**Aplicación de Modelos de Machine Learning a Búsquedas de Materia Oscura con Datos del Satélite Fermi-LAT de la NASA**

*Machine Learning Applications to Dark Matter Searches with Data of the NASA Satellite Fermi-LAT*

Autor: Marta Canino Romero

Tutor: Viviana Gammaldi

[1. Introducción 2](#_Toc1876097651)

[1.2 Objetivos del proyecto 3](#_Toc1347012744)

[1.2 Estructura de la memoria 4](#_Toc1145314208)

[2. Marco teórico 5](#_Toc210509842)

[2.1. El problema 6](#_Toc491302612)

[2.2. Fuente de datos 9](#_Toc487643711)

[2.3. Introducción al aprendizaje automático 11](#_Toc511976204)

[2.4. Modelos de detección de anomalías 13](#_Toc1846417048)

[2.5. Estado del arte de métodos aplicados en el campo 16](#_Toc1843824899)

[3. Gestión del Proyecto 17](#_Toc333133745)

[3.1. Metodología y enfoque de trabajo (Agile, Scrum, XP) 18](#_Toc995964746)

[3.2. Modelo de Ciclo de Vida 18](#_Toc193480296)

[3.3. Papeles desempeñados en el proyecto 20](#_Toc1563793694)

[3.4. Planificación y Organización (épicas, sprints, entregables) 21](#_Toc315003135)

[3.5. Ejecución práctica 26](#_Toc1880219853)

[2.5 Ejecución 28](#_Toc227153367)

[4. Desarrollo experimental 29](#_Toc576102189)

[4.1. Datos utilizados y preprocesamiento 29](#_Toc785360008)

[4.2. Desarrollo y análisis del modelo base (UNN) 29](#_Toc932963393)

[4.3. Desarrollo de los modelos OCSVM (2F, 3F, 4F) 29](#_Toc1897069291)

[4.4. Validación y evaluación de resultados 29](#_Toc1920435949)

[4.4. Comparación entre modelos 29](#_Toc379197857)

[4.4. Discusión de resultados 29](#_Toc139524356)

[5. Conclusiones y líneas futuras 29](#_Toc2147405212)

[6. Entregables 30](#_Toc648471334)

[6.1. Repositorio del proyecto 31](#_Toc827640537)

[6.2. Manual de uso 31](#_Toc360601314)

[7. Bibliografía 31](#_Toc641373793)

[8. Anexos 34](#_Toc948946302)

[8.2. Planificación detallada de tareas 35](#_Toc1298672016)

[8.3. Criterios de aceptación 38](#_Toc268202748)

[8.4. Código o fragmentos relevantes 39](#_Toc69073229)

[8.5. Otros materiales de apoyo 40](#_Toc748176842)

# **1. Introducción**

El presente Trabajo de Fin de Grado (TFG) aborda el estudio de la **materia oscura** (Dark Matter, DM), uno de los mayores enigmas de la física moderna. Aunque su existencia no ha sido confirmada mediante detección directa en experimentos de laboratorio, su influencia gravitacional es evidente a escalas astrofísicas, afectando la dinámica de galaxias, cúmulos galácticos y la evolución del universo (Bertone & Hooper, 2018).

El **catálogo 4FGL** del telescopio espacial de rayos gamma **Fermi-LAT r**ecopila información sobre miles de **fuentes puntuales de rayos gamma** detectadas en el cielo. Sin embargo, aproximadamente un tercio de estas fuentes no presentan una **asociación clara con fenómenos astrofísicos conocidos** y permanecen **sin identificar**. Estas fuentes no identificadas (unassociated gamma-ray sources, unIDs) representan una oportunidad para explorar posibles señales indirectas de materia oscura, que podrían manifestarse como **emisión gamma producto de la aniquilación o decaimiento de partículas masivas débilmente interactuantes (WIMPs)** (Cirelli et al., 2011).

Estudios previos han abordado este problema mediante técnicas de **aprendizaje automático supervisado**, utilizando **redes neuronales artificiales (ANN)** para intentar distinguir entre fuentes astrofísicas conocidas y posibles candidatos a materia oscura. Este trabajo parte de dichos estudios, con el objetivo de replicar y evaluar el modelo base, y de explorar alternativas mediante técnicas no supervisadas, específicamente **modelos de detección de anomalías**, con el fin de superar limitaciones identificadas y mejorar la clasificación de las fuentes no identificadas.

El desarrollo del proyecto incluye la **comprensión y adaptación del código base existente**, la **replicación de resultados previos** y la **implementación y evaluación de nuevos modelos de aprendizaje automático**. Los experimentos se llevaron a cabo utilizando **Python**, principalmente en **Jupyter Notebooks**, con el apoyo de librerías como **scikit-learn**, entre otras herramientas.

## 1.2 Objetivos del proyecto

El **objetivo general** de este Trabajo de Fin de Grado es aplicar y evaluar técnicas de **aprendizaje automático** sobre los datos del catálogo 4FGL del telescopio Fermi-LAT, con el fin de identificar posibles candidatos a materia oscura entre las fuentes de rayos gamma no identificadas.

Dentro de este marco general, se plantearon los siguientes **objetivos específicos**:

* Analizar y comprender el modelo base de red neuronal desarrollado en estudios previos, replicando sus resultados sobre el catálogo 4FGL.
* Realizar un análisis exploratorio de los datos, identificando las variables más relevantes para la tarea de clasificación.
* Investigar y comparar distintas metodologías de aprendizaje automático, evaluando su aplicabilidad al problema de clasificación de fuentes no identificadas.
* Desarrollar un modelo de aprendizaje automático alternativo, basado en técnicas de **detección de anomalías**, y evaluar su desempeño.
* Comparar los resultados obtenidos por el nuevo modelo con los resultados del modelo base supervisado, identificando coincidencias y discrepancias en la selección de posibles candidatos a materia oscura.
* Documentar de manera estructurada y reproducible el proceso de desarrollo, análisis de resultados y conclusiones.

Cabe destacar que la selección final del modelo **One-Class Support Vector Machine (OCSVM)** fue resultado de un proceso iterativo de investigación y experimentación, tras evaluar distintas aproximaciones al problema.

## 1.2 Estructura de la memoria

La memoria del presente Trabajo de Fin de Grado se organiza en siete capítulos principales. En el **Capítulo 1** se presenta la introducción, que incluye el contexto del problema, los objetivos del proyecto y la estructura del documento. El **Capítulo 2** expone el marco teórico, abordando los fundamentos del aprendizaje automático, los modelos de detección de anomalías y el estado del arte de la aplicación de estas técnicas al análisis de datos del Fermi-LAT.

En el **Capítulo 3** se describe la gestión del proyecto, detallando la metodología adoptada, el modelo de ciclo de vida, los roles desempeñados, la planificación y la ejecución. El **Capítulo 4** recoge el desarrollo experimental, incluyendo el análisis exploratorio de datos, la implementación de los modelos, la validación y la comparación de resultados.

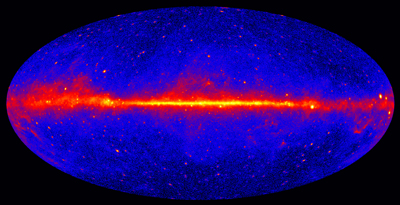
El **Capítulo 5** presenta las conclusiones obtenidas y las posibles líneas de trabajo futuro. Finalmente, se incluyen los apartados de bibliografía y anexos, que recopilan materiales complementarios como la planificación detallada, criterios de aceptación y fragmentos relevantes de código.

# **2. Marco teórico**

## 2.1. El problema

El **telescopio espacial Fermi-LAT** (Large Area Telescope), en operación desde 2008, ha realizado un seguimiento continuo del cielo en la banda de los **rayos gamma de alta energía**. Gracias a sus observaciones, se han catalogado **miles de fuentes gamma** en diversos catálogos públicos, siendo el más reciente el 4FGL (Abdollahi et al., 2020). La mayoría de estas fuentes se han asociado con **clases astrofísicas conocidas**, como **púlsares** o **núcleos galácticos activos** (Active Galactic Nuclei, AGN), mediante su correlación con observaciones en otras longitudes de onda.

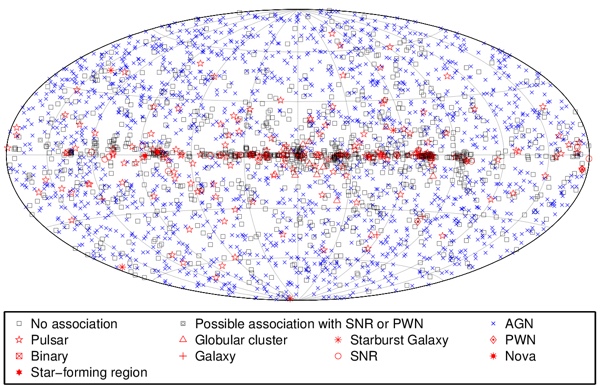
La **Figura 1.2** muestra la distribución de las fuentes de rayos gamma detectadas por el telescopio Fermi-LAT en coordenadas galácticas, donde se observa una mayor concentración en el plano de la Vía Láctea.



***Figura 2.1.*** *Mapa del cielo en rayos gamma obtenido por el satélite Fermi-LAT. Fuente: NASA/DOE/Fermi LAT Collaboration (*[*https://fermi.gsfc.nasa.gov/ssc/*](https://fermi.gsfc.nasa.gov/ssc/)*)*

Sin embargo, un porcentaje significativo de las fuentes detectadas por Fermi-LAT permanece **sin identificación**: no presentan **contrapartes claras** en otras bandas del espectro electromagnético y no encajan fácilmente en las categorías conocidas. Estas **fuentes no identificadas** (unassociated gamma-ray sources, unIDs) representan tanto un desafío como una oportunidad para la astrofísica, ya que podrían corresponder a fenómenos aún no comprendidos o incluso a manifestaciones indirectas de **materia oscura** (Saz Parkinson & Farrar, 2017).

La **Figura 2.1** ilustra la distribución de las fuentes del catálogo 3FGL de Fermi-LAT en coordenadas galácticas, clasificadas según su posible asociación con distintos tipos de objetos astronómicos. Se observa una concentración de fuentes en el plano galáctico y una notable proporción de fuentes sin identificar, representadas en gris, que constituyen el foco de estudio de este proyecto.

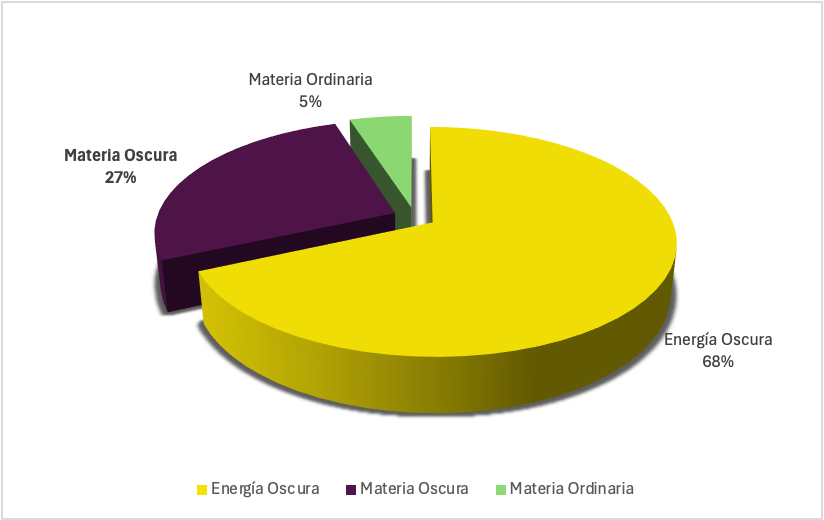


***Figura 2.2.*** *Distribución de fuentes de rayos gamma del catálogo Fermi-LAT 3FGL en coordenadas galácticas.*

*Fuente: NASA/Fermi Science Support Center (*[*https://fermi.gsfc.nasa.gov/science/eteu/catalogs/*](https://fermi.gsfc.nasa.gov/science/eteu/catalogs/)*)*

La **materia oscura** es uno de los mayores enigmas de la física contemporánea. Diversas observaciones astronómicas, como las curvas de rotación de galaxias y la dinámica de cúmulos galácticos, solo pueden explicarse si existe una gran cantidad de **masa invisible** que no interactúa con la luz pero ejerce **gravitación** (Bertone & Hooper, 2018). Una de las estrategias para su detección indirecta es la búsqueda de **emisión de rayos gamma** producto de la **aniquilación o decaimiento de partículas de materia oscura** en regiones de alta densidad, como el centro galáctico o las galaxias enanas satélites (Cirelli et al., 2011).

Como se muestra en la **Figura 1.3**, los resultados de la misión Planck indican que aproximadamente e**l 26.8% del universo** está compuesto por **materia oscura**, mientras que la energía oscura constituye la mayor fracción con un 68.3%.”



***Figura 1.3.*** *Composición estimada del universo según los resultados de la misión Planck: 68.3% energía oscura, 26.8% materia oscura y 4.9% materia ordinaria.*

*Fuente de datos: Planck Collaboration (2013); elaboración propia.*

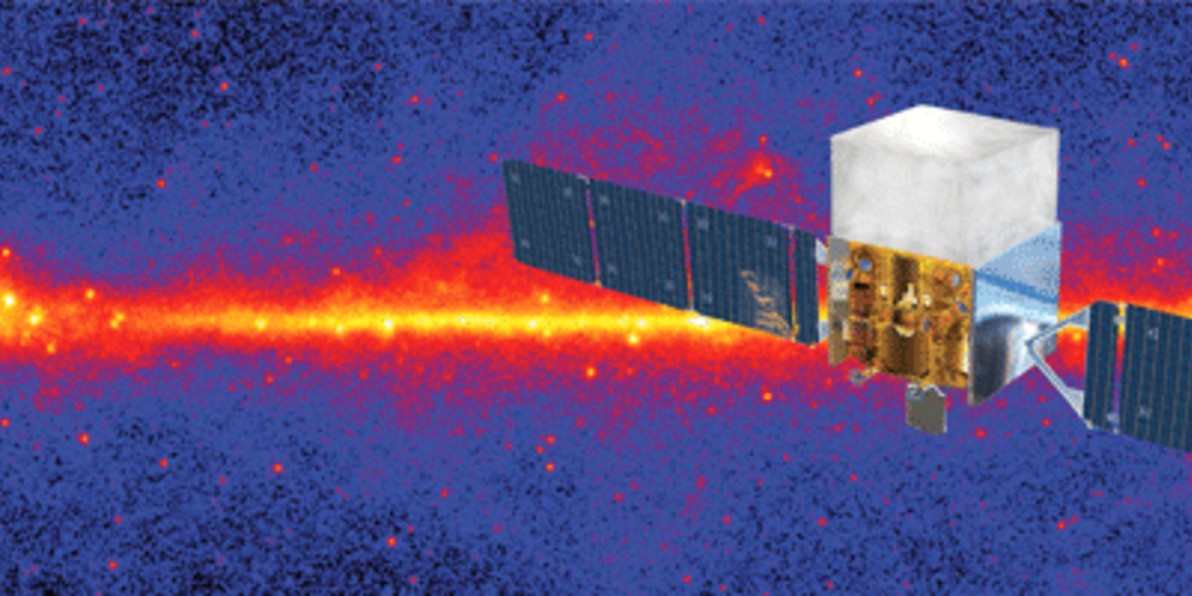
En este contexto, las fuentes no identificadas del catálogo Fermi-LAT podrían contener **candidatos a señales de materia oscura**, aunque también podrían corresponder a poblaciones astrofísicas aún no clasificadas. La gran cantidad de datos, la complejidad de los parámetros espectrales y la ausencia de etiquetas confiables para las fuentes no identificadas dificultan su análisis mediante métodos tradicionales.

En los últimos años, los métodos de **aprendizaje automático** (Machine Learning, ML) han demostrado ser herramientas prometedoras para abordar este problema. Estos métodos permiten construir **modelos predictivos capaces de clasificar automáticamente las fuentes gamma** en categorías conocidas o de **detectar anomalías o patrones atípicos** que podrían señalar fenómenos no explicados. Aplicar técnicas de ML al catálogo de Fermi-LAT abre la posibilidad de **priorizar fuentes no identificadas para estudios de seguimiento** y de aportar una herramienta de ayuda para la búsqueda indirecta de materia oscura desde un enfoque computacional (Mirabal et al., 2012).

Desde la perspectiva de la **ingeniería en sistemas de información**, este proyecto se enmarca en los desafíos de la **gestión**, **procesamiento** y **análisis de grandes volúmenes de datos** **científicos**, integrando técnicas avanzadas de minería de datos y aprendizaje automático para la extracción de conocimiento en un dominio altamente interdisciplinar.

## 2.2. Fuente de datos

La fuente principal de datos utilizada en este proyecto es e**l catálogo público 4FGL** (Fourth Fermi Large Area Telescope Source Catalog), publicado por la colaboración Fermi-LAT (Abdollahi et al., 2020). Este catálogo compila información sobre más de 5.000 **fuentes de radiación gamma detectadas en el espacio** por el telescopio Fermi-LAT durante sus primeros ocho años de operación.



***Figura 2.3.*** *Esquema ilustrativo del telescopio espacial Fermi-LAT.*

*Fuente: Stanford University / GLAST Collaboration (*[*https://glast.sites.stanford.edu/*](https://glast.sites.stanford.edu/)*)*

Cada registro del catálogo representa una fuente detectada y contiene múltiples atributos o variables relevantes, entre ellos:

* **Posición celeste** (coordenadas en el cielo)
* **Significancia de detección** (Test Statistic, TS)
* **Flujo observado** en distintas bandas de energía
* **Parámetros espectrales** que describen la forma de su emisión
* I**ndicadores de variabilidad temporal** y, en su caso, **curvas de luz** para fuentes variables

Los datos están disponibles de forma abierta a través del **Fermi Science Support Center (FSSC)** de la NASA, lo cual facilita su descarga y análisis mediante herramientas estándar. En este proyecto, se ha construido un **conjunto de datos específico** combinando los datos reales del catálogo 4FGL con un **conjunto simulado de fuentes hipotéticas de materia oscura**. Este conjunto simulado se generó a partir de modelos teóricos que predicen cómo se manifestaría la materia oscura en las observaciones de rayos gamma (Cirelli et al., 2011). El objetivo es disponer de un dataset

Las **variables seleccionadas para el análisis** incluyen:

* **Eₚₑₐₖ (energía pico)**: energía donde la emisión es máxima
* **β (curvatura espectral)**: indica cómo varía la intensidad con la energía
* **σ\_d (significancia de detección)**: nivel de confianza estadística en la detección de la fuente
* **Incertidumbre en β**: margen de error asociado al parámetro de curvatura

La elección de estas variables responde a su **potencial discriminativo**: estudios previos han mostrado que, al incluir variables relacionadas con la fiabilidad y precisión de la medición, se mejora el rendimiento de los modelos de clasificación automática, especialmente para **identificar posibles fuentes no catalogadas previamente** (Saz Parkinson & Farrar, 2017).

Desde el punto de vista técnico, el dataset final representa una **tabla de datos estructurados**, con cada fila correspondiente a una fuente y cada columna a un atributo. Este tipo de datos es idóneo para ser procesado mediante herramientas de **machine learning supervisado y no supervisado**, permitiendo entrenar modelos capaces de clasificar o identificar patrones en las fuentes no identificadas.

## 2.3. Introducción al aprendizaje automático

El **aprendizaje automático** (Machine Learning, ML) es una disciplina surgida como subcampo de la **inteligencia artificial** (IA) en la década de 1950. Su objetivo principal es desarrollar algoritmos y modelos que permitan a las computadoras **aprender automáticamente a partir de los datos**, sin estar explícitamente programadas para realizar una tarea específica (Samuel, 1959).

Según Çelik y Altunaydin (2018), el aprendizaje puede definirse como “el proceso de cambio y mejora del comportamiento mediante la exploración de nueva información a lo largo del tiempo”. Cuando este proceso de aprendizaje es llevado a cabo por máquinas, hablamos de **aprendizaje automático**.

En la última década, las técnicas basadas en aprendizaje automático han experimentado un **crecimiento exponencial,** impulsando avances significativos en diversas áreas, como la **conducción autónoma**, la **medicina personalizada**, las **finanzas**, la **industria manufacturera** y la **energía renovable** (Carleo et al., 2019). El aprendizaje automático se considera una de las tecnologías más disruptivas de la actualidad, con un impacto comparable al de la introducción de los ordenadores personales en los años ochenta y noventa.

El objetivo esencial del aprendizaje automático es **identificar patrones en los datos** para resolver problemas complejos o realizar predicciones. Por ejemplo, en el contexto de un **vehículo autónomo**, grandes cantidades de datos de sensores deben procesarse y traducirse en decisiones, como frenar o girar, mediante un sistema que ha aprendido a identificar patrones asociados a situaciones de “peligro” (Carleo et al., 2019).

**Tipos de aprendizaje automático**

El aprendizaje automático se puede clasificar de diversas maneras, siendo una de las más comunes aquella basada en la **disponibilidad de datos etiquetados**. Según Bobadilla (2019), los principales tipos de aprendizaje automático son los siguientes:

* **Aprendizaje supervisado**: los algoritmos se entrenan con datos que incluyen tanto las **entradas** como las **salidas esperadas**. Se utiliza para tareas como:
  + **Regresión**: predicción de valores continuos (por ejemplo, el precio de una casa).
  + **Clasificación**: asignación de una categoría a una entrada (por ejemplo, detectar si una imagen contiene un gato o un perro).
* **Aprendizaje no supervisado**: los datos de entrenamiento **no están etiquetados**. El modelo busca descubrir patrones ocultos o estructuras subyacentes. Entre las técnicas más comunes se encuentran:
  + **Clustering** (agrupamiento): agrupar datos similares en conjuntos.
  + **Reducción de dimensionalidad**: simplificar los datos manteniendo la mayor cantidad de información relevante posible.
* **Aprendizaje semi-supervisado**: combinación de un pequeño conjunto de datos etiquetados con una gran cantidad de datos no etiquetados, para mejorar el rendimiento del modelo.
* **Aprendizaje por refuerzo**: los algoritmos aprenden mediante **interacción con un entorno**, recibiendo **recompensas** o **penalizaciones** en función de las acciones realizadas, con el objetivo de maximizar una señal de recompensa acumulada.

Estos enfoques constituyen la base del aprendizaje automático y han sido aplicados con éxito en campos tan diversos como la **física de partículas**, la **astronomía** y la **búsqueda de materia oscura**, donde los datos son abundantes, complejos y de difícil interpretación mediante métodos tradicionales.

## 2.4. Modelos de detección de anomalías

La **detección de anomalías** es un enfoque dentro del aprendizaje automático que busca **identificar patrones o comportamientos inusuales** que se desvían significativamente del comportamiento esperado o "normal" en un conjunto de datos. Las observaciones identificadas como anómalas pueden representar **errores, fraudes, fallos técnicos o fenómenos raros**, dependiendo del contexto de aplicación (Chandola, Banerjee & Kumar, 2009).

Este tipo de técnicas se utiliza ampliamente en campos tan diversos como la **detección de fraudes financieros**, la **ciberseguridad** (por ejemplo, detección de intrusiones en redes), el **diagnóstico médico asistido** (como el monitoreo cardíaco), la **industria aeroespacial** (detección de fallos en sistemas críticos), o la **procesamiento de imágenes hiperespectrales** en aplicaciones de teledetección (Bou Nassif et al., 2022).

En términos generales, la detección de anomalías puede considerarse como una **tarea de clasificación binaria**, en la que el objetivo es distinguir entre instancias "normales" y "anómalas". Sin embargo, a diferencia de los problemas clásicos de clasificación, en muchos escenarios la clase anómala está **poco representada o directamente ausente** en los datos de entrenamiento, lo que limita la aplicabilidad de los modelos supervisados tradicionales (Zimek et al., 2012).

Dependiendo del tipo de información disponible para el entrenamiento, los enfoques de detección de anomalías pueden clasificarse en tres categorías principales (Bou Nassif et al., 2022):

* **Detección de anomalías supervisada**

Requiere un conjunto de datos etiquetado que contenga tanto instancias normales como anómalas. En este caso, se entrena un modelo predictivo convencional (por ejemplo, regresión logística o redes neuronales) para aprender a clasificar ambas clases. Sin embargo, en muchos casos reales, las **anomalías son escasas o difíciles de etiquetar**, lo que limita la eficacia de este enfoque.

* **Detección de anomalías semisupervisada**

En este enfoque, el modelo se entrena **únicamente con instancias normales**. Se asume que cualquier observación futura que difiera significativamente del patrón aprendido puede considerarse anómala. Este método es especialmente útil cuando **no se dispone de ejemplos de anomalías**, como ocurre en muchos problemas del mundo real, incluyendo la identificación de fuentes astrofísicas inusuales.

* **Detección de anomalías no supervisada**

Este enfoque **no requiere etiquetas** previas. El modelo infiere qué observaciones son "normales" y cuáles no, basándose en su frecuencia o en la densidad de los datos. Se asume que las **anomalías son estadísticamente raras**. Técnicas como clustering, k-nearest neighbors, Isolation Forest o autoencoders suelen aplicarse en esta categoría. Aunque es versátil, este enfoque puede ser propenso a falsos positivos si la distribución de los datos no cumple ciertos supuestos.

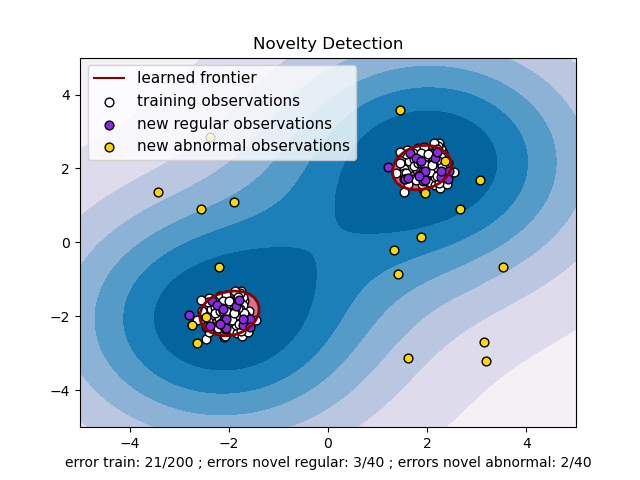
En este proyecto se ha optado por un enfoque **semisupervisado**, en el que se parte de un conjunto de fuentes gamma **ya clasificadas** como "normales" (por ejemplo, AGN o púlsares), y se entrena un modelo para detectar patrones que se desvíen significativamente de estos. En concreto, se ha utilizado el algoritmo **One-Class Support Vector Machine** (**OCSVM**), una técnica basada en aprendizaje automático que aprende los límites de una clase positiva y **detecta desviaciones respecto a esa frontera** como posibles anomalías (Schölkopf et al., 2001). Esta metodología resulta especialmente adecuada cuando se busca identificar **instancias inusuales en grandes volúmenes de datos no etiquetados,** como ocurre con las fuentes no identificadas del catálogo Fermi-LAT.

**Enfoque en One-Class SVM (OCSVM)**

El algoritmo **One-Class Support Vector Machine (OCSVM)** es una técnica de aprendizaje automático semisupervisada, utilizada para la detección de anomalías cuando solo se dispone de datos de una única clase (la clase "normal"). A diferencia de los clasificadores tradicionales, el OCSVM no busca separar dos o más clases distintas, sino **modelar el contorno de una única clase** en el espacio de características, de manera que cualquier punto que quede fuera de ese contorno sea considerado como potencialmente anómalo (Schölkopf et al., 2001).

OCSVM es una extensión del algoritmo SVM convencional. Utiliza una función **kernel** (generalmente el kernel radial o RBF) para proyectar los datos en un espacio de mayor dimensión, donde se construye una **▪ Enfoque en One-Class SVM (OCSVM)**

que rodea la mayoría de los datos de entrenamiento. Este modelo intenta **maximizar el margen respecto al origen**, lo que permite identificar observaciones fuera del contorno como outliers.



*Figura 2.5. Visualización de la detección de anomalías utilizando One-Class SVM con kernel RBF.*

*Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Duchesnay, É. (2011). Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research, 12, 2825–2830.*

Los parámetros clave de OCSVM incluyen:

* **ν (nu)**: controla la fracción de outliers permitidos en los datos de entrenamiento y el número de vectores de soporte. Un valor más alto de ν permite mayor flexibilidad, pero puede generar más falsos positivos.
* **Kernel**: la función utilizada para mapear los datos al espacio de características. El kernel RBF (función gaussiana) es el más común en detección de anomalías, ya que permite modelar fronteras no lineales.
* **γ (gamma)**: parámetro del kernel RBF que determina la influencia de cada punto individual. Valores pequeños generan fronteras suaves; valores grandes producen fronteras más ajustadas.

OCSVM es especialmente adecuado para contextos en los que **se dispone de muchos datos bien clasificados de una clase conocida**, pero **las observaciones anómalas son escasas o desconocidas**, como ocurre en este proyecto con las fuentes astrofísicas del catálogo Fermi-LAT. En este caso, el modelo se entrena con un conjunto de fuentes etiquetadas (por ejemplo, AGNs y púlsares) y se aplica posteriormente a las fuentes no identificadas, con el objetivo de detectar aquellas que se alejan significativamente de las clases conocidas y que podrían ser candidatas a fenómenos aún no clasificados, incluida la materia oscura.

En este proyecto, el modelo OCSVM se ha utilizado bajo el enfoque de **detección de novedades** (novelty detection), en el que se entrena con un conjunto de fuentes gamma conocidas y clasificadas, para luego aplicarlo a nuevas observaciones (las fuentes no identificadas) y determinar si presentan propiedades significativamente diferentes a las clases conocidas. Esto difiere del enfoque clásico de detección de anomalías (anomaly detection), que trabaja directamente sobre un conjunto mixto, en el que las anomalías están embebidas entre los datos normales. En el contexto del catálogo Fermi-LAT, donde se busca evaluar si las fuentes no clasificadas podrían representar una clase distinta (como materia oscura), el paradigma de novelty detection resulta más apropiado.

## 2.5. Estado del arte de métodos aplicados en el campo

# **3. Gestión del Proyecto**

## 3.1. Metodología y enfoque de trabajo (Agile, Scrum, XP)

Para la planificación y ejecución de este Trabajo de Fin de Grado se ha optado por adoptar un enfoque basado en metodologías ágiles, ampliamente utilizadas en proyectos de desarrollo de software y en el ámbito tecnológico. Aunque inicialmente estas metodologías podrían parecer poco adaptables a un proyecto de investigación científica con componentes de Machine Learning (ML), su naturaleza iterativa y su capacidad de adaptación ante la incertidumbre las hacen especialmente adecuadas para abordar problemas complejos y dinámicos, como los asociados al aprendizaje automático.

En el ámbito de Machine Learning, el proceso ETL (Extract, Transform, Load) estructura de manera iterativa las fases de definición del problema, extracción de datos, preparación de datos, desarrollo y evaluación de modelos, y, en su caso, despliegue. Este ciclo, al ser iterativo, guarda paralelismos conceptuales con la estructura de sprints en metodologías ágiles, lo que ha permitido extrapolarlo y adaptarlo al esquema de planificación del presente trabajo.

El enfoque ágil aporta beneficios clave en este contexto, entre ellos la posibilidad de abordar un problema inicialmente percibido como una “caja negra” —una metáfora frecuente en la descripción de los modelos de aprendizaje automático— mediante una estrategia incremental y de aprendizaje continuo. Esta metodología favorece la incorporación progresiva de conocimiento, la revisión constante de los avances y la posibilidad de ajustar objetivos y enfoques conforme se obtienen resultados parciales.

Con este enfoque metodológico, se espera optimizar el desarrollo del proyecto y alcanzar resultados significativos que contribuyan al estudio de la materia oscura.

## 3.2. Modelo de Ciclo de Vida

El ciclo de vida del proyecto describe las fases principales por las que transcurre su desarrollo, desde su concepción inicial hasta su finalización, incluyendo los hitos, entregables y mecanismos de control que lo estructuran. En el presente Trabajo de Fin de Grado, el ciclo de vida adoptado sigue un modelo iterativo e incremental, en coherencia con la metodología ágil seleccionada (Scrum y prácticas de eXtreme Programming), y adaptado a las particularidades de un proyecto de investigación científica con aplicación de técnicas de Machine Learning.

A diferencia de los modelos tradicionales en cascada, que se desarrollan de manera lineal y secuencial, el modelo de ciclo de vida adoptado para este proyecto se basa en iteraciones sucesivas (sprints) que permiten la entrega progresiva de resultados parciales, la revisión continua y la adaptación dinámica a nuevos hallazgos, dificultades o necesidades emergentes.

El ciclo de vida del proyecto se estructura en las siguientes fases principales, que se corresponden con las épicas definidas en la planificación:

1. **Definición y organización inicial (EPIC-01):**

*En esta fase se establecen los objetivos generales, el alcance del proyecto y el marco teórico inicial. Se organiza el backlog inicial y se priorizan las tareas. Constituye la base sobre la que se construirá el trabajo posterior, asegurando una comprensión clara del problema y de los recursos disponibles.*

1. **Análisis y validación del modelo base (EPIC-02):**

*Esta fase se centra en la ejecución, análisis y documentación del modelo preexistente (UNN), con el objetivo de comprender sus fundamentos, reproducir resultados y establecer una línea base sobre la que se construirá el nuevo modelo.*

1. **Desarrollo iterativo del nuevo modelo (EPIC-03):**

*Corresponde a la fase de diseño, implementación, entrenamiento y optimización del nuevo modelo de detección de anomalías. Esta etapa se aborda de forma incremental mediante varios sprints, permitiendo evaluar resultados parciales, realizar ajustes y mejorar progresivamente el rendimiento del modelo.*

1. **Integración, análisis comparativo y documentación final (EPIC-04):**

*En esta fase se integran los resultados del modelo base y del nuevo modelo, se realiza un análisis comparativo y se preparan los productos finales del proyecto, incluyendo la redacción del informe definitivo y la preparación de la presentación para la defensa.*

Cada una de estas fases se desarrolla a través de ciclos iterativos (sprints) de aproximadamente dos semanas, que incluyen planificación, ejecución de tareas, seguimiento, revisión de resultados y retrospectiva. Este ciclo se repite para cada fase, promoviendo la mejora continua y la adaptación a los desafíos del proyecto.

Además, el ciclo de vida incorpora puntos de control y revisión con el tutor (Product Owner) al cierre de cada sprint y en momentos clave, que permiten evaluar el avance, recoger retroalimentación y definir posibles ajustes en los objetivos o en la estrategia de trabajo.

En conjunto, el modelo de ciclo de vida adoptado combina las ventajas de un enfoque ágil —flexibilidad, iteración, entrega progresiva de valor— con las necesidades propias de un proyecto de investigación, asegurando una gestión adaptativa, centrada en resultados y orientada al aprendizaje continuo.

## 3.3. Papeles desempeñados en el proyecto

El desarrollo del presente Trabajo de Fin de Grado ha seguido un enfoque basado en metodologías ágiles, adoptando el marco de Scrum como guía principal de organización. En este contexto, los papeles desempeñados en el proyecto se han adaptado a las características de un proyecto académico con un único desarrollador.

Los roles principales han sido los siguientes:

* **Product Owner (Tutor)**: El tutor del TFG ha asumido el papel de Product Owner, siendo responsable de orientar la definición de objetivos, priorizar las tareas principales, proporcionar retroalimentación durante las reuniones de seguimiento y supervisar la calidad de los entregables parciales y finales. Su intervención ha sido clave para asegurar que el proyecto mantuviera la alineación con los objetivos académicos y de investigación.
* **Scrum Team (Alumno)**: El alumno ha desempeñado de manera integral el rol de Scrum Team, asumiendo todas las funciones operativas del proyecto. Entre sus responsabilidades principales se incluyen la planificación de los sprints, la ejecución de las tareas definidas en las historias de usuario, la documentación de los resultados, la implementación y evaluación de los modelos de aprendizaje automático, así como la redacción de la memoria final.
* **Scrum Master (Autogestionado)**: Dado el carácter individual del proyecto, las funciones propias del rol de Scrum Master —facilitar los procesos, eliminar impedimentos y velar por el cumplimiento de la metodología— han sido asumidas también por el alumno, en un ejercicio de autogestión y autorregulación, contando con la supervisión puntual del tutor para resolver bloqueos específicos.

La adaptación de estos roles ha permitido mantener una organización ágil y flexible, favoreciendo la autonomía en la ejecución y la iteración continua mediante revisiones periódicas con el tutor, alineadas con las prácticas de revisión continua y pair programming adaptado propias de eXtreme Programming (XP).

## 3.4. Planificación y Organización (épicas, sprints, entregables)

La aplicación de esta metodología ágil se ha materializado en una planificación inicial estructurada en cuatro épicas principales, cada una con sus respectivos sprints y entregables definidos. Este esquema permite gestionar el proyecto de manera iterativa, incremental y flexible, adaptándose a los retos y descubrimientos propios de un proceso investigativo. La planificación detallada de las épicas, sprints y tareas específicas se describe en el apartado correspondiente, y se complementa con un Anexo que incluye la descomposición completa de las tareas, sus dependencias y criterios de aceptación.

La metodología ágil adoptada se fundamenta principalmente en el marco de Scrum, complementado con prácticas específicas de eXtreme Programming (XP), adaptadas al contexto académico e investigativo. La organización del trabajo se estructura en épicas, entendidas como grandes bloques funcionales o fases principales del proyecto, subdivididas a su vez en sprints de dos semanas de duración.

Cada sprint incluye las siguientes fases principales:

* **Definición de User Stories**: Se formulan descripciones claras y concisas de las tareas a realizar, alineadas con los objetivos específicos de cada fase. Estas historias de usuario contemplan actividades como la revisión del código base, la replicación de resultados previos y la experimentación con nuevos modelos. Cada historia incorpora criterios de aceptación que permiten evaluar objetivamente su grado de finalización.
* **Planificación del Sprint**: Antes de iniciar cada sprint, se priorizan las historias de usuario en función de su impacto y viabilidad, estableciendo un alcance realista y orientado a la consecución de entregables parciales.
* **Seguimiento y Control**: Durante el sprint se realiza un seguimiento continuo del estado de las tareas mediante un tablero Kanban implementado en Excel, que permite visualizar el progreso, detectar bloqueos y replanificar si fuera necesario.
* **Revisión y Retrospectiva**: Al finalizar cada sprint se evalúa el cumplimiento de los objetivos, se documentan los logros y se reflexiona sobre las áreas de mejora, con el fin de aplicar los aprendizajes en los ciclos siguientes.

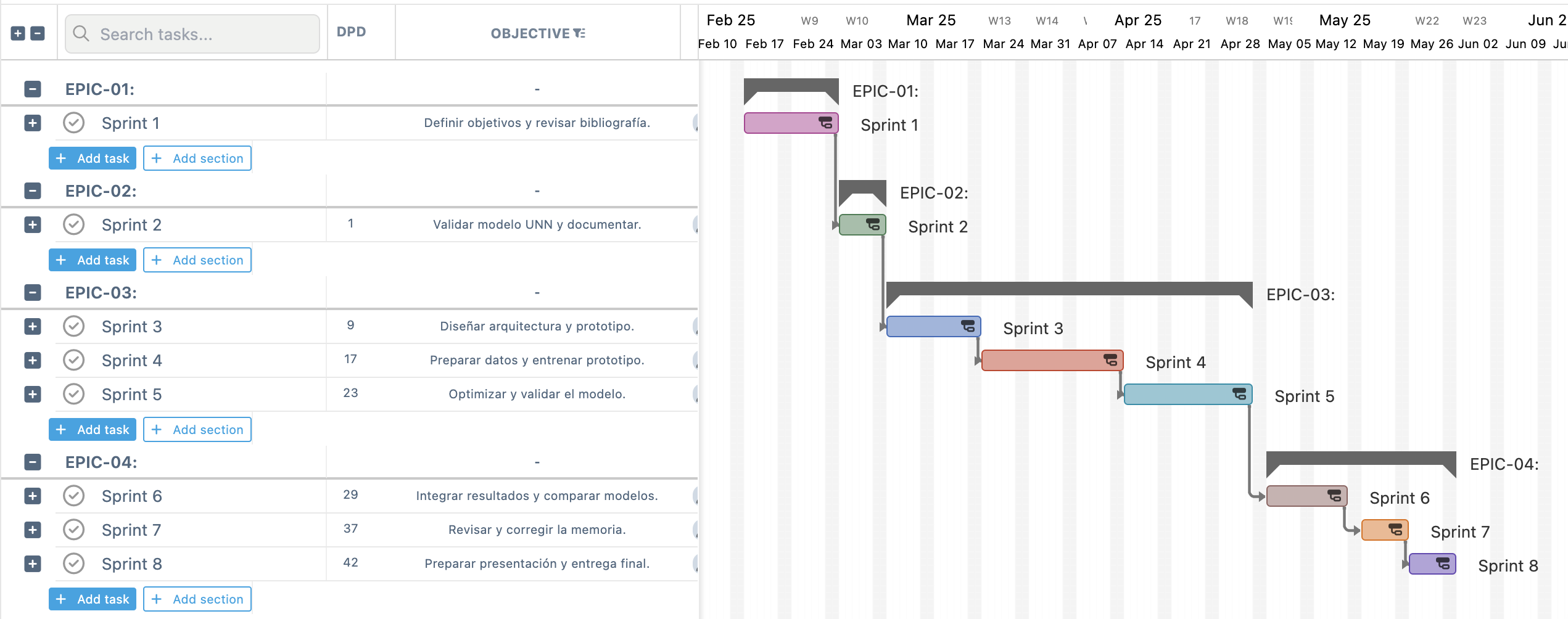
En paralelo, se integran prácticas de eXtreme Programming (XP) como:

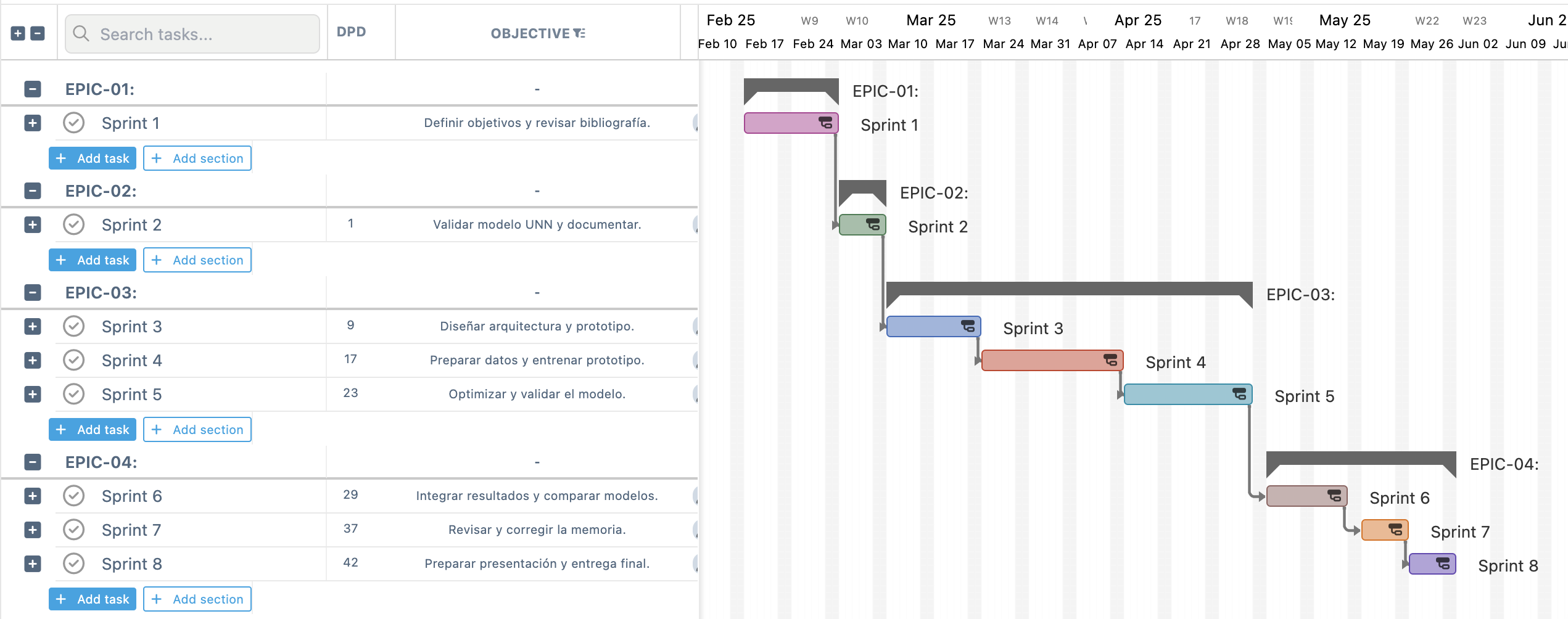
* Revisión continua del código y documentación de funciones (integrada en las tareas de análisis y comprensión del modelo base).
* Validación incremental de los prototipos mediante pruebas funcionales durante las fases de entrenamiento y optimización.
* Revisiones periódicas con el tutor, entendidas como una forma adaptada de “pair programming” orientada al control de calidad y mejora continua.
* Criterios de aceptación explícitos para cada tarea y sprint, que aseguran una definición clara de los entregables y sus estándares de calidad.

Este enfoque metodológico se acompaña de reuniones semanales con el tutor (Product Owner), en las que se revisa el progreso, se priorizan nuevas actividades y se definen posibles ajustes en los objetivos o estrategias.

A continuación, se detalla la organización en épicas y sprints de manera compacta. En el Anexo se incluye una versión extensa y detallada con las tareas que abarca cada sprint.

|  |  |  |  |
| --- | --- | --- | --- |
| **Sprint** | **Épica** | **Objetivo** | **Entregables** |
| **Sprint 1** | **EPIC-01** | Definir alcance, organizar backlog, revisión bibliográfica | Alcance definido,  backlog priorizado,  marco teórico inicial |
| **Sprint 2** | **EPIC-02** | Validar y documentar modelo base UNN | Código ejecutado y documentado,  resultados iniciales, metodología preliminar |
| **Sprint 3** | **EPIC-03 (Parte I)** | Diseñar arquitectura y prototipo del nuevo modelo | Arquitectura definida, prototipo base, metodología redactada |
| **Sprint 4** | **EPIC-03 (Parte II)** | Entrenar prototipo y obtener resultados preliminares | Prototipo entrenado, resultados preliminares, redacción resultados iniciales |
| **Sprint 5** | **EPIC-03 (Parte III)** | Optimizar y validar modelo | Modelo optimizado y validado,  documentación mejoras, revisión parcial con tutor |
| **Sprint 6** | **EPIC-04 (Parte I)** | Integrar y comparar resultados | Resultados integrados y comparados,  análisis preliminar, documentación análisis, revisión parcial con tutor |
| **Sprint 7** | **EPIC-04 (Parte II)** | Revisión y corrección del borrador final | Borrador revisado y corregido |
| **Sprint 8** | **EPIC-04 (Parte III)** | Preparación defensa y entrega final | Presentación final, memoria definitiva |





## 3.5. Ejecución práctica

“usando **datos reales del universo para buscar posibles señales de materia oscura con algoritmos de IA”**

La ejecución técnica del proyecto se centró en el desarrollo y evaluación de una serie de modelos de detección de anomalías aplicados al catálogo Fermi-LAT, con el objetivo de identificar posibles candidatos a materia oscura. Para ello, se utilizó como base la librería scikit-learn en Python, trabajando inicialmente en entornos Jupyter Notebook, lo que permitió realizar un análisis interactivo y exploratorio de los datos, así como una visualización directa de los resultados intermedios.

El desarrollo comenzó con la implementación de un modelo One-Class Support Vector Machine (OCSVM) entrenado exclusivamente con datos de fuentes astrofísicas identificadas (ASTRO) del catálogo Fermi-LAT, utilizando inicialmente dos características principales (2F): Eₚₑₐₖ y Beta. Este modelo fue utilizado para realizar predicciones sobre el conjunto de fuentes no identificadas (UNIDs), con el fin de clasificar aquellas que mostraban características anómalas respecto al conjunto de entrenamiento.

Posteriormente, se ampliaron las características utilizadas a tres (3F) y cuatro (4F), incorporando nuevas variables disponibles en los datos, lo que permitió evaluar el impacto de incluir mayor información en la capacidad de detección y clasificación del modelo. Cada versión del modelo OCSVM se ejecutó al menos tres veces, utilizando diferentes particiones de datos de entrenamiento y prueba, con el objetivo de analizar la estabilidad de las predicciones y la sensibilidad del modelo frente a la selección de datos.

De manera complementaria, se realizó un análisis del modelo de red neuronal previamente desarrollado en estudios anteriores, cuyo código fue revisado y validado para asegurar su correcto funcionamiento. Este modelo sirvió como referencia comparativa frente a los modelos OCSVM implementados en este trabajo, permitiendo contrastar los resultados y evaluar posibles coincidencias en la selección de candidatos a materia oscura.

A lo largo de la ejecución se emplearon métricas de evaluación estándar en problemas de clasificación, incluyendo la matriz de confusión, la puntuación F1 y la precisión, aplicadas en los casos donde se contaba con etiquetas de referencia. La validación cualitativa de los resultados se apoyó también en la visualización gráfica de los datos y las predicciones, facilitando la interpretación de los patrones detectados y la identificación de resultados coherentes o inesperados.

Todo el proceso se documentó de manera incremental, registrando los experimentos realizados, los parámetros utilizados en cada caso y los resultados obtenidos, con el fin de asegurar la trazabilidad y la reproducibilidad del trabajo. Actualmente se valora la posibilidad de migrar el código desarrollado en notebooks a un formato de software más estructurado en Python, con el objetivo de mejorar su organización, facilitar su reutilización y ampliar su aplicabilidad en futuros trabajos.

El modelo OneClassSVM entra dentro de los. Modelos no supevisados de **detección de anomalías**, los cuales engloban tanto la detección de valores atípicos como la detección de novedades.

En el caso concreto de OneClassSVM, se utiliza para la **detección de novedades**, es decir, los datos de entrenamiento se asumen que no están “contaminados” por valores atípicos, y en lo que se está interesado es en si una **nueva observación** se trata de un valor atípico. En este contexto, al valor atípico también se le llama novedad.

Se eligió este modelo no supervisado como modelo a entrenar con la intención de, en un primer lugar, entrenar al modelo tan solo con datos “**ASTRO**”. Es decir, se entrena al modelo con la fuente de datos del catálogo del satélite Fermi-LAT que contiene **fuentes identificadas**, con sus correspondientes valores para uno determinados parámetros que se explicarán en detalle más adelante (E peak y beta). Así, una vez e modelo haya aprendido lo que es “normal” (considerando las fuentes astrofísicas identificadas como lo normal), se le alimenta con los datos **UNIDs**, o fuentes no identificadas del catálogo.

Una vez hecho esto, se entrena otro modelo pero específicamente con los datos simulados **DM**, o de materia oscura, y se le alimenta al modelo las fuentes ASTRO, con la intención de ver si alguna fuente identificada la considera como “normal”, o en este caso, materia oscura.

Para el desarrollo del modelo se ha utilizado la librería de código abierto para Python Scikit-Learn, que permite construir modelos clásicos de ML. Se trata de una *suite* que incluye una variedad de herramientas cubriendo todas las fases del ciclo de desarrollo de un modelo de Machine Learning (preprocesamiento, entrenamiento y pruebas, ajuste de hiperparámetros, …) por lo que permite construir el pipeline completo del modelo de aprendizaje automático solo utilizando su librería.

# 4. Desarrollo experimental

## 4.1. Datos utilizados y preprocesamiento

↳ Incluir descripción de los conjuntos: ASTRO, UNIDs, DM simulado y nuevo dataset 3F.

## 4.2. Estudio preliminar del modelo base (ANN)

↳ Exploración del código heredado, estructura, y primeros resultados.

## 4.3. Desarrollo de los modelos OCSVM

En este trabajo se utiliza el algoritmo One-Class Support Vector Machine (OCSVM) como herramienta para modelar el comportamiento de fuentes astrofísicas conocidas (ASTRO) y detectar posibles fuentes anómalas entre los objetos no identificados (UNIDs) del catálogo 4FGL.

El modelo OCSVM está diseñado específicamente para tareas de detección de anomalías cuando se dispone únicamente de datos pertenecientes a una sola clase (en este caso, fuentes astrofísicas conocidas). El objetivo es aprender una frontera de decisión que delimite la región del espacio de características donde se concentran los datos normales, de forma que cualquier punto que caiga fuera de esta región pueda considerarse una posible anomalía.

Hiperparámetros y su interpretación

El comportamiento del modelo OCSVM depende principalmente de dos hiperparámetros clave:

|  |  |  |
| --- | --- | --- |
| Parámetro | Descripción | Impacto |
| ν (nu) | Proporción máxima de **datos de entrenamiento** que el **modelo puede considerar anómalos**.  También sirve como una cota inferior del porcentaje de soporte. | Si **nu** es **pequeño**, la frontera se ajusta ampliamente para abarcar casi todos los datos ASTRO.  Si **nu** es **grande**, se **toleran más** "**outliers**" dentro del entrenamiento. |
| γ (gamma) | Define la influencia de cada muestra sobre la **forma de la frontera** (solo para kernel RBF). | **Valores** **altos** de gamma producen **fronteras muy ajustadas** y sinuosas, mientras que valores bajos dan lugar a fronteras más suaves. |

Objetivo del modelo en este contexto

El objetivo del modelo OCSVM es detectar **candidatos a materia oscura** entre los UNIDs, partiendo del supuesto de que las fuentes ASTRO representan el patrón de "normalidad". Por tanto:

* Es deseable que la **frontera de decisión englobe la mayor parte de las fuentes ASTRO**.
* Si el modelo empieza a clasificar como anómalas muchas fuentes ASTRO, está **sobreajustando** (falsos positivos).
* Si la frontera es demasiado amplia, se corre el riesgo de que también **los UNIDs anómalos** queden clasificados como normales (falsos negativos).

Criterio de ajuste

En este trabajo se ha optado por **ajustar la frontera de decisión** de forma que:

* Se incluyan la mayoría (idealmente > 99%) de fuentes ASTRO como normales.
* Se identifiquen como anómalos únicamente los UNIDs cuya representación en el espacio de características se aleje significativamente del comportamiento aprendido.

Este enfoque busca un equilibrio entre **minimizar falsos positivos** sobre datos conocidos y **maximizar la detección de fuentes potencialmente interesantes**, como candidatas a materia oscura.

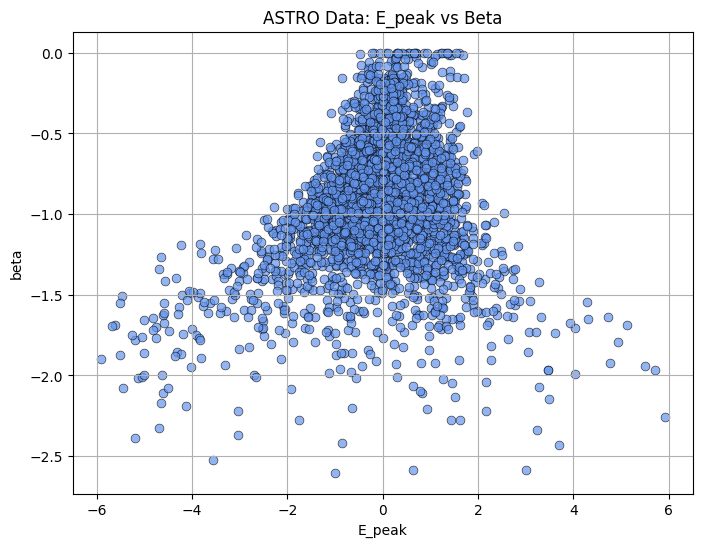
### 4.3.1 Modelo OCSVM con 2F

El modelo de detección de anomalías se entrenó utilizando un algoritmo One-Class Support Vector Machine (OCSVM), aplicando únicamente datos de fuentes astrofísicas conocidas (ASTRO) como ejemplos de comportamiento normal. Para esta configuración, se seleccionaron **dos características espectrales clave**:

* E\_peak: energía del pico espectral,
* beta: pendiente de la distribución energética.

Ambas fueron **transformadas al espacio logarítmico** y **escaladas mediante StandardScaler** para garantizar una correcta convergencia del modelo y comparabilidad entre magnitudes.

La **Figura 4.1** muestra la distribución de las fuentes astrofísicas conocidas (ASTRO) en el espacio de características seleccionado, E\_peak y beta, tras aplicar la transformación logarítmica. Esta representación permite observar la forma y densidad del conjunto de entrenamiento que el modelo One-Class SVM tomará como referencia del comportamiento normal.



***Figura 4.1.*** *Distribución bidimensional de las fuentes astrofísicas conocidas (ASTRO) en el espacio de características E\_peak y beta. Se observan concentraciones de puntos en torno a regiones centrales, lo que sugiere una forma estructurada útil para el aprendizaje de patrones normales por parte del modelo.*

¿Qué hace el modelo?

1. Aprende el espacio ocupado por fuentes astro (E\_peak, beta).
2. Cuando se aplica a nuevos datos (por ejemplo, UNIDs), decide:

* Si están dentro del “espacio de normalidad” → inlier (astro-like)
* Si están fuera → outlier (potencialmente anomalía → candidata a materia oscura)

Hipótesis del modelo

Dado que OCSVM es un **modelo no supervisado**, se entrenó exclusivamente sobre ejemplos etiquetados como normales, sin utilizar datos simulados de materia oscura (DM) en esta fase. Se empleó una **estrategia de validación estratificada** dividiendo el conjunto ASTRO en:

* 60% entrenamiento,
* 20% validación,
* 20% test.

La **selección de hiperparámetros** (nu, gamma) se realizó mediante un **barrido en malla** (grid search), evaluando el rendimiento en el conjunto de validación con el objetivo de que la frontera aprendida englobase al mayor número posible de ejemplos normales.

Dado que todos los datos de entrenamiento provienen de fuentes ASTRO (normalidad conocida), el modelo OCSVM aprende a:

* Estimar la región del espacio donde se encuentran las fuentes astrofísicas.
* Rechazar cualquier observación significativamente diferente como anómala.

OCSVM está diseñado para aprender la "forma" o distribución del conjunto de datos de una sola clase (normalidad) y luego identificar instancias que se desvían significativamente de ese patrón como anomalías.

Evaluación del modelo sobre datos normales

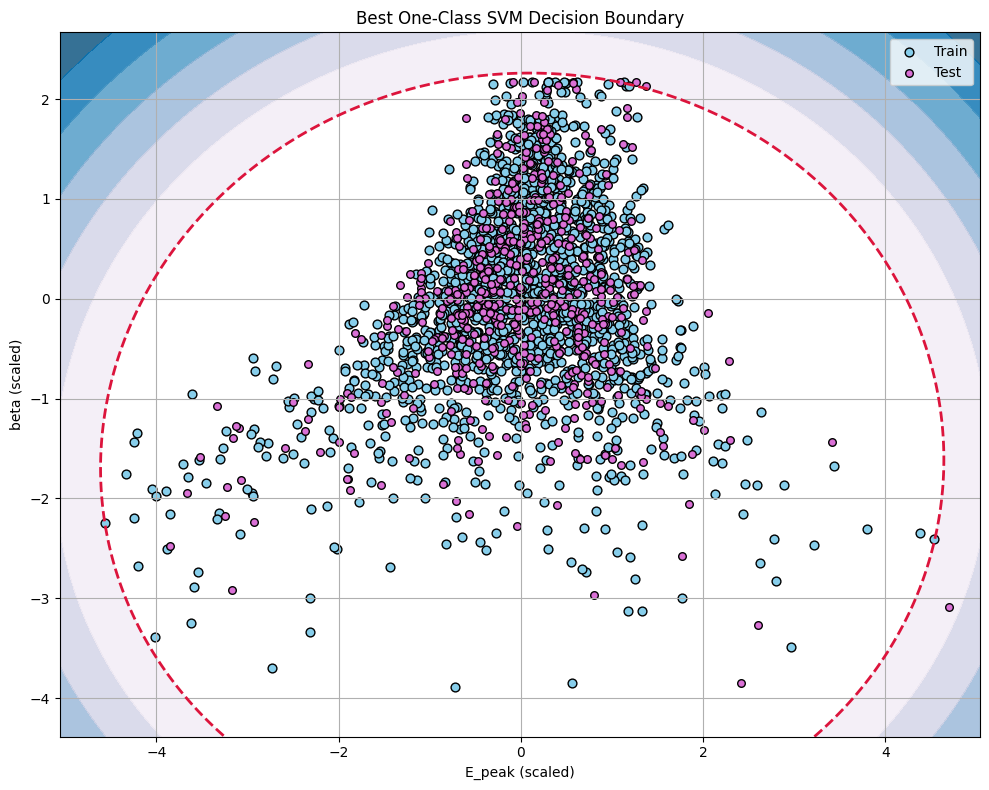
Una vez entrenado, se evaluó el modelo sobre los conjuntos de validación y test. Dado que todos los datos pertenecen a la misma clase "normal", el objetivo era **verificar que el modelo reconocía correctamente esa distribución sin identificar excesivos outliers**.

Se obtuvieron los siguientes resultados:

* **Conjunto de validación**: 2 ejemplos clasificados como anomalías.
* **Conjunto de prueba**: 4 ejemplos clasificados como anomalías.

Estos valores son **consistentes y aceptables**, ya que representan una **pequeñísima fracción del total**, mostrando que el modelo ha aprendido correctamente la distribución general del conjunto ASTRO, sin ser demasiado restrictivo ni permisivo.

La **Figura 4.2** muestra la frontera de decisión aprendida por el modelo One-Class SVM entrenado sobre las fuentes astrofísicas conocidas (clase única). Esta frontera representa el límite que el modelo considera como el comportamiento 'normal' aprendido.



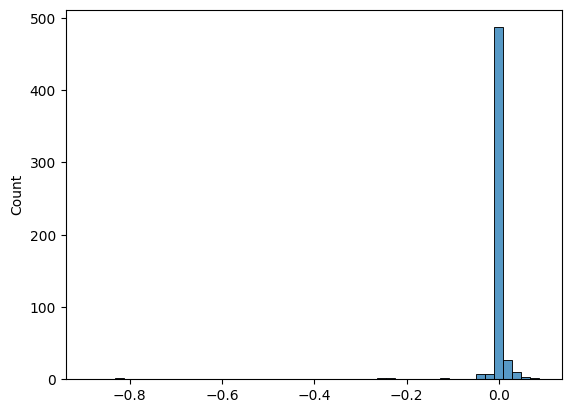
***Figura 4.2.*** *Frontera de decisión aprendida por el modelo One-Class SVM entrenado con dos características (E\_peak y beta), mostrando los conjuntos de entrenamiento y prueba. Los puntos dentro del área delimitada por la línea roja discontinua son considerados normales; los que quedan fuera, anomalías.*

Análisis de scores del modelo

Se construyó un histograma de las puntuaciones (decision\_function) del modelo sobre los datos normales. El histograma muestra una distribución concentrada en valores positivos, lo que indica que **la mayoría de los puntos fueron correctamente considerados normales**, con solo unos pocos con valores negativos (es decir, ligeramente alejados de la frontera de decisión).

Este comportamiento confirma visualmente la robustez del modelo y su adecuada calibración en este espacio de 2 características.

La **Figura 4.3** muestra el histograma de las puntuaciones de decisión sobre el conjunto de validación. Estas puntuaciones indican qué tan 'normal' o 'anómalo' considera el modelo cada punto.



***Figura 4.3.*** *Distribución de puntuaciones de decisión en el conjunto de validación. Valores cercanos a 0 indican alta similitud con el comportamiento aprendido por el modelo; valores negativos indican mayor desviación.*

### 4.3.2 Comparación con ANN (2F)

### 4.3.3 Modelo OCSVM con 4F

### 4.3.4 Comparación con ANN (4F)

### 4.3.5 Modelo OCSVM con 3F (nuevo dataset)

## 4.4. Validación y análisis de resultados

↳ Decisión boundaries, scores, interpretación de outliers, repetición y robustez.

## 4.5. Discusión final

↳ Análisis cruzado de modelos, observaciones clave y solidez de los resultados.

# 5. Conclusiones y líneas futuras

# 6. Entregables

## 6.1. Repositorio del proyecto

## 6.2. Manual de uso

# 7. Bibliografía

* Bobadilla, J. (2021). *Machine learning y deep learning usando Python, Scikit y Keras*. Ediciones Paraninfo.
* Carleo, G., Cirac, I., Cranmer, K., Daudet, L., Schuld, M., Tishby, N., Vogt-Maranto, L., & Zdeborová, L. (2019). *Machine learning and the physical sciences.* Reviews of Modern Physics, 91(4), 045002. <https://doi.org/10.1103/RevModPhys.91.045002>
* Çelik, Ö., & Altunaydin, S. S. (s.f.). *A Research on Machine Learning Methods and Its Applications*. Osmangazi University.
* NASA/DOE/Fermi LAT Collaboration. (s.f.)*. Fermi Large Area Telescope (LAT) Data*. <https://fermi.gsfc.nasa.gov/ssc/>
* Abdollahi, S., Acero, F., Ackermann, M., Ajello, M., Atwood, W. B., Baldini, L., ... & Zimmer, S. (2020). Fermi Large Area Telescope Fourth Source Catalog. *The Astrophysical Journal Supplement Series*, 247(1), 33. <https://doi.org/10.3847/1538-4365/ab6bcb>
* Bertone, G., & Hooper, D. (2018). History of dark matter. *Reviews of Modern Physics*, 90(4), 045002. <https://doi.org/10.1103/RevModPhys.90.045002>
* Cirelli, M., Corcella, G., Hektor, A., Hütsi, G., Kadastik, M., Panci, P., ... & Strumia, A. (2011). PPPC 4 DM ID: A Poor Particle Physicist Cookbook for Dark Matter Indirect Detection. *Journal of Cosmology and Astroparticle Physics*, 2011(03), 051. <https://doi.org/10.1088/1475-7516/2011/03/051>
* Fermi-LAT Collaboration. (2020). Fermi Science Support Center (FSSC). Retrieved from <https://fermi.gsfc.nasa.gov/ssc/>
* Mirabal, N., Nieto, D., & Pardo, S. (2012). Constraining dark matter properties with unidentified gamma-ray sources. *Monthly Notices of the Royal Astronomical Society: Letters*, 424(1), L64-L68. <https://doi.org/10.1111/j.1745-3933.2012.01287>.
* Saz Parkinson, P. M., & Farrar, G. R. (2017). Machine learning approaches to gamma-ray source classification. *The Astrophysical Journal*, 835(2), 195. <https://doi.org/10.3847/1538-4357/835/2/195>
* European Space Agency. (2015). Planck reveals an almost perfect universe. Retrieved May 7, 2025, from <https://www.esa.int/Science_Exploration/Space_Science/Planck/Planck_reveals_an_almost_perfect_Universe>
* Planck Collaboration. (2013). Planck 2013 results. I. Overview of products and scientific results. Astronomy & Astrophysics, 571, A1. <https://doi.org/10.1051/0004-6361/201321529>
* Bou Nassif, A., Abu Talib, M., Nasir, Q., & Dakalbab, F. M. (2022). Machine learning for anomaly detection: A systematic review. IEEE Access, 10, 59771–59794. <https://doi.org/10.1109/ACCESS.2022.3178659>
* Chandola, V., Banerjee, A., & Kumar, V. (2009). Anomaly detection: A survey. ACM Computing Surveys, 41(3), 1–58. <https://doi.org/10.1145/1541880.1541882>
* Schölkopf, B., Platt, J., Shawe-Taylor, J., Smola, A. J., & Williamson, R. C. (2001). Estimating the support of a high-dimensional distribution. Neural Computation, 13(7), 1443–1471. <https://doi.org/10.1162/089976601750264965>
* Zimek, A., Schubert, E., & Kriegel, H.-P. (2012). A survey on unsupervised outlier detection in high-dimensional numerical data. Statistical Analysis and Data Mining: The ASA Data Science Journal, 5(5), 363–387.

Fermi Large Area Telescope Fourth Source Catalog Data Release 4 (4FGL-DR4)

<https://ar5iv.labs.arxiv.org/html/2307.12546#:~:text=We%20present%20an%20incremental%20version,are%20updated%20for%20all%20sources>

Fermi Gamma-ray Space Telescope Currently Available Data Products

<https://fermi.gsfc.nasa.gov/ssc/data/access/#:~:text=,51>

How and Why to Use Agile for Machine Learning

<https://medium.com/qash/how-and-why-to-use-agile-for-machine-learning-384b030e67b6>

Machine Learning Steps

<https://www.simplilearn.com/tutorials/machine-learning-tutorial/machine-learning-steps>

What is Agile Iteration

<https://www.wrike.com/agile-guide/faq/what-is-agile-iteration/>

Life Cycle of a Machine Learning Project

<https://neptune.ai/blog/life-cycle-of-a-machine-learning-project>

Navigating Through a Machine Learning Project: A Step-by-Step Guide

<https://medium.com/@careerInAI/navigating-through-a-machine-learning-project-a-step-by-step-guide-d673f72a80ff>

Classification of Fermi -LAT sources with deep learning using energy and time spectra

<https://www.researchgate.net/publication/354086181_Classification_of_Fermi_-LAT_sources_with_deep_learning_using_energy_and_time_spectra>

# 8. Anexos

## 8.2. Planificación detallada de tareas

**EPIC-01: Definición del Alcance y Organización del Proyecto**

**Objetivo**: Definir el marco de trabajo, los objetivos generales y organizar el backlog inicial.

Sprint 1 (Semanas 1-2)

|  |  |  |
| --- | --- | --- |
| ID Tarea | Descripción | Dependencias |
| TSK-EP1-S1-01 | Definir alcance y objetivos | - |
| TSK-EP1-S1-02 | Crear backlog y priorizar tareas | TSK-EP1-S1-01 |
| TSK-EP1-S1-03 | Revisión de literatura sobre UNN | - |
| TSK-EP1-S1-04 | Revisión de literatura sobre técnicas de detección de anomalías | - |
| TSK-EP1-S1-05 | Redacción preliminar de introducción y marco teórico | TSK-EP1-S1-03 + TSK-EP1-S1-04 |

**EPIC-02: Validación y Comprensión del Modelo Existente (UNN)**

**Objetivo**: Ejecutar, comprender y documentar el modelo base UNN.

Sprint 2 (Semanas 3-4)

|  |  |  |
| --- | --- | --- |
| ID Tarea | Descripción | Dependencias |
| TSK-EP2-S2-01 | Ejecutar código UNN | - |
| TSK-EP2-S2-02 | Revisar código fuente | TSK-EP2-S2-01 |
| TSK-EP2-S2-03 | Comentar y documentar funciones | TSK-EP2-S2-02 |
| TSK-EP2-S2-04 | Documentar resultados del modelo UNN | TSK-EP2-S2-01 |
| TSK-EP2-S2-05 | Redactar metodología preliminar | TSK-EP2-S2-04 |

**EPIC-03: Desarrollo del Nuevo Modelo de Detección de Anomalías**

Objetivo: Diseñar, implementar, entrenar y optimizar el nuevo modelo.

Sprint 3 (Semanas 5-6)

|  |  |  |
| --- | --- | --- |
| ID Tarea | Descripción | Dependencias |
| TSK-EP3-S3-01 | Revisar y definir requisitos técnicos | TSK-EP1-S1-01 |
| TSK-EP3-S3-02 | Diseñar arquitectura preliminar | TSK-EP3-S3-01 |
| TSK-EP3-S3-03 | Implementar estructura base prototipo | TSK-EP3-S3-02 |
| TSK-EP3-S3-04 | Realizar spikes experimentales | TSK-EP3-S3-02 |
| TSK-EP3-S3-05 | Redactar metodología (parte diseño modelo) | TSK-EP3-S3-02 |

Sprint 4 (Semanas 7-8)

|  |  |  |
| --- | --- | --- |
| ID Tarea | Descripción | Dependencias |
| TSK-EP3-S4-01 | Preparar y dividir datos | - |
| TSK-EP3-S4-02 | Configurar entorno de experimentación | - |
| TSK-EP3-S4-03 | Entrenar prototipo modelo | TSK-EP3-S4-01 + TSK-EP3-S4-02 |
| TSK-EP3-S4-04 | Evaluar resultados preliminares | TSK-EP3-S4-03 |
| TSK-EP3-S4-05 | Redacción preliminar de resultados iniciales | TSK-EP3-S4-04 |

Sprint 5 (Semanas 9-10)

|  |  |  |
| --- | --- | --- |
| ID Tarea | Descripción | Dependencias |
| TSK-EP3-S5-01 | Optimizar hiperparámetros | TSK-EP3-S4-04 |
| TSK-EP3-S5-02 | Validación cruzada | TSK-EP3-S5-01 |
| TSK-EP3-S5-03 | Evaluar en set independiente | TSK-EP3-S5-02 |
| TSK-EP3-S5-04 | Documentar mejoras y resultados | TSK-EP3-S5-03 |
| TSK-EP3-S5-05 | Revisión parcial con tutor | TSK-EP3-S5-03 |

**EPIC-04: Integración, Análisis Comparativo y Documentación Final**

**Objetivo**: Integrar resultados, realizar análisis comparativo y finalizar documentación y presentación.

Sprint 6 (Semanas 11-12)

|  |  |  |
| --- | --- | --- |
| ID Tarea | Descripción | Dependencias |
| TSK-EP4-S6-01 | Integrar resultados UNN + modelo propio | TSK-EP2-S2-04 + TSK-EP3-S5-03 |
| TSK-EP4-S6-02 | Análisis preliminar comparativo | TSK-EP4-S6-01 |
| TSK-EP4-S6-03 | Documentar integración y análisis | TSK-EP4-S6-02 |
| TSK-EP4-S6-04 | Revisión parcial con tutor (análisis comparativo) | TSK-EP4-S6-02 |

Sprint 7 (Semanas 13-14)

|  |  |  |
| --- | --- | --- |
| ID Tarea | Descripción | Dependencias |
| TSK-EP4-S7-01 | Revisión y ajustes del borrador del informe | - |
| TSK-EP4-S7-02 | Incorporar feedback y correcciones finales | TSK-EP4-S7-01 |

Sprint 8 (Semana 15)

|  |  |  |
| --- | --- | --- |
| ID Tarea | Descripción | Dependencias |
| TSK-EP4-S8-01 | Revisión final y preparación presentación | - |
| TSK-EP4-S8-02 | Pruebas de contingencia y ajustes finales | TSK-EP4-S8-01 |

## 8.3. Criterios de aceptación

|  |  |
| --- | --- |
| **ID** | **Criterios de aceptación** |
| TSK-EP1-S1-01 | Documento con alcance aprobado por tutor, listado de objetivos generales claros. |
| TSK-EP2-S2-02 | Código fuente revisado con comentarios explicativos y sin errores de compilación/ejecución. |
| TSK-EP3-S3-03 | Prototipo ejecuta correctamente un caso de prueba simple y devuelve resultados sin errores. |
| TSK-EP3-S4-03 | El prototipo ha sido entrenado sin errores y genera métricas básicas (accuracy, loss) documentadas. |
| TSK-EP3-S5-02 | Validación cruzada completada con reporte de métricas y comparativa con baseline inicial. |
| TSK-EP4-S6-01 | Resultados del modelo propio y modelo base integrados en un mismo archivo, con tablas y gráficas listas. |
| TSK-EP4-S7-02 | Feedback del tutor aplicado al menos al 90% de los comentarios recibidos. |

## 8.4. Código o fragmentos relevantes

## 8.5. Otros materiales de apoyo