

En la siguiente Figura Nº 51, La clase 1 y 2 perteneciente a los ataques Exploit y Backdoor, al aplicar el algoritmo Maquinas de Soporte Vectorial se debe encontrar cual es la línea más optima que maximice la separación de estas 2 clases.

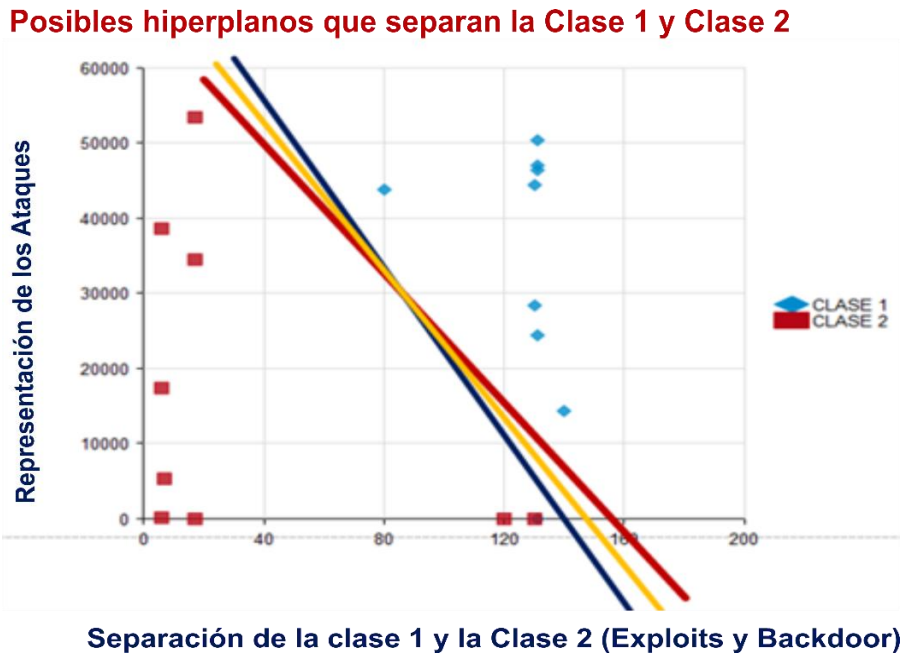

*Figura 49:*Posibles hiperplanos que separan la Clase 1 y Clase 2 (Exploits y Backdoor).

Fuente: Elaboración Propia.

El algoritmo Maquinas de Soporte Vectorial lo primero que hace, es elegir pivotes, que vendrían hacer los vectores de soporte, el cual determina cual sería ángulo de nuestra red, y se hace calculando la distancia euclidiana siguiendo la siguiente ecuación Nº 01 y con ello definimos se define cuáles son las distancias más cortas.

$$d= \sqrt{(x_2-x_{1)}{}^2+(y_2-y_1)^2} \qquad (1)$$

El siguiente proceso realizó, cálculos entre todos los puntos, evaluando cada punto de la clase 1(Exploit) con la clase 2 (Backdoor), definiendo cuales son las distancias más cortas, como se puede observar en las Tablas Nº 38, 39, 40, 41,42, 43, 44, 45, 46 y 47 y en la Tabla Nº 38 vemos que el 131 y el 50379 es un vector característico, el 130 y 44320 es otro vector característico y así

sucesivamente, para simplificar esto, se tiene que sacar de la clase 1 del punto A con toda la clase del punto B, tal y como se presenta en las figuras.

Tabla 38*:*

Evaluación Punto de la Clase 1 (Exploit) con la clase 2.

CLASE 1 (Exploits)			**CLASE 2 (Backdoor)**		
Protocol	**Destination_Port**	**Punto**	**Protocol**	**Destination_Port**	**Punto**
131	50379	A)	7	5353	J)
130	44320	B)	6	38507	L)
131	46320	C)	6	80	M)
131	24312	D)	130	7	N)
140	14325	E)	17	53	O)
131	24	F)	120	15	P)
131	46886	G)	17	34447	Q)
130	28342	H)	6	17364	R)
80	43809	I)	17	53302	S)

Nota: Elaboración Propia

Tabla 39:

Resultado de la simplificación de la Clase 1 con la Clase A para obtener las distancias más cortas

CLASE 1 (Exploit)			**CLASE 2(Backdoor)**			**Distancia**
131	50379	A)	7	5353	J)	45026.17075
131	50379	A)	6	38507	K)	11872.65804
131	50379	A)	6	80	L)	50299.15532
131	50379	A)	130	7	M)	50372.00001
131	50379	A)	17	53	N)	50326.12912
131	50379	A)	120	15	O)	50364.0012
131	50379	A)	17	34447	P)	15932.40785
131	50379	A)	6	17364	Q)	33015.23663
131	50379	A)	17	53302	R)	2925.222214

Nota: Elaboración Propia

Tabla 40*:*

Resultado de la simplificación de la Clase 1 con la Clase B para obtener las distancias más cortas.

CLASE 1(Exploit)			CLASE 2(Backdoor)			Distancia
130	44320	B)	7	5353	J)	38967.1941
130	44320	B)	6	38507	K)	5814.3224
130	44320	B)	6	80	L)	44240.1738
130	44320	B)	130	7	M)	44313
130	44320	B)	17	53	N)	44267.1442
130	44320	B)	120	15	O)	44305.0011
130	44320	B)	17	34447	P)	9873.64664
130	44320	B)	6	17364	Q)	26956.2852
130	44320	B)	17	53302	R)	8982.71078

Nota: Elaboración Propia

Tabla 41:

Resultado de la simplificación de la Clase 1 con la Clase C para obtener las distancias más cortas.

CLASE 1(Exploit)			CLASE 2(Backdoor)			Distancia
131	46320	C)	7	5353	J)	40967.1877
131	46320	C)	6	38507	K)	7813.99987
131	46320	C)	6	80	L)	46240.169
131	46320	C)	130	7	M)	46313
131	46320	C)	17	53	N)	46267.1404
131	46320	C)	120	15	O)	46305.0013
131	46320	C)	17	34447	P)	11873.5473
131	46320	C)	6	17364	Q)	28956.2698
131	46320	C)	17	53302	R)	6982.93062

Nota: Elaboración Propia

Tabla 42:

Resultado de la simplificación de la Clase 1 con la Clase D para obtener las distancias más cortas.

CLASE 1(Exploit)			**CLASE 2(Backdoor)**			**Distancia**
131	24312	D)	7	5353	J)	18959.4055
131	24312	D)	6	38507	K)	14195.5504
131	24312	D)	6	80	L)	24232.3224
131	24312	D)	130	7	M)	24305
131	24312	D)	17	53	N)	24259.2679
131	24312	D)	120	15	O)	24297.0025
131	24312	D)	17	34447	P)	10135.6411
131	24312	D)	6	17364	Q)	6949.12433
131	24312	D)	17	53302	R)	28990.2241

Nota: Elaboración Propia

Tabla 43:

Resultado de la simplificación de la Clase 1 con la Clase E para obtener las distancias más cortas.

CLASE 1(Exploit)			CLASE 2(Backdoor)			Distancia
140	14325	E)	7	5353	J)	8972.98573
140	14325	E)	6	38507	K)	24182.3713
140	14325	E)	6	80	L)	14245.6302
140	14325	E)	130	7	M)	14318.0035
140	14325	E)	17	53	N)	14272.53
140	14325	E)	120	15	O)	14310.014
140	14325	E)	17	34447	P)	20122.3759
140	14325	E)	6	17364	Q)	3041.95283
140	14325	E)	17	53302	R)	38977.1941

Nota: Elaboración Propia

Tabla 44:

Resultado de la simplificación de la Clase 1 con la Clase F para obtener las distancias más cortas.

CLASE 1(Exploit)			**CLASE 2(Backdoor)**			**Distancia**
131	24	F)	7	5353	J)	5330.44248
131	24	F)	6	38507	K)	38483.203
131	24	F)	6	80	L)	136.9708
131	**24**	**F)**	**130**	**7**	**M)**	**17.0293864**
131	24	F)	17	53	N)	117.630778
131	**24**	**F)**	**120**	**15**	**O)**	**14.2126704**
131	24	F)	17	34447	P)	34423.1888
131	24	F)	6	17364	Q)	17340.4505
131	24	F)	17	53302	R)	53278.122

Nota: Elaboración Propia

Tabla 45:

Resultado de la simplificación de la Clase 1 con la Clase F para obtener las distancias más cortas.

CLASE 1(Exploit)			CLASE 2(Backdoor)			Distancia
131	46886	G)	7	5353	J)	41533.1851
131	46886	G)	6	38507	K)	8379.93234
131	46886	G)	6	80	L)	46806.1669
131	46886	G)	130	7	M)	46879
131	46886	G)	17	53	N)	46833.1387
131	46886	G)	120	15	O)	46871.0013
131	46886	G)	17	34447	P)	12439.5224
131	46886	G)	6	17364	Q)	29522.2646
131	46886	G)	17	53302	R)	6417.0127

Nota: Elaboración Propia

Tabla 46:

Resultado de la simplificación de la Clase 1 con la Clase H para obtener las distancias más cortas.

CLASE 1(Exploit)			CLASE 2(Backdoor)			Distancia
130	28342	H)	7	5353	J)	22989.329
130	28342	H)	6	38507	K)	10165.7563
130	28342	H)	6	80	L)	28262.272
130	28342	H)	130	7	M)	28335
130	28342	H)	17	53	N)	28289.2257
130	28342	H)	120	15	O)	28327.0018
130	28342	H)	17	34447	P)	6106.04569
130	28342	H)	6	17364	Q)	10978.7003
130	28342	H)	17	53302	R)	24960.2558

Nota: Elaboración Propia.

Tabla 47:

Resultado de la simplificación de la Clase 1 con la Clase I para obtener las distancias más cortas.

CLASE 1			CLASE 2			Distancia
80	43809	I)	7	5353	J)	38456.0693
80	43809	I)	6	38507	K)	5302.51638
80	43809	I)	6	80	L)	43729.0626
80	43809	I)	130	7	M)	43802.0285
80	43809	I)	17	53	N)	43756.0454
80	43809	I)	120	15	O)	43794.0183
80	43809	I)	17	34447	P)	9362.21197
80	43809	I)	6	17364	Q)	26445.1035
80	43809	I)	17	53302	R)	9493.20905

Nota: Elaboración Propia

De los resultados obtenidos de los cálculos de todas las distancias obtenidas de los puntos A, B, C, D, E, F, G, H, I con todos los puntos de la clase 2, procediéndose después a realizar la búsqueda de la distancia más pequeña con los 2 vectores de soporte, el cual corresponden a los puntos F) y O) con un valor de 14.2126704.

Luego, se seleccionan los puntos de los valores F) y el valor más pequeño de O), tomándose de estos dos el valor más corto, que en este caso se presenta el valor obtenido en el punto F) y M) con 17.0293864. Ver Figurar Nº 50.

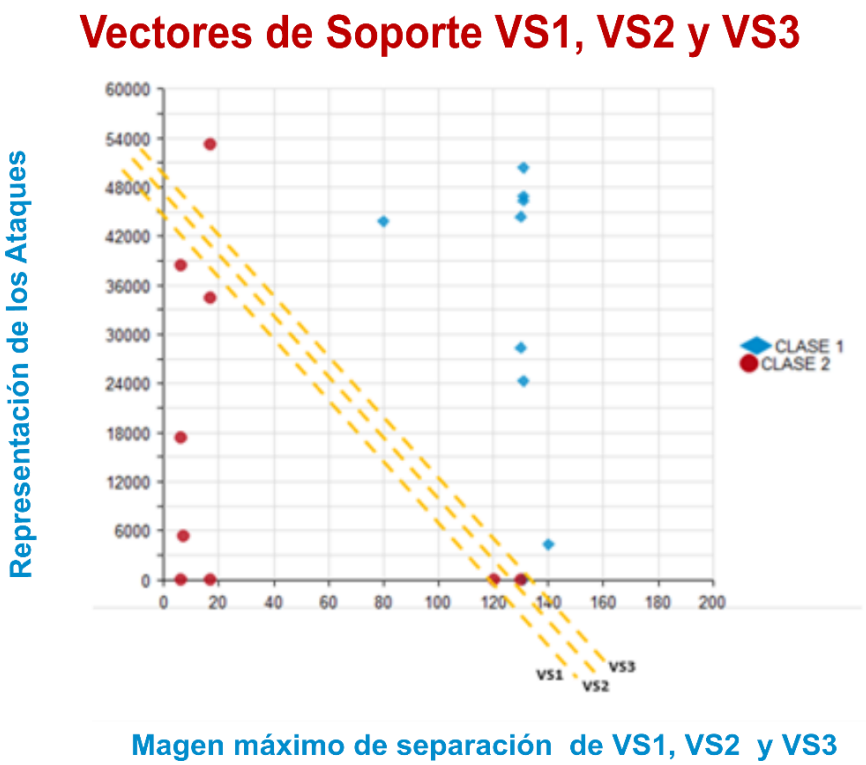

Figura 50:Vectores de soporte VS1, VS2 y VS3.

Fuente: Elaboración Propia.

Como se puede observar en la Figura Nº 61, tenemos a los vectores de soporte Vs1, Vs2 y Vs3 identificados con sus respectivas coordenadas, luego de haberlo graficado, se realiza los siguientes pasos para la representación de los vectores de soporte en su coordenada.

Paso Nº 01: Representación de Soporte por su respectiva coordenada.

$$Vs1= \begin{pmatrix} 131 \\ 24 \end{pmatrix}$$

$$Vs2= \begin{pmatrix}120\\15\end{pmatrix}$$

$$Vs3= \begin{pmatrix}130\\7\end{pmatrix}$$

Paso Nº 02: Se le agrega el valor de 1 al final del arreglo o del vector de soporte, este 1 apoya como un inicio del ángulo de la pendiente que la recta va a tomar.

$$Vs1= \begin{pmatrix}131\\24\end{pmatrix} \longrightarrow Vs1= \begin{pmatrix}131\\24\\1\end{pmatrix}$$

$$Vs2= \begin{pmatrix}120\\15\end{pmatrix} \longrightarrow Vs2= \begin{pmatrix}12O\\15\\1\end{pmatrix}$$

$$Vs3= \begin{pmatrix}130\\7\end{pmatrix} \longrightarrow Vs3= \begin{pmatrix}130\\7\\1\end{pmatrix}$$

Paso N° 03: Se aplicó la ecuación lineal para cada clase:

$$\propto_1 S_1.S_1+ \propto_2 S_2.S_1+ \propto_3 S_3.S_1= -1 \quad (2)$$

$$\propto_1 S_1.S_2+ \propto_2 S_2.S_2+ \propto_3 S_3.S_2= -1 \quad (3)$$

$$\propto_1 S_1.S_3+ \propto_2 S_2.S_3+ \propto_3 S_3.S_3= +1 \quad (4)$$

Paso N° 04: Se comenzó a reemplazar los valores de la ecuación lineal, tal y como se presenta a continuación.

$$\propto_1 \begin{pmatrix}131\\24\\1\end{pmatrix}. \begin{pmatrix}131\\24\\1\end{pmatrix} +\propto_2 \begin{pmatrix}120\\15\\1\end{pmatrix}. \begin{pmatrix}131\\24\\1\end{pmatrix} +\propto_3 \begin{pmatrix}130\\7\\1\end{pmatrix}. \begin{pmatrix}131\\24\\1\end{pmatrix} = -1$$

$$\propto_1 \begin{pmatrix}131\\24\\1\end{pmatrix}. \begin{pmatrix}120\\15\\1\end{pmatrix} +\propto_2 \begin{pmatrix}120\\15\\1\end{pmatrix}. \begin{pmatrix}120\\15\\1\end{pmatrix} +\propto_3 \begin{pmatrix}130\\7\\1\end{pmatrix}. \begin{pmatrix}120\\15\\1\end{pmatrix} = -1$$

$$\propto_1 \begin{pmatrix}131\\24\\1\end{pmatrix}. \begin{pmatrix}130\\7\\1\end{pmatrix} +\propto_2 \begin{pmatrix}120\\15\\1\end{pmatrix}. \begin{pmatrix}130\\7\\1\end{pmatrix} +\propto_3 \begin{pmatrix}130\\7\\1\end{pmatrix}. \begin{pmatrix}130\\7\\1\end{pmatrix} = -1$$

Paso Nº 05: Se procedió a simplificar la ecuación lineal obteniendo una reducción de la ecuación con los siguientes resultados.

$$\propto_1 17737+\propto_2 16081+\propto_3+17199= -1 \quad (5)$$

$$\propto_1 16081 + \propto_2 14626 + \propto_3 15706 = -1 \quad (6)$$

$$\propto_1 17199 + \propto_2 15706 + \propto_3 16950 = -1 \quad (7)$$

Paso Nº 06: Como se puede observar tenemos un sistema de ecuaciones con 3 ecuaciones y 3 incógnitas, en este caso se aplicará el método de la regla de Kramer para obtener los valores de $\propto_1, \propto_2, \propto_3$ y se presenta en la siguiente Figuras Nº 51,52,53,54 y 55.

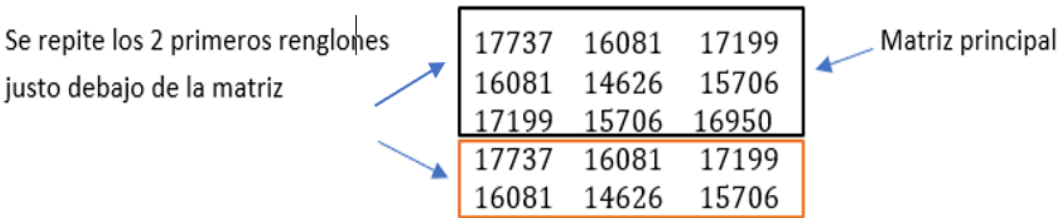

Figura 51: Representación del método de Regla.

Fuente: Elaboración Propia.

Paso Nº 7: Se resolvió la ecuación de Kramer con sus respectivos incrementos, obteniendo los siguientes valores de $\Delta, \Delta_1, \Delta_2, \Delta_3$ como se observa en las Tablas Nº Tablas Nº 38, 39, 40, 41,42, 43, 44, 45, 46 y 47 y Figura Nº 64.

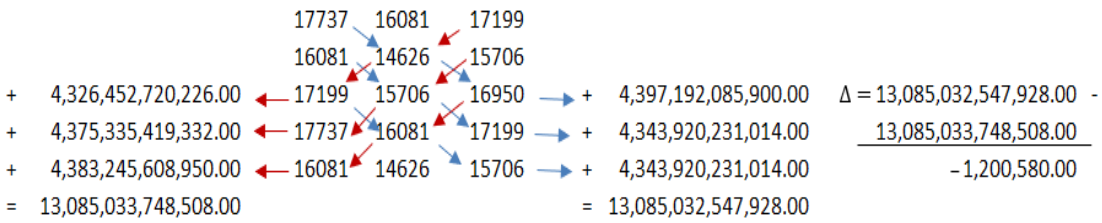

Figura 52: Incremento de Ecuación de Kramer.

Fuente: Elaboración Propia.

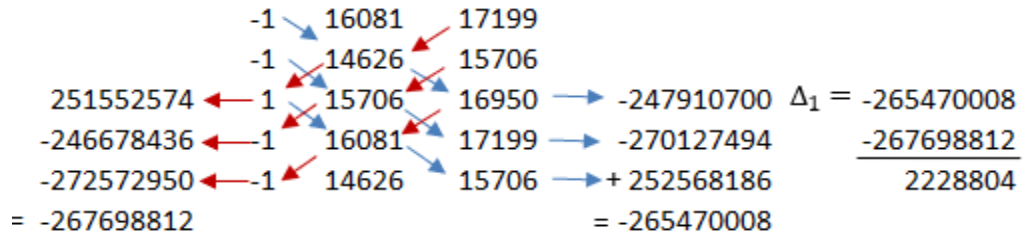

Figura 53: Primer Incremento aplicando la Ecuación de Kramer.

Fuente: Elaboración Propia.

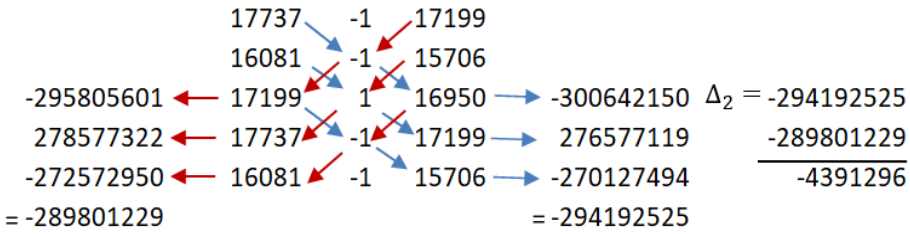

Figura 54: Segundo Incremento aplicando la Ecuación de Kramer.

Fuente: Elaboración Propia.

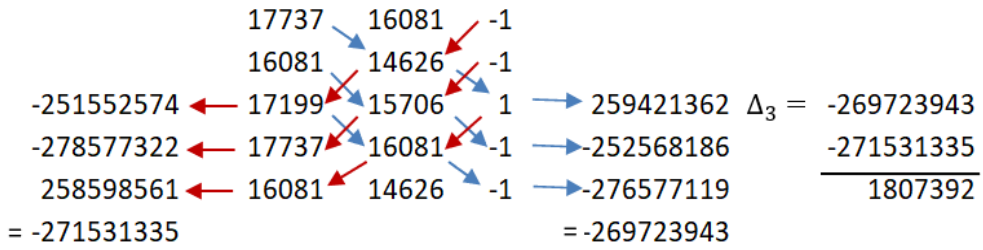

Figura 55: Tercer Incremento de Ecuación de Kramer.

Fuente: Elaboración Propia.

Paso Nº 08: Se realizó el cálculo del sistema de ecuaciones obteniendo como resultados los valores de $\propto_{1,}\propto_{2,}\propto_{3}$.

$$\propto_1 = -128$$

$$\propto_2 = 3.657645471$$

$$\propto_3 = -1.50542374$$

Paso Nº 09: Luego de haber realizado el cálculo del sistema de ecuaciones y determinando los vectores de soporte (Vs1, Vs2, Vs3), se procederá a calcular con la siguiente ecuación el hiperplano más óptimo que diferencie a las dos clases.

$$\hat{W} = \propto_1 . Sv_1 + \propto_2 . Sv_2 + \propto_3 . Sv_3 = \begin{bmatrix} wx \\ wy \\ wb \end{bmatrix} \quad (8)$$

Paso Nº 10: De la ecuación Nº 08, se remplazaron los valores de $\propto_{1,}\propto_{2,}\propto_{3}$ y valores de los vectores de soporte Vs1, Vs2 y Vs3, obteniendo el valor de $\hat{W}$

$$\hat{W} = -128. \begin{bmatrix} 131 \\ 24 \\ 1 \end{bmatrix} + 3.657645471. \begin{bmatrix} 120 \\ 15 \\ 1 \end{bmatrix} + -1.50542374. \begin{bmatrix} 130 \\ 7 \\ 1 \end{bmatrix}$$

$$\hat{W}= \begin{bmatrix} 17050.00 \\ -3118.44 \\ 130.45 \end{bmatrix}$$

Por consiguiente, la ecuación del hiperplano más óptimo de separación de la clase, representando al ataque Exploit y Clase 2 representa al ataque Backdoor y expresa e n la siguiente ecuación y = wx +b, con w= $\left(\frac{17050}{3118}\right)$ y un desplazamiento de b=130.45 y se puede observar en la siguiente Figura Nº 56 con el hiperplano más óptimo.

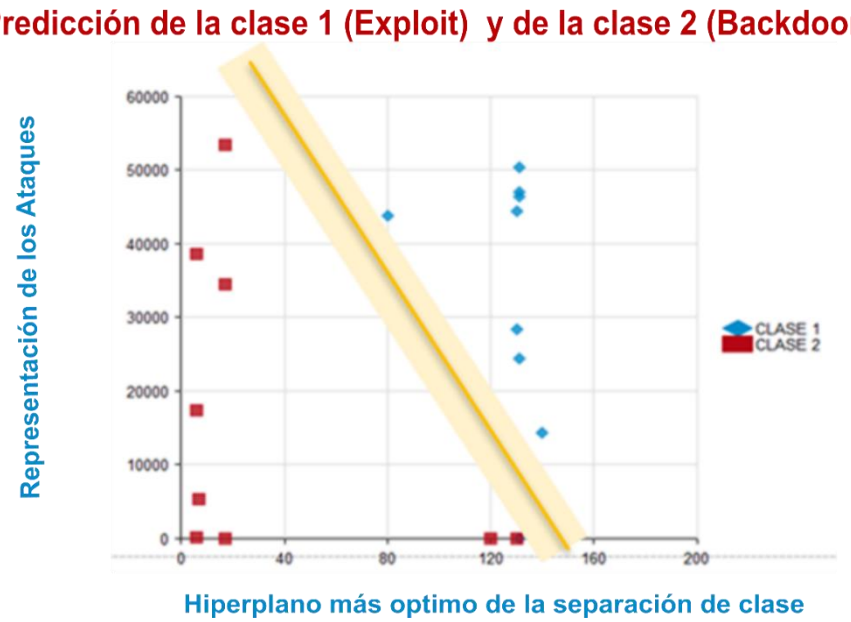

Figura 56: Predicción de la Clase 1 (Exploit) y de la Clase (Backdoor).

Fuente: Elaboración Propia.

Para la realización de la implementación del algoritmo Maquinas de Soporte Vectorial, se procedió a obtener las variables dependientes y variables independientes del conjunto de datos.

Después se procedió a dividir los datos en el conjunto de entrenamiento y el conjunto de pruebas, para ellos importamos la librería de Sci-Kit Learn, también importamos la función train_test_split para dividir los datos, en la función utilizamos las variables x,y, más dos parámetros adicionales : test_size y random_state. Con el primero (test_size) indicamos que el tamaño para el conjunto de prueba, guardados en las variables x_test y y_test tendrán 30% del conjunto total, es decir 186394 registros, dado que el conjunto total, es de

621311 y los 186394 seleccionados estarán escogidos de manera aleatoria de entre esos 621311, dado que el último parámetro de la función es random_state=0.Luego de eso se llama al clasificador LinearSVC que es el algoritmo Máquinas de Soporte Vectorial, utilizando el enfoque uno contra el resto.

Este algoritmo toma como entrada dos matrices: una matriz X que son las muestras de entrenamiento y una matriz y de etiquetas de clases (tamaño: [n_nuestras,n_ características]) y se presentan en la figura Nº 57.

Entrada $\{(\overline{X}, Y_1), \ldots\ldots (\overline{X}_m, Y_m)\}$

Inicializa: $\overline{T} = \overline{0}, \ldots\ldots, \overline{T}_m = \overline{0};$

Loop

1. **Elija un ejemplo p.**
2. **Calcule las constantes para el problema reducido:**

   . $A_p = K(\overline{X}_p, \overline{X}_p)$

   . $B_p = \sum_{i \neq p} K(\overline{X}_i, \overline{X}_p)\ \overline{T}_i - \boldsymbol{\beta}\overline{1}_{yp}$

3. **Establezca $\overline{T}_p$ para ser la solución del problema reducido:**

$$\min_{T_p} Q(\overline{T}_p) = \frac{1}{2} A_p (\overline{T}_p . \overline{T}_p) + \overline{B}_p . \overline{T}_p$$

$$\text{Subjeto a} : \overline{T}_p \leq \overline{1}_{yp} \text{ and } \overline{T}_p . \overline{1} = 0$$

Salida: $H(\bar{X}) = arg \max_{r=1}^{k} \left\{ \sum_i T_{i,r} K(\overline{X}, \overline{X}_i) \right\}$

Figura 57: Algoritmo de Máquina de Soporte Vectorial Multiclase.
Fuente: (Crammer & Singer , 2001)

Para evidenciar la implementación del algoritmo Support Vector Machine Multicapa de una forma automatizada, se ha utilizado la librería Scikit-learn implementada en el lenguaje de implementación en Python ver **Anexo N°12.**

3.3.3.2. Implementación del Algoritmo Perceptrón Multicapa.

Para lograr el objetivo de la implementación del algoritmo Perceptrón Multicapa, se tomó algunos datos del Dataset_Attack_20018, tomando variables predictivas para explicar el proceso de la técnica mencionada tal y como se muestra en la Tabla Nº 48 el ejemplo de la Implementación con los datos del dataset_Attackt_2018.

Tabla 48:

MLP Características de demostración del Dataset_Attack_2018.

Source_IP	Source_Port	Etiqueta
0	1	Phishing
1	1	Phishing
0	0	Phishing
1	0	Phishing
0	1	Backdoor
0	0	Backddoor
1	0	Backddoor
1	1	Backddoor

Nota: Elaboración Propia.

Como se puede observar en la Tabla Nº 49 tenemos a las entradas y salidas deseadas del conjunto de Datos Dataset_Attack_2018.

Tabla 49*:*

Representación Entradas y Salidas deseadas del Dataset_Attack_2018.

ENTRADAS		**SALIDAS DESEADAS**
Source_IP	**Source_Port**	**Etiqueta**
0	**1**	**1**
1	**0**	**1**
0	**0**	**1**
1	**1**	**1**
0	**1**	**0**
0	**0**	**0**
1	**0**	**0**
1	**1**	**0**

Nota: Elaboración Propia.

Luego de haber presentado las entradas y salidas deseadas se procederá a realizar el cálculo de la propagación hacia adelante en el siguiente Paso N° 01.

Paso N° 01: Propagación hacia adelante

En este paso tenemos a las siguientes entradas: Source_IP que representa a X_1, Source_Port representado por X_2.

La salida deseadas tenemos a las etiquetas representado por 0 y 1, determinando asi la salida de la red neuronal.

1. Capa de Entrada: En este caso hemos utilizado el primer vector de entrenamiento

Vector Entrada 1: X_1 = 0, X_2= 1

2. Capa oculta: Se realizó un conjunto de operaciones para calcular la salida y para ello se hizo uso de la siguiente ecuaciones: Salida de la neurona N° 09 y la función de activacion sigmoide en la ecuación N° 10.

$$Y = F(\sum_{i=1}^{n} W_i X_i) \quad (9)$$

$$F(x) = \frac{1}{1+e^{-x}} \quad (10)$$

Paso N° 02: Luego se inicializan y se determian los pesos de forma aletoria ejemplo:W_{11} = 0.1, W_{12} =0.8, W_{21} =0.6, W_{22} =0.3

$$W=\begin{bmatrix} 0.1 & 0.6 \\ 0.8 & 0.3 \end{bmatrix}$$

Paso N° 03 : Se realizó los calculos utilizando las siguiente ecuaciones 11 y 12.

$$H1 = X_1 * w_{11} + X_2 * w_{12} \quad (11)$$

$$H2 = X_1 * w_{21} + X_2 * w_{22} \quad (12)$$

Paso N° 04: Se reemplaza los valores de las ecuaciones de las ecuaciones 11 y 12 y se aplica las funcion de activación para obtener el calculo de las Neuronas 1 y 2 y las salidas de las neuronas.

Neurona O1:

Entrada: = (0.1) * (0) + (0.8) * (1) = 0.8

Salida = $\frac{1}{1+e^{-0.8}}$ = 0.689974

Neurona O2:

Entrada: = (0.6)*(1)+ (0.3)*(0) = 0.6

Salida = $\frac{1}{1+e^{-0.6}}$ = 0.645656

Como se pudo observar en este paso tenemos los resultados de las salidas de las neuronas, por lo tanto tenemos a las salidas de las neuronas con los valores de 0.689974 y 0.645656.

Paso N° 05: A continuación se calcula en la tercera capa de salida la Neurona S1, el cual entra 2 señales uno de la Neurona O1 y la otra de la Neurona O2, dada por la ecuación N° 13.

$$NS1 = O_1 * W_{21} + O_2 * W_{22} \quad (13)$$

Paso N° 06: Se reemplazó los valores de la ecuación de las 2 señales de salida de la Neurona O1 y Neurona 02, multiplicado por los pesos que son de forma aleatoria.

Capa de Salida O1:

Entrada= (0.2) * (0.689974) + (0.4) * (0.645656) = 0.396257

Salida **=** $\frac{1}{1+e^{-0.396257}}$ = 0.597

Propagación hacia atrás – determinando los nuevos pesos de la Red Neuronal.

Paso N° 07: Como se puede observar a partir de ahora en adelante, veremos la propagación hacia atrás, el cual tiene como propósito determinar los nuevos pesos de la red Neuronal, para ello haremos uma propagación en la capa de salida y en la capa oculta.

Capa de Salida (Neurona S1) : Aqui se calcula el error cometido

Error Real = 1 - 0.598 = 0.402

Por consiguiente el error cometido que obtenemos es 0.402.

Paso N° 08: En este paso **s**e calcula Δ_k, para obtener la varación que debe sufrir los pesos de la capa de salida y se calcula con la siguiente ecuación N° 14 y se reemplaza los valores.

$$\Delta_k = g'(S_k)(D_k - S_k) \quad (14)$$

$$\Delta_k = 0.597 * (1 - 0.597) * (1 - 0.598) = 0.0967$$

El valor 0.0967 es el resultado de la variación que deben sufir los pesos de la capa de salida.

Paso N° 09: A continuacion se procedió a calcular los nuevos pesos de la capa de salida haciendo uso de la ecuación N° 15 y se reemplazo los valores.

$$W_{ij} = W_{ij} + \alpha E_j \Delta_k \quad (15)$$

$$W2 = \begin{vmatrix} 0.2 + 0.25 * 0.689974 * 0.0967 \\ 0.4 + 0.25 * 0.645656 * 0.0967 \end{vmatrix} = \begin{vmatrix} 0.216 \\ 0.415 \end{vmatrix}$$

Como se puede observar, luego de haber realizado el caculo se dedujo que hubo un incremento en los pesos con valores obtenidos de 0.316 y 0.516.

Paso N° 10: Se procedió a realizar el calculo para la capa Oculta, primeramente calcularemos el error estimado Δ_1 para la Neurona O1, utilizando para ello la ecuación N° 16 y reemplazamos valores.

$$\Delta_k = g'(S_J \sum_{k=1}^{p} (W_{jk} \Delta_k) \quad (16)$$

Error Estimado:

Δ_k= 0.689974*(1-0.689974)* 0.216*0.0967 = 0.0044

Luego de haber calculado el el error estimado tenemos um valor de 0.000668

Paso N° 11: Se procede a calcular los nuevos pesos utilizando la ecuación N° 17, reemplazando los valores para obtener los nuevos pesos de la Neurona O1.

$$W_{ij} = W_{ij} + \alpha E i_j \Delta_j \quad (17)$$

Nuevos Pesos:

$$W1 = \begin{vmatrix} 0.1+0.25*0* 0.0044 \\ 0.8+0.25*1* 0.0044 \end{vmatrix} = \begin{vmatrix} 0.100 \\ 0.801 \end{vmatrix}$$

Se obtuvo los nuevos pesos de la Neurona O1 obteniendo los resultados de 0.100 y 0.801.

Paso N° 12: En este paso se procede a calcular Δ_2, obteniendo el error estimado de la Neruona 02, , utilizando para ello la ecuación N° 15, expresion mencionada líneas arriba y se reemplaza valores.

Error Estimado:

$$\Delta_k = 0.645656 * (1 - 0.645656) * 0.415 * 0.0967 = 0.0091$$

Paso N° 13: Finalmente se procedió a calcular los nuevos pesos utilizando la ecuación N° 16 y reemplazando los valores para obtener los nuevos pesos de la Neurona O2.

Nuevos Pesos

$$W1 = \begin{vmatrix} 0.6 + 0.25 * 1. * 0.0091 \\ 0.3 + 0.25 * 0 * 0.0091 \end{vmatrix} = \begin{vmatrix} 0.602 \\ 0.300 \end{vmatrix}$$

Con esto se habria hecho el cálculo de los nuevos pesos de W2 y W1, ya que el propósito de progagación hacias atras consiste em calcular estos pesos.

$$W1 = \begin{bmatrix} 0.1 & 0.6 \\ 0.8 & 0.3 \end{bmatrix}$$

Para evidenciar la implementación del algoritmo Perceptrón Multicapa de una forma automatizada, se ha utilizado la librería Scikit-learn implementada en el lenguaje de programación de Python ver **Anexo N°13.**

La arquitectura del esquema de trabajo consta de tres capas: 1 capa de entrada, 1 capas oculta y una capa de salida: Ver Figura Nº 62.

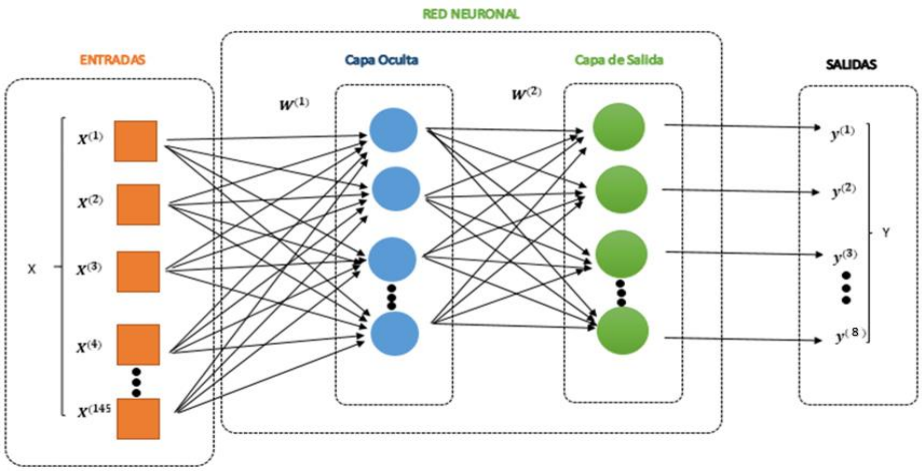

Figura 58: Esquema de Red de Trabajo del Algoritmo Perceptrón Multicapa. Fuente: Elaboración Propia.

Para esta investigación en la capa de entrada ingresan el vector con las 145 características, En la capa oculta se trabajó con 100 neuronas de la capa oculta, en la capa de salida se trabajó con 8 salidas.

Este algoritmo tiene dos fases, la primera fase: propagación hacia delante y la otra fase: propagación hacia atrás, el cual sirve para determinar los nuevos pesos de la red Neuronal. En la capa de entrada se presenta los patrones de entrada a la red neuronal, en este caso las neuronas o patrones de entrada son los datos entrenamiento que tiene el 70%, conteniendo las 145 características. características y las salidas deseadas que son las variables objetivas en este caso ver **Anexo N° 09.**

Durante la primera la capa de entrada es presentada a la red y se propaga a través de las capas hasta llegar a la capa de salida que son las variables objetivas. Luego de haber obtenidos los valores de salida de la red, se inicia la segunda fase, comparándose estos valores con la salida esperada para obtener el error. También se ajustan los pesos de la última capa proporcionalmente al error. Se pasa a la capa anterior con una retropopagación del error, ajustando los pesos y continuando con este proceso hasta llegar a la primera capa. De esta manera se cambian los pesos de las conexiones de la red para cada patrón de aprendizaje del problema, del que conocíamos su valor de entrada y la salida deseada que debería generar la red ante dicho patrón.

3.3.4. Fase de Evaluación de las técnicas de Aprendizaje Automático.

En esta última fase de evaluación, se evaluaron las técnicas de aprendizaje automático aplicadas en esta investigación, los algoritmos evaluados fueron las Máquinas de Soporte Vectorial y Perceptrón Multicapa, empleando para ello las métricas de desempeño (exactitud, precisión, sensibilidad y especificidad), el tipo de evaluación fue realizada con la clasificación multiclase, con 8 clases de salida. Por otro lado, también se evaluó el tiempo de entrenamiento, para ajustarse a los datos de entrenamiento en la creación de los modelos de las técnicas mencionadas. (Belouch, El Hadaja, & Idhammadb, 2018).

Esta investigación empleo la matriz de confusión multiclase, para la obtención de la clase a la que pertenece cada tipo de ataque cibernético, obteniendo de la matriz los resultados de las predicciones que estas correctas y las que se encuentran erradas.

3.3.4.1. Evaluación del Algoritmo Maquinas de Soporte Vectorial.

En la tabla Nº 50, se obtuvo los resultados de la matriz de confusión ver **Anexo N° 10**, se deduce que la clase Real de Backdoor de 699 registros, el modelo ha clasificado incorrectamente 695 registros Exploits y clasificó correctamente 4 registros Backdoor. De la clase real Exploits de 13395 registros, el modelo ha clasificado correctamente todos los registros como Exploits. De le clase real HTTP-Flood de 1237 registros, el modelo ha clasificado incorrectamente 69 registros SIDDOS, también clasificó correctamente 1168 registros HTTP-Flood.

De la clase real Phishing de 212 registros, el modelo ha clasificado correctamente todos los registros Phishing. Así mismo de la clase real Ransomware, el modelo ha clasificado correctamente todos los registros Ransomware. De la clase real SIDDOS de 2008 registros el modelo ha clasificado incorrectamente 1 registro HTTP-Flood, 116 registros UDP-Flood, clasifico correctamente 1891 registros SIDDOS. De la clase real Smurf de 3809 registros, el modelo ha clasificado incorrectamente 18 registros HTTP-Flood, 140 registros SIDDOS y 2344 registros UD-Flood, también clasificó correctamente 1307 registros Smurf, Así mismo de la clase real UDP-Flood de 60385 registros, el modelo ha clasificado incorrectamente 4 ataques HTTP-Flood, 7 registros SIDDOS, 44 registros Smurf, y clasificó correctamente 60330 registros UDP-Flood, llegando a la conclusión que este modelo tuvo muchas equivocaciones con respecto a sus clases reales.

Tabla 50:

Resultados de la matriz de confusión o error del Algoritmo Máquinas de Soporte Vectorial.

Clase	FN	FP	TN	TP
Backdoor	695	1	185663	4
Exploits	0	699	172269	13395
HTTP-Flood	69	23	185103	1168
Phishing	0	0	186151	212
Ransomware	0	0	81745	104618
SIDDOS	117	216	184139	1891
Smurf	2502	44	182510	1307
UDP-Flood	55	2460	123518	60330

Nota: Elaboración Propia.

En la tabla Nº 51, se obtuvo los resultados de la evaluación de la eficiencia del algoritmo Maquinas de Vectores de Soporte Vectorial, utilizando para ello las métricas de exactitud, precisión, sensibilidad y especificidad. Las métricas que se obtuvieron son por clase, porque pertenece a una clasificación clase, también se logró sacar el promedio general sus métricas. La exactitud promedio es del 98. %, precisión es del 97%, la sensibilidad del 63% y la especificidad del 96 %. El tiempo de entrenamiento del algoritmo Máquinas de Soporte Vectorial tuvo una duración de 114 segundos ver **Anexo Nº 12.**

Tabla 51:

Resultados de la matriz de confusión o error del Algoritmo Máquinas de Soporte Vectorial.

Clase	Exactitud	Precisión	Sensibilidad	Especificidad	Tiempo Entren.
Backdoor	0.99625	0.75555	0.0	1.0	
Exploits	0.99625	0.9504	1.0	0.99596	
HTTP-Flood	0.99951	0.98069	0.94422	0.99988	
Phishing	1.0	1.0	1.0	1.0	
Ransomware	1.0	1.0	1.0	1.0	114.83
SIDDOS	0.99821	0.89748	0.94173	0.99883	segund
Smurf	0.98634	0.96743	0.34313	0.99976	os
UDP-Flood	0.9865	0.96082	0.99909	0.98047	
Promedio	**0.98**	**0.96**	**0.63**	**0.97**	

Nota: Elaboración Propia.

En la Figura Nº 59, se calculó la exactitud para cada clase, utilizando los valores de la tabla Nº 51, mostrando los resultados de Exactitud del Algoritmo Maquinas de Soporte Vectorial, obtenidos de la matriz de confusión Multiclase.

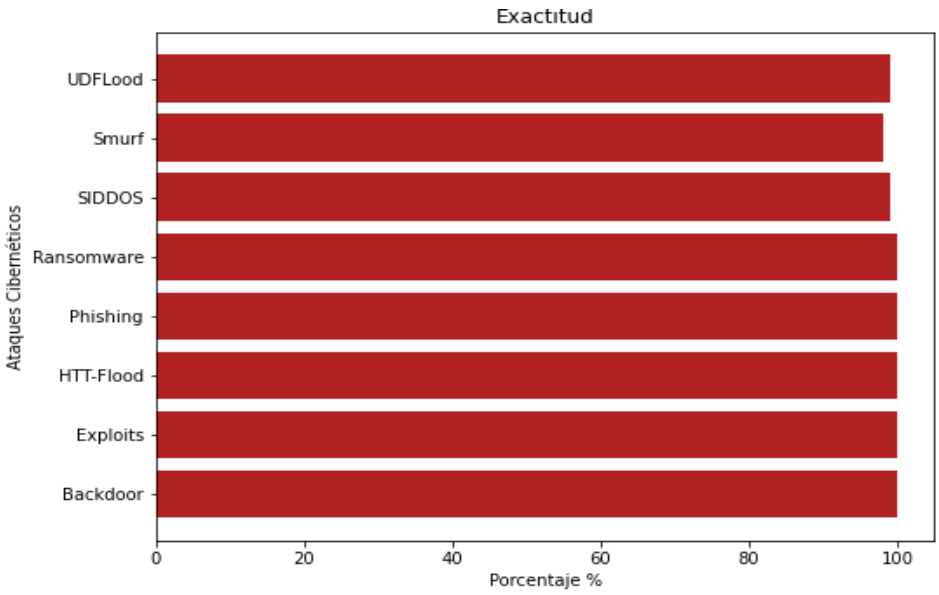

Figura 59: Exactitud del Algoritmo Máquinas de Soporte Vectorial.
Fuente: Elaboración Propia.

De la Figura Nº 60, se calculó la precisión para cada clase, utilizando los valores de la tabla Nº 51, mostrando los resultados de precisión del Algoritmo Maquinas de Soporte Vectorial, obtenidos de la matriz de confusión Multiclase.

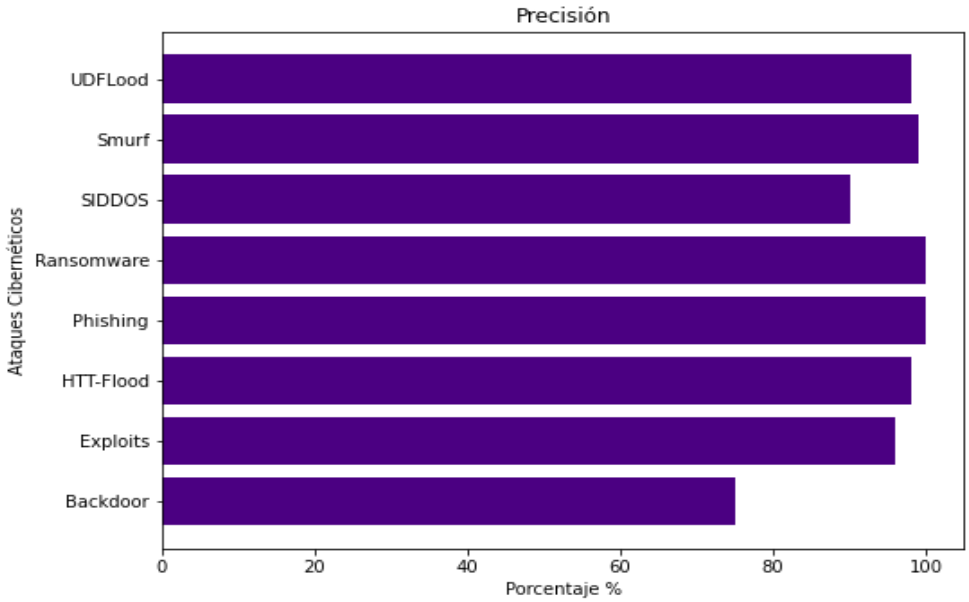

Figura 60: Precisión del Algoritmo Máquinas de Soporte Vectorial.

Fuente: Elaboración Propia.

De la Figura Nº 61, se calculó la sensibilidad para cada clase, utilizando los valores de la tabla Nº 51, mostrando los resultados de sensibilidad del Algoritmo Maquinas de Soporte Vectorial, obtenidos de la matriz de confusión Multiclase.

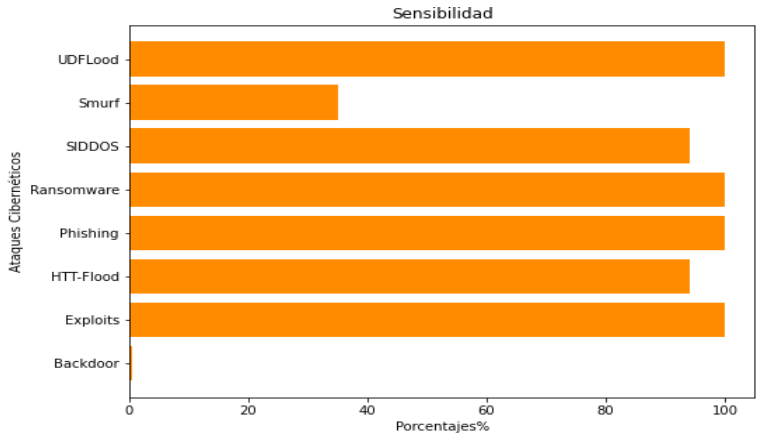

*Figura 61:*Precisión del Algoritmo Máquinas de Soporte.

Fuente: Elaboración Propia.

En la siguiente a Figura Nº 62, se calculó la especificidad para cada clase, utilizando los valores de la tabla Nº 51, mostrando los resultados de especificidad del Algoritmo Maquinas de Soporte Vectorial, obtenidos de la matriz de confusión Multiclase.

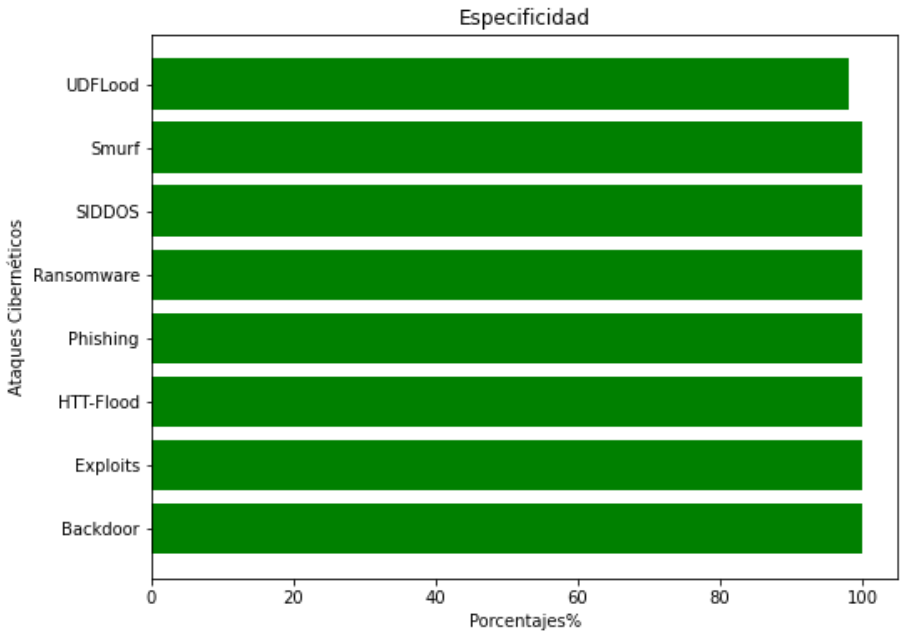

Figura 62: Especificidad del Algoritmo Máquinas de Soporte Vectorial.

Fuente: Elaboración Propia.

3.3.4.2. Evaluación del Algoritmo Perceptrón Multicapa.

De la tabla Nº 52 se obtuvo los resultados de la matriz de confusión ver **Anexo N° 11**, se deduce que la clase Real de Backdoor de 699 registros, el modelo ha clasificado incorrectamente 694 registros Exploits y correctamente clasifico 5 ataques Backdoor. De la clase Real Exploits de 13395 registros, el modelo ha clasificado incorrectamente 1 registro Backdoor y correctamente clasifico 13394 registros Exploits. Así mismo de la clase Real HTTP-Flood de 1237 registros, el modelo ha clasificado incorrectamente 69 registros SIDDOS y clasifico correctamente 1168 registros HTTP-Flood. De la clase real Phishing de 212 registros el modelo ha clasificado correctamente todos sus registros como Phishing. De la clase real Ransomware de 104618 registros, el modelo ha clasificado correctamente 104618 registros como ataque Ransomware. De la clase real SIDDOS de 2008 registros, el modelo ha clasificado incorrectamente 1 registro Smurf y 116 registros UDP-Flood. De la clase real

Smurf de 3688 registros, el modelo ha clasificado incorrectamente 121 registros SIDDOS y 2358 registros UDP-Flood y clasifico correctamente 1318 como ataque Smurf. De la clase real UDP-Flood de 60385 registros, el modelo ha clasificado incorrectamente 19 registros Smurf y clasificó correctamente 60366 registros UDP-Flood.

Tabla 52:

Resultados de la matriz de confusión o error del Algoritmo Perceptrón Multicapa.

Clase	FN	FP	TN	TP
Backdoor	694	1	185663	5
Exploits	1	694	172274	13394
HTTP-Flood	69	12	185114	1168
Phishing	0	0	186151	212
Ransomware	0	0	81745	104618
SIDDOS	117	190	184165	1891
Smurf	2491	20	182534	1318
UDP-Flood	19	2474	123504	60366

Nota: Elaboración Propia.

En la tabla Nº 53, se obtuvo los resultados de la evaluación de la eficiencia del algoritmo Perceptrón Multicapa utilizando para ello las métricas de exactitud, precisión, especificidad y sensibilidad. Las métricas que se obtuvieron son por clase, porque pertenece a una clasificación, también se logró sacar el promedio general sus métricas. La exactitud promedio es del 99 %, precisión del 98%, la sensibilidad alcanzó el 67% y la especificidad del 98%. El tiempo de entrenamiento del algoritmo Perceptrón Multicapa duró 141 segundos ver **Anexo Nº 13**. Determinando que este algoritmo es más eficiente tanto en la exactitud, precisión y especificidad.

Tabla 53:

Resultado de las métricas de desempeño evaluando la eficiencia del Algoritmo Perceptrón Multicapa.

Clase	Exactitud	Precisión	Sensibilidad	Especificidad	Tiempo Entr.
Backdoor	0.99627	0.83333	0.00715	0.99999	
Exploits	0.99627	0.95074	0.99993	0.99599	
HTTP-Flood	0.99957	0.98983	0.94422	0.99994	
Phishing	1.0	1.0	1	1.0	
Ransomware	1.0	1.0	1	1.0	141.52
SIDDOS	0.99835	0.9087	0.94173	0.99897	segundos
Smurf	0.98653	0.98505	0.34602	0.99989	
UDP-Flood	0.98662	0.96063	0.99969	0.98036	
Promedio	**0.99**	**0.98**	**0.67**	**0.98**	

Nota: Elaboración Propia.

En la Figura Nº 63, se calculó la exactitud para cada clase, utilizando los valores de la tabla Nº 53, mostrando los resultados de Exactitud del Algoritmo Perceptrón Multicapa, obtenidos de la matriz de confusión Multiclase.

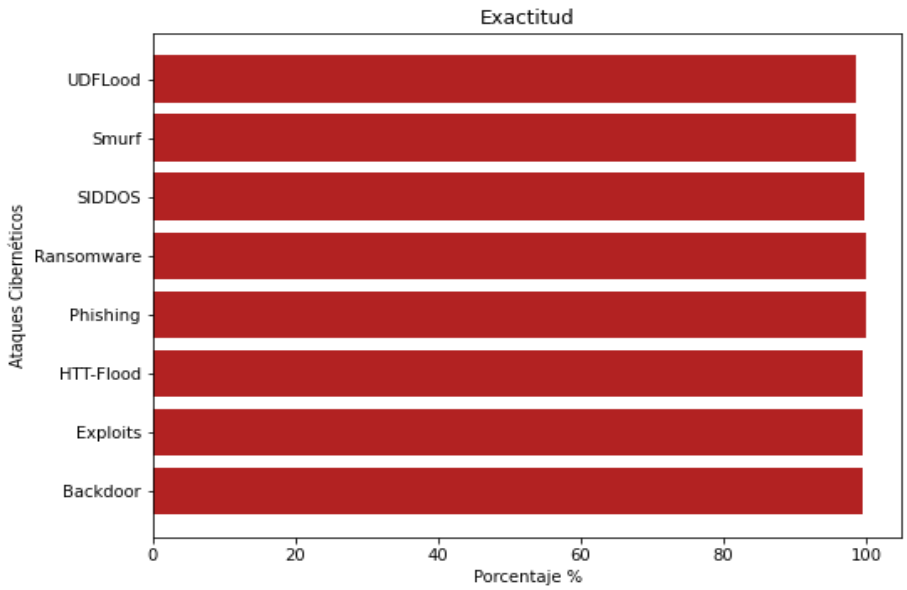

Figura 63: Exactitud del Algoritmo Perceptrón Multicapa.

Fuente: Elaboración Propia.

En la siguiente Figura Nº 64, se calculó la precisión para cada clase, utilizando los valores de la tabla Nº 53, mostrando los resultados de Precisión del Algoritmo Perceptrón Multicapa, obtenidos de la matriz de confusión Multiclase.

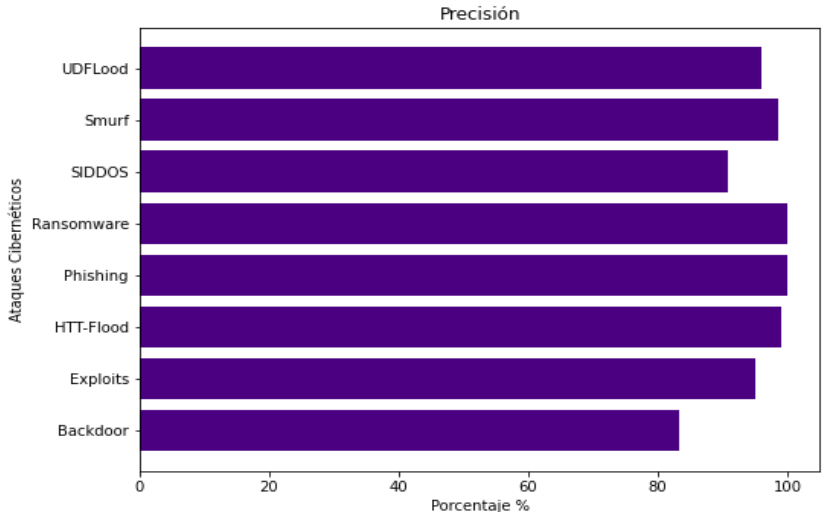

Figura 64: Precisión del Algoritmo Perceptrón Multicapa.
Fuente: Elaboración Propia.

En la Figura Nº 65, se calculó la Sensibilidad para cada clase, utilizando los valores de la tabla Nº 53, mostrando los resultados de sensibilidad del Algoritmo Perceptrón Multicapa, obtenidos de la matriz de confusión Multiclase.

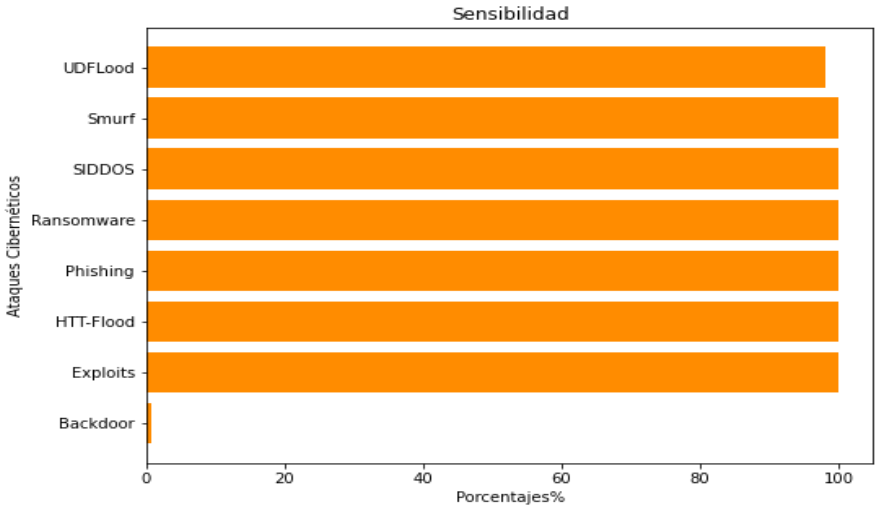

*Figura 65:*Sensibilidad del Algoritmo Perceptrón Multicapa.
Fuente: Elaboración Propia.

En la Figura Nº 66, se calculó la Especificidad para cada clase, utilizando los valores de la tabla Nº 53, mostrando los resultados de especificidad del Algoritmo Perceptrón Multicapa, obtenidos de la matriz de confusión Multiclase.

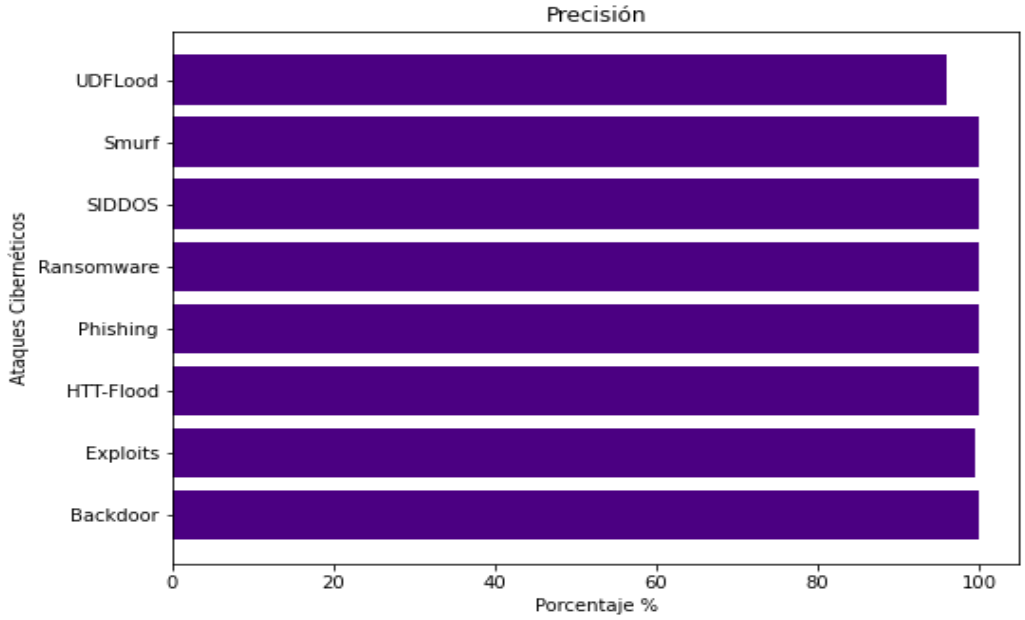

Figura 66: Sensibilidad del Algoritmo Perceptrón Multicapa.
Fuente: Elaboración Propia.

IV. CONCLUSIONES Y RECOMENDACIONES.

4.1. Conclusiones.

La presente investigación recolectó patrones de ataque de los 4 conjuntos de datos, estos conjuntos de datos son los siguientes: UNSW-NB15, dataset Phishing, dataset DDOS, y el último conjunto de datos recolectado fue el CICAndMal2017, formando así el nuevo conjunto de datos llamado Dataset_Attack_2018, conteniendo un total de 621, 209 registros, con 8 clases de diferentes tipos de ataque, guardados en un archivo de Excel con la extensión .csv, el tamaño de este archivo es de 392 Megabytes.

Se analizaron 14 artículos científicos de clasificación, obtenidos de los repositorios IEE, SciELO, Springer Link, Dialnet, siendo evaluados mediante la métrica de exactitud y como resultado esta investigación formó un top de 6 técnicas de Aprendizaje Automáticos, seleccionando dos técnicas Perceptrón Multicapa y Máquinas de Soporte Vectorial.

Esta investigación evaluó las técnicas Perceptrón Multicapa y Maquinas de Soporte Vectorial, con la finalidad de medir la eficiencia de los algoritmos en la detección de intrusos, siendo medidos con las métricas de exactitud, precisión, sensibilidad y especificidad, clasificando 8 tipos de ataque cibernéticos.

Se concluyó que el algoritmo Perceptrón Multicapa es el más eficiente con una exactitud del 99%, precisión del 98 %, sensibilidad del 67%, y especificidad del 98% a pesar que el algoritmo tiene dificultades en clasificar ataques.

4.2. Recomendaciones.

Esta investigación recomienda a futuros científicos, implementar nuevas técnicas de aprendizaje automático, para predecir ataques cibernéticos y aplicarlos en el conjunto de datos Multiclase ya propuesto, realizando así una comparación de los resultados.

Se recomienda a futuros investigadores realizar la implementación de ambas técnicas propuestas en esta investigación, en un lenguaje diferente a Python, debido a la gran demanda que presentan estas técnicas de aprendizaje automático.

REFERENCIAS

Belouch, M., El Hadaja, S., & Idhammadb, M. (2018). Performance evaluation of intrusion detection based on machine learning using Apache Spark. 1-5. Retrieved from https://www.sciencedirect.com/science/article/pii/S1877050918301029

KeyCDN. (2018). *DDoS Attack*. Retrieved from https://www.keycdn.com/support/ddos-attack

Wanga , C., Xub, R., Leec, S., & Leed, C. (2017). Network Intrusion Detection Using Equality Constrained-Optimization-Based Extreme Learning Machines. 42. doi:https://doi.org/10.1016/j.knosys.2018.02.015

Abdelaal, H., Elmahdy, A., Halawa, A., & Youness, H. (2018). Improve the automatic classification accuracy for Arabic tweets using ensemble methods. Retrieved from https://www.sciencedirect.com/science/article/pii/S2314717218300266

Acántara, R. (207). *DESARROLLO DE UN PROCESO DE SEGURIDAD PARA LA PREVENCIÓN DE INTRUSIONES EN UNA RED PRIVADA.* México.

Acedo, E. (2015). *OpenWebinars*. Retrieved from https://openwebinars.net/blog/top-10-de-ataques-dos-denial-of-service-o-denegacion-de-servicios/

Ahmad , D., Kusrini , K., & Sudarmawan , S. (2017). Classification of Intrusion Detection System (IDS) Based on Computer Network. Retrieved from https://ieeexplore.ieee.org/document/8285566

Alcántara, A., López, L., & Rueda, J. (2017). computacionales, Generación de un vector característico para la detección de intrusos en redes. 19-31. Retrieved from https://www.semanticscholar.org/paper/Generaci%C3%B3n-de-un-vector-caracter%C3%ADstico-para-la-de-Alc%C3%A1ntara-Ram%C3%ADrez-L%C3%B3pez-Garc%C3%ADa/cbbf52fad19e82fb6c4aab8baada1dbe1f785aed

Alcántara, A., López, L., & Rueda, J. (2017). Generación de un vector característico para la detección de intrusos en redes computacionales.

Alegre Ramos , M., & Cervigón Hurtado, A. G. (2011). *Seguridad Informática.* Madrid, España. Retrieved from https://books.google.com.pe/books?id=c8kni5g2Yv8C&printsec=frontcover&dq=Seguridad+Inform%C3%A1tica&hl=es&sa=X&ved=0ahUKEwiat8mQoeXaAhUFx1kKHeD4BNAQ6AEISTAH#v=onepage&q=Seguridad%20Inform%C3%A1tica&f=false

Alkasassbeh, M., Al-Naymat, G., Hassanat, A., & Almseidein, M. (2016). *Detecting Distributed Denial of Service Attacks Using Data Mining Techniques.* Princess Sumaya University for technology, Jordan. Retrieved from http://thesai.org/Downloads/Volume7No1/Paper_59-Detecting_Distributed_Denial_of_Service_Attacks_Using_Data_Mining_Techniques.pdf

Almansob, S., & Lomte, S. (2017). Addressing Challenges for Intrusion Detection System using Naive Bayes and PCA Algorithm. 1-4. doi:https://ieeexplore.ieee.org/document/8226193

Álvarez, G. (2010). *Investigación Básica e Investigación Aplicada.* Universidad Señor Bolivar, Caracas.

Amazon Web Service. (2018). *Clasificación multiclase*. Retrieved from https://docs.aws.amazon.com/es_es/machine-learning/latest/dg/multiclass-classification.html

Antolín Ayuso, M., & Barcenilla Mancha, M. (2017). Tipos de ataque en datos de entrenamiento. Retrieved from http://www.it.uc3m.es/jvillena/irc/practicas/07-08/IntrusionesDeRed.pdf

Arash, H., Andi, F., Laya, T., & Ali, A. (2018). *Toward Developing a Systematic Approach to Generate Benchmark Android Malware Datasets and Classification.* Canadian Institute for Cybersecurity (CIC), University of New Brunswick (UNB). doi:https://ieeexplore.ieee.org/document/8585560/

Assolini, F. (2018). *Karpesky Lab*. Retrieved from https://latam.kaspersky.com/about/press-releases/2018_panorama-de-amenazas-phishing

Aziz, A., Hanafi, S., & Hassanien, A. (2017). Comparison of classification techniques applied for network intrusion detection and classification. 109-118. Retrieved from https://www.sciencedirect.com/science/article/pii/S1570868316300738

Baca, G. (2016). *Introducción a la Seguridad Informática.* México: Grupo Editorial Patria. Retrieved from https://books.google.com.pe/books?id=IhUhDgAAQBAJ&printsec=frontcover&dq=Seguridad+inform%C3%A1tica&hl=es&sa=X&ved=0ahUKEwj_uojJluXaAhVIoFMKHUwLD_UQ6AEIKzAB#v=onepage&q=Seguridad%20inform%C3%A1tica&f=false

Baca, G. (2016). *Introducción a la Seguridad Informática.* México: Grupo Editorial Patria. Retrieved from https://books.google.com.pe/books?id=IhUhDgAAQBAJ&printsec=frontcover&dq=Seguridad+inform%C3%A1tica&hl=es&sa=X&ved=0ahUKEwj_uojJluXaAhVIoFMKHUwLD_UQ6AEIKzAB#v=onepage&q=Seguridad%20inform%C3%A1tica&f=false

Belavagi, M., & Muniyal, B. (2016). Performance Evaluation of Supervised Machine Learning Algorithms for Intrusion Detection. 117-123. doi:https://doi.org/10.1016/j.procs.2016.06.016

Bonilla, B. (2015). Diseño e Implementación de una Auditoría de Seguridad a la Información de los Datos Sensibles de un Centro de Educacion Superior. Ecuador. Obtenido de http://repositorio.upse.edu.ec/handle/46000/3709

Calvo, D. (2017, Julio). *Clasificación de redes neuronales (topología de red)*. Retrieved from http://www.diegocalvo.es/clasificacion-de-redes-neuronales-topologia-de-red/

Cannady, J. (1998). Artificial Neural Networks for Misuse Detection. 1-14.

Cannady, J. (n.d.). Artificial Neural Networks for Misuse Detection. 1-14.

Caraballo, D. (n.d.). *Introducción a las Redes Neuronales - Parte 1: Elementos básicos de una Red Neuronal*. Retrieved from Mi diario Python:

http://www.pythondiario.com/2018/07/introduccion-las-redes-neuronales-parte_12.html

Cárdenes, A. (2015). *Inteligencia Artificial.* España.

Carvajal, C. (2015). Sistema de Detección de Intrusos en Redes mediante el Método Heurístico para el Gobierno Municipal de la Ciudad de Otavalo. Ibarra, Ecuador. Retrieved from http://dspace.uniandes.edu.ec/bitstream/123456789/1528/1/TUISIS020-2015.pdf

Castro, J. (2016). Redes Neuronales. Retrieved from http://grupo.us.es/gtocoma/pid/pid10/RedesNeuronales.htm

Chauhan, H., Kumar, V., Pundir, S., & Pilli, E. (2013). A Comparative Study of Classification Techniques for Intrusion Detection. 40-43. doi:https://doi.org/10.1109/ISCBI.2013.16

Chih-Chung, C., & Chih-Jen, L. (2001). *LIBSVM: A Library for Support Vector Machines.* Taiwan.

Choudhury, S., & Bhowal, A. (2015). Comparative Analysis of Machine Learning Algorithms along with Classifiers for Network Intrusion Detection. 40-43. doi:https://doi.org/10.1109/ICSTM.2015.7225395

Cimpanu, C. (2017). *Koler Android Ransomware Targets the US with Fake PornHub Apps*. Retrieved from https://www.bleepingcomputer.com/news/security/koler-android-ransomware-targets-the-us-with-fake-pornhub-apps/

Costas, J. (2012). *Principios de seguridad y alta disponibilidad.* España: RA-MA.

Crammer , K., & Singer , Y. (2001). On the Algorithmic Implementation of Multiclass Kernel-based Vector Machines. 273.

Cristianini, N., & Shawe-Taylor, J. (2000). *An Introduction to Support Vector Machines and other Kernel-Based Learning Methods.* Cambridge University Press.

De la Hoz, E. (2015). Sistemas de Detección de Intrusos con Mapas Autorganizativos Probabilísticos y Optimización Multiobjetivo. Retrieved from https://dialnet.unirioja.es/servlet/tesis?codigo=56512

De la Hoz, E., De la Hoz, M., Ortiz, A., & Ortega, J. (2012). Modelo de detección de intrusiones en sistemas de red, realizando selección de características con FDR y entrenamiento y clasificación con SOM. *Revista INGE CUC*, 85-116. Retrieved from https://revistascientificas.cuc.edu.co/ingecuc/article/view/225/214

DELL-EMC. (2017, 11 4). *Total Global Losses from Phishing Attacks*. Retrieved from https://spain.emc.com/microsites/rsa/phishing/index.htm

Depren, O., Topallar, M., Anarim, E., & Kemal Ciliz, M. (2005). An intelligent intrusion detection system (IDS) for anomaly and misuse detection in computer networks. *Sciencedirect*.

Dezhen, Z., & Kai, Y. (208). *Genetic Algorithm based Optimization for AdaBoost.* China.

Domiguez , E. (2015). *Neuronal Networks Framework*. Retrieved from http://www.redes-neuronales.com.es/tutorial-redes-neuronales/Las-redes-neuronales-monoCapa.htm

Domínguez, M. (2015). *Porn Droid, nuevo malware que ataca a Android*. Retrieved from https://www.parentesis.co/noticias/software_aplicaciones/Porn_Droid_nuevo_malware_que_ataca_a_Android

ESET. (2017). *ESET Security Report Latinoamérica 2017.* Equipo de Investigación de ESET Latinoamérica. Retrieved from https://www.welivesecurity.com/wp-content/uploads/2017/04/eset-security-report-2017.pdf

ESET. (2017). ESET Security Report Latinoamérica 2017.

ESET. (2019). ESET SECURITY REPORT Latinoamérica. Retrieved from https://www.welivesecurity.com/wp-content/uploads/2019/07/ESET-security-report-LATAM-2019.pdf

ESET SECURITY. (2018). *ESET SECURITY REPORT*. Retrieved from https://www.welivesecurity.com/wp-content/uploads/2018/06/ESET_security_report_LATAM2018.pdf

Esmaily, J., Moradinezhad, R., & Ghasemi, J. (2015). Intrusion Detection System Based on Multi-Layer Perceptron Neural Networks and Decision Tree. Retrieved from https://ieeexplore.ieee.org/document/7288736/

Esquivel. (2012). *Sistema de detección de intrusos sobre la red basado en redes neuronales.* Cartago. Retrieved from https://studylib.es/doc/7593166/sistema-de-detecci%C3%B3n-de-intrusos-basado-en-redes-neuronales.

Esquivel, H. (2012). Sistema de detección de intrusos sobre la red basado en redes neuronales. Cartago, Costa Rica. Retrieved from https://rua.ua.es/dspace/bitstream/10045/15687/1/JDARE-08-H.pdf

Evgeniou , T., & Pontil, M. (2001). Workshop on Support Vector Machines: Theory and Applications.

Frampton, M. (2015). *Mastering Apache Spark.* Mumbai-India: Open Source.

Galán, H., & Martínez, A. (2016). *Inteligencia artificial.Redes neuronales y aplicaciones.* Madrid: Universidad Carlos III .

García, S., Ramírez, S., Luengo, J., & Herrera, F. (2016). Big Data: Preprocesamiento y calidad de datos. Retrieved from https://sci2s.ugr.es/sites/default/files/ficherosPublicaciones/2133_Nv237-Digital-sramirez.pdf

Gestal, M., & Pérez, J. (2015). Seguridad electrónica en la gestión de información. Retrieved from https://ruc.udc.es/dspace/bitstream/handle/2183/13116/CC-116_art_6.pdf?sequence=1&isAllowed=y

Gil, D., & Gil, J. (2017). Seguridad informática organizacional: un modelo de simulación basado en dinámica de sistemas. Retrieved from https://revistas.utp.edu.co/index.php/revistaciencia/article/view/11371/10511

Gómes Beites, A. (2011). *Enciclopedia de la Seguridad Informática.* España: Ra-Ma. Retrieved from https://books.google.com.pe/books?id=Bq8-DwAAQBAJ&printsec=frontcover&dq=Seguridad+inform%C3%A1tica&hl=es&sa=X&ved=0ahUKEwj2hdnp0OXaAhVB2VMKHYvID_AQ6wEIMjAC#v=onepage&q=Seguridad%20inform%C3%A1tica&f=false

Gómez, A., Keever, V., & Novales, M. (2016). *El protocolo de investigación III: la población de estudio.* México.

Gonzales, R. (2017). *Las fases del Hacking Ético*. Retrieved from http://ehack.info/las-fases-del-hacking-etico/

Graupe, D. (2013). *Principles of Artificial Neuronal Networks.* USA: World Scientific Publishing Company.

Guevara, A. (2010). *El Hacking Ético y la Seguridad de La Información de Empresas En México*. Retrieved from Seguridad Cultura de Prevencion para T.I.: https://revista.seguridad.unam.mx/numero-13/el-hacking-%C3%A9tico-y-la-seguridad-de-la-informaci%C3%B3n-de-empresas-en-m%C3%A9xico-parte-ii

Guevara, C., Santos, M., & Lòpez, V. (2013). *Sistema de Inmune Artificial para detección de Intrusos aplicando Selección negativa a perfiles de comportamiento de usuario.* España. Retrieved from https://cybercamp.es/cybercamp2015/sites/default/files/contenidos/material/3_sistema_de_inmune_artificial_para_deteccion_de_intrusos.pdf

Gutiérrez, C. (2018). *WebliveSecurity*. Retrieved from https://www.welivesecurity.com/la-es/2018/08/22/gandcrab-nueva-familia-ransomware-crece-latinoamerica/

Hamidian, B., & Ospino, G. (2015). ¿Por qué los sistemas de información son esenciales? *ANUARIO*, 161-183. Retrieved from http://servicio.bc.uc.edu.ve/derecho/revista/idc38/art07.pdf

Henao, J., & Espinosa, J. (2013). Machine learning techniques applied to intruder detection in networks. Retrieved from https://ieeexplore.ieee.org/document/6922081

Henao, R. (2012). *Definición De Un Modelo De Seguridad En Redes De Cómputo, Mediante El Uso De Técnicas De Inteligencia Artificial.* Colombia. Retrieved from https://repositorio.unal.edu.co/bitstream/handle/unal/11602/7107005.2012.pdf?sequence=1&isAllowed=y

Herrero, Á., & Corchado, E. (2014). *Mobile Hybrid Intrusion Deteccion.* Newelska: Springer.

Hoyos, L. (2015). *Prototipo de Detección de Ataques Distribuidos de Denegación de Servicios (ddos1) a Partir de Máquinas de Aprendizaje.* Universidad Autónoma de Manizales, Manizales. Retrieved from http://repositorio.autonoma.edu.co/jspui/bitstream/11182/924/1/InformeManuelSebastianHoyos%20v3.1.pdf

IBM. (2016). *El modelo de redes neuronales.* IBM Kniwledge Center. Retrieved from https://www.ibm.com/support/knowledgecenter/es/SS3RA7_15.0.0/com.ibm.spss.modeler.help/neuralnet_model.htm

Imperva. (2017). *HTTP Flood Attack*. Retrieved from https://www.incapsula.com/ddos/attack-glossary/http-flood.html

INCIBED. (n.d.). *Ransomware: una guía de aproximación para el empresario.* Retrieved from https://www.incibe.es/sites/default/files/contenidos/guias/doc/guia_ransomware_metad.pdf

ISO 27001. (2017). *La norma ISO 27001: Aspectos claves de su diseño e implantación.*

Karim, R., & Kaysar, M. (2016). *Large Scale Machine Learning with Spark.* Reino Unido: Packt Publishing.

Karpesky. (2018). Ataques DDoS en el cuarto trimestre de 2018. Retrieved from https://securelist.lat/ddos-attacks-in-q4-2018/88346/

Karspesky. (2019). IT threat evolution Q3 2019. Statistics. Retrieved from https://securelist.com/it-threat-evolution-q3-2019-statistics/95269/

Kaspersky. (2017, Mayo). Retrieved from https://arstechnica.com/information-technology/2017/05/windows-7-not-xp-was-the-reason-last-weeks-wcry-worm-spread-so-widely/

Kaysar, M. (2016). *Large Scale Machine Learning with.* UK: Packt Publishing.

Kevin, L., Priddy, Paull, & Keller, E. (2005). *Artificial neural networks : An Introduction.* USA: SPI Press.

Khaing, W., Mie Mie, S. T., Thi Thi , Z., Wei Lin, J. C., Shyang Pan, J., Tin, P., & Yokota, M. (2016). *Genetic and Evolutionary Computing: Proceedings of the Ninth International Conference on Genetic and Evolutionary Computing, August 26-28, 2015, Yangon, Myanmar - Volume I.* Poland: Springer. Retrieved from https://books.google.com.pe/books?id=pl51CgAAQBAJ&pg=PA137&dq=nsl+kdd+dataset&hl=es&sa=X&ved=0ahUKEwjD5Oeat4PbAhXrRd8KHTjqAEgQ6AEILDAB#v=onepage&q=nsl%20kdd%20dataset&f=false

Khandelwal, S. (2015). *LockerPin Ransomware Resets PIN and Permanently Locks Your SmartPhones.* Retrieved from https://thehackernews.com/2015/09/android-lock-ransomware.html

Khattab M, A. (2010). *Intrusion Detection System and Artificial Intelligent.* Iraq. Retrieved from https://www.intechopen.com/books/intrusion-detection-systems/intrusion-detection-system-and-artificial-intelligent

Kim, D., & Solomon, M. (2018). *Fundamentals of Information Systems Security.* Information Systems Security & Assurance.

Kim, P. (2017). *MATLAB Deep Learning: con Machine Learning, Neural Networks e Artificial Intelligence.* Korea: Appress.

Kubat, M. (2017). *An Introduction to Machine Learning.* Springer.

Kumar, B., & Kalita, K. (2013). *Network Anomaly Detection A Machine Learning Perspective.* USA. Retrieved from https://books.google.com.pe/books?id=c2nSBQAAQBAJ&printsec=frontcover&dq=Network+Anomaly+Detection+A+Machine+Learning++Perspective&hl=es&sa=X&ved=0ahUKEwjs7uWD_oPbAhUJPN8KHYqdAN8Q6AEIKTAA#v=onepage&q=Network%20Anomaly%20Detection%20A%20Machine%20Learning%

Kumar, N., & Babu, S. (2018). A Study on Machine Learning Techniques Towards the Detection of Distributed Denial of Service Attacks. Retrieved from https://acadpubl.eu/hub/2018-120-6/6/510.pdf

Kumar, S., Verma, K., & Thoke, A. (2015). Investigations on Impact of Feature Normalization Techniques on Classifier's Performance in Breast Tumor Classification. 11-12.

Kunal, J. (2015). *Machine Learning basics for a newbie.* Retrieved from https://www.analyticsvidhya.com/blog/2015/06/machine-learning-basics/#

Lipovsky, R. (2014). *Análisis de Simplocker, primer ransomware para Android activado en TOR*. Retrieved from https://www.welivesecurity.com/la-es/2014/06/04/analisis-primer-ransomware-android-activado-tor/

Lipovský, R., & Stefanko, L. (2018). *Android Ransomware From Android Defender To Doublelocker.* Eslovakia: Eset. Retrieved from https://www.welivesecurity.com/wp-content/uploads/2018/02/Android_Ransomware_From_Android_Defender_to_Doublelocker.pdf

Liska, A., & Gallo, T. (2016). Introduction to Ransomware.

López. (2016). *El ataque DDoS de Cyberbunker a Spamhaus*. Retrieved from http://www.eoi.es/blogs/ciberseguridad/2016/04/17/el-ataque-ddos-de-cyberbunker-a-spamhaus/

Luna, D. (2015). *Sistema Detector de Intrusiones Ocupando una Red Neuronal Artificial.* México. Retrieved from http://ri.uaemex.mx/bitstream/handle/20.500.11799/49966/Sistema%20Detector%20de%20Intrusiones%20Ocupando%20una%20Red%20Neuronal%20Artificial.pdf?sequence=1

Manjula, C., & Muniyal, B. (2016). *Performance Evaluation of Supervised Machine Learning Algorithms for Intrusion Detection.* India: ScienceDirect. Retrieved from https://www.sciencedirect.com/science/article/pii/S187705091631081X

Marroquín, R. (2012). Metodología de la Investigación. 3.

Martinez. (2004). Ideas centrales de la metodología cualitativa.

Matich, D. (2001). *Redes Neuronales: Conceptos Básicos y Aplicaciones.* Argentina. Retrieved from Redes Neuronales: Conceptos Básicos y Aplicaciones

Mazini, M., Shirazi, B., & Mahdavi, I. (2018). Anomaly network-based intrusion detection system using a reliable hybrid artificial bee colony and AdaBoost algorithms. 1-30. doi:https://doi.org/10.1016/j.jksuci.2018.03.011

Mendoza, P. (2013). *Aplicación de Selección De Características, Métricas De Aprendizaje Y Reducción de Dimensión en Sistemas de Detección de intrusos.* Cartagena de Indias. Retrieved from http://biblioteca.utb.edu.co/notas/tesis/0065092.pdf

Misud, E. (2016). *MONOGRÁFICO: Introducción a la seguridad informática*. Retrieved from http://recursostic.educacion.es/observatorio/web/es/software/software-general/1040-introduccion-a-la-seguridad-informatica?start=1

Mohammad, R., Thabtah, F., & McCluskey, L. (2012). *An Assessment of Features Related to Phishing Websites using an Automated Technique.* University of

California, Irvine (UCI) . Retrieved from https://ieeexplore.ieee.org/document/6470857

Moustafa, N., & Slay, J. (2015). *UNSW-NB15: A Comprehensive Data set for Network Intrusion Detection systems.* Australia. Retrieved from https://ieeexplore.ieee.org/document/7348942

Muller, A., & Guido, S. (2015). Introduction to machine learning with Python. Kaggle's. Retrieved from https://github.com/justmarkham/scikit-learn-videos

Mustapha, B., Salah , E., & Mohamed, I. (2018). Performance evaluation of intrusion detection based on machine learning using Apache Spark. 1-5.

Mustapha, B., Salah, E., & Mohamed , I. (2018). Performance evaluation of intrusion detection based on machine learning using Apache Spark. *Science*, 3.

Mustapha, B., Salah, E., & Mohamed , I. (2018). Performance evaluation of intrusion detection based on machine learning using Apache Spark . *Science*, 3.

Muzammil, M., Qazi, S., & Ali, T. (2013). Comparative Analysis of Classification Algorithms Performance for Statistical based Intrusion Detection System. 282-288. doi:https://doi.org/10.1109/PST.2016.7906975

Mylavarapu, G., Johnson, T., & Kumar, T. (2015). Real-time Hybrid Intrusion Detection System using Apache Storm. Retrieved from https://ieeexplore.ieee.org/document/7336370

Nadiammai, G., & Hemalatha, M. (2012). *Perspective analysis of machine learning algorithms for detecting network intrusions.* Coimbatore.

Nadiammai, G., & Hemalatha, M. (2012). Perspective Analysis of Machine Learning Algorithms for Detecting Network Intrusions. Retrieved from https://ieeexplore.ieee.org/document/6395949/

Networks, A. (2017). *Worldwide Infrastructure Security Report. Burlington.* Estados Unidos: Arbor Networks.

Novillo, C., & Guaño, M. (2012). *Implementación De un Sistema De Detección De Intrusos utilizando inteligencia artificial.* Escuela Politécnica Nacional, Quito. Retrieved from http://bibdigital.epn.edu.ec/bitstream/15000/4566/1/CD-4187.pdf

Nunes, I., Hernane, D., Andrade, R., Bartocci, L., & Reis, S. (2017). Artificial Neural Networks. 21-28.

Olanrewaju, R., Ku Zahir, K., Asnawi, A., Sanni, M., & Adekunle, A. (2017). Modelling of Intelligent Intrusion Detection System Making a case for Snort. Retrieved from https://ieeexplore.ieee.org/document/8267152/

Padial Diaz, O. (2014). *Objetivos Principales de la Seguridad de la Información de Redes*. Retrieved from http://oscarpadial.com/objetivos-principales-de-la-seguridad-de-la-informacion-en-las-redes/

Palacios, E., & Sánchez, B. (2010). *El Sistema de Detección de Intrusos: Snort. (Windows y Linux)*. Retrieved from http://www.linux-party.com/index.php/6000-el-sistema-de-deteccion-de-intrusos-snort--windows-y-linux

Palmer, D. (2017). *This Android banking malware steals data by exploiting smartphone accessibility services*. Retrieved from https://www.zdnet.com/article/this-android-banking-malware-steals-data-by-exploiting-smartphone-accessibility-services/

Panda Security. (2013, Noviembre). *¿Qué es un Ransomware?* Retrieved from https://www.pandasecurity.com/es/mediacenter/malware/que-es-un-ransomware/#:~:text=El%20Ransomware%20es%20un%20software,la%20informaci%C3%B3n%20y%20datos%20almacenados.

Panda Security. (2020). Threat Insights Report 2020. 12. Retrieved from https://www.pandasecurity.com/emailhtml/2004-report-threath-20/Threat-Insights-Report-en.pdf

Pérez, A., & Deyban, A. (2017). *Diseño, Implementación y Evaluación de Técnicas Híbridas de Aprendizaje Automático en la Detección de Intrusos en Redes de Computadoras.* Universidad Central de Venezuela, Caracas.

Petersen, R. (2015). A comparison of data mining algorithms and an analysis of relevant features for detecting cyber-attacks. 1-61. Retrieved from https://www.diva-portal.org/smash/get/diva2:939697/FULLTEXT01.pdf

Ray, S. (2017). *Understanding Support Vector Machine algorithm from examples (along with code)*. Retrieved from https://www.analyticsvidhya.com/blog/2017/09/understaing-support-vector-machine-example-code/

REPORT INTERNET CRIME. (2019). *INTERNET CRIME REPORT*. Retrieved from https://pdf.ic3.gov/2019_IC3Report.pdf

Rivero Pérez, J. L. (2014). Técnicas de aprendizaje automático para la detección de intrusos. 52-73.

Rivero, M. (2018). *Phishing*. Retrieved from https://www.infospyware.com/articulos/que-es-el-phishing/

Romero, C. (2010). *Tipos de amenazas en los Sistemas Informáticos*. Retrieved from http://christianromero.es/blog/2010/07/18/tipos-de-amenazas-en-los-sistemas-informaticos/

Sandoval, E. J. (2017). *Seguridad de la Información*. Retrieved from https://revista.seguridad.unam.mx/numero-10/ingenier%C3%AD-social-corrompiendo-la-mente-humana

Sanjeev, k., Paramdeep, S., Rakesh, S., & Bhatia, J. (2012). Distributed Honeynet System Using Gen III Virtual Honeynet. *International Journal of Computer Theory and Engineering,*, 537-540. Retrieved from http://ijcte.org/papers/527-N30013.pdf

Santa, J., Veloza, J., & Montoya, R. (2013). *Aplicación del aprendizaje automático con árboles de decisión al estudio de las variables del modelo de indicadores de gestión de las universidades públicas .* Pereira.

Saranya, C., & Manikandan, G. (2013). A Study on Normalization Techniques for Privacy Preserving Data Mining . 2701-2704.

Sarkar, D., Bali, R., & Sharma, S. (2017). *A Problem-Solver's Guide to Building Real-World Intelligent Systems.* India: Apress.

Sarkar, D., Bali, R., & Sharma, T. (2018). A Problem-Solver's Guide to Building Real-World Intelligent Systems. India: Aprres.

Shah, S., & Issac, B. (2016). Performance comparison of intrusion detection systems and application of machine learning to Snort system. 1-29. Retrieved from https://www.sciencedirect.com/science/article/abs/pii/S0167739X17323178

Su Twin, M., & Shwe Wutyi, K. (2015). *Heuristic Rules for Attack Detection Charged by NSL KDD Dataset.* Retrieved from https://link.springer.com/chapter/10.1007/978-3-319-23204-1_15

Sullivan, W. (2017). *Machine Learning Beginners Guide Algorithms: Supervised & Unsupervised Learning, Decision Tree & Random Forest Introduction.*

Sultana, A., & Jabbar , M. (2016). Intelligent network intrusion detection system using data mining techniques. Retrieved from https://ieeexplore.ieee.org/document/7912017

Syed, R., & Bijú, I. (2016). Performance comparison of intrusion detection systems and application of machine learning to Snort system. 1-29. Retrieved from https://www.sciencedirect.com/science/article/pii/S0167739X17323178

Tamayo, L., & Silva, S. (2017). Técnicas e instumentos de recolección de datos.

Tchakoucht , T., & Ezziyyani, M. (2018). Building A Fast Intrusion Detection System For High-SpeedNetworks - Probe and DoS Attacks Detection. 520-530. Retrieved from https://www.sciencedirect.com/science/article/pii/S1877050918301637

Tchakoucht , T., & Ezziyyani , M. (2018). Building A Fast Intrusion Detection System For High-SpeedNetworks: Probe and DoS Attacks Detection. 1-10. Retrieved from https://www.sciencedirect.com/science/article/pii/S1877050918301637

Teare, D. (1999). *Designing Cisco Networks.* Cisco System. Retrieved from http://docwiki.cisco.com/wiki/Internetworking_Basics

Theobald, O. (2017). *Machine Learning for Absolute.*

Trend Micro. (2018). *Exploit*. Retrieved from https://www.trendmicro.com/vinfo/us/security/definition/exploit

TrendLabs. (2015). *Backdoor*. Retrieved from https://www.trendmicro.com/vinfo/us/security/definition/backdoor

Tribak. (2012). *Análisis estadístico de Distintas Técnicas de Inteligencia Artificial en Detección*. Retrieved from https://dialnet.unirioja.es/servlet/tesis?codigo=61513

Tribak, H. (2012). *Análisis Estadístico de Distintas Técnicas de Inteligencia Artificial en Detección de Intrusos.* Universidad de Granada, España. Retrieved from https://hera.ugr.es/tesisugr/20758340.pdf

Tribak, H. (2012). *Análisis Estadístico de Distintas Técnicas de Inteligencia Artificial en Detección de Intrusos.* Universidad de Granada, España. Retrieved from https://hera.ugr.es/tesisugr/20758340.pdf

Verisign. (2017). *HTTP Flood Attack*. Retrieved from https://www.verisign.com/en_US/security-services/ddos-protection/ddos-attack/index.xhtml

VERSIGIN. (2018). *Q2 2018 DDOS Trends Report*. Retrieved from https://blog.verisign.com/security/ddos-protection/q2-2018-ddos-trends-report-52-percent-of-attacks-employed-multiple-attack-types/

Wang, C., Xu, R., Lee, S., & Lee, C. (2017). Network Intrusion Detection Using Equality Constrained-Optimization-Based Extreme Learning Machines. Retrieved from https://www.sciencedirect.com/science/article/abs/pii/S0950705118300613

WIRED. (2018). *GITHUB SOBREVIVIÓ AL MAYOR ATAQUE DDOS JAMÁS REGISTRADO*. Retrieved from https://www.wired.com/story/github-ddos-memcached/

Yihunie, F., Abdelfattah, E., & Regmi, A. (2019). Applying Machine Learning to Anomaly-Based Intrusion Detection Systems. Retrieved from https://ieeexplore.ieee.org/abstract/document/8817340

Yuquilema, N. (2016). Estudio de Técnicas y Herramientas para la prevención y deteccion de intrusos a nivel de aplicación en la red de datos de la UNACH. Riobamba, Ecuador. Retrieved from http://dspace.unach.edu.ec/bitstream/51000/1501/1/UNACH-EC-ISC-2016-0014.pdf

Zenodermus, J. (2014). *DDOS mediante inyección SQL (SiDDOS)*. Retrieved from http://www.securityidiots.com/Web-Pentest/SQL-Injection/ddos-website-with-sqli-siddos.html

Zhenwei , Y., & Tsai, J. (2011). *Intrusion Detection A Machine Learning Approach.* Chicago: Imperial College Press.

Anexos.

Anexo 1: Resolución de Aprobación del Proyecto de Investigación.

UNIVERSIDAD SEÑOR DE SIPÁN

FACULTAD DE INGENIERÍA, ARQUITECTURA Y URBANISMO

RESOLUCIÓN N° 0036-2019/FIAU-USS

Chiclayo, 23 de enero de 2019

VISTO:

El Acta de Reunión N° S/N de fecha 21 de setiembre de 2018, del Comité Evaluador de la Escuela Académico Profesional de **INGENIERÍA DE SISTEMAS**, donde se propone la aprobación de la modificación de título de la tesis presentada por el(los) tesista(s) **SANTISTEBAN AYASTA LEONARDO EULER** ; y

CONSIDERANDO:

Que, de conformidad con la Ley Universitaria N° 30220 en su artículo 48° que a la letra dice: *"La investigación constituye una función esencial y obligatoria de la universidad, que la fomenta y realiza, respondiendo a través de la producción de conocimiento y desarrollo de tecnologías a las necesidades de la sociedad, con especial énfasis en la realidad nacional. Los docente, estudiantes y graduados participan en la actividad investigadora en su propia institución o en redes de investigación nacional o internacional, creadas por las instituciones universitarias públicas o privadas."*;

Que, mediante Resolución de Facultad N° 0952-2017/FIAU-USS de fecha 19 de diciembre de 2017 se aprobó el Proyecto de Tesis titulado "IMPLEMENTACIÓN DE UN SISTEMA DE DETECCIÓN INTRUSOS PARA MEJORAR LA PROTECCIÓN DE UNA RED INFORMÁTICA FRENTE A ATAQUES 2018 ", presentado por el(los) tesista(s) SANTISTEBAN AYASTA LEONARDO EULER ;

Que, es necesario facilitar el adecuado desarrollo de las Tesis aprobadas con la finalidad de dar continuidad al proceso de investigación; y

Estando a lo expuesto, en uso de las atribuciones conferidas y de conformidad con las normas y reglamentos vigentes;

SE RESUELVE:

ARTÍCULO 1°: APROBAR la modificación de título de la Tesis denominada: *"IMPLEMENTACIÓN DE UN SISTEMA DE DETECCIÓN INTRUSOS PARA MEJORAR LA PROTECCIÓN DE UNA RED INFORMÁTICA FRENTE A ATAQUES 2018 "*, por el siguiente: ***"EVALUACIÓN DE MÁQUINA DE SOPORTE VECTORIAL Y RED NEURONAL ARTIFICIAL PARA DETERMINAR LA EFICIENCIA EN LA DETECCIÓN DE INTRUSOS EN UNA RED INFORMÁTICA"***, presentada por el(los) tesista(s) **SANTISTEBAN AYASTA LEONARDO EULER** , de la Escuela Académico Profesional de **INGENIERÍA DE SISTEMAS.**

ARTÍCULO 2°: MODIFICAR la Resolución de Asesor Especialista N° 0904-2017/FIAU-USS de fecha 18 de diciembre de 2017 y Resolución de Aprobación de Proyecto N° 0952-2017/FIAU-USS de fecha 19 de diciembre de 2017, en el extremo que dice: *"IMPLEMENTACIÓN DE UN SISTEMA DE DETECCIÓN INTRUSOS PARA MEJORAR LA PROTECCIÓN DE UNA RED INFORMÁTICA FRENTE A ATAQUES 2018 "*, por lo siguiente: ***"EVALUACIÓN DE MÁQUINA DE SOPORTE VECTORIAL Y RED NEURONAL ARTIFICIAL PARA DETERMINAR LA EFICIENCIA EN LA DETECCIÓN DE INTRUSOS EN UNA RED INFORMÁTICA".***

REGÍSTRESE, COMUNÍQUESE Y ARCHÍVESE

Cc: CPGTT, Interesados, Archivo

ADMISIÓN E INFORMES
074 481610 - 074 481632
CAMPUS USS
Km. 5, carretera a Pimentel
Chiclayo, Perú

www.uss.edu.pe

Anexo 2: Matriz de consistencia del informe de investigación.

TITULO	EVALUACIÓN DE LAS TÉCNICAS DE MÁQUINAS DE SOPORTE VECTORIAL Y REDES NEURONALES PARA DETERMINAR SU EFICIENCIA EN LA DETECCIÓN DE INTRUSOS EN UNA RED DE COMPUTADORAS.						
TIPO DE INVESTIGACIÓN	**PROBLEMA**	**VARIABLES**	**INDICADORES**	**POBLACIÓN**	**MUESTRA**	**MÉTODO DE RECOLECCIÓN DE DATOS**	**TÉCNICAS DE PROCESAMIENTO DE DATOS**
Aplicada	¿Qué técnica de aprendizaje automático es la más eficiente en la detección de intrusos en una red de Informática?	**Variable Dependiente:** Detección de Intrusos **Variable Independiente:** Técnicas de Aprendizaje Automático	Precisión (P)= VP/(VP+FP) Sensibilidad = VP/(VP+FN) Especificidad = VN/(VN+FP) Tiempo de Entrenamiento = T=TF - TI	La población de la presente investigación está conformada por un top de 6 técnicas de aprendizaje automático, que han sido evaluados a través de las métricas de desempeño, para determinar la eficiencia de intrusos en una red informática.	Por conveniencia, la presente investigación trabajará con dos técnicas de aprendizaje automático, Máquinas de Soporte Vectorial y Perceptrón Multicapa, ya que se realizó un estudio del top, de las técnicas de aprendizaje automático	Análisis Documental Observación	Matriz de Confusión

Diseño de investigación	Hipótesis	Objetivo General	Objetivos específicos	Método propuesto y desarrollado	Resultados preliminares
Cuasi-Experimental	La técnica Perceptrón Multicapa es la más eficiente en la detección de intrusos de una Red Informática	Evaluar las técnicas de Máquinas de Soporte Vectorial y Perceptrón Multicapa para determinar su eficiencia en la detección de Intrusos de una Red Informática.	a) Seleccionar las técnicas de mayor eficiencia en la Detección de Intrusos. b) Preparar la base de datos para entrenamiento y pruebas. c) Implementar las Técnicas de Máquinas de Soporte Vectorial y Perceptrón Multicapa en la detección de Intrusos. d) Evaluar las Técnicas de Perceptrón Multicapa y Máquinas de Soporte Vectorial.	a) Fase de Selección de algoritmos. b) Fase de Preparación del conjunto de datos. C) Fase de Implementación de los algoritmos d) Fase Evaluación de los algoritmos	Para la detección de intrusos la técnica Perceptrón Multicapa tuvo mejores resultados, obteniendo una exactitud del (99%), precisión del (98 %), sensibilidad del 67%, especificidad del (98%) y su tiempo de entrenamiento fue de 141 segundos a diferencia de Máquinas de Soporte Vectorial que tuvo una exactitud del (98%), en precisión (98%) ,sensibilidad del 67%, especificidad (97%) y su tiempo de entrenamiento fue de 114 segundos

Fuente: Elaboración Propia

Anexo 3: Instrumentos de Recolección De Datos.

Anexo 3.1: Constancia de validación por Juicio De Expertos.

USS | UNIVERSIDAD SEÑOR DE SIPÁN

CONSTANCIA DE VALIDACIÓN POR JUICIO DE EXPERTOS

Quien suscribe, Mg. Ing. **Victor Alexci Tuesta Monteza**, con documento Nacional de Identidad N° 42722929, de profesión Ingeniero de Sistemas con Grado de **Ingeniero de Sistemas,** docente de la escuela profesional de Ingeniería de Sistemas de la Universidad Señor de Sipán, mediante la presente hago constatar que el instrumento utilizado para la recolección de datos del proyecto de tesis para obtener el grado de **Ingeniero de Sistemas**, denominado **"EVALUACIÓN DE MÁQUINAS DE SOPORTE VECTORIAL Y RED NEURONAL ARTIFICIAL PARA DETERMINAR LA EFICIENCIA EN LA DETECCIÓN DE INTRUSOS DE UNA RED INFORMÁTICA.**", elaborado por el Bach. **LEONARDO EULER SANTISTEBAN AYASTA** identificado con DNI N° 43760643 de la Universidad Señor de Sipán, reúne los requisitos suficientes y necesarios para ser considerados válidos y confiables, por tanto, están aptos para se aplicados en el logro de los objetivos que se plantearon en la investigación.

Atentantemnte

Fecha 07 de Diciembre del 2018

FIRMA DEL JUEZ DE EXPERTO

Mg. Ing: Victor Alexci Tuesta Monteza

Cargo : Docente de la EP Ing. Sistemas

Anexo 3.2 : Ficha de observación resumen de la técnica Maquinas de Soporte Vectorial.

RESUMEN DE ANÁLISIS DE LA TÉCNICA MAQUINAS DE SOPORTE VECTORIAL

FICHA DE OBSERVACIÓN			
ALGORITMO: MÁQUINAS DE SOPORTE VECTORIAL			
Computador	**Investigador:** Leonardo Euler Santisteban Ayasta		
	Fecha: 05 de Diciembre 2018		
Core i3	**Institución :** Universidad Señor de Sipán		
	Resumen del Análisis		
RESUMEN DEL ANÁLISIS			
Número de análisis	1		
Tipo de Prueba	Porcentaje split (Training= 70%, testing= 30%)		
Tiempo de Predicción	114.83 segundos		
EVALUACIÓN DE MÉTRICAS			
EXACTITUD	**PRECISIÓN**	**SENSIBILIDAD**	**ESPECIFICIDAD**
0.98	0.96	0.63	0.97

Fuente: Elaboración Propia

Anexo 3.3: Ficha de observación resumen de la técnica Redes Neuronales Artificiales.

RESUMEN DE ANÁLISIS DE LA TÉCNICA REDES NEURONALES ARTIFICIALES

FICHA DE OBSERVACIÓN
ALGORITMO: REDES NEURONALES ARTIFICIALES

Computador	**Investigador:** Leonardo Euler Santisteban Ayasta
	Fecha: 05 de Diciembre 2018
Core i3	**Institución :** Universidad Señor de Sipán
	Resumen del Análisis

RESUMEN DEL ANÁLISIS

Número de análisis	1
Tipo de Prueba	Porcentaje split (Training= 70%, testing= 30%)
Tiempo de Predicción	141.52 segundos

EVALUACIÓN DE MÉTRICAS

EXACTITUD	PRECISIÓN	SENSIBILIDAD	ESPECIFICIDAD
0.98	0.98	0.67	0.98

Fuente: Elaboración Propia

Anexo 3.4: Ficha de observación resumen del análisis del ataque cibernético Backdoor implementado con la técnica Máquinas de Soporte Vectorial.

RESUMEN DE ANÁLISIS DEL ATAQUE CIBERNÉTICO BACKDOOR

FICHA DE OBSERVACIÓN

ALGORITMO: MÁQUINAS DE SOPORTE VECTORIAL

Computador	**Investigador:** Leonardo Euler Santisteban Ayasta
	Fecha: 05 de Diciembre 2018
Core i3	**Institución :** Universidad Señor de Sipán
	Resumen del Análisis

RESUMEN DEL ANÁLISIS

Número de análisis	1
Tipo de Prueba	Porcentaje split (Training= 70%, testing= 30%)
Tiempo de Predicción	114.83 segundos

EVALUACIÓN DE MÉTRICAS

EXACTITUD	PRECISIÓN	SENSIBILIDAD	ESPECIFICIDAD
0.99	0.75	0.0	1.0

Fuente: Elaboración Propia

Anexo 3.5: Ficha de observación resumen del análisis del ataque cibernético Exploits implementado con la técnica Maquinas de Soporte Vectorial.

RESUMEN DE ANÁLISIS DEL ATAQUE CIBERNÉTICO EXPLOITS

FICHA DE OBSERVACIÓN ALGORITMO: MÁQUINAS DE SOPORTE VECTORIAL	
Computador	**Investigador:** Leonardo Euler Santisteban Ayasta **Fecha:** 05 de Diciembre 2018
Core i3	**Institución :** Universidad Señor de Sipán
	Resumen del Análisis

RESUMEN DEL ANÁLISIS	
Número de análisis	1
Tipo de Prueba	Porcentaje split (Training= 70%, testing= 30%)
Tiempo de Predicción	114.83 segundos

EVALUACIÓN DE MÉTRICAS			
EXACTITUD	**PRECISIÓN**	**SENSIBILIDAD**	**ESPECIFICIDAD**
0.99	0.95	1.0	0.99

Fuente: Elaboración Propia.

Anexo 3.6: Ficha de observación resumen del análisis del ataque cibernético HTTP-Flood implementado con la técnica Maquinas de Soporte Vectorial.

RESUMEN DE ANÁLISIS DEL ATAQUE CIBERNÉTICO HTT-FLOOD

FICHA DE OBSERVACIÓN

ALGORITMO: MÁQUINAS DE SOPORTE VECTORIAL

Computador	**Investigador:** Leonardo Euler Santisteban Ayasta
	Fecha: 05 de Diciembre 2018
Core i3	**Institución :** Universidad Señor de Sipán
	Resumen del Análisis

RESUMEN DEL ANÁLISIS

Número de análisis	1
Tipo de Prueba	Porcentaje split (Training= 70%, testing= 30%)
Tiempo de Predicción	114.83 segundos

EVALUACIÓN DE MÉTRICAS

EXACTITUD	PRECISIÓN	SENSIBILIDAD	ESPECIFICIDAD
0.99	0.98	0.94	0.99

Fuente: Elaboración Propia

Anexo 3.7: Ficha de observación resumen del análisis del ataque cibernético Phishing implementado con la técnica Maquinas de Soporte Vectorial.

RESUMEN DE ANÁLISIS DEL ATAQUE CIBERNÉTICO PHISHING

FICHA DE OBSERVACIÓN

ALGORITMO: MÁQUINAS DE SOPORTE VECTORIAL

Computador	**Investigador:** Leonardo Euler Santisteban Ayasta
	Fecha: 05 de Diciembre 2018
Core i3	**Institución :** Universidad Señor de Sipán
	Resumen del Análisis

RESUMEN DEL ANÁLISIS

Número de análisis	1
Tipo de Prueba	Porcentaje split (Training= 70%, testing= 30%)
Tiempo de Predicción	114.83 segundos

EVALUACIÓN DE MÉTRICAS

EXACTITUD	PRECISIÓN	SENSIBILIDAD	ESPECIFICIDAD
1.0	1.0	1.0	1.0

Fuente: Elaboración Propia

Anexo 3.8: Ficha de observación resumen del análisis del ataque cibernético Ransomware implementado con la técnica Maquinas de Soporte Vectorial.

RESUMEN DE ANÁLISIS DEL ATAQUE CIBERNÉTICO RANSOMWARE

FICHA DE OBSERVACIÓN

ALGORITMO: MÁQUINAS DE SOPORTE VECTORIAL

Computador	**Investigador:** Leonardo Euler Santisteban Ayasta
Fecha:	**Fecha: 0**5 de Diciembre 2018
Core i3	**Institución :** Universidad Señor de Sipán
	Resumen del Análisis

RESUMEN DEL ANÁLISIS

Número de análisis	1
Tipo de Prueba	Porcentaje split (Training= 70%, testing= 30%)
Tiempo de Predicción	114.83 segundos

EVALUACIÓN DE MÉTRICAS

EXACTITUD	PRECISIÓN	SENSIBILIDAD	ESPECIFICIDAD
1.0	1.0	1.0	1.0

Fuente: Elaboración Propia

Anexo 3.9: Ficha de observación resumen del análisis del ataque cibernético SIDDOS implementado con la técnica Maquinas de Soporte Vectorial.

RESUMEN DE ANÁLISIS DEL ATAQUE CIBERNÉTICO SIDDOS

FICHA DE OBSERVACIÓN

ALGORITMO: MÁQUINAS DE SOPORTE VECTORIAL

Computador	**Investigador:** Leonardo Euler Santisteban Ayasta
Fecha:	**Fecha: 0**5 de Diciembre 2018
Core i3	**Institución :** Universidad Señor de Sipán
	Resumen del Análisis

RESUMEN DEL ANÁLISIS

Número de análisis	1
Tipo de Prueba	Porcentaje split (Training= 70%, testing= 30%)
Tiempo de Predicción	114.83 segundos

EVALUACIÓN DE MÉTRICAS

EXACTITUD	PRECISIÓN	SENSIBILIDAD	ESPECIFICIDAD
0.99	0.89	0.94	0.99

Fuente: Elaboración Propia

Anexo 3.10: Ficha de observación resumen del análisis del ataque cibernético Smurf implementado con la técnica Maquinas de Soporte Vectorial.

RESUMEN DE ANÁLISIS DEL ATAQUE CIBERNÉTICO SMURF

FICHA DE OBSERVACIÓN

ALGORITMO: MÁQUINAS DE SOPORTE VECTORIAL

Computador	**Investigador:** Leonardo Euler Santisteban Ayasta
	Fecha: 05 de Diciembre 2018
Core i3	**Institución :** Universidad Señor de Sipán
	Resumen del Análisis

RESUMEN DEL ANÁLISIS

Número de análisis	1
Tipo de Prueba	Porcentaje split (Training= 70%, testing= 30%)
Tiempo de Predicción	114.83 segundos

EVALUACIÓN DE MÉTRICAS

EXACTITUD	PRECISIÓN	SENSIBILIDAD	ESPECIFICIDAD
0.98	0.96	0.34	0.99

Fuente: Elaboración Propia

Anexo 3.11: Ficha de observación resumen del análisis del ataque cibernético UDP-Flood implementado con la técnica Maquinas de Soporte Vectorial.

RESUMEN DE ANÁLISIS DEL ATAQUE CIBERNÉTICO UDP-Flood

FICHA DE OBSERVACIÓN

ALGORITMO: MÁQUINAS DE SOPORTE VECTORIAL

Computador	**Investigador:** Leonardo Euler Santisteban Ayasta
Fecha:	**Fecha: 0**5 de Diciembre 2018
Core i3	**Institución :** Universidad Señor de Sipán
	Resumen del Análisis

RESUMEN DEL ANÁLISIS

Número de análisis	1
Tipo de Prueba	Porcentaje split (Training= 70%, testing= 30%)
Tiempo de Predicción	114.83 segundos

EVALUACIÓN DE MÉTRICAS

EXACTITUD	PRECISIÓN	SENSIBILIDAD	ESPECIFICIDAD
0.98	0.96	0.99	0.98

Fuente: Elaboración Propia

Anexo 3.12: Ficha de observación resumen del análisis del ataque cibernético Backdoor implementado con la técnica Redes Neuronales Artificiales.

RESUMEN DE ANÁLISIS DEL ATAQUE CIBERNÉTICO BACKDOOR

FICHA DE OBSERVACIÓN

ALGORITMO: REDES NEURONALES ARTIFICIALES

Computador	**Investigador:** Leonardo Euler Santisteban Ayasta
	Fecha: 05 de Diciembre 2018
Core i3	**Institución :** Universidad Señor de Sipán
	Resumen del Análisis

RESUMEN DEL ANÁLISIS

Número de análisis	1
Tipo de Prueba	Porcentaje split (Training= 70%, testing= 30%)
Tiempo de Predicción	141.52 segundos

EVALUACIÓN DE MÉTRICAS

EXACTITUD	PRECISIÓN	SENSIBILIDAD	ESPECIFICIDAD
0.99	0.83	0.0	0.99

Fuente: Elaboración Propia

Anexo 3.13: Ficha de observación resumen del análisis del ataque cibernético Exploits implementado con la técnica Redes Neuronales Artificiales.

RESUMEN DE ANÁLISIS DEL ATAQUE CIBERNÉTICO EXPLOITS

FICHA DE OBSERVACIÓN

ALGORITMO: REDES NEURONALES ARTIFICIALES

Computador	**Investigador:** Leonardo Euler Santisteban Ayasta
	Fecha: 05 de Diciembre 2018
Core i3	**Institución :** Universidad Señor de Sipán
	Resumen del Análisis

RESUMEN DEL ANÁLISIS

Número de análisis	1
Tipo de Prueba	Porcentaje split (Training= 70%, testing= 30%)
Tiempo de Predicción	141.52 segundos

EVALUACIÓN DE MÉTRICAS

EXACTITUD	PRECISIÓN	SENSIBILIDAD	ESPECIFICIDAD
0.99	0.95	0.99	0.99

Anexo 3.14: Ficha de observación resumen del análisis del ataque cibernético HTTP-Flood implementado con la técnica Redes Neuronales Artificiales.

RESUMEN DE ANÁLISIS DEL ATAQUE CIBERNÉTICO HTTP-FLOOD

FICHA DE OBSERVACIÓN

ALGORITMO: REDES NEURONALES ARTIFICIALES

Computador	**Investigador:** Leonardo Euler Santisteban Ayasta
	Fecha: 05 de Diciembre 2018
Core i3	**Institución :** Universidad Señor de Sipán
	Resumen del Análisis

RESUMEN DEL ANÁLISIS

Número de análisis	1
Tipo de Prueba	Porcentaje split (Training= 70%, testing= 30%)
Tiempo de Predicción	141.52 segundos

EVALUACIÓN DE MÉTRICAS

EXACTITUD	PRECISIÓN	SENSIBILIDAD	ESPECIFICIDAD
0.99	0.98	0.94	0.99

Anexo 3.15: Ficha de observación resumen del análisis del ataque cibernético Phishing implementado con la técnica Redes Neuronales Artificiales.

RESUMEN DE ANÁLISIS DEL ATAQUE CIBERNÉTICO PHISHING

FICHA DE OBSERVACIÓN

ALGORITMO: REDES NEURONALES ARTIFICIALES

Computador	**Investigador:** Leonardo Euler Santisteban Ayasta
	Fecha: 05 de Diciembre 2018
Core i3	**Institución :** Universidad Señor de Sipán
	Resumen del Análisis

RESUMEN DEL ANÁLISIS

Número de análisis	1
Tipo de Prueba	Porcentaje split (Training= 70%, testing= 30%)
Tiempo de Predicción	141.52 segundos

EVALUACIÓN DE MÉTRICAS

EXACTITUD	PRECISIÓN	SENSIBILIDAD	ESPECIFICIDAD
1.0	1.0	1.0	1.0

Fuente: Elaboración Propia

Anexo 3.16: Ficha de observación resumen del análisis del ataque cibernético Ransomware implementado con la técnica Redes Neuronales Artificiales.

RESUMEN DE ANÁLISIS DEL ATAQUE CIBERNÉTICO RANSOMWARE

FICHA DE OBSERVACIÓN

ALGORITMO: REDES NEURONALES ARTIFICIALES

Computador	**Investigador:** Leonardo Euler Santisteban Ayasta
	Fecha: 05 de Diciembre 2018
Core i3	**Institución :** Universidad Señor de Sipán
	Resumen del Análisis

RESUMEN DEL ANÁLISIS

Número de análisis	1
Tipo de Prueba	Porcentaje split (Training= 70%, testing= 30%)
Tiempo de Predicción	141.52 segundos

EVALUACIÓN DE MÉTRICAS

EXACTITUD	PRECISIÓN	SENSIBILIDAD	ESPECIFICIDAD
1.0	1.0	1.0	1.0

Fuente: Elaboración Propia

Anexo 3.17: Ficha de observación resumen del análisis del ataque cibernético SIDDOS implementado con la técnica Redes Neuronales Artificiales.

RESUMEN DE ANÁLISIS DEL ATAQUE CIBERNÉTICO SIDDOS

FICHA DE OBSERVACIÓN

ALGORITMO: REDES NEURONALES ARTIFICIALES

Computador	**Investigador:** Leonardo Euler Santisteban Ayasta
	Fecha: 05 de Diciembre 2018
Core i3	**Institución :** Universidad Señor de Sipán

Resumen del Análisis

RESUMEN DEL ANÁLISIS

Número de análisis	1
Tipo de Prueba	Porcentaje split (Training= 70%, testing= 30%)
Tiempo de Predicción	141.52 segundos

EVALUACIÓN DE MÉTRICAS

EXACTITUD	PRECISIÓN	SENSIBILIDAD	ESPECIFICIDAD
0.99	0.90	0.94	0.99

Fuente: Elaboración Propia

Anexo 3.18: Ficha de observación resumen del análisis del ataque cibernético Smurf implementado con la técnica Redes Neuronales Artificiales.

RESUMEN DE ANÁLISIS DEL ATAQUE CIBERNÉTICO SMURF

FICHA DE OBSERVACIÓN

ALGORITMO: REDES NEURONALES ARTIFICIALES

Computador	**Investigador:** Leonardo Euler Santisteban Ayasta
	Fecha: 05 de Diciembre 2018
Core i3	**Institución :** Universidad Señor de Sipán
	Resumen del Análisis

RESUMEN DEL ANÁLISIS

Número de análisis	1
Tipo de Prueba	Porcentaje split (Training= 70%, testing= 30%)
Tiempo de Predicción	141.52 segundos

EVALUACIÓN DE MÉTRICAS

EXACTITUD	PRECISIÓN	SENSIBILIDAD	ESPECIFICIDAD
0.98	0.98	0.34	0.99

Fuente: Elaboración Propia

Anexo 3.19: Ficha de observación resumen del análisis del ataque cibernético UDP-Flood implementado con la técnica Redes Neuronales Artificiales.

RESUMEN DE ANÁLISIS DEL ATAQUE CIBERNÉTICO UDP-FLOOD

FICHA DE OBSERVACIÓN

ALGORITMO: REDES NEURONALES ARTIFICIALES

Computador	**Investigador:** Leonardo Euler Santisteban Ayasta
	Fecha: 05 de Diciembre 2018
Core i3	**Institución :** Universidad Señor de Sipán
	Resumen del Análisis

RESUMEN DEL ANÁLISIS

Número de análisis	1
Tipo de Prueba	Porcentaje split (Training= 70%, testing= 30%)
Tiempo de Predicción	141.52 segundos

EVALUACIÓN DE MÉTRICAS

EXACTITUD	PRECISIÓN	SENSIBILIDAD	ESPECIFICIDAD
0.98	0.96	0.99	0.98

Fuente: Elaboración Propia

Anexo 4: Proceso de la Selección de las técnicas de mayor eficiencia en la detección de intrusos.

N°	Clasificador	Precision	Excatitud (%)	Sensibilidad (%) / DR / Recall	Especificidad (%)	FPR (%)	Tiempo (s)
1	Random Forest		99.746	99.881	99.591		6.77
2	Arbol de decisiones C4.5 / J48		99.7352				64.23
3	Random Forest	0.951	91.5236	88.68	88.04	0.12	
4	Naive Bayes	0.859		80.56			
	MultiLayer perceptron	0.9856		62.74			
5	Arbol de decisiones C4.5 / J48		98.8				0.02
6	Random Forest		97.49	93.53	97.75	3.1	5.69
7	Random Forest	0.99	99	99			
8	Naive Bayes		87.0				
9	Naive Bayes			99.8		2.7	0.17
	Arbol de decisiones C4.5 / J48			99.64		0.3	2.61
10	SMO	0.986	98.60	98,6/0.986			
11	AdaBoost		98.9	99.61		0.01	

12	MultiLayer perceptron	te pe	99.90	99.94			4545.51
	MultiLayer perceptron	0.8203	81.61	99.48		0.52	3344.90
	Support Vector Machines	0.9878	99.04	99.24			2939.88
13	k-NN		77.0892				
14	Naive Bayes			96.8		0.7	
	Naive Bayes			97		0.5	
	Naive Bayes			97.3		3.1	

	Clasificador	**Precision**	**Exactitud (%)**	**Sensibilidad (%) / DR / Recall**	**Especificidad (%)**	**FPR (%)**	**Tiempo (s)**
2	Random Forest		99.746	99.881	99.591		6.77
3	Naive Bayes			99.8		2.7	0.17
4	Arbol de decisiones C4.5 / J48			99.64		0.3	2.61
6	SMO	0.986	98.60	98.6			
5	AdaBoost		98.9	99.61		0.01	
1	**MultiLayer perceptron**	**0.9994**	**99.90**	**99.94**			**4545.51**
7	**Support Vector Machines**	**0.9878**	**99.04**	**99.24**			**2939.88**
8	k-NN		77.0892				

Fuente: Elaboración Propia.

Anexo 5: Listado de las características del conjunto de datos UNSW-NB15

N°	Tipo de Característica	Característica	Descripción	Tipo
1		Srcip	Dirección IP de origen	Nominal
2		Sport	Número de puerto de origen	Entero
3	**Flujo**	Dstip	Dirección IP de destino	Nominal
4		Dsport	Número de puerto de destino	Entero
5		Proto	Protocolo de transacción	Nominal
6		State	Indica el estado y su protocolo dependiente, por ejemplo, ACC, CLO, CON, ECO, ECR, FIN, INT, MAS, PAR, REQ, RST, TST, TXD, URH, URN, y (-) (si no se utiliza el estado)	Nominal
7		Dur	Registrar la duración total	Flotación
8		Sbytes	Bytes de la transacción de Origen a destino. El total de bytes de origen	Entero
9	**Basicas**	Dbytes	Destino a los bytes de transacción de origen. Los bytes de destino	Entero
10		Sttl	Tiempo de origen a destino para valor de vida. Tiempo de origen a destino para el valor en vivo	Entero
11		Dttl	Destino al tiempo de origen para valor en vivo	Entero
12		Sloss	Paquetes fuente retransmitidos o descartados	Entero
13		Dloss	Paquetes de destino retransmitidos o descartados	Entero

14		Servicio	Http, ftp, smtp, ssh, dns, ftp-data, irc y (-) si no se usa mucho el servicio	Nominal
15		Sload	Fuente de bits por segundo	Flotante
16		Dload	Bits de destino por segundo	Float
17		Spkts	Recuento de paquetes de origen a destino	Entero
18		Dpkts	Recuento de paquetes de destino a origen	Entero
19		swin	Valor de anuncio de la ventana TCP de origen	Integer
20		dwin	Valor de anuncio de la ventana TCP de destino	Entero
21		stcpb	Número de secuencia base de TCP de origen	Entero
22		dtcpb	Número de secuencia base TCP de destino	Entero
23	**Contenido**	smeansz	La Media del tamaño del paquete de flujo transmitido por la fuente.	Entero
24		dmeansz	Media del tamaño del paquete de flujo transmitido por el dst	Entero
25		trans_depth	Representa la profundidad canalizada en la conexión de la transacción de solicitud / respuesta http	Entero
26		res_bdy_len	Tamaño de contenido real sin comprimir de los datos transferidos desde el servicio http del servidor.	Entero
27		Sjit	Fuente jitter (mSec). variabilidad del tiempo de ejecución de los paquetes. Si el jitter es	Float

			demasiado grande, no puede asegurarse que las informaciones críticas de proceso lleguen a tiempo.	
28		Djit	Jitter de destino (mSec)	Flotador
29		Stime	Tiempo de inicio de registro	marca de tiempo
30		Ltime	Record la última vez	timestamp
31		Sintpkt	Hora de llegada del paquete de origen (mSec)	Flotador
32	**Tiempo**	Dintpkt	Hora de llegada del paquete intermedio de destino (mSec)	Float
33		tcprtt	Tiempo de ida y vuelta de la configuración de la conexión TCP, la suma de 'synack' y 'ackdat	Float
34		synack	"Tiempo de configuración de la conexión TCP, el tiempo entre los paquetes SYN y SYN_ACK	Float
35		ackdat	"Tiempo de configuración de la conexión TCP,el tiempo entre los paquetes SYN_ACK y ACK	Float
36	**Propósito General**	is_sm_ips_ports	Si las direcciones IP de origen (1) y de destino (3) son iguales y los números de puerto (2) (4) son iguales, esta variable toma el valor 1 más 0.	Binary
37		ct_state_ttl	No. para cada estado (6) de acuerdo con el rango específico de valores para el tiempo de origen / destino para vivir (10) (11).	Integer
38		ct_flw_http_mthd	No. de flujos que tiene métodos como Get y Post en el servicio http.	Entero
39		is_ftp_login	Si se accede a la sesión ftp por usuario y contraseña, entonces 1 más 0.	Binary

40		ct_ftp_cmd	No de flujos que tiene un comando en la sesión ftp.	Entero
41		ct_srv_src	No. de conexiones que contienen el mismo servicio (14) y dirección de origen (1) en 100 conexiones según la última vez (26).	Entero
43		ct_srv_dst	No. de conexiones que contienen el mismo servicio (14) y dirección de destino (3) en 100 conexiones según la última vez (26).	Entero
43		ct_dst_ltm	No. de conexiones de la misma dirección de destino (3) en 100 conexiones según la última vez (26).	Entero
44	**Conexión**	ct_src_ltm	No. de conexiones de la misma dirección de origen (1) en 100 conexiones según la última vez (26).	Entero
45		ct_src_dport_ltm	No de conexiones de la misma dirección de origen (1) y el puerto de destino (4) en 100 conexiones según la última vez (26).	Entero
46		ct_dst_sport_ltm	No de conexiones de la misma dirección de destino (3) y el puerto de origen (2) en 100 conexiones según la última vez (26).	entero
47		ct_dst_src_ltm	No de conexiones de la misma fuente (1) y la dirección de destino (3) en 100 conexiones según la última vez (26).	entero
48	**Label**	**attack_cat**	El nombre de cada categoría de ataque. En este conjunto de datos, nueve categorías, por ejemplo, Backdoors, DoS y Exploits	nominal
49		Etiqueta	0 para registros normales y 1 para ataques	binario

Fuente: (Moustafa & Slay, 2015)

Anexo 6: Listado de las características del conjunto de datos Dataset_ddos

N°	Característica	Descripción	Tipo
1	SRC_ADD	Puerto de dirección de origen	continuous
2	DES_ADD	Puerto de dirección de destino	continuous
3	PKT_ID	Identificador de paquete	continuous
4	FROM_NODE	Definir cliente enviando paquete	continuous
5	TO_NODE	Definir el paquete receptor del cliente	continuous
6	PKT_TYPE	Tipo de paquete basado en protocolo	continuous
7	PKT_SIZE	Tamaño total del paquete en bytes	continuous
8	FLAGS	Una cadena de bandera de 7 dígitos	Symbolic
9	FID	Identificador de flujo	continuous
10	SEQ_NUMBER	Secuencia de números	continuous
11	NUMBER_OF_PKT	Número total de paquetes	continuous
12	NUMBER_OF_BYTE	Número total de bytes	continuous
13	NODE_NAME_FROM	Nombre del cliente que envía el paquete.	Symbolic
14	NODE_NAME_TO	Nombre del cliente que recibe el paquete	Symbolic
15	PKT_IN	Tiempo total del paquete dentro de la cola	continuous
16	PKT_OUT	Tiempo total del paquete fuera de la cola	continuous
17	PKT_R	Tiempo de paquete recibido	continuous
18	PKT_DELAY_NODE	Tiempo de retraso del paquete dentro del nodo	continuous
19	PKT_RATE	Tasa media de paquetes	continuous

20	BYTE_RATE	Velocidad media de bytes	continuous
21	PKT_AVG_SIZE	Tamaño promedio del paquete	continuous
22	UTILIZATION	Utilización del ancho de banda	continuous
23	PKT_DELAY	Tiempo total de retraso del paquete	continuous
24	PKT_SEND_TIME	Hora de envío del paquete.	continuous
25	PKT_RESEVED_TIME	Tiempo de recepción del paquete	continuous
26	FIRST_PKT_SENT	Hora del primer paquete enviado.	continuous
27	LAST_PKT_RESEVED	Hora del último paquete recibido	continuous
28	PKT_CLASS	THTTP Flood, SIDDOS, UDP Flood, and Smurf y Normal	

Fuente: (Alkasassbeh, Al-Naymat, Hassanat, & Almseidein, 2016)

Anexo 7: Listado de las características del conjunto de datos Dataset_Phishing

N°	Cracterística	Descripción
1	URL_of_Anchor	Los enlaces dentro de la página web pueden apuntar a un dominio diferente del dominio escrito en la barra de direcciones de URL.
2	Request_URL	La URL de solicitud examina si los objetos externos contenidos en una página web, como imágenes, videos y sonidos, se cargan desde otro dominio. En las páginas web legítimas, la dirección de la página web y la mayoría de los objetos incrustados dentro de la página web comparten el mismo dominio.
3	Server Form Handle	Una vez que el usuario presentó su información; la página web transferirá la información a un servidor para que pueda procesarla. Normalmente, la información se procesa desde el mismo dominio donde se encuentra la página web. siendo cargado. Los phishers recurren para que el controlador de formularios del servidor esté vacío o la información se transfiera a un lugar diferente al dominio legítimo.
4	URL_Length	Los phishers pueden usar una URL larga para ocultar la parte dudosa en la barra de direcciones.
5	URL's having @ symbol	El uso del símbolo "@" en la URL hace que el navegador ignore todo lo que precede al símbolo "@" y la dirección real a menudo sigue al símbolo "@".

6	Prefix and suffix	Los phishers intentan estafar a los usuarios cambiando la forma de la URL sospechosa para que se vea legítima. Una técnica utilizada es agregar un prefijo o sufijo a la URL legítima. Por lo tanto, el usuario puede no notar ninguna diferencia.
7	IP address:	El uso de una dirección IP en el nombre de dominio de la URL es un indicador de que alguien está tratando de acceder a la información personal. Este truco involucra enlaces que pueden comenzar con un Dirección IP que la mayoría de las empresas ya no suelen utilizar. En el análisis de frecuencia realizado anteriormente, el 20% de los datos contiene una dirección de "IP" y todos ellos están asociados con sitios web phishy. Una dirección IP es como http: // 91.121.10.211/~chems/webscr/verify A veces, la IP La dirección se transforma en hexadecimal como http: // 0x58.0xCC.0xCA.0x62..
8	Sub Domain	Otra técnica utilizada por los phishers para estafar a los usuarios es agregar un subdominio a la URL para que los usuarios crean que están tratando con un sitio web auténtico. Un ejemplo: http: //www.paypal.it.asce ndancethe atrearts.co.uk

9	Website traffic	Los sitios web legítimos generalmente tienen mucho tráfico ya que se visitan con regularidad. Desde phishing Los sitios web normalmente tienen una vida relativamente corta; No tienen tráfico web o tienen bajo ranking. Se ha recomendado que una página web legítima tenga un rango menor o igual a 150,000 en la Base de datos de Alex
10	Age of domain	Sitios web que tienen presencia en línea de Menos de 1 año, puede considerarse arriesgado.
11	**Classe**	Donde los valores "1", "0" y "-1" denotan "Legitimate", "Suspicious" y "Phishy" respectivamente.

Fuente: (Mohammad, Thabtah, & McCluskey, 2012)

Anexo 08: Listado de las características del conjunto de datos CICAndMal2017

N°	Característica	Descripción
1	Flow ID	Identifación del flujo
2	Source IP	Ip de Origen
3	Source Port	Puerto de origen
4	Destination IP	IP de destino
5	Destination Port	Puerto de destino
6	Protocol	Protocolo
7	Timestamp	Marca de tiempo
8	Flow Duration	Duración del flujo
9	Total Fwd Packets	Total de paquetes en la dirección de avance (de origen a destino).
10	Total Backward Packets	Tota, de paquetes hacia atrás (destino a origen).
11	Total Length of Fwd Packets	Tamaño total del paquete hacia adelante (de origen a destino).
12	Total Length of Bwd Packets	Tamaño total del paquete hacia atrás (destino a origen).
13	Fwd Packet Length Max	Tamaño máximo del paquete en dirección hacia adelante
14	Fwd Packet Length Min	Tamaño mínimo del paquete en dirección hacia adelante
15	Fwd Packet Length Mean	Tamaño medio del paquete en dirección hacia adelante

16	Fwd Packet Length Std	Tamaño de desviación estándar del paquete en dirección hacia adelante
17	Bwd Packet Length Max	Tamaño máximo del paquete hacia atrás
18	Bwd Packet Length Min	Tamaño mínimo del paquete hacia atrás
19	Bwd Packet Length Mean	Tamaño medio del paquete hacia atrás
20	Bwd Packet Length Std	Tamaño de desviación estándar del paquete en dirección hacia atrás
21	Flow Bytes/s	Velocidad de bytes de flujo, es elNúmero de bytes de flujo por segundo
22	Flow Packets/s	Velocidad de paquetes de flujo que es el Número de paquetes de flujo por segundo.
23	Flow IAT Mean	Tiempo medio entre dos flujos.
24	Flow IAT Std	Tiempo de desviación estándar dos flujos
25	Flow IAT Max	Tiempo máximo entre dos flujos.
26	Flow IAT Min	Tiempo mínimo entre dos flujos.
27	Fwd IAT Total	Tiempo total entre dos paquetes enviados en la dirección de avance
28	Fwd IAT Mean	Tiempo medio entre dos paquetes enviados en la dirección de avance
29	Fwd IAT Std	Tiempo de desviación estándar entre dos paquetes enviados en la dirección de avance
30	Fwd IAT Max	Tiempo máximo entre dos paquetes enviados en la dirección de avance
31	Fwd IAT Min	Tiempo mínimo entre dos paquetes enviados en la dirección de avance.

32	Bwd IAT Total	Tiempo total entre dos paquetes enviados hacia atrás.
33	Bwd IAT Mean	Tiempo medio entre dos paquetes enviados hacia atrás
34	Bwd IAT Std	Tiempo de desviación estándar entre dos paquetes enviados hacia atrás
35	Bwd IAT Max	Tiempo máximo entre dos paquetes enviados hacia atrás
36	Bwd IAT Min	Tiempo mínimo entre dos paquetes enviados hacia atrás.
37	Fwd PSH Flags	Número de veces que se estableció el indicador PSH en paquetes que viajan en la dirección de avance (0 para UDP)
38	Bwd PSH Flags	Número de veces que se estableció el indicador PSH en paquetes que viajan en la dirección hacia atrás (0 para UDP)
39	Fwd URG Flags	Número de veces que se estableció el indicador URG en paquetes que viajan en la dirección de avance (0 para UDP)
40	Bwd URG Flags	Número de veces que el indicador URG se configuró en paquetes que se desplazan hacia atrás (0 para UDP)
41	Fwd Header Length	Total de bytes utilizados para los encabezados en la dirección de avance
42	Bwd Header Length	Total de bytes utilizados para los encabezados en la dirección hacia atraz
43	Fwd Packets/s	Número de paquetes hacia adelante por segundo

44	Bwd Packets/s	Número de paquetes hacia atrás por segundo
45	Min Packet Length	Longitud mínima de un flujo
46	Max Packet Length	Longitud máxima de un flujo
47	Packet Length Mean	Longitud media de un flujo
48	Packet Length Std	Longitud de desviación estándar de un flujo
49	Packet Length Variance	Tiempo mínimo de llegada entre paquetes.
50	FIN Flag Count	Número de paquetes con FIN
51	SYN Flag Count	Número de paquetes con SYN
52	RST Flag Count	Número de paquetes con RST
53	PSH Flag Count	Número de paquetes con PUSH
54	ACK Flag Count	Número de paquetes con ACK
55	URG Flag Count	Número de paquetes con URG
56	CWE Flag Count	Número de paquetes con CWE
57	ECE Flag Count	Número de paquetes con ECE
58	Down/Up Ratio	Descargar y subir el ratio
59	Average Packet Size	Tamaño promedio del paquete
60	Avg Fwd Segment Size	Tamaño promedio observado en la dirección hacia adelante
61	Avg Bwd Segment Size	Tamaño medio observado en la dirección hacia atrás.
62	Fwd Header Length	Longitud del encabezado en la direccion de avance
63	Fwd Avg Bytes/Bulk	Número promedio de bytes de velocidad masiva en la dirección de avance

64	Fwd Avg Packets/Bulk	Número promedio de paquetes a granel en la dirección de avance
65	Fwd Avg Bulk Rate	Número promedio de tasa de volumen en la dirección de avance
66	Bwd Avg Bytes/Bulk	Número promedio de bytes de velocidad masiva en la dirección hacia atrás
67	Bwd Avg Packets/Bulk	Número promedio de paquetes a granel en la dirección hacia atrás
68	Bwd Avg Bulk Rate	Número promedio de tasa de volumen en la dirección hacia atrás
69	Subflow Fwd Packets	El número promedio de paquetes en un subflujo en la dirección de avance
70	Subflow Fwd Bytes	El número promedio de bytes en un subflujo en la dirección de avance
71	Subflow Bwd Packets	El número promedio de paquetes en un subflujo en la dirección hacia atrás
72	Subflow Bwd Bytes	El número promedio de bytes en un subflujo en la dirección hacia atrás
73	Init_Win_bytes_forward	El número total de bytes enviados en la ventana inicial en la dirección de avance
74	Init_Win_bytes_backward	El número total de bytes enviados en la ventana inicial en la dirección hacia atrás
75	act_data_pkt_fwd	Recuento de paquetes con al menos 1 byte de carga útil de datos TCP en la dirección de avance
76	min_seg_size_forward	Tamaño mínimo del segmento observado en la dirección hacia adelante

77	Active Mean	Tiempo medio que un flujo estuvo activo antes de estar inactivo
78	Active Std	Tiempo de desviación estándar que un flujo estuvo activo antes de estar inactivo
79	Active Max	Tiempo máximo que un flujo estuvo activo antes de estar inactivo
80	Active Min	Tiempo mínimo que un flujo estuvo activo antes de estar inactivo
81	Idle Mean	Tiempo medio que un flujo estuvo inactivo antes de activarse
82	Idle Std	Tiempo de desviación estándar de un flujo inactivo antes de activarse
83	Idle Max	Tiempo máximo que un flujo estuvo inactivo antes de activarse
84	Idle Min	Tiempo mínimo que un flujo estuvo inactivo antes de activarse
85	Label	Charge, Jisut, Koler, Lockerpin, pletor, PornDroid, RansomBO, Simplocker,Svpeng y WannaLocker

Fuente: (Arash, Andi, Laya, & Ali, 2018)

Anexo 9: Lista de características de Dataset_Attack_2018

	Característica	Descripción
1	Source_IP	Ip de Origen
2	Source_Port	Puerto de origen
3	Destination_IP	IP de destino
4	Destination_Port	Puerto de destino
5	Protocol	Protocolo
6	Timestamp	marca del tiempo
7	Flow_Duration	Duración del flujo
8	Total_Fwd_Packets	Total de paquetes en la dirección de avance (de origen a destino).
9	Total_Backward_Packets	Total de paquetes hacia atrás (destino a origen).
10	Total_Length_of_Fwd_Packets	Tamaño total del paquete hacia adelante (de origen a destino).
11	Total_Length_of_Bwd_Packets	Tamaño total del paquete hacia atrás (destino a origen).
12	Fwd_Packet_Length_Max	Tamaño máximo del paquete en dirección hacia adelante
13	Fwd_Packet_Length_Min	Tamaño mínimo del paquete en dirección hacia adelante
14	Fwd_Packet_Length_Mean	Tamaño medio del paquete en dirección hacia adelante
15	Fwd_Packet_Length_Std	Tamaño de desviación estándar del paquete en dirección hacia adelante
16	Bwd_Packet_Length_Max	Tamaño máximo del paquete hacia atrás
17	Bwd_Packet_Length_Min	Tamaño mínimo del paquete hacia atrás
18	Bwd_Packet_Length_Mean	Tamaño medio del paquete hacia atrás

19	Bwd_Packet_Length_Std	Tamaño de desviación estándar del paquete en dirección hacia atrás
20	Flow_Bytes/s	Velocidad de bytes de flujo, es el número de paquetes transferidos por segundo.
21	Flow_Packets/s	Velocidad de paquetes de flujo que es el número de paquetes transferidos por segundo.
22	Flow_IAT_Mean	Tiempo medio entre dos flujos.
23	Flow_IAT_Std	Tiempo de desviación estándar dos flujos
24	Flow_IAT_Max	Tiempo máximo entre dos flujos.
25	Flow_IAT_Min	Tiempo mínimo entre dos flujos.
26	Fwd_IAT_Total	Tiempo total entre dos paquetes enviados en la dirección de avance
27	Fwd_IAT_Mean	Tiempo medio entre dos paquetes enviados en la dirección de avance
28	Fwd_IAT_Std	Tiempo de desviación estándar entre dos paquetes enviados en la dirección de avance
29	Fwd_IAT_Max	Tiempo máximo entre dos paquetes enviados en la dirección de avance
30	Fwd_IAT_Min	Tiempo mínimo entre dos paquetes enviados en la dirección de avance.
31	Bwd_IAT_Total	Tiempo total entre dos paquetes enviados hacia atrás.
32	Bwd_IAT_Mean	Tiempo medio entre dos paquetes enviados hacia atrás
33	Bwd_IAT_Std	Tiempo de desviación estándar entre dos paquetes enviados hacia atrás
34	Bwd_IAT_Max	Tiempo máximo entre dos paquetes enviados hacia atrás
35	Bwd_IAT_Min	Tiempo mínimo entre dos paquetes enviados hacia atrás.
36	Fwd_PSH_Flags	Número de veces que se estableció el indicador PSH en paquetes que viajan en la dirección de avance (0 para UDP)

37	Bwd_PSH_Flags	Número de veces que se estableció el indicador PSH en paquetes que viajan en la dirección hacia atrás (0 para UDP)
38	Fwd_URG_Flags	Número de veces que se estableció el indicador URG en paquetes que viajan en la dirección de avance (0 para UDP)
39	Bwd_URG_Flags	Número de veces que el indicador URG se configuró en paquetes que se desplazan hacia atrás (0 para UDP)
40	Fwd_Header_Length	Total de bytes utilizados para los encabezados en la dirección de avance
41	Bwd_Header_Length	Total de bytes utilizados para los encabezados en la dirección hacia atrás.
42	Fwd_Packets/s	Número de paquetes hacia adelante por segundo
43	Bwd_Packets/s	Número de paquetes hacia atrás por segundo
44	Min_Packet_Length	Longitud mínima de un flujo
45	Max_Packet_Length	Longitud máxima de un flujo
46	Packet_Length_Mean	Longitud media de un flujo
47	Packet_Length_Std	Longitud de desviación estándar de un flujo
48	Packet_Length_Variance	Tiempo mínimo de llegada entre paquetes.
49	FIN_Flag_Count	Número de paquetes con FIN
50	SYN_Flag_Count	Número de paquetes con SYN
51	RST_Flag_Count	Número de paquetes con RST
52	PSH_Flag_Count	Número de paquetes con PUSH
53	ACK_Flag_Count	Número de paquetes con ACK
54	URG_Flag_Count	Número de paquetes con URG
55	CWE_Flag_Count	Número de paquetes con CWE
56	ECE_Flag_Count	Número de paquetes con ECE
57	Down/Up_Ratio	Descargar y subir el ratio
58	Average_Packet_Size	Tamaño medio del paquete.
59	Avg_Fwd_Segment_Size	Tamaño promedio observado en la dirección hacia adelante

60	Avg_Bwd_Segment_Size	Tamaño medio observado en la dirección hacia atrás.
61	Fwd_Header_Length.1	Longitud del encabezado en la dirección de avance
62	Fwd_Avg_Bytes/Bulk	Número promedio de bytes de velocidad masiva en la dirección de avance
63	Fwd_Avg_Packets/Bulk	Número promedio de paquetes a granel en la dirección de avance
64	Fwd_Avg_Bulk_Rate	Número promedio de tasa de volumen en la dirección de avance
65	Bwd_Avg_Bytes/Bulk	Número promedio de bytes de velocidad masiva en la dirección hacia atrás
66	Bwd_Avg_Packets/Bulk	Número promedio de paquetes a granel en la dirección hacia atrás
67	Bwd_Avg_Bulk_Rate	Número promedio de tasa de volumen en la dirección hacia atrás
68	Subflow_Fwd_Packets	El número promedio de paquetes en un subflujo en la dirección de avance
69	Subflow_Fwd_Bytes	El número promedio de bytes en un subflujo en la dirección de avance
70	Subflow_Bwd_Packets	El número promedio de paquetes en un subflujo en la dirección hacia atrás
71	Subflow_Bwd_Bytes	El número promedio de bytes en un subflujo en la dirección hacia atrás
72	Init_Win_bytes_forward	El número total de bytes enviados en la ventana inicial en la dirección de avance
73	Init_Win_bytes_backward	El número total de bytes enviados en la ventana inicial en la dirección hacia atrás
74	act_data_pkt_fwd	Recuento de paquetes con al menos 1 byte de carga útil de datos TCP en la dirección de avance
75	min_seg_size_forward	Tamaño mínimo del segmento observado en la dirección hacia adelante
76	Active_Mean	Tiempo medio que un flujo estuvo activo antes de estar inactivo

77	Active_Std	Tiempo de desviación estándar que un flujo estuvo activo antes de estar inactivo
78	Active_Max	Tiempo máximo que un flujo estuvo activo antes de estar inactivo
79	Active_Min	Tiempo mínimo que un flujo estuvo activo antes de estar inactivo
80	Idle_Mean	Tiempo medio que un flujo estuvo inactivo antes de activarse
81	Idle_Std	Tiempo de desviación estándar de un flujo inactivo antes de activarse
82	Idle_Max	Tiempo máximo que un flujo estuvo inactivo antes de activarse
83	Idle_Min	Tiempo mínimo que un flujo estuvo inactivo antes de activarse
84	PKT_ID	Identificador de paquete
85	FROM_NODE	Definir cliente enviando paquete
86	TO_NODE	Definir el paquete receptor del cliente
87	PKT_SIZE	Tamaño total del paquete en bytes
88	SEQ_NUMBER	Secuencia de números
89	NUMBER_OF_PKT	Número total de paquetes
90	NUMBER_OF_BYTE	Número total de bytes
91	PKT_IN	Tiempo total del paquete dentro de la cola
92	PKT_OUT	Tiempo total del paquete fuera de la cola
93	PKT_R	Tiempo de paquete recibido
94	PKT_DELAY_NODE	Tasa media de paquetes
95	PKT_RATE	Velocidad media de bytes
96	BYTE_RATE	Tamaño promedio del paquete
97	UTILIZATION	Utilización del ancho de banda
98	PKT_DELAY	Tiempo total de retraso del paquete
99	PKT_SEND_TIME	Hora de envío del paquete.
100	PKT_RESEVED_TIME	Tiempo de recepción del paquete
101	FIRST_PKT_SENT	Hora del primer paquete enviado.
102	LAST_PKT_RESEVED	Hora del último paquete recibido

103	state	Indica el estado y su protocolo dependiente, por ejemplo, ACC, CLO, CON, ECO, ECR, FIN, INT, MAS, PAR, REQ, RST, TST, TXD, URH, URN, y (-) (si no se utiliza el estado)
104	sttl	Tiempo de origen a destino para valor de vida. Tiempo de origen a destino para el valor en vivo
105	dttl	destino al tiempo de origen para valor en vivo
106	sloss	paquetes fuente retransmitidos o descartados
107	dloss	paquetes de destino retransmitidos o descartados
108	service	http, ftp, smtp, ssh, dns, ftp-data, irc y (-) si no se usa mucho el servicio
109	Sload	Fuente de bits por segundo
110	Dload	bits de destino por segundo
111	swin	valor de anuncio de la ventana TCP de origen
112	dwin	valor de anuncio de la ventana TCP de destino
113	stcpb	número de secuencia base de TCP de origen
114	dtcpb	número de secuencia base TCP de destino
115	trans_depth	Representa la profundidad canalizada en la conexión de la transacción de solicitud / respuesta http
116	res_bdy_len	Tamaño de contenido real sin comprimir de los datos transferidos desde el servicio http del servidor.
117	Sjit	Fuente jitter (mSec).variabilidad del tiempo de ejecución de los paquete. Si el jitter es demasiado grande, no puede asegurarse que las informaciones críticas de proceso lleguen a tiempo.
118	Djit	jitter de destino (mSec)
119	Ltime	tiempo de inicio de registro
120	Sintpkt	Hora de llegada del paquete de origen (mSec)
121	Dintpkt	Hora de llegada del paquete intermedio de destino (mSec)
122	tcprtt	Tiempo de ida y vuelta de la configuración de la conexión TCP, la suma de 'synack' y 'ackdat’

123	synack	Tiempo de configuración de la conexión TCP, el tiempo entre los paquetes SYN y SYN_ACK
124	ackdat	Tiempo de configuración de la conexión TCP, el tiempo entre los paquetes SYN_ACK y ACK
125	is_sm_ips_ports	Si las direcciones IP de origen (1) y de destino (3) son iguales y los números de puerto (2) (4) son iguales, esta variable toma el valor 1 más 0.
126	ct_state_ttl	No. para cada estado (6) de acuerdo con el rango específico de valores para el tiempo de origen / destino para vivir (10) (11).
127	ct_flw_http_mthd	No. de flujos que tiene métodos como Get y Post en el servicio http.
128	is_ftp_login	Si se accede a la sesión ftp por usuario y contraseña, entonces 1 más 0.
129	ct_ftp_cmd	No de flujos que tiene un comando en la sesión ftp.
130	ct_srv_src	No. de conexiones que contienen el mismo servicio (14) y dirección de origen (1) en 100 conexiones según la última vez (26).
131	ct_srv_dst	No. de conexiones que contienen el mismo servicio (14) y dirección de destino (3) en 100 conexiones según la última vez (26).
132	ct_dst_ltm	No. de conexiones de la misma dirección de destino (3) en 100 conexiones según la última vez (26).
133	ct_src_ltm	No. de conexiones de la misma dirección de origen (1) en 100 conexiones según la última vez (26).
134	ct_src_dport_ltm	No de conexiones de la misma dirección de origen (1) y el puerto de destino (4) en 100 conexiones según la última vez (26).
135	ct_dst_sport_ltm	No de conexiones de la misma dirección de destino (3) y el puerto de origen (2) en 100 conexiones según la última vez (26).

136	ct_dst_src_ltm	No de conexiones de la misma fuente (1) y la dirección de destino (3) en 100 conexiones según la última vez (26).
137	SFH	Los SFH que contienen una cadena vacía o "about: blank" se consideran dudosos porque se debe tomar una acción sobre la información presentada. Además, si el nombre de dominio en SFHs es diferente del nombre de dominio de la página web, esto revela que la página web es sospechosa porque la información enviada rara vez es manejada por dominios externos.
138	popUpWidnow	Es inusual encontrar un sitio web legítimo que solicite a los usuarios que envíen su información personal a través de una ventana emergente. Por otro lado, esta función se ha utilizado en algunos sitios web legítimos y su objetivo principal es advertir a los usuarios sobre actividades fraudulentas o emitir un anuncio de bienvenida, aunque no se solicitó información personal a través de estas ventanas emergentes.
139	SSLfinal_State	Un ancla es un elemento definido por la etiqueta <a>. Esta característica se trata exactamente como "URL de solicitud". Sin embargo, para esta función, examinamos: 1. Si las etiquetas <a> y el sitio web tienen nombres de dominio diferentes. Esto es similar a la función de solicitud de URL. 2. Si el ancla no se enlaza con ninguna página web, por ejemplo: A. <a href=templace> B. <a href==##contentID> C. <a href==## skipva> D. <a href = "JavaScript :: void (0)">
140	Request_URL	La URL de solicitud examina si los objetos externos contenidos en una página web, como imágenes, videos y sonidos, se cargan desde otro dominio. En

		las páginas web legítimas, la dirección de la página web y la mayoría de los objetos incrustados dentro de la página web comparten el mismo dominio.
141	URL_of_Anchor	La existencia de HTTPS es muy importante para dar la impresión de legitimidad del sitio web, pero esto claramente no es suficiente. Los autores en (Mohammad, Thabtah y McCluskey 2012) (Mohammad, Thabtah y McCluskey 2013) sugieren verificar el certificado asignado con HTTPS, incluida la extensión del emisor del certificado de fideicomiso y la antigüedad del certificado. Las Autoridades de Certificación que están constantemente listadas entre los nombres más confiables incluyen: "GeoTrust, GoDaddy, Network Solutions, Thawte, Comodo, Doster y VeriSign". Además, al probar nuestros conjuntos de datos, encontramos que la edad mínima de un certificado acreditado es de dos años.
142	web_traffic	Esta función mide la popularidad del sitio web al determinar el número de visitantes y el número de páginas que visitan. Sin embargo, dado que los sitios web de suplantación de identidad (phishing) viven durante un breve período de tiempo, es posible que la base de datos de Alexa no los reconozca (Alexa the Web Information Company, 1996). Al revisar nuestro conjunto de datos, encontramos que en los peores escenarios, los sitios web legítimos se encuentran entre los 100.000 principales. Además, si el dominio no tiene tráfico o no está reconocido por la base de datos de Alexa, se clasifica como "Phishing". De lo contrario, se clasifica como “Sospechoso”.
143	URL_Length	Los phishers pueden usar una URL larga para ocultar la parte dudosa en la barra de direcciones.

		Los resultados mostraron que si la longitud de la URL es mayor o igual a 54 caracteres, la URL se clasifica como phishing
144	age_of_domain	Esta característica se puede extraer de la base de datos WHOIS (Whois 2005). La mayoría de los sitios web de phishing viven por un corto período de tiempo. Al revisar nuestro conjunto de datos, encontramos que la edad mínima del dominio legítimo es de 6 meses.
145	having_IP_Address	Se utiliza una dirección IP como alternativa del nombre de dominio en la URL.
146	Clase	Exploits: 1, Backdoor: 2, Smurf:3, UDP-Flood: 4, SIDDOS:5, HTTP-FLOOD:6, Ransomware:7, Phishing:8.

Fuente: Elaboración propia.

Anexo 10: Matriz de Confusión Multiclase de Máquinas de Soporte Vectorial usando PCYM

Predicción

Actual	Backdoor	Exploits	HTTP-Flood	Phishing	Ransomware	SIDDOS	Smurf	UDP-Flood
Backdoor	4	695	0	0	0	0	0	0
Exploits	0	13395	0	0	0	0	0	0
HTTP-Flood	0	0	1168	0	0	69	0	0
Phishing	0	0	0	212	0	0	0	0
Ransomware	0	0	0	0	104618	0	0	0
SIDDOS	0	0	1	0	0	1891	0	116
Smurf	0	0	18	0	0	140	1307	2344
UDP-Flood	0	0	4	0	0	7	44	60330

Fuente: Elaboración Propia

Anexo 11: Matriz de Confusión Multiclase del Perceptrón Multicapa usando PCYM

Predicción

Actual	Clase	Backdoor	Exploits	HTTP-Flood	Phishing	Ransomware	SIDDOS	Smurf	UDP-Flood
	Backdoor	5	694	0	0	0	0	0	0
	Exploits	1	13394	0	0	0	0	0	0
	HTTP-Flood	0	0	1168	0	0	69	0	0
	Phishing	0	0	0	212	0	0	0	0
	Ransomware	0	0	0	0	104618	0	0	0
	SIDDOS	0	0	0	0	0	1891	1	116
	Smurf	0	0	12	0	0	121	1318	2358
	UDP-Flood	0	0	0	0	0	0	19	60366

Fuente: Elaboración Propia.

Anexo 12: Código de implementación del algoritmo Máquinas de Soporte Vectorial

```
In [1]: import pandas as pd
        from sklearn.model_selection import cross_val_score
        from sklearn.model_selection import train_test_split
        import seaborn as sb
        import matplotlib.pyplot as plt
        from pycm import*
        import time
        from sklearn.multiclass import OneVsRestClassifier
        from sklearn.metrics import precision_score, accuracy_score
        import psutil
        from sklearn.svm import LinearSVC

In [2]: data = pd.read_csv('attackes_2018_finalito.csv')
        y = data['Etiqueta']
        X = data.drop('Etiqueta', axis = 1)

In [31]: X_train, X_test, y_train, y_test = train_test_split (X, y, test_size = 0.30, random_state = 0)

In [32]: clasificador = LinearSVC()
         print (clasificador)
         start = time.time()
         clasificador.fit(X_train,y_train)
         end = time.time()
         print('Tiempo del entrenamiento:{0:.2f}seg'.format(end - start))
         print('USO de CPU = {}'.format(psutil.cpu_percent()))
         memory =psutil.virtual_memory().used / (1024.0 ** 3)
         print('Consumo de Memoria Física= {}'.format(memory))

         y_pred = clasificador.predict(X_test)
         clasificador.score(X_test,y_test)

         LinearSVC(C=1.0, class_weight=None, dual=True, fit_intercept=True,
              intercept_scaling=1, loss='squared_hinge', max_iter=1000,
              multi_class='ovr', penalty='l2', random_state=None, tol=0.0001,
              verbose=0)
         Tiempo del entrenamiento:114.83seg
         USO de CPU = 57.9
         Consumo de Memoria Física= 4.979671478271484

Out[32]: 0.9815306686413076
```

Fuente: Elaboración Propia

Anexo 13: Código de Implementación del Algoritmo Perceptrón Multicapa

In [2]:
```
import pandas as pd
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import train_test_split
import seaborn as sb
import matplotlib.pyplot as plt
from pycm import*
import time
from sklearn.multiclass import OneVsRestClassifier
from sklearn.metrics import precision_score, accuracy_score
import psutil
from sklearn.neural_network import MLPClassifier
```

In [3]:
```
data = pd.read_csv('attackes_2018_finalito.csv')
y = data['Etiqueta']
X = data.drop('Etiqueta', axis = 1)
```

In [4]:
```
X_train, X_test, y_train, y_test = train_test_split (X, y, test_size = 0.30, random_state = 0)
```

In [5]:
```
clasificador = MLPClassifier()
print (clasificador)
start = time.time()
clasificador.fit(X_train,y_train)
end = time.time()
print('Tiempo del entrenamiento:{0:.2f}seg'.format(end - start))
print(psutil.cpu_percent())
print(psutil.virtual_memory().used)
print(psutil.cpu_times())
y_pred = clasificador.predict(X_test)
clasificador.score(X_test,y_test)
```

```
MLPClassifier(activation='relu', alpha=0.0001, batch_size='auto', beta_1=0.9,
       beta_2=0.999, early_stopping=False, epsilon=1e-08,
       hidden_layer_sizes=(100,), learning_rate='constant',
       learning_rate_init=0.001, max_iter=200, momentum=0.9,
       nesterovs_momentum=True, power_t=0.5, random_state=None,
       shuffle=True, solver='adam', tol=0.0001, validation_fraction=0.1,
       verbose=False, warm_start=False)
Tiempo del entrenamiento:141.52seg
44.3
4157788160
scputimes(user=3937.140625, system=2760.734375, idle=21343.3125, interrupt=84.296875, dpc=41.46875)
```

Out[5]:
```
0.9817721328804537
```

Fuente: Elaboración Propia