Universidad de Valladolid

Escuela de Ingeniería Informática

TRABAJO FIN DE GRADO

Grado en Ingeniería Informática
Mención en Tecnologías de la Información

Diseño de un modelo neuronal para la detección y la clasificación de intrusiones en redes informáticas

Alumno:
Hugo López Álvarez

Tutores:
Diego García Álvarez

...

Agradecimientos

...

Resumen

Resumen

Abstract

Abstract

Índice general

Lista de Figuras

Lista de Tablas

Capítulo 1

Introducción

Este documento corresponde con la memoria del Trabajo de Fin de Grado (TFG) del grado en Informática de la Universidad de Valladolid. Este trabajo se centra en la creación de un modelo neuronal capaz de detectar intrusiones en una red informática. La principal ventaja de utilizar un modelo neuronal para la detección de intrusiones en una red, frente a los algoritmos tradicionales (como firmas basadas en reglas o análisis estadísticos), radica en su capacidad para aprender patrones complejos y no lineales en los datos, lo que le permite identificar amenazas desconocidas o variantes de ataques existentes (zero-day attacks). Mientras que los métodos tradicionales dependen de reglas predefinidas y actualizaciones manuales para detectar intrusiones (limitándose a ataques conocidos), las redes neuronales pueden analizan grandes volúmenes de tráfico de red, detectando anomalías sutiles y correlaciones ocultas mediante capas de abstracción.

1.1. Explicación del problema

En la actualidad, los sistemas informáticos reciben muchos más ataques de denegación de servicio y de intrusión que hace unos años, esto se debe en parte a los avances en los modelos de IA.

Los sistemas informáticos enfrentan actualmente graves amenazas debido al uso malintencionado de la Inteligencia Artificial (IA) por parte de ciberdelincuentes. Una de las principales problemáticas es la automatización de ataques, donde herramientas basadas en IA permiten ejecutar campañas de ataques informáticos con mayor precisión y escala. Estas IAs pueden generar mensajes convincentes, imitar patrones de comportamiento legítimos y evadir medidas de seguridad tradicionales, lo que incrementa la frecuencia y sofisticación de los ataques.

Otro desafío crítico es la explotación de vulnerabilidades mediante IA, que acelera la identificación de fallos en sistemas sin intervención humana. Existen algoritmos de machine

learning que analizan grandes volúmenes de datos para descubrir brechas de seguridad en tiempo récord, facilitando ataques dirigidos incluso contra infraestructuras críticas como hospitales.

La IA también complica la defensa, ya que los sistemas de detección tradicionales no siempre pueden anticipar tácticas adaptativas generadas por algoritmos hostiles. Esto obliga a las organizaciones y empresas a invertir en soluciones de IA defensiva, como sistemas de respuesta autónoma. Sin embargo, esto genera una carrera tecnológica desigual donde actores maliciosos aprovechan herramientas accesibles y de bajo costo. La falta de regulación global agrava este escenario, dificultando la mitigación de riesgos asociados.

Además, los modelos neuronales son adaptativos: mejoran su precisión con el tiempo al entrenarse con nuevos datos, lo que es crucial en entornos dinámicos donde los ciberataques evolucionan rápidamente. Por ejemplo, pueden distinguir entre comportamientos legítimos inusuales (como un empleado accediendo a recursos fuera de horario) y actividades maliciosas (como filtración de datos), reduciendo falsos positivos. En cambio, los enfoques tradicionales suelen ser rígidos y requieren ajustes manuales frecuentes para mantener su eficacia.

Sin embargo, el uso de modelos nueronales para la defensa de los sistemas conlleva grandes desafíos, como la necesidad de grandes conjuntos de datos etiquetados y recursos computacionales intensivos. Aun así, en escenarios donde la sofisticación de los ataques supera las capacidades de detección convencionales, los modelos neuronales representan un salto cualitativo en proactividad y escalabilidad.

https://www.wsj.com/articles/the-ai-effect-amazon-sees-nearly-1-billion-cyber-threats-a-day-15434edd

1.2. Motivación

A continuación, se explica cual ha sido la motivación para realizar este proyecto. La motivación representa la fuerza impulsora o el conjunto de razones que justifican su inicio y continuidad. La motivación puede originarse de la necesidad de resolver un problema específico, aprovechar una oportunidad identificada, cumplir con requisitos normativos, alcanzar metas estratégicas o generar un impacto positivo

Durante mi formación universitaria en el Grado en Ingeniería Informática, como alumno de la mención de tecnologías de la información, he aprendido a administrar grandes sistemas de computación en aspectos como: la seguridad, la garantía de la información, la evaluación de dichos sistemas y el almacenamiento de los datos. Además de cierto componente de desarrollo de software.

Revisar

Sin embargo, uno de los conocimientos que no he podido adquirir durante mis estudios, es uno de los temas más importantes en la actualidad, la Inteligencia Artificial. Con el objetivo de expandir mis conocimientos sobre este tema, decidí implementar un modelo neuronal

que facilitase la detección de ataques a redes informáticas que tantas complicaciones está generando a los encargados de la administración de estos sistemas.

1.3. Objetivos del proyecto

En esta sección se listan los objetivos del proyecto, que constituyen las metas específicas, medibles, alcanzables, relevantes y con plazos definidos que se persiguen con la ejecución del mismo. Dichos objetivos describen los resultados concretos que se espera lograr al finalizar el proyecto y proporcionan un marco de referencia para la planificación, la ejecución, el seguimiento y la evaluación de su progreso.

- Investigar las mejores opciones de arquitectura y de elección de hiperparámetros.
- Entendimiento de los problemas que enfrentan los sistemas informáticos en la actualidad
- Generación de modelos basados en Deep Learning.
- Mitigar riesgos de seguridad, reduciendo los tiempos de respuesta ante incidentes.

1.4. Objetivos académicos

En esta sección se enumeran los objetivos académicos del presente estudio, los cuales representan las metas específicas, susceptibles de evaluación, realizables, pertinentes para el ámbito del conocimiento, que se pretenden alcanzar a través del desarrollo de este proyecto.

- Comprender el funcionamiento de los modelos neuronales
- Aprender las características de varios de los tipo de modelos neruonales que existen.
- Descubrir el potencial de las redes neuronales para optimizar y mejorar las tecnologías de la información, incluyendo la administración de sistemas.

1.5. Estrucutra de la memoria

Este documento se estructura de la siguiente forma:

Capítulo 2 Metodología: En este capítulo se definen cuales son las fases de la metodología CRISP-DM que se utiliza para entrenar los modelos que se deaarrollan en este proyecto. Se describe la aplicación de cada una de las seis fases al contexto específico del entrenamiento de redes neuronales.

Capítulo 3 Planificación: Este capítulo presenta la planificación del proyecto. Se definen los recursos necesarios, se identifican las tareas principales, se estiman los plazos y se establece el cronograma. También se aborda la gestión de riesgos inicial y la asignación de roles.

Capítulo 4 Entendimiento del problema: En este capítulo se describe el problema que el proyecto busca abordar. Se presenta el contexto, la relevancia y los objetivos generales.

Capítulo 5 Entendimiento de los datos: Este capítulo se dedica a la exploración y comprensión del conjunto de datos utilizado para el entrenamiento de los modelos desarrollados. Se describe la fuente, el formato, el tamaño y las variables de los datos. Se presenta un análisis exploratorio para identificar patrones, problemas de calidad y la distribución de las variables.

Capítulo 6 Modelos: En este capítulo se detallan los modelos de redes neuronales desarrollados y entrenados. Se describe la arquitectura, la justificación de su elección, los hiperparámetros, la función de pérdida y el optimizador. Se incluye la estrategia de entrenamiento y las métricas de evaluación.

Capítulo 7 Test: Este capítulo se centra en la evaluación final de los modelos entrenados. Se describe el conjunto de datos de prueba, el proceso de evaluación, la presentación de los resultados de las métricas y el análisis de las fortalezas y debilidades de los modelos.

Capítulo 8 Despliegue: Este capítulo aborda la fase de despliegue de los modelos entrenados. Se describe la integración en un entorno operativo, las consideraciones técnicas, los posibles desafíos y las estrategias de monitorización y mantenimiento.

Capítulo 9 Tecnologías utilizadas: En este capítulo se listan y describen las tecnologías de software y hardware empleadas en el proyecto. Se incluyen lenguajes de programación, bibliotecas de aprendizaje automático, herramientas de visualización y plataformas de seguimiento de experimentos.

Capítulo 10 Seguimiento del proyecto: Este capítulo describe cómo se realiza el seguimiento del progreso del proyecto. Se definen los indicadores clave de rendimiento, las metodologías de seguimiento, las herramientas de gestión y los mecanismos para la identificación y resolución de desviaciones.

Capítulo 11 Conclusiones: En este capítulo final se presentan las conclusiones del proyecto. Se resumen los principales hallazgos, se evalúa el cumplimiento de los objetivos académicos, se discuten las implicaciones de los resultados, las limitaciones y las posibles líneas de trabajo futuro.

Anexo A Manuales:

Anexo B Resumen de enlaces adicionales:

Capítulo 2

Metodología

En este capítulo se explica la metodología CRISP-DM (Cross-Industry Standard Process for Data Mining), que se utiliza en el desarrollo del resto del proyecto para alcanzar los objetivos propuestos.

La adopción de metodologías estructuradas es fundamental en el desarrollo de proyectos informáticos, puesto que proporcionan un marco sistemático para garantizar la calidad, eficiencia y trazabilidad del proyecto. En particular, metodologías como CRISP-DM, permiten: alinear objetivos técnicos con necesidades de negocio,reducir riesgos mediante fases iterativas y documentadas, y facilitar la colaboración entre equipos multidisciplinares.

Según algunos estudios, los proyectos que utilizan metodologías estandarizadas incrementan un 35 % su probabilidad de éxito, frente a aproximaciones *ad-hoc*, al minimizar desviaciones en costes y plazos [7]. En el ámbito de la ciberseguridad, donde los requisitos legales y técnicos son críticos, este enfoque metodológico resulta indispensable para asegurar soluciones robustas y auditables

2.1. CRISP-DM

La metodología CRISP-DM, es un marco de trabajo estandarizado para guiar proyectos de minería de datos y aprendizaje automático. Su estructura cíclica y flexible la hace aplicable en diversos dominios, desde marketing hasta ciberseguridad. Está compuesta por las siguientes fases:

1. Comprensión del negocio: La primera fase de CRISP-DM establece los cimientos estratégicos del proyecto mediante un proceso de alineación entre los objetivos técnicos y las necesidades organizacionales. Para lograr establecer los cimientos, se lleva a cabo un análisis exhaustivo del contexto empresarial para identificar los problemas clave que el proyecto debe abordar, así como las oportunidades de mejora que podrían aprovecharse. Se realiza

un proceso de recopilación y documentación de requisitos que involucra a todas las partes interesadas relevantes. El resultado de esta fase es una definición precisa del alcance del proyecto, que incluye no solo los objetivos cuantificables sino también los criterios de éxito que permitirán evaluar el impacto real de la solución propuesta. Además, se establecen las limitaciones operativas y estratégicas que condicionarán el desarrollo del proyecto, asegurando que todas las fases posteriores se ejecuten dentro de un marco bien definido y alineado con las prioridades organizacionales.

2. Comprensión de los datos: Esta fase se centra en el análisis detallado de los datos disponibles para el proyecto, con el objetivo de evaluar su idoneidad y calidad para abordar los problemas identificados en la fase anterior. Este proceso implica un examen minucioso de las diversas fuentes de información, su estructura y sus características fundamentales. Durante esta etapa, se identifican y documentan aspectos críticos como la complejidad de los datos, la presencia de posibles sesgos y la representatividad de la información en relación con los objetivos del proyecto. La comprensión profunda de los datos permite anticipar desafíos potenciales y establecer estrategias adecuadas para su tratamiento en fases posteriores. Además, esta fase proporciona perspectivas que pueden influir en decisiones técnicas importantes, como la selección de algoritmos o el diseño de características. El resultado es un conocimiento del potencial y las limitaciones de los datos disponibles, que sirve como base para las transformaciones que se realizan en la siguiente fase.

3. Preparación de los Datos: Se trata de una fase crítica donde los datos brutos se transforman en un conjunto adecuado para modelado. Esta etapa implica una serie de operaciones fundamentales que garantizan la calidad y consistencia de los datos que alimentan a los modelos analíticos. Las actividades realizadas en esta fase son cruciales para el éxito del proyecto, ya que determinan en gran medida la capacidad de los algoritmos para extraer patrones significativos y generar resultados confiables. Se aplican técnicas especializadas para abordar problemas comunes en los datos, asegurando que la información sea representativa, completa y se encuentre adecuadamente estructurada para los análisis posteriores. Cualquier deficiencia en la preparación de los datos puede comprometer significativamente la efectividad de las siguientes fases. Al finalizar este proceso, se obtiene un conjunto de datos optimizado que conserva la esencia de la información original mientras elimina ruido y distorsiones que podrían afectar negativamente a los resultados del modelado.

4. Modelado: Constituye el núcleo técnico del proceso CRISP-DM, donde se desarrollan y evalúan los algoritmos diseñados para extraer conocimiento de los datos preparados. Esta etapa comienza con la selección cuidadosa de las técnicas de modelado más apropiadas para los objetivos específicos del proyecto y las características de los datos disponibles. Durante el proceso de modelado, se exploran diferentes enfoques algorítmicos, ajustando meticulosamente sus parámetros para optimizar su rendimiento. En esta fase se incluyen procesos de validación diseñados para garantizar que los modelos desarrollados sean robustos y generalizables, capaces de mantener su efectividad cuando se enfrenten a datos nuevos y no vistos previamente. El modelado es un proceso iterativo que puede requerir volver a fases anteriores para refinar la preparación de datos o incluso reconsiderar algunos aspectos del planteamiento inicial del problema. El resultado de esta fase es uno o varios modelos validados que cumplen con los criterios de calidad establecidos y están listos para su evaluación en el contexto de los objetivos empresariales definidos inicialmente.

5. Evaluación: Esta fase representa un examen exhaustivo de los modelos desarrollados, contrastando su desempeño técnico con los objetivos empresariales establecidos en la primera fase del proyecto. Este proceso va más allá de las métricas estadísticas tradicionales para incorporar una valoración del impacto potencial de la solución propuesta. Durante la evaluación, se analiza minuciosamente la capacidad de los modelos para resolver el problema de negocio original, considerando tanto su precisión técnica como su aplicabilidad práctica en el contexto organizacional. Se identifican y documentan las limitaciones de los modelos, así como los posibles riesgos asociados a su implementación. Esta fase también incluye la validación de los resultados con las partes interesadas clave, asegurando que la solución cumpla con las expectativas y requisitos operativos. La evaluación termina con una decisión fundamentada sobre la idoneidad de los modelos para su implementación, junto con recomendaciones para su posible mejora o adaptación a escenarios futuros. También se valida su robustez en escenarios realistas.

6. Despliegue: Se trata de la fase final de CRISP-DM, esta se centra en la transición del modelo analítico desde un entorno de desarrollo a un sistema operativo donde pueda generar valor tangible para la organización. Este proceso implica una serie de actividades cuidadosamente planificadas que garantizan la integración efectiva de la solución en los procesos empresariales existentes. El despliegue incluye aspectos técnicos como la implementación de la infraestructura necesaria, el desarrollo de interfaces adecuadas y la creación de mecanismos de monitoreo continuo. También se ha de tener en cuenta la capacitación de los usuarios finales y la documentación exhaustiva de la solución, asegurando su adopción efectiva y su uso óptimo. La fase de despliegue también establece procesos para el mantenimiento y actualización periódica del modelo, puesto que las soluciones analíticas requieren evolución continua para mantener su relevancia y efectividad. Como en el resto de metodologçias, se implementan mecanismos para medir el impacto real de la solución una vez en producción, cerrando el ciclo al proporcionar retroalimentación valiosa que puede ser la base de futuros proyectos analíticos.

Como se ha explicado, CRISP-DM es una metodología iterativa, esto significa que los resultados de fases posteriores pueden revelar la necesidad de ajustes en etapas anteriores (como recolectar más datos o redefinir objetivos). Su enfoque estructurado minimiza riesgos y maximiza el valor entregado, siendo especialmente útil en proyectos complejos donde la alineación entre técnica y negocio es esencial.

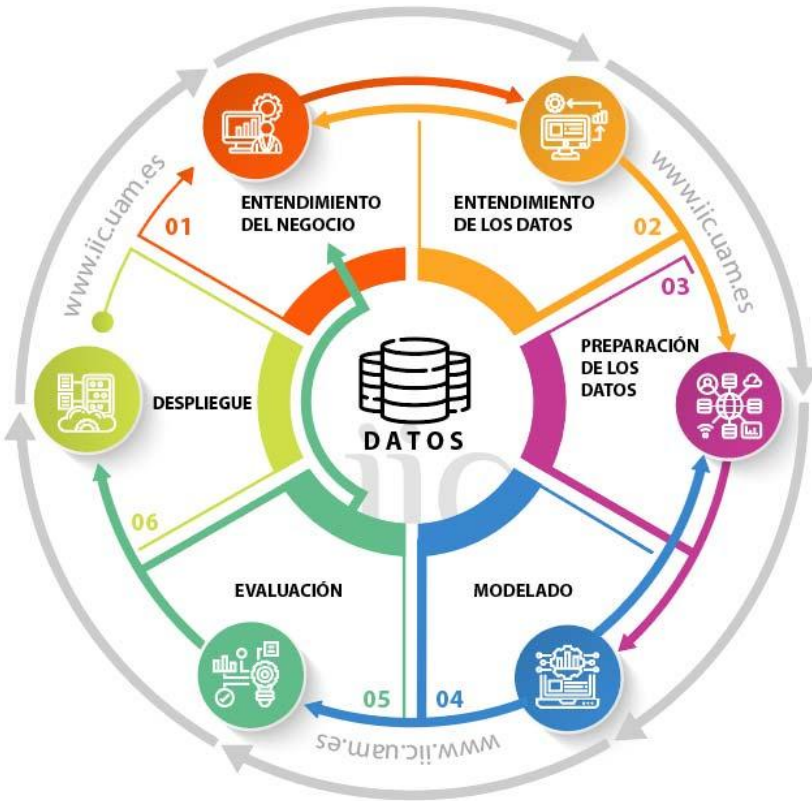

Figura 2.1: Esquema del ciclo CRISP-DM estándar. [1]

Capítulo 3

Planificación

Este capítulo aborda la organización detallada de un Trabajo de Fin de Grado, cubriendo desde su diseño inicial hasta la implementación y el seguimiento durante su desarrollo. Una planificación rigurosa resulta fundamental para sentar las bases del proyecto, ya que permite definir con claridad los objetivos, los recursos necesarios, los plazos de entrega y las actividades clave para alcanzar los resultados esperados.

En primer lugar, se establece una planificación temporal preliminar, donde se estiman los tiempos requeridos para cada etapa. Este cronograma se estructura en torno a las fases de la metodología CRISP-DM, complementadas con etapas específicas propias de un Trabajo de Fin de Grado. A continuación, se realiza un análisis de riesgos exhaustivo, evaluando tanto la probabilidad como el impacto de cada posible contingencia.

Además, se elabora un presupuesto detallado para las tareas del proyecto, abordado desde dos perspectivas. Por un lado, se incluye una estimación realista de los costes asociados a la ejecución del trabajo en el ámbito académico. Por otro lado, se plantea una proyección teórica de los gastos que implicaría un proyecto equivalente en un contexto profesional.

Por último, se contrasta la planificación inicial con el desarrollo real del trabajo, lo que permite evaluar posibles desviaciones y los aprendizajes obtenidos durante el proceso.

3.1. Planificación temporal

La planificación temporal constituye un elemento fundamental en la ejecución de un proyecto fin de grado, ya que permite estructurar de manera sistemática todas las actividades necesarias para alcanzar los objetivos propuestos. En el contexto de un trabajo académico que combine el desarrollo de software con una metodología de investigación, como es el caso de CRISP-DM para el proceso analítico y SCRUM para la gestión del proyecto, una adecuada planificación garantiza la distribución equilibrada del tiempo disponible entre las distintas

fases del trabajo. Esta organización temporal resulta especialmente relevante cuando se deben coordinar aspectos teóricos, desarrollo técnico y validación de resultados, asegurando que cada componente reciba la atención necesaria sin comprometer la calidad global del proyecto.

El empleo de un diagrama de Gantt como herramienta de planificación ofrece ventajas significativas para visualizar la secuencia de actividades y su superposición temporal. Este tipo de representación gráfica facilita la identificación de hitos críticos y dependencias entre tareas, aspectos particularmente importantes cuando se combinan metodologías diferentes como CRISP-DM y SCRUM. La primera, con sus fases bien definidas, proporciona la estructura para el desarrollo del núcleo analítico del proyecto, mientras que SCRUM, con sus sprints iterativos, permite adaptar el trabajo a los descubrimientos que vayan surgiendo durante la investigación. La integración de ambas aproximaciones en un único cronograma exige una cuidadosa coordinación que el diagrama de Gantt ayuda a materializar de forma clara y comprensible.

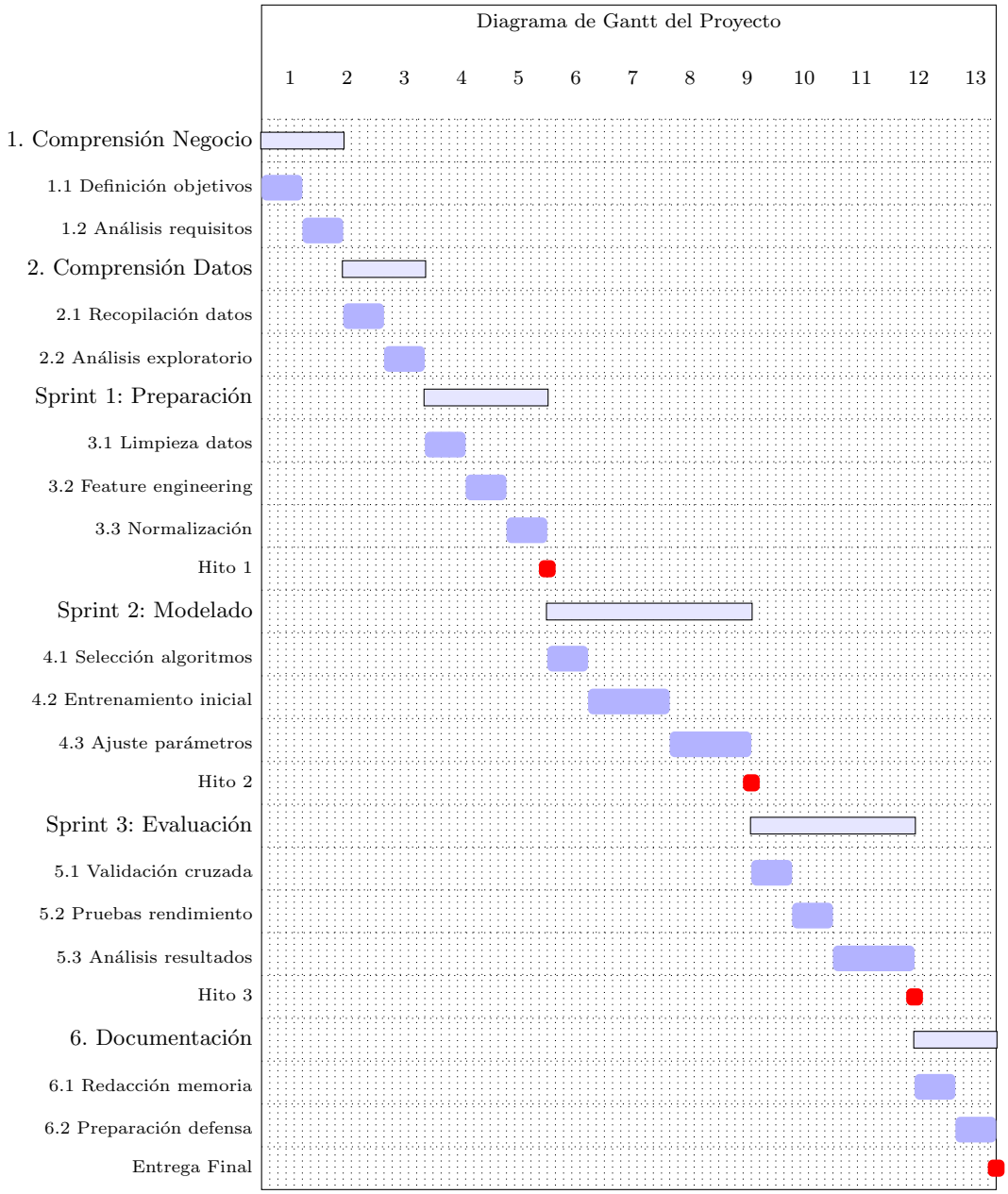

Figura 3.1: Diagrama de Gantt para la planificación del proyecto.

3.2. Gestión de riesgos

En este apartado se presentan los principales riesgos potenciales del proyecto junto con sus correspondientes planes de mitigación. De acuerdo con el *PMBOK (Project Management Body of Knowledge)* [8], los riesgos en gestión de proyectos se clasifican en las siguientes categorías:

Hitos del Proyecto

Hito	**Horas hasta el hito**	**Horas acumuladas**
Finalización de formación y trabajos previos	20	20
Finalización de la planificación inicial	30	50
Finalización de la comprensión de datos	40	90
Finalización del modelado	80	170
Obtención de modelos óptimos	40	210
Finalización y entrega de la memoria	90	300

Tabla 3.1: Cronograma de hitos y horas de trabajo

Tipología de Riesgos

1. **Riesgos Técnicos:**
   - Limitaciones en la infraestructura de hardware
   - Incompatibilidad entre sistemas o tecnologías
   - Problemas de rendimiento o escalabilidad
   - Deficiencias en el diseño o implementación de software
2. **Riesgos de Gestión:**
   - Deficiencias en la comunicación entre los miembros del equipo
   - Modificaciones en los requisitos del proyecto
   - Inestabilidad del equipo por conflictos internos o rotación de personal
   - Retrasos en la disponibilidad de recursos críticos
3. **Riesgos de Mercado:**
   - Aparición de competencia no anticipada

- Variaciones en las condiciones del mercado que afectan la demanda
- Cambios regulatorios que impactan la ejecución del proyecto

4. **Riesgos Financieros:**
   - Limitaciones en la disponibilidad de fondos¡
   - Excesos presupuestarios no previstos
   - Fluctuaciones en los tipos de cambio
5. **Riesgos Externos:**
   - Fenómenos meteorológicos adversos
   - Interrupciones en la cadena de suministro
   - Eventos naturales catastróficos

Metodología de Evaluación

La identificación y valoración de riesgos se realiza mediante criterios cualitativos, al no disponer de métricas cuantitativas suficientemente fiables para un análisis más exhaustivo. Este enfoque permite priorizar los riesgos según su impacto potencial y probabilidad de ocurrencia.

		Impacto				
		Mínimo	**Bajo**	**Medio**	**Alto**	**Extremo**
Probabilidad	**Extrema**					
	Alta					
	Media					
	Baja					
	Muy Baja					

Nivel de Riesgo	**Color**
Extremo	
Alto	
Moderado	
Bajo	
Mínimo	

Tabla 3.2: Matriz Probabilidad-Impacto

Probabilidad
Grado de posibilidad de que un riesgo se materialice. Se suele cuantificar en escala del 1 (muy improbable) al 5 (casi seguro). En la matriz, determina el eje vertical y se combina con el impacto para priorizar riesgos.

Impacto
Consecuencia o efecto potencial que tendría la materialización del riesgo. Se valora del 1 (impacto mínimo) al 5 (impacto catastrófico). Representa el eje horizontal en la matriz y mide la severidad del riesgo.

Plan de Mitigación
Acciones proactivas para reducir la probabilidad o impacto del riesgo antes de que ocurra. Incluye:

- Prevención: Eliminar las causas del riesgo.
- Reducción: Disminuir su probabilidad o impacto.
- Transferencia: Trasladar el riesgo a terceros.

Plan de Contingencia
Medidas reactivas que se implementan cuando el riesgo se materializa. Contiene:

- Activación: Criterios para ejecutar el plan.
- Respuesta: Acciones específicas de contención.
- Recuperación: Cómo volver a la normalidad.

Nivel de Riesgo
Resultado de multiplicar la probabilidad por el impacto. Clasifica riesgos en:

- Alto (15-25): Requieren acción inmediata.
- Medio (5-14): Necesitan monitoreo.
- Bajo (1-4): Aceptables con supervisión mínima.

Umbral de Riesgo
Límite máximo aceptable de riesgo para el proyecto. Determina cuándo se deben implementar planes de mitigación o contingencia.

Propietario del Riesgo
Persona o equipo responsable de monitorear cada riesgo y ejecutar los planes correspondientes.

Riesgos identificados

1. **R01**: Conflicto con periodos de exámenes y entregas de otras asignaturas.3.3
2. **R02**: Fallos hardware en equipos de desarrollo.3.4
3. **R03**: Limitaciones de capacidad de procesamiento para el entrenamiento de los modelos. 3.5
4. **R04**: Pérdida o corrupción de los datasets. 3.6
5. **R05**: Disponibilidad limitada del tutor académico.3.7
6. **R06**: Desviaciones en la planificación temporal inicial.3.8
7. **R07**: Cambios en los requisitos técnicos.3.9
8. **R08**: Dependencia de tecnologías inestables o no documentadas.3.10
9. **R09**: Dificultades en la integración de componentes.3.11
10. **R10**: Problemas de licencias de software.3.12

Riesgo R01	
Título	Conflicto con periodos de exámenes y entregas de otras asignaturas.
Descripción	Sobrecarga académica que dificulta la dedicación y el rendimiento en todas las tareas, aumentando el estrés y la presión estudiantil.
Probabilidad	4 (Alta)
Impacto	3 (Media)
Matriz P/I	Alta/Media (12)
Plan Mitigación	▪ Coordinar calendario académico anticipadamente. ▪ Avanzar trabajo en periodos de menor estrés.
Plan Contingencia	▪ Dedicar horas extra en los asuntos académicos. ▪ Reorganizar prioridades temporales.

Tabla 3.3: R01: Conflicto con periodos de exámenes y entregas de otras asignaturas.

Riesgo R02	
Título	Fallos hardware en equipos de desarrollo.
Descripción	Problemas causados por el mal funcionamiento de los componentes físicos de un ordenador, como la placa base, la tarjeta gráfica, la memoria RAM, el disco duro o la fuente de alimentación en equipos de desarrollo.
Probabilidad	3 (Media)
Impacto	4 (Alto)
Matriz P/I	Media/Alto (12)
Plan Mitigación	▪ Mantenimiento preventivo mensual. ▪ Uso de equipos redundantes.
Plan Contingencia	▪ Utilizar equipos alternativos. ▪ Acceder a laboratorios universitarios.

Tabla 3.4: R02: Fallos hardware en equipos de desarrollo.

Riesgo R03	
Título	Limitaciones de capacidad de procesamiento para el entrenamiento de los modelos.
Descripción	Restricciones de hardware (cálculo, memoria) que impactan la velocidad, viabilidad y calidad del entrenamiento, influyendo en el tamaño y complejidad de los modelos.
Probabilidad	4 (Alta)
Impacto	5 (Extremo)
Matriz P/I	Alto/Extremo (20)
Plan Mitigación	▪ Optimización temprana del código. ▪ Uso de técnicas de muestreo.
Plan Contingencia	▪ Utilizar servicios en la nube académicos. ▪ Reducir complejidad de modelos.

Tabla 3.5: R03: Limitaciones de capacidad de procesamiento para el entrenamiento de los modelos.

Riesgo R04	
Título	Pérdida o corrupción de los datasets.
Descripción	Extraviación o daño en los conjuntos de datos del proyecto que se utilizan para en entrenamiento y la validación del modelo.
Probabilidad	2 (Baja)
Impacto	5 (Extremo)
Matriz P/I	Baja/Extremo (10)
Plan Mitigación	▪ Almacenamiento redundante de los datos en distintos medios. ▪ Verificación de la inegridad de los datos con checksums.
Plan Contingencia	▪ Recuperar datasets desde backups externos. ▪ Regenerar datos sintéticos.

Tabla 3.6: R04: Pérdida o corrupción de los datasets.

Riesgo R05	
Título	Disponibilidad limitada del tutor académico.
Descripción	Restricciones de tiempo y acceso al profesor guía, afectando la frecuencia y profundidad de la retroalimentación y el apoyo al estudiante en su proceso de aprendizaje.
Probabilidad	3 (Media)
Impacto	3 (Media)
Matriz P/I	Media/Media (9)
Plan Mitigación	▪ Agendar reuniones con anticipación. ▪ Preparar preguntas concretas.
Plan Contingencia	▪ Consultar con profesores alternativos. ▪ Usar foros académicos.

Tabla 3.7: R05: Disponibilidad limitada del tutor académico

Riesgo R06	
Título	Desviaciones en la planificación temporal inicial.
Descripción	Variaciones o retrasos respecto al cronograma original, impactando los plazos de entrega, la gestión del tiempo y la consecución de los objetivos previstos.
Probabilidad	4 (Alta)
Impacto	4 (Alto)
Matriz P/I	Alto/Alto (16)
Plan Mitigación	▪ Incluir días asignados a descanso como días dedicados al proyecto. ▪ Revisiones semanales de progreso.
Plan Contingencia	▪ Reorganizar del diagrama de Gantt. ▪ Eliminar funcionalidades no críticas.

Tabla 3.8: R06: Desviaciones en la planificación temporal inicial.

Riesgo R07	
Título	Cambios en los requisitos técnicos.
Descripción	Modificaciones o alteraciones en las especificaciones necesarias para un proyecto o tarea, que pueden afectar al diseño, a la implementación, a los recursos y a los plazos.
Probabilidad	4 (Alta)
Impacto	4 (Alto)
Matriz P/I	Alto/Alto (16)
Plan Mitigación	▪ Documentar requisitos iniciales con precisión. ▪ Establecer procedimiento de cambio formal.
Plan Contingencia	▪ Revisar el alcance con tutor. ▪ Asignar tiempo adicional para cambios.

Tabla 3.9: R07: Cambios en los requisitos técnicos.

Riesgo R08	
Título	Dependencia de tecnologías inestables o no documentadas.
Descripción	Riesgos por la falta de fiabilidad, soporte o información clara, pudiendo generar problemas de funcionamiento, mantenimiento y escalabilidad del sistema.
Probabilidad	3 (Media)
Impacto	5 (Extremo)
Matriz P/I	Media/Extremo (15)
Plan Mitigación	▪ Investigar alternativas estables. ▪ Aislar componentes críticos.
Plan Contingencia	▪ Implementar soluciones temporales. ▪ Buscar soporte comunitario.

Tabla 3.10: R08: Dependencia de tecnologías inestables o no documentadas.

Riesgo R09	
Título	Dificultades en la integración de componentes.
Descripción	Problemas o complicaciones al combinar diferentes partes o sistemas, generando errores, incompatibilidades o un funcionamiento incorrecto del conjunto.
Probabilidad	3 (Media)
Impacto	4 (Alto)
Matriz P/I	Media/Alto (12)
Plan Mitigación	▪ Definir interfaces claras desde el inicio. ▪ Pruebas unitarias frecuentes.
Plan Contingencia	▪ Desarrollar adaptadores o intermediarios. ▪ Reimplementar componentes críticos.

Tabla 3.11: R09: Dificultades en la integración de componentes.

Riesgo R10	
Título	Problemas de licencia de software.
Descripción	Inconvenientes o restricciones legales relacionadas con el uso, la distribución o la activación de software, pudiendo causar interrupciones, costos adicionales o incluso acciones legales.
Probabilidad	2 (Baja)
Impacto	3 (Media)
Matriz P/I	Baja/Media (6)
Plan Mitigación	▪ Verificar licencias antes de usarlas para su implementación. ▪ Priorizar la utilización software open-source.
Plan Contingencia	▪ Buscar alternativas equivalentes. ▪ Solicitar licencias académicas.

Tabla 3.12: R10: Problemas de licencia de software.

3.3. Estimación de costes

En esta sección se presenta la estimación de costes, que comprende la identificación y valoración de los recursos necesarios para el desarrollo del proyecto. Este proceso implica la cuantificación de los gastos previsibles asociados a materiales, software específico y acceso a bases de datos. También se considera un coste el tiempo dedicado por el estudiante a la realización del trabajo..

La precisión en la estimación de costes facilita la elaboración de un presupuesto realista y la planificación financiera del proyecto. Permite anticipar las necesidades económicas, buscar posibles fuentes de financiación si fuese necesario y gestionar eficientemente los recursos disponibles. Una estimación detallada contribuye a evitar desviaciones presupuestarias y a asegurar la viabilidad económica del proyecto.

3.3.1. Costes materiales

A continuación, se hace un recuento de los costes materiales en hardware y software que han sido utilizados para el desarrollo de este proyecto.

Hardware

El hardware comprende el conjunto de componentes físicos y tangibles que constituyen un sistema informático. Proporciona la infraestructura física necesaria para la ejecución del software y el procesamiento de la información.

Para realizar este proyecto se han utilizado los siguientes componentes:

- **CPU**: AMD Ryzen 7 6800HS with Radeon Graphics (16) @ 4.785GHz
- **RAM**: 16GB SO-DIMM DDR5 4800MH
- **Memoria**: 512GB PCIe® 4.0 NVMe™ M.2 SSD
- **GPU1**: NVIDIA GeForce RTX 3050 Mobile
- **GPU2**: AMD ATI Radeon 680M

Debido al tamaño del conjunto de datos, el componente que más ha ralentizado el proyecto es la RAM, que en ciertas ocasiones durante el entrenamiento de los modelos se quedaba algo escasa en capacidad.

Teniendo en cuenta que el coste del ordenador en el momento de la compra fue de 829€, que la vida útil aproximada es de 8 años y que para el desarrollo de este proyecto se ha estado utilizando durante 3,5 meses, la amortización del hardware es:

$$\text{Amortización del Hardware} = 829\,€ \times \frac{1}{8} \times \frac{3{,}5}{12} = 30,22\,€ \tag{3.1}$$

Software

El software constituye el conjunto intangible de programas, datos e instrucciones que habilitan el funcionamiento de un sistema informático. Es el responsable de definir la funcionalidad, el comportamiento y la interacción del sistema con el usuario y con otros sistemas.

Funcionalidad	Software	Coste Mensual	Duración	Coste Total
Sistema Operativo (SO)	Kubuntu 24.10 x86_64	0 €	3,5 meses	0 €
Lenguaje (memoria)	Latex	0 €	3 meses	0 €
Editor latex	TexMaker	0 €	3 meses	0 €
IDE	MS Visual Studio Code	0 €	1 mes	0 €
Lenguaje (modelos)	Python	0 €	1 mes	0 €
IDE de Python	Jupyeter Notebooks	0 €	1 mes	0 €
Plataforma MLOps[1]	Weights&Biases	0 €	1 mes	0 €
Control de versiones	GitHub	0 €	1 mes	0 €
IA generativa (código)	DeepSeek	0 €	1 mes	0 €
IA generativa (memoria)	Gemini	0 €	2 mes	0 €
Comunicación 1	MS Outlook	0 € [2]	3,5 mes	0 €
Comunicación 2	MS Teams	0 €	3,5 mes	0 €

Tabla 3.13: Costes de Software

Debido a que el proyecto se realiza en un ámbito académico, se han minimizado los costes software del proyecto utilizando exclusivamente herramientas cedidas por la entidad académica o bien herramientas con licencia open-source que no suponen un coste monetario para el desarrollo del proyecto.

3.3.2. Costes humanos

Como proyecto académico, los costes humanos representan el valor del tiempo y el esfuerzo personal invertido en la planificación, investigación, redacción y presentación del trabajo. Estos costes se manifiestan en las horas dedicadas al proyecto, el esfuerzo intelectual requerido, el aplazamiento o anulación de otras actividades personales o profesionales y el estrés asociado al proceso.

En el caso de la simulación de los costes monetarios de un proyecto similar a este, sería necesario contar con personas cualificadas para los siguientes puestos:

- Ingeniero de Machine Learching
- Data Scientist

- Data Analyst

Fase 1: Definición y Preparación (75 horas)

Profesional	€/hora	Horas	Total (€)
Ingeniero ML	42	12	504
Científico Datos	34.5	38	1 311
Analista Datos	18.5	25	462.5
Total Fase 1		**75**	**2 277.5**

Tabla 3.14: Costes de Profesionales - Fase 1

Fase 2: Desarrollo y Entrenamiento (150 horas)

Profesional	€/hora	Horas	Total (€)
Ingeniero ML	42	95	3 990
Científico Datos	34.5	38	1 311
Analista Datos	18.5	17	314.5
Total Fase 2		**150**	**5 615,5**

Tabla 3.15: Costes de Profesionales - Fase 2

Fase 3: Validación y Evaluación (75 horas)

Profesional	€/hora	Horas	Total (€)
Ingeniero ML	42	30	1 260
Científico Datos	34.5	37	1 276.5
Analista Datos	18.5	8	148
Total Fase 3		**75**	**2 684.5**

Tabla 3.16: Costes de Profesionales - Fase 3

Coste Total del Proyecto (300 horas)

Profesional	Coste Total (€)
Ingeniero ML	5 754
Científico Datos	3 898,5
Analista Datos	925
Coste Total del Proyecto	**10 577,5**

Tabla 3.17: Coste Total por Profesional

Las estimaciones salariales proporcionadas se basan en los salarios en el sector tecnológico y de análisis de datos en España. Esta información se encuentra publicada en portales de empleo (Talent.com), escuelas de negocio (Aicad Business School, KSchool, Esden Business School) y noticias del sector (Tokio School).

Capítulo 4

Entendimiento del problema

En este capítulo se trata el entendimiento del problema. Tal y como se comenta en el capítulo dos, es la fase inicial de la metología CRISP-DM. A continuación, se alinean los objetivos técnicos con las necesidades del negocio y con el problema a resolver. Se definen requisitos, se identifican métricas de éxito y se trata de dar comprensión sobre el contexto organizacional.

4.1. ¿Qué es un ataque a un sistema informático?

Un ataque a un sistema informático constituye una acción deliberada y no autorizada que explota vulnerabilidades con el objetivo de comprometer la confidencialidad, integridad o disponibilidad de los datos y recursos del sistema. Esta actividad maliciosa puede manifestarse a través de diversas técnicas, incluyendo la inyección de código malicioso, la denegación de servicio, el acceso no autorizado y la ingeniería social. Su ejecución busca obtener beneficios ilícitos, interrumpir operaciones o dañar la infraestructura tecnológica.

La consecuencia de un ataque puede variar desde la pérdida o alteración de información sensible hasta la paralización completa de los servicios ofrecidos por el sistema. La identificación, análisis y mitigación de estas amenazas representan un aspecto fundamental en la seguridad informática, requiriendo la implementación de medidas preventivas y reactivas para proteger los activos digitales de una organización o individuo.

4.2. Tipos de ataque a sistemas informáticos

4.3. ¿Qué es TCP?

El Protocolo de Control de Transmisión (TCP) constituye uno de los protocolos fundamentales de la capa de transporte del modelo TCP/IP, sobre el cual se sustenta gran parte de la comunicación en redes IP, incluyendo Internet. Su diseño se orienta a proporcionar un servicio de transferencia de datos fiable, ordenado y con detección de errores entre aplicaciones que se ejecutan en sistemas finales diferentes. Para lograr esta fiabilidad, TCP establece una conexión virtual punto a punto entre las aplicaciones comunicantes mediante un proceso de "three-way handshake", lo que permite la negociación de parámetros de la conexión y la sincronización de los números de secuencia iniciales.

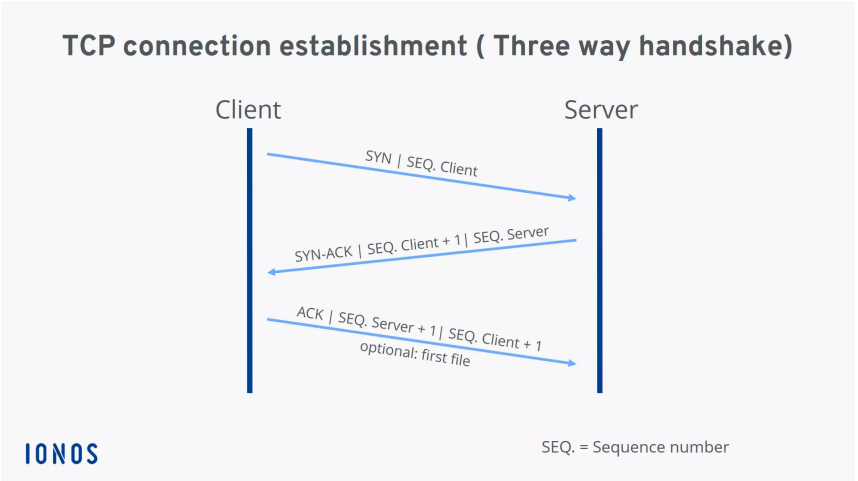

Figura 4.1: Esquema funcionamiento TCP. [2]

El protocolo garantiza la entrega ordenada de la información al receptor mediante la asignación de números de secuencia a cada byte transmitido, permitiendo así la reordenación en caso de que la información no llegue al receptor en el orden correcto. La fiabilidad se logra a través de un mecanismo de acuse de recibo (acknowledgment, ACK) positivo con retransmisión, donde el receptor confirma la recepción correcta de los paquetes de información, y el emisor retransmite aquellos partes de la información para los que no recibe confirmación dentro de un tiempo límite (timeout).

4.3.1. ¿Qué es un segmento TCP?

Una vez establecida la conexión, TCP divide los datos de la aplicación en unidades más pequeñas denominadas segmentos. Un segmento o paquete TCP constituye la unidad de datos fundamental que se intercambia a través de una red utilizando el mencionado protocolo TCP.

Este segmento encapsula una porción de los datos de la capa de aplicación, precedida por una cabecera TCP.

La cabecera TCP contiene información de control esencial para la funcionalidad del protocolo, incluyendo los números de puerto de origen y destino que identifican las aplicaciones comunicantes, los números de secuencia y de acuse de recibo (ACK) que garantizan la entrega ordenada y fiable, las banderas de control que indican el propósito del segmento (establecimiento de conexión, finalización, ACK, entre otros muchos), y otros campos como la ventana de recepción para el control de flujo y la suma de verificación para la detección de errores.

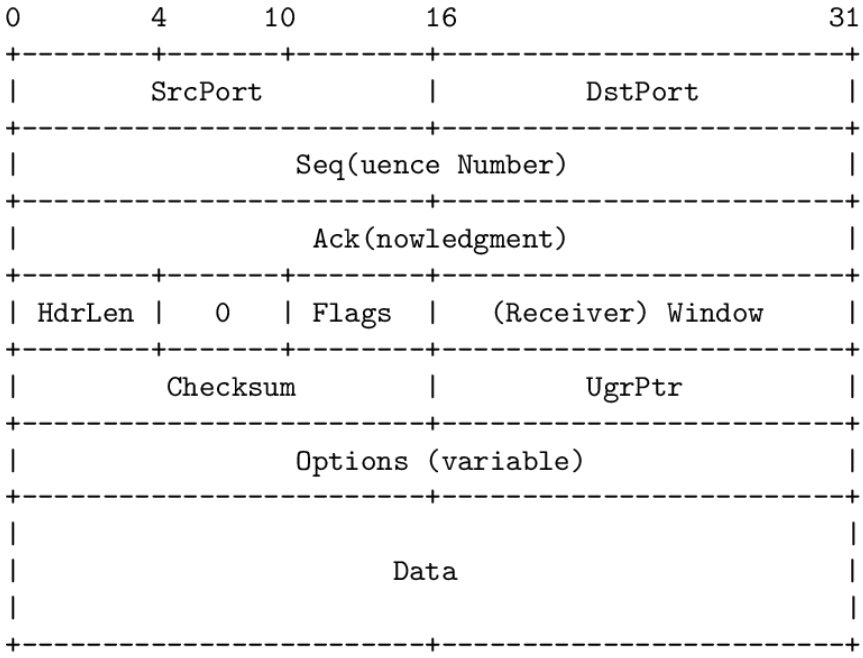

Figura 4.2: Esquema segmento TCP. [3]

En el proceso de transmisión, el segmento TCP se encapsula a su vez dentro de un paquete IP (Protocolo de Internet) para su enrutamiento a través de la red. El paquete IP añade su propia cabecera con las direcciones IP de origen y destino, entre otra información necesaria para el transporte a nivel de red.

4.4. Importancia de protegerse frente a un ataque

La importancia de protegerse frente a ataques informáticos radica en la salvaguarda de activos digitales críticos, la garantía de la continuidad operativa y la preservación de la confianza y la reputación. En un entorno digital cada vez más interconectado, los ataques informáticos representan una amenaza significativa para individuos, organizaciones y la sociedad en su conjunto, pudiendo acarrear consecuencias devastadoras.

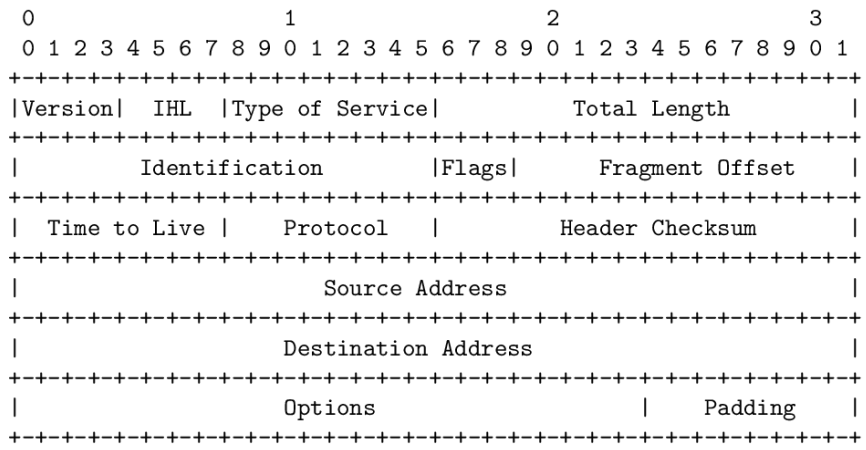

Figura 4.3: Formato de la cabecera en IPv4. [4]

Para las organizaciones, las implicaciones de un ataque informático pueden ser aún más costosas. Estas implicaciones incluyen pérdidas financieras directas debido al robo de fondos, la interrupción de las operaciones comerciales, los costes de recuperación y las posibles sanciones regulatorias. Además, se puede producir un daño significativo a la reputación y la pérdida de la confianza de los clientes, lo que a largo plazo afecta la viabilidad del negocio. Los ataques también pueden resultar en el robo de propiedad intelectual, secretos comerciales e información estratégica, otorgando ventajas competitivas a adversarios.

Por otra parte, la interrupción de servicios críticos, como energía, comunicaciones o sanidad, puede tener consecuencias graves para la sociedad en su conjunto.

La protección frente a ataques informáticos no es solo una cuestión de seguridad tecnológica, sino una necesidad imperante en la actualidad para proteger activos valiosos, asegurar la continuidad de las actividades, mantener la confianza de los usuarios y garantizar la estabilidad y el bienestar en el mundo digital actual. La implementación de prácticas de seguridad robustas y la concienciación sobre las amenazas cibernéticas son fundamentales en a la hora de defenderse de estos ataques.

4.5. Importancia de detectar los ataques rápidamente

La detección temprana de ataques informáticos constituye un pilar fundamental en la ciberseguridad moderna debido a su capacidad para mitigar consecuencias críticas. Cuando un sistema logra identificar intrusiones o actividades maliciosas en sus fases iniciales, se reducen significativamente los daños operativos y económicos. Esta rapidez de respuesta permite contener amenazas antes de que comprometan infraestructuras completas, preservando tanto la integridad de los datos como la continuidad del negocio.

Desde una perspectiva técnica, la identificación inmediata limita la superficie de ataque, impidiendo que los actores maliciosos escalen privilegios o se propaguen lateralmente por la

red. En el ámbito regulatorio, cumple con los estrictos plazos que exigen normativas como el Reglamento General de Protección de Datos (RGPD), que obliga a notificar violaciones de seguridad en un máximo de 72 horas. Además, desde el punto de vista económico, reduce los costes asociados a las reparaciones, que suelen multiplicarse exponencialmente cuando los ataques permanecen indetectados durante largos períodos.

La capacidad de detectar rápidamente anomalías en el tráfico de red, accesos no autorizados o patrones de comportamiento sospechosos no solo protege los activos digitales, sino que también salvaguarda la reputación institucional. Organizaciones con sistemas de detección temprana robustos demuestran proactividad ante clientes y socios comerciales, generando confianza en su capacidad para manejar información sensible. Esta anticipación resulta especialmente crítica en entornos donde la disponibilidad del servicio es primordial, como en las infraestructuras críticas anteriormente mencionadas.

4.6. Soluciones comerciales o actuales a estos problemas

En esta sección se comentan algunas de las soluciones y software que se utilizan en la actualiadad para detectar y neutralizar posibles ataques informáticos. Estas herramientas protegen los sistemas informáticos analizando y controlando el tráfico de la red.

Los firewalls de próxima generación (NGFW) como Palo Alto Networks, Check Point o Cisco Firepower, inspeccionan el tráfico de red a un nivel profundo (Deep Packet Inspection - DPI), analizando el contenido de los paquetes más allá de los puertos y protocolos tradicionales. Esto permite identificar y bloquear amenazas sofisticadas, malware, y tráfico de aplicaciones maliciosas, además de ofrecer funcionalidades como prevención de intrusiones (IPS) y control de aplicaciones. [9]

Los sistemas de detección y prevención de intrusiones o IDS e IPS, como: Snort, Suricata o Trend Micro TippingPoint, monitorean el tráfico de red en tiempo real en busca de patrones sospechosos o firmas de ataques conocidos. Los IDS alertan sobre posibles intrusiones, mientras que los IPS tienen la capacidad de bloquear o mitigar activamente el tráfico malicioso detectado, interrumpiendo los ataques en curso. [10]

La microsegmentación de red con herramientas como VMware NSX, Cisco ACI o Illumio, divide la red en segmentos más pequeños y aislados, aplicando políticas de seguridad granular a cada segmento. Esto limita el movimiento lateral de los atacantes dentro de la red una vez que han comprometido un punto inicial. Al controlar el tráfico entre estos segmentos, se reduce la superficie de ataque y se contiene la propagación de las amenazas. [11]

4.7. Requisitos

Como se ha comentado en la sección 1.3Objetivos del proyecto, el principal objetivo del proyecto es desarrollar un modelo neuronal que detecte la presencia de ataques en una red

informática y los clasifique según su tipo. Para cumplir con dicho objetivo, se considera imprescindible cumplir con los requisitos que se listan a continuación.

4.7.1. Requisitos Funcionales

Primera versión de requisitos, no me convencen mucho

- **RF-1**: El sistema deberá detectar cuales de las conexiones podrían ser potenciales intrusiones en la red.
- **RF-2**: El sistema deberá clasificará las conexiones en 10 categorías predefinidas en Clasificación de amenazas de seguridad.
- **RF-3**: El sistem deberá ser capaz de procesar formatos estándar de logs como son Syslog, NetFlow y PCAP.
- **RF-4**: El sistema deberá diferenciar entre ataques conocidos (basados en firmas) y desconocidos (basados en anomalías).
- **RF-5**: El sistema deberá ofrecer API REST para conexión con SIEMs (Splunk, IBM QRadar)
- **RF-6**: Generar alertas automatizadas con nivel de criticidad (bajo/medio/alto).
- **RF-7**: Proveer recomendaciones de mitigación básicas (ej. bloquear IPs maliciosas)

  ¿Debería integrar el modelo en algún sistema o crear un script o alguna forma para comunicarme con él?

4.7.2. Requisitos No Funcionales

- **RNF-1**: Latencia ¡50 ms en redes de 10Gbps (requisito crítico para SOC [12]).
- **RNF-2**: Interfaz accesible para usuarios no técnicos (evaluado con test SUS [13]).

4.8. Contexto organizacional

4.9. Objetivos del proyecto

Capítulo 5

Entendimiennto de los datos

Este capítulo se corresponde con la segunda etapa de la metodología CRISP-DM, En el se explicará la naturaleza de los datos y sus características, así como los valores atípicos que presentan y sus sesgos.

5.1. Origen de los datos

Los datos que se han utilizado para desarrollar este trabajo, se han obtenido de conjuntos de datos diseñados para entrenar Sistemas de Detección de Intrusión de Red (NIDS) basados en el aprendizaje automático. El dataset en cuenstión forma parte de un análisis realizado en la Universidad de Queensland, Australia.[14]

El dataset utilizado es NF-UNSW-NB15-v3, este es una versión basada en NetFlow del conocido conjunto de datos UNSW-NB15, mejorada con características adicionales de NetFlow y etiquetada de acuerdo con sus respectivas categorías de ataque.

5.2. Tipos de ataques registrados en los datos

En esta sección se explican los tipos de datos presentes en el Dataset que se utiliza para entrenar a los modelos del proyecto. Se explica en que consiste cada tipo de ataque registrado así como el número exacto de ataques de cada tipo presente.

El conjunto de datos consiste en un total de 2 365 424 flujos de datos, donde 127 639 (5,4 %) son muestras de ataque y 2 237 731 (94,6 %) son benignos. Los flujos de ataque se clasifican en nueve clases, cada una representando una amenaza a la red distinta. La siguiente tabla proporciona una distribución detallada del conjunto de datos:

Clase	Cantidad	Descripción
Benigno	2 237 731	Flujos normales no maliciosos.
Fuzzers	33 816	Tipo de ataque en el que el atacante envía grandes cantidades de datos aleatorios que hacen que un sistema se bloquee y también apuntan a descubrir vulnerabilidades de seguridad en un sistema.
Analysis	2 381	Un grupo que presenta una variedad de amenazas que se dirigen a aplicaciones web a través de puertos, correos electrónicos y scripts.
Backdoor	1 226	Una técnica que tiene como objetivo eludir los mecanismos de seguridad respondiendo a aplicaciones específicas de clientes construidos.
DoS	5 980	La denegación de servicio es un intento de sobrecargar los recursos de un sistema informático con el objetivo de evitar el acceso o la disponibilidad de sus datos.
Exploits	42 748	Son secuencias de comandos que controlan el comportamiento de un host a través de una vulnerabilidad conocida.
Generic	19 651	Un método que se dirige a la criptografía y causa una colisión con cada cifrado de bloques.
Reconnaissance	17 074	Una técnica para recopilar información sobre un host de red, también se conoce como sonda.
Shellcode	4 659	Un malware que penetra en un código para controlar el host de una víctima.
Worms	158	Ataques que se replican y se extienden a otros sistemas.

Tabla 5.1: Clasificación de amenazas de seguridad

5.3. Parámetros de los datos

En esta sección se explicarán el significado de cada uno de los 55 parámetros que componen cada fila del Dataset seleccionado.

`https://arxiv.org/pdf/2503.04404`

- **IPV4_SRC_ADDR**: Dirección IPv4 de origen.
- **IPV4_DST_ADDR**: Dirección IPv4 de destino.

- **L4_SRC_PORT**: Número de puerto de origen de la capa 4.
- **L4_DST_PORT**: Número de puerto de destino de la capa 4.
- **PROTOCOL**: Byte identificador del protocolo IP.
- **L7_PROTO**: Protocolo de aplicación (numérico) de la capa 7.
- **IN_BYTES**: Número de bytes entrantes.
- **OUT_BYTES**: Número de bytes salientes.
- **IN_PKTS**: Número de paquetes entrantes.
- **OUT_PKTS**: Número de paquetes salientes.
- **FLOW_DURATION_MILLISECONDS**: Duración del flujo en milisegundos.
- **TCP_FLAGS**: Acumulativo de todos los flags TCP.
- **CLIENT_TCP_FLAGS**: Acumulativo de todos los flags TCP del cliente.
- **SERVER_TCP_FLAGS**: Acumulativo de todos los flags TCP del servidor.
- **DURATION_IN**: Duración del flujo Cliente a Servidor (mseg).
- **DURATION_OUT**: Duración del flujo Cliente a Servidor (mseg).
- **MIN_TTL**: TTL mínimo del flujo.
- **MAX_TTL**: TTL máximo del flujo.
- **LONGEST_FLOW_PKT**: Paquete más largo (bytes) del flujo.
- **SHORTEST_FLOW_PKT**: Paquete más corto (bytes) del flujo.
- **MIN_IP_PKT_LEN**: Longitud del paquete IP más pequeño del flujo observado.
- **MAX_IP_PKT_LEN**: Longitud del paquete IP más grande del flujo observado.
- **SRC_TO_DST_SECOND_BYTES**: Bytes/segundo de origen a destino.
- **DST_TO_SRC_SECOND_BYTES**: Bytes/segundo de destino a origen.
- **RETRANSMITTED_IN_BYTES**: Número de bytes TCP retransmitidos del flujo (src a dst).
- **RETRANSMITTED_IN_PKTS**: Número de paquetes TCP retransmitidos del flujo (src a dst).
- **RETRANSMITTED_OUT_BYTES**: Número de bytes TCP retransmitidos del flujo (dst a src).
- **RETRANSMITTED_OUT_PKTS**: Número de paquetes TCP retransmitidos del flujo (dst a src).

- **SRC_TO_DST_AVG_THROUGHPUT**: Tasa de transferencia promedio de origen a destino (bps).
- **DST_TO_SRC_AVG_THROUGHPUT**: Tasa de transferencia promedio de destino a origen (bps).
- **NUM_PKTS_UP_TO_128_BYTES**: Paquetes cuyo tamaño IP es ¡= 128.
- **NUM_PKTS_128_TO_256_BYTES**: Paquetes cuyo tamaño IP es ¿128 y ¡= 256.
- **NUM_PKTS_256_TO_512_BYTES**: Paquetes cuyo tamaño IP es ¿256 y ¡= 512.
- **NUM_PKTS_512_TO_1024_BYTES**: Paquetes cuyo tamaño IP es ¿512 y ¡= 1024.
- **NUM_PKTS_1024_TO_1514_BYTES**: Paquetes cuyo tamaño IP es ¿1024 y ¡= 1514.
- **TCP_WIN_MAX_IN**: Ventana TCP máxima (src a dst).
- **TCP_WIN_MAX_OUT**: Ventana TCP máxima (dst a src).
- **ICMP_TYPE**: Tipo ICMP * 256 + Código ICMP.
- **ICMP_IPV4_TYPE**: Tipo ICMP.
- **DNS_QUERY_ID**: ID de transacción de la consulta DNS.
- **DNS_QUERY_TYPE**: Tipo de consulta DNS (ej., 1=A, 2=NS..).
- **DNS_TTL_ANSWER**: TTL del primer registro A (si existe).
- **FTP_COMMAND_RET_CODE**: Código de retorno del comando del cliente FTP.
- **FLOW_START_MILLISECONDS**: Marca de tiempo de inicio del flujo en milisegundos.
- **FLOW_END_MILLISECONDS**: Marca de tiempo de fin del flujo en milisegundos.
- **SRC_TO_DST_IAT_MIN**: IAT mínimo (src a dst).
- **SRC_TO_DST_IAT_MAX**: IAT máximo (src a dst).
- **SRC_TO_DST_IAT_AVG**: IAT promedio (src a dst).
- **SRC_TO_DST_IAT_STDDEV**: Desviación estándar del IAT (src a dst).
- **DST_TO_SRC_IAT_MIN**: IAT mínimo (dst a src).
- **DST_TO_SRC_IAT_MAX**: IAT máximo (dst a src).
- **DST_TO_SRC_IAT_AVG**: IAT promedio (dst a src).
- **DST_TO_SRC_IAT_STDDEV**: Desviación estándar del IAT (dst a src).
- **Label**: Toma valor 0 si el flujo es benigno y 1 si se trata de un ataque.
- **Attack**: Clase de flujo, perteneciente a las clases mencionadas en 5.1 Clasificación de amenazas de seguridad

5.4. Patrones preliminares, valores atípicos y sesgos

En esta sección se comentan los patrones prelinimares, así como los valores y los sesgos presentes en el Dataset. La identificación de estas irregularidades es fundamental para el correcto desarrollo del proyecto.

Al analizar los datos originales del dataset, se encuentran características que afectan de forma negativa al entrenamiento del modelo y por lo tanto, a su correcto funcionamiento. A continuación, se mencionan cuales son las características problematicas de los datos que se utilizan.

Tras estudiar los datos, se descubre que algunos parámetros presentan valores infinitos, estos valores alteran la distribución inherente de las variables, introduciendo sesgos significativos en el proceso de aprendizaje. Los algoritmos de entrenamiento, diseñados para operar con valores numéricos finitos, pueden comportarse de manera impredecible o inestable ante la presencia de infinitos, dificultando la convergencia hacia una solución óptima. Asimismo, la interpretación de las características con valores infinitos se vuelve ambigua, comprometiendo la capacidad del modelo para establecer relaciones significativas con otras variables y para generalizar correctamente a datos futuros que no contengan tales valores extremos. En consecuencia, la presencia de infinitos puede degradar sustancialmente el rendimiento y la fiabilidad del modelo entrenado.

Como se comenta en la sección anterior 5.3 Parámetros de los datos, existen dos parámetros que registran la IP origen y la IP destino de la conexión. En principio, si los datos no fuesen sintéticos, la dirección IP de la máquina que origina la comunicación sería aleatoria. En este conjunto de datos, se puede observar como todos los ataques provienen de IP con máscara 175.45.176.255, lo que no encaja con la realiadad. Independientemente de este patrón, que no se manifiesta de forma natural en las conexiones entre sistemas, el uso de las IPs de las máquinas en el entranamiento de los modelos provoca que el algoritmo encargado de este entrenamiento asigne pesos de manera incorrecta a un parámetro que no influye en la clasificación que se propone para este trabajo.

5.5. Preparación de los datos ¿SEPARAR EN OTRO CAPÍTULO?

En esta sección se modificarán los datos que presentan patrones, valores atípicos o sesgos que se comentan en la sección anterior. 5.4 Patrones preliminares, valores atípicos y sesgos. Así como el tratamiento necesario de los datos para poder entrenar los modelos.

Una de las características problemáticas, es la existencia de parámetros con valores infinito. Para tratar con esta problemática existen dos enfoques, o bien se sustituyen los valores infinitos por el máximo valor posible para ese parámetro, o bien, se elimina el flujo al que pertenece cada valor infinito. Para este trabajo, se opta por ...

5.6. DECIDIR SI ELIMINAR LOS DATOS CON VALORES INFINITOS O NO

Para entrenar un modelo es necesario que en primer lugar todos los parámetros sean numéricos. En los datos originales del dataset, se encuentran tres parámetros que no presentan un formato numérico: IPV4_SRC_ADDR, IPV4_DST_ADDR y Attack.

- Para dar un formato correcto al parámetro Attack, se utiliza el codificador LabelEncoder de la biblioteca sklearn.preprocessing y el método fit_transform de la biblioteca sklearn. La primera función, LabelEncoder codifica las etiquetas de características categóricas en valores numéricos entre 0 y el número de clases menos 1. Una vez se instancian las categorías del parámtero, el método fit_transform nos permite entrenar el codificador y transformar el conjunto de datos en un único paso.
- Las direcciones IPv4 están formadas por 4 octetos separados por puntos. Obviamnete este formato no es numérico, por ese motivo se opta por dividir las direcciones IPv4 en 4 parámetros diferentes, uno por cada octeto. De esta manera, si el valor de un flujo para el parámetro IPV4_SRC_ADDR es 175.45.176.23, se separa en cuatro nuevos valores que son: 175, 45, 176 y 23. Para realizar esta transformación se utiliza la función 5.1 Función de transformación para los parámetros IPv4.

```python
def split_ip_column(df, ip_column_name):

    # Divide la IP en cuatro partes
    ip_parts = df[ip_column_name].str.split('.', expand=True)

    # Crea nombres de columnas basados en el nombre original
    new_columns = {
        0: f"{ip_column_name}_part1",
        1: f"{ip_column_name}_part2",
        2: f"{ip_column_name}_part3",
        3: f"{ip_column_name}_part4"
    }

    # Se elimina la columna de ip_column_name
    df = df.drop(columns=[ip_column_name])

    # Añade las nuevas columnas al DataFrame
    for part, col_name in new_columns.items():
        df[col_name] = pandas.to_numeric(ip_parts[part])  # Convierte a numérico

    return df
```

Figura 5.1: Función de transformación para los parámetros IPv4

Una vez que todos los parámetros presentan valores numéricos, se separan los parámetros ente los que formarán parte de la entrada del modelo neuronal (a partir de ahora se les menciona X) y los que contienen el valor de la salida que debe proporcionar el modelo neuronal (a partir de ahora se les menciona Y).

Los parámetros que pertenecen a Y son Label y Attack. Sin embargo, en función del modelo que se entrena se utiliza o bien Label, o bien Attack, pero nunca los dos al mismo tiempo.

Por su parte, X la conforman el total de los parámetros del Dataseet a excepción de:

- **Label y Attack**: Se tratan de los parámetros que conforman Y, si estos parámetros formasen parte de los datos de entrada del modelo, el algoritmo encargado de entrenarlo, les asignaría a estos parámetros unos pesos muy elevados y tendría una tasa teórica de éxito del 100 %, puesto que, se estaría introduciendo la respuesta al problema que se aborda directamente como entrada al modelo. En un sistema informático real esta práctica es imposible.
- **IPV4_SRC_ADDR y IPV4_DST_ADDR**: Tal y como se menciona en la sección 5.3 Parámetros de los datos, estos parámetros presentan un patrón irregular y sintético. Al introducir estos parámetros como entradas para el entrenamiento del modelo, de forma análoga a como sucede con los parámetros Label y Attack se estarían introduciendo valores que no se pueden obtener en una situación realista y condicionando el comportamiento del modelo. Esto implica como se menciona en los siguiente capítulos que el modelo resultante obtendría muy buenas métricas en validación pero muy malas al someterlo a datos que nunca ha visto durante su entrenamiento.
- **FLOW_START_MILLISECONDS y FLOW_END_MILLISECONDS**: A diferencia de los anteriores parámetros, estos muestran valores que puedan condicionar el entrenamiento de los modelos ni sus resultados, pero se trata de parámetros que ofrecen información redundante, pues el parámetro FLOW_DURATION_MILLISECONDS es combinación lineal de estos dos. Al eliminar estos parámetros se agiliza el entrenamiento del modelo y su tiempo de respuesta una vez entrenado sin perjudicar su precisión.

Para finalizar con el tratamiento de los datos, se normalizan los valores de X para que no haya tanta varianza entre los parámetros que lo conforman. Para normalizar los valores se utiliza el escalador MinMaxScaler que forma parte de la biblioteca sklearn.preprocessing junto con el método fit_transform que se menciona en la explicación de la transformación del parámetro Attck. Llamando a MinMaxScaler con un rango (0,1), tras haber tratado los datos con el método fit_transform, se obtienen valores escalados en un rango entre 0 y 1, lo que facilita la comparación y el procesamiento por parte de los algoritmos de aprendizaje automático sin alterar las relaciones proporcionales inherentes en los datos originales.

Capítulo 6

Modelos

6.1. ¿Qué es un modelo neuronal?

En esta sección, se explica qué es un modelo neuronal, sus características y cómo funciona en el contexto de la inteligencia artificial.

Los modelos neuronales son estructuras computacionales inspiradas en el cerebro humano, diseñadas para procesar información mediante una red de unidades interconectadas que imitan, de manera simplificada, la forma en que las neuronas biológicas se comunican entre sí citegoodfellow2016deep.

Los descurbrimientos del Premio Nobel de biología español, Santiago Ramón y Cajal sobre la estructura y funcionamiento de las neuronas, proporcionaron las bases para la comprensión del sistema nervioso en el que se basan los modelos de inteligencia artificial. A finales del siglo XIX, Cajal formuló la teoría de que las neuronas son células individuales conectadas por sinapsis, una idea que revolucionó la neurociencia y sirvió de inspiración para los modelos computacionales de redes neuronales. Su trabajo permitió comprender cómo las señales eléctricas viajan entre las neuronas y cómo se pueden formar conexiones adaptativas, conceptos que más tarde serían adoptados en el diseño de redes neuronales artificiales.

Un modelo neuronal es una arquitectura matemática que busca resolver problemas complejos de aprendizaje mediante el procesamiento de datos. Está compuesto por una serie de unidades de procesamiento, conocidas como neuronas artificiales, organizadas en capas. Cada capa recibe la salida de la capa anterior, aplica una función matemática sobre los datos y transmite el resultado a la siguiente capa. Este proceso se repite de manera secuencial hasta que se alcanza la capa de salida, que proporciona el resultado final del modelo [15].

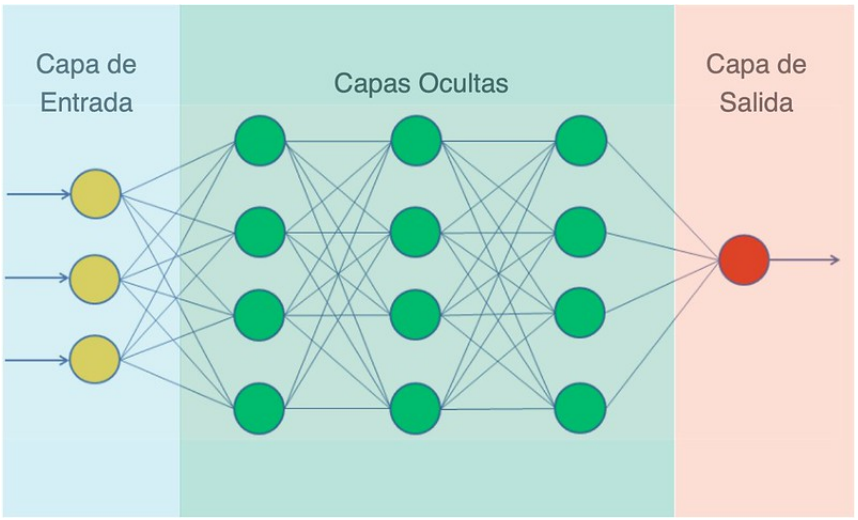

Figura 6.1: Esquema de redes neuronales. [5]

Las neuronas dentro de un modelo neuronal no operan de forma aislada, sino que están conectadas entre sí a través de enlaces denominados "pesos". Estos pesos determinan la importancia de la señal que se transmite de una neurona a otra. Durante el proceso de entrenamiento, los pesos se ajustan con el objetivo de minimizar el error en la salida del modelo, lo que permite que el modelo .ªprenda"de los datos y mejore su capacidad para predecir o clasificar nueva información [16].

El proceso de entrenamiento de un modelo neuronal implica la retroalimentación o back-propagation, donde el error de la predicción se calcula y se distribuye hacia atrás a través de la red para ajustar los pesos de manera que el modelo se optimice progresivamente. En este sentido, el modelo neuronal tiene la capacidad de adaptarse a distintos tipos de datos y mejorar su rendimiento con el tiempo [17].

A grandes rasgos, un modelo neuronal es una estructura computacional que imita el funcionamiento del cerebro humano para procesar y aprender de datos, y se utiliza en tareas como clasificación, predicción y reconocimiento de patrones.

6.1.1. ¿Qué tipos de modelos neuronales existen?

En esta sección se explican cuales son tipos de modelos neuronales existen y sus características principales.

Los modelos neuronales pueden clasificarse según su estructura, la forma en que procesan la información y la aplicación específica a la que están destinados. Estos modelos neuronales son fundamentales para resolver una variedad de problemas en áreas relacionadas con la inteligencia artificial como la visión por computadora, el procesamiento del lenguaje natural y el reconocimiento de patrones.

Uno de los tipos más comunes de modelos neuronales es el perceptrón multicapa (MLP), que está formado por varias capas de neuronas organizadas en una estructura jerárquica. Cada capa en un MLP recibe la salida de la capa anterior y la procesa mediante una función de activación antes de pasar el resultado a la siguiente capa. Este tipo de red se utiliza principalmente para tareas de clasificación y regresión, y su entrenamiento se realiza utilizando algoritmos como el de retropropagación. Es el tipo de modelo que se utiliza en este proyecto para obtener un modelo capaz de detectar instrusiones en una red informática. [18]

Otro tipo importante de modelo neuronal es la red neuronal convolucional (CNN), que está diseñada específicamente para procesar datos con una estructura de cuadricula, como las imágenes. En una CNN, las neuronas están organizadas en capas convolucionales que aplican filtros a los datos de entrada para extraer características relevantes, como bordes, texturas o formas. Esta estructura permite que las CNN sean altamente eficaces para tareas como el reconocimiento de imágenes y la visión por computadora [18].

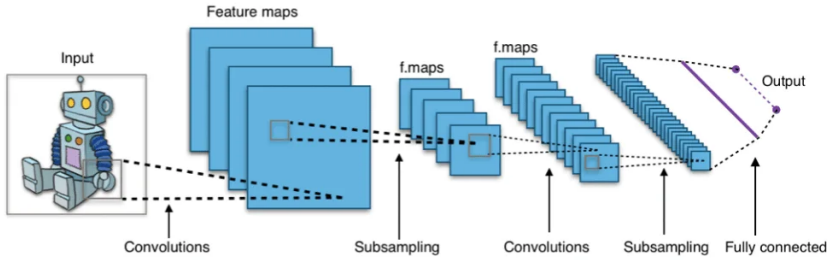

Figura 6.2: Esquema de redes neuronal convolucional. [6]

Las redes neuronales recurrentes (RNN), por otro lado, son adecuadas para procesar secuencias de datos, como el texto o el audio. Las RNN son únicas porque sus neuronas tienen conexiones que permiten que la información fluya hacia atrás a lo largo del tiempo, lo que las hace útiles para modelar dependencias temporales en los datos. Este tipo de red es comúnmente utilizado en tareas de procesamiento de lenguaje natural y en modelos de predicción de series temporales [16].

En un enfoque más avanzado, existen las redes generativas antagónicas (GAN), que se componen de dos redes neuronales: un generador y un discriminador. El generador crea datos falsos a partir de ruido, mientras que el discriminador intenta diferenciar entre los datos reales y los generados. A través de un proceso de entrenamiento competitivo, ambas redes mejoran en sus respectivas tareas. Las GANs se utilizan principalmente en la generación de imágenes, música y otros tipos de contenido artístico [17].

6.1.2. Función de perdida

En esta sección se explica que es una función de pérdida y su importancia en el proceso de entrenamiento de un modelo neuronal. También se justifica el uso de las funciones de pérdida que se emplean para el entrenamiento de los modelos de este trabajo.

Una función de pérdida es un componente fundamental en el entrenamiento de modelos de aprendizaje automático, cuya finalidad consiste en cuantificar la discrepancia entre las predicciones generadas por el modelo y los valores reales esperados. Esta medida permite guiar el proceso de optimización, ya que el objetivo durante el entrenamiento es minimizar dicha pérdida para mejorar la precisión del modelo.[19]

Las funciones de pérdida desempeñan un papel central en la formulación matemática del aprendizaje supervisado, al establecer una métrica que penaliza el error cometido por el modelo. Esta penalización permite que los algoritmos de optimización, como el descenso por gradiente, ajusten iterativamente los parámetros del modelo en la dirección que reduce la pérdida total. Para que esta estrategia sea efectiva, la función de pérdida debe poseer ciertas propiedades fundamentales:

- **Diferenciabilidad**: Permite el cálculo de gradientes necesarios para optimizar los parámetros mediante técnicas basadas en derivadas, como el descenso por gradiente.
- **Convexidad (deseable)**: Favorece la convergencia hacia un mínimo global, aunque en la práctica, muchos modelos no lineales presentan funciones de pérdida no convexas.
- **Estabilidad numérica**: Previene errores computacionales al manejar valores extremos o transformaciones exponenciales, manteniendo precisión durante el entrenamiento.
- **Sensibilidad al error**: Garantiza que los errores mayores se penalicen de forma más significativa, orientando el modelo hacia mejores predicciones.
- **Escalabilidad**: Permite su aplicación eficiente en contextos con grandes volúmenes de datos y arquitecturas de red complejas.

Estas propiedades aseguran que la función de pérdida cumpla su objetivo central, actuar como un mecanismo confiable y eficiente para guiar la actualización de los parámetros del modelo. [20]

En el caso específico del modelo de clasificación binaria, se opta por la función BCEWithLogitsLoss (Binary Cross Entropy with Logits Loss), que está especialmente recomendada para utilizarse en entornos de aprendizaje profundo como PyTorch, entorno utilizado en el desarrollo práctico de este proyecto. Esta función combina en una sola operación dos pasos fundamentales: la aplicación de la función sigmoide y el cálculo de la entropía cruzada binaria. Al integrar ambos procedimientos en una única función, se obtienen varias ventajas prácticas.[21]

En primer lugar, se mejora la estabilidad numérica, ya que evita operaciones redundantes que podrían dar lugar a pérdidas de precisión, especialmente al manejar las salidas no normalizadas del modelo (logits) con valores extremos.

En segundo lugar, se optimiza el rendimiento computacional, al reducir el número de transformaciones necesarias antes del cálculo de la pérdida.

Finalmente, permite trabajar directamente con logits, lo cual simplifica la implementación y reduce errores potenciales derivados de transformaciones incorrectas. [22]

6.1.3. Algoritmo de optimización

6.2. Modelo neuronal de clasificación binaria

6.3. Modelo neuronal de clasificación multiclase

6.4. Métricas

6.4.1. Matriz de confusión

En esta sección se explica en que consiste una matriz de confusión. Una matriz de confusión es una herramienta fundamental utilizada en el campo del aprendizaje automático y la clasificación, especialmente cuando se evalúan modelos de clasificación como el que se propone en este proyecto.

La matriz de confusión es una representación tabular que permite evaluar el rendimiento de un modelo de clasificación. Esta matriz compara las predicciones del modelo con los valores reales (verdaderos) y proporciona una visión detallada sobre los errores cometidos. Esta matriz permite calcular diversas métricas de evaluación del modelo, que son esenciales para entender la efectividad del modelo en tareas de clasificación.

En el caso de los modelos neuronales de clasificación binaria, la matriz tiene una estructura 2x2, donde cada celda en la matriz representa la cantidad de veces que una combinación específica de clase real y clase predicha ocurrió. Los valores posibles para la clasificación binaria son:

- Verdaderos positivos (VP): son las instancias que pertenecen a la clase positiva y que el modelo ha clasificado correctamente como positivas.
- Falsos positivos (FP): corresponden a las instancias que no pertenecen a la clase positiva, pero que el modelo ha etiquetado incorrectamente como positivas.
- Falsos negativos (FN): se refieren a las instancias que deberían ser clasificadas como positivas, pero que el modelo ha predicho como negativas.
- Verdaderos negativos (VN): son las instancias que pertenecen a la clase negativa y que el modelo ha clasificado correctamente como negativas.

	Predicción Positiva	**Predicción Negativa**
Real Positivo	Verdaderos Positivos (VP)	Falsos Negativos (FN)
Real Negativo	Falsos Positivos (FP)	Verdaderos Negativos (VN)

Tabla 6.1: Matriz de confusión para clasificación binaria.

Para un modelo con múltiples clases, como el caso de un modelo neuronal con 9 salidas, la matriz tiene las siguientes interpretaciones:

- Diagonal principal: Cada celda de la diagonal principal de la matriz representa cuántas veces la clase real fue correctamente predicha como clase. Este es el caso de las instancias que fueron correctamente clasificadas, y es lo más cercano a un "verdadero positivo"para esa clase específica. Sin embargo, en clasificación multiclase, se suele hablar de .ªciertos.º ïnstancias correctamente clasificadas"para cada clase.
- Fuera de la diagonal: Las celdas fuera de la diagonal representan falsas clasificaciones. Es decir, representan cuántas veces una instancia de la clase real fue predicha incorrectamente como otra clase.

	Predicción Clase 1	**Predicción Clase 2**	**Predicción Clase 3**	**Predicción Clase 4**	**Predicción Clase 5**	**Predicción Clase 6**	**Predicción Clase 7**	**Predicción Clase 8**	**Predicción Clase 9**
Real Clase 1	VP_1	FP_{12}	FP_{13}	FP_{14}	FP_{15}	FP_{16}	FP_{17}	FP_{18}	FP_{19}
Real Clase 2	FP_{21}	VP_2	FP_{23}	FP_{24}	FP_{25}	FP_{26}	FP_{27}	FP_{28}	FP_{29}
Real Clase 3	FP_{31}	FP_{32}	VP_3	FP_{34}	FP_{35}	FP_{36}	FP_{37}	FP_{38}	FP_{39}
Real Clase 4	FP_{41}	FP_{42}	FP_{43}	VP_4	FP_{45}	FP_{46}	FP_{47}	FP_{48}	FP_{49}
Real Clase 5	FP_{51}	FP_{52}	FP_{53}	FP_{54}	VP_5	FP_{56}	FP_{57}	FP_{58}	FP_{59}
Real Clase 6	FP_{61}	FP_{62}	FP_{63}	FP_{64}	FP_{65}	VP_6	FP_{67}	FP_{68}	FP_{69}
Real Clase 7	FP_{71}	FP_{72}	FP_{73}	FP_{74}	FP_{75}	FP_{76}	VP_7	FP_{78}	FP_{79}
Real Clase 8	FP_{81}	FP_{82}	FP_{83}	FP_{84}	FP_{85}	FP_{86}	FP_{87}	VP_8	FP_{89}
Real Clase 9	FP_{91}	FP_{92}	FP_{93}	FP_{94}	FP_{95}	FP_{96}	FP_{97}	FP_{98}	VP_9

Tabla 6.2: Matriz de confusión para clasificación con 9 clases.

6.4.2. Fórmulas e Interpretación

Para evaluar el desempeño del modelo de detección y clasificación de ataques, se utilizan las siguientes métricas derivadas de la matriz de confusión.

- **Exactitud (*Accuracy*)**:

$$\text{Accuracy} = \frac{VP + VN}{VP + FP + VN + FN} \tag{6.1}$$

  En el entrenamiento de modelos neuronales para la detección de intrusiones, esta métrica representa la proporción del total de las clasificaciones realizadas correctamente. Indica la capacidad general del modelo para distinguir entre tráfico normal e intrusivo.

Si bien ofrece una visión global del rendimiento, su valor disminuye en escenarios donde la cantidad de tráfico normal supera significativamente al tráfico malicioso, ya que el modelo puede obtener una alta exactitud simplemente clasificando la mayoría de las instancias como normales.

- **Precisión (*Precision*)**:

$$\text{Precision} = \frac{VP}{VP + FP} \tag{6.2}$$

Esta métrica evalúa la capacidad del modelo neuronal para evitar la identificación incorrecta de tráfico normal como intrusivo. En la detección de intrusiones, una alta precisión es crucial para minimizar las falsas alarmas, las cuales pueden generar una sobrecarga operativa en los equipos de seguridad, requiriendo la revisión de eventos benignos y distrayendo la atención de amenazas reales. Un modelo preciso reduce la fatiga de alertas y permite una respuesta más eficiente a incidentes genuinos.

- **Sensibilidad (*Recall*)**:

$$\text{Recall} = \frac{VP}{VP + FN} \tag{6.3}$$

La sensibilidad mide la habilidad del modelo neuronal para detectar todas las instancias de intrusión presentes en el tráfico de red. En el contexto de la seguridad, un alto recall es de suma importancia, ya que implica una menor probabilidad de que ataques reales pasen desapercibidos. Un modelo con baja sensibilidad puede tener consecuencias graves, permitiendo que actividades maliciosas se infiltren y comprometan la integridad y la confidencialidad de los sistemas.

- **Puntuación F1 (*F1-Score*)**:

$$F1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{6.4}$$

Esta métrica proporciona una evaluación equilibrada del rendimiento del modelo neuronal al calcular la media armónica entre la precisión y el recall. En la detección de intrusiones, donde a menudo existe un desequilibrio entre el tráfico normal y el malicioso, el F1-score ofrece una métrica más robusta que la exactitud, ya que considera tanto la capacidad de evitar falsas alarmas como la de detectar todas las intrusiones. Un valor alto de F1-score indica un buen compromiso entre ambas capacidades.

- **Puntuación F2 (*F2-Score*)**:

$$F2 = 5 \times \frac{\text{Precision} \times \text{Recall}}{4 \times \text{Precision} + \text{Recall}} \tag{6.5}$$

Esta variante de la puntuación F pondera la sensibilidad más que la precisión. En el ámbito de la detección de intrusiones, el F2-score resulta útil cuando las consecuencias de no detectar un ataque (falso negativo) se consideran significativamente más perjudiciales que generar una falsa alarma (falso positivo). Al asignar un mayor peso al recall, se prioriza la identificación de la mayor cantidad posible de actividades maliciosas, incluso a expensas de un posible aumento en las falsas alertas.

6.4.3. Aplicación en Seguridad

En el contexto de detección de intrusiones:

- Un recall alto (¿95 %) asegura que pocos ataques pasan desapercibidos.
- La precisión debe optimizarse para reducir la carga operativa de analistas (falsos positivos ¡10 %).
- El F2-Score es preferible al F1 cuando la prioridad es minimizar riesgos de ataques no detectados.

Capítulo 7

Test

Capítulo 8

Despliegue

Capítulo 9

Tecnologías usadas

Capítulo 10

Seguimiento del proyecto

Capítulo 11

Conclusiones

Apéndice A

Manuales

A.1. Manual de despliegue e instalación

A.2. Manual de mantenimiento

A.3. Manual de usuario

Apéndice B

Resumen de enlaces adicionales

Los enlaces útiles de interés en este Trabajo Fin de Grado son:

- Repositorio del código: `https://gitlab.inf.uva.es/`.

Bibliografía

[1] Pablo Haya. La metodología crisp-dm en ciencia de datos, noviembre 2021. Consultado el 18 de mayo de 2025.

[2] IONOS España. Tcp protocol: así funciona el protocolo de transmisión, 2020.

[3] Vicente González Ruiz. Tcp (transmission control protocol), December 2014.

[4] Vicente González Ruiz. El ip (internet protocol), December 2014.

[5] AprendeIA. ¿qué es deep learning?, 2021.

[6] Daniel Nielson Unite.AI. ¿qué son las redes neuronales convolucionales (cnn)?, 2020.

[7] Pete Chapman, Julian Clinton, Randy Kerber, Thomas Khabaza, Thomas Reinartz, Colin Shearer, and Rüdiger Wirth. *CRISP-DM 1.0: Step-by-step data mining guide*. SPSS Inc., 2000.

[8] Project Management Institute. *A Guide to the Project Management Body of Knowledge (PMBOK Guide)*. Project Management Institute, 2021.

[9] Cosmikal. ¿qué es y cómo funciona un firewall? guía básica 2025, december 2024.

[10] Geekflare. Ids vs ips: A comprehensive guide to network security solutions, december 2024.

[11] Palo Alto Networks. ¿qué es la microsegmentación?

[12] National Institute of Standards and Technology (NIST). Adversarial machine learning: A taxonomy and terminology of attacks and mitigations, 2021.

[13] John Brooke. SUS-A quick and dirty usability scale. *Usability evaluation in industry*, 189(194), 1996.

[14] Majed Luay, Siamak Layeghy, Seyedehfaezeh Hosseininoorbin, Mohanad Sarhan, Nour Moustafa, and Marius Portmann. Temporal analysis of netflow datasets for network intrusion detection systems, 2025.

[15] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

[16] Simon Haykin. *Neural Networks and Learning Machines*. Pearson, 2009.

[17] Michael Nielsen. *Neural Networks and Deep Learning*. Determination Press, 2015.

[18] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016.

[19] EITCA Academy. ¿cuál es el papel de la función de pérdida en el aprendizaje automático? Accedido: 2025-05-22.

[20] Ultralytics. Función de pérdida. Accedido: 2025-05-22.

[21] DataCamp. Explicación de las funciones de pérdida en el machine learning. Accedido: 2025-05-22.

[22] BigDataFran. 7. introducción problemas de clasificación — trabajando con pytorch. Accedido: 2025-05-22.