Especialización en Ciencia de Datos

Trabajo Final Integrador

Modelos de regresión para la predicción de la probabilidad de colisión entre dos cuerpos orbitando alrededor de la Tierra en una trayectoria de encuentro

Esteban Federico Wenger

Tutora:

Dra. Marcela Riccillo

Buenos Aires, 2022

Resumen

El número de satélites y chatarra espacial orbitando alrededor de la Tierra ha crecido en las últimas décadas. La chatarra espacial además de ser un tipo de contaminación ambiental, es también un inconveniente para las empresas operadoras de satélites. La industria conocida como New Space también ha crecido en los últimos años. Los satélites de estas empresas realizan órbitas que eventualmente tienen eventos de conjunción o de encuentro con otros cuerpos, ya sean otros satélites o chatarra espacial. Por este motivo, existe un Ente de Monitoreo que se encarga de emitir reportes conocidos como Conjunction Data Messages (CDM) que le avisa a las empresas operadoras de satélites cuando uno de sus satélites se encuentra dentro de una trayectoria de colisión con otro objecto. Para evaluar un evento de conjunción es fundamental la probabilidad de colisión y la mínima distancia a la que se encontrarán los cuerpos en el momento conocido como TCA (time of closest approach). En este Trabajo de Investigación se propone aplicar Técnicas de Machine Learning para predecir una de estas variables, la cual es fundamental a la hora de tomar la decisión de si el riesgo de un evento de encuentro amerita realizar una maniobra o no. Este Trabajo encara el problema planteando un Modelo Lineal y luego se avanza con Modelos No Lineales de Árboles de Decisión. Se aplican técnicas de Modelos de Ensemble de Boosting con dos algoritmos AdaBoost y Light Gradient Boosting Machines. Para entrenar estos Modelos se aplica un ETL que resulta interesante para los reportes CDM en formato estándar y es extensible a otros Modelos que se deseen aplicar. Los resultados obtenidos permiten obtener conclusiones que son contundentes en cuanto a la utilización de Técnicas de Machine Learning para aportar valor a los datos, en este caso CDM, y para la evaluación de este tipo eventos de conjunción.

Abstract

The number of satellites and Space Debris orbiting round the Earth has grown a lot in the past decades. Space Debris besides of polluting the space environment, it is also an issue for the companies that operates satellites. The New Space Industry has also grown a lot during the last years. The satellites of these companies shape orbits that eventually have Conjunction Events or Encounter Events with other bodies, satellites or debris. For that reason, there is a Monitoring Organization, which is in charge of releasing reports known as Conjunction Data Messages (CDM) that shows when one satellite of the fleet of a company is in a collision trajectory with other object. In order to evaluate a Conjunction Event it is essential the collision probability and the minimun distance between the objects at the time of closest approach (TCA). In this Final Research Project it is proposed the aplication of Machine Learning Techniques for the prediction of one of theses variables, which is one of the most important for making the decision of maneuvering or not. In this Project the problem is first addressed with a Linear Model and then with Decision Trees Non Linear Models. Ensemble Learning Technique with Boosting is applied using two algorithms AdaBoost and Light Gradient Boosting Machines. For the training of the Models an interesting Extraction, Transformation and Load (ETL) process is developed, that fits any CDM in standard format. The ETL could be use as a first step in case a researcher wants to apply any other Machine Learning Techinque. The results of this Project are overwhelming. It shows how the application of Machine Learning can add value to existing data, in this case to CDM. The results also manifest that it is feasible to apply this algorithms to evaluate Satellite Conjunction Events.

Contenidos

Sección 1

Introducción

Este trabajo aborda la temática de chatarra espacial (space debris) y automatización de maniobras disuasivas de colisión (automatic collision avoidance) a través de técnicas de aprendizaje automático. En los últimos años, ha aumentado la cantidad de objetos orbitando alrededor de la Tierra y en el futuro próximo la tendencia seguirá creciendo (Acciarini y col., 2021). Por tal motivo, las empresas operadoras de satélites deben monitorear sus naves para evitar colisiones entre ellas o con chatarra espacial. El monitoreo se realiza a través de mensajes de alerta conocidos como Mensajes de Datos de Conjunción (Conjunction Data Messages). En la industria espacial es común referirse a ellos por su nombre en inglés o las siglas CDM. El problema existente, es que al momento de tomar la decisión se usa el último CDM disponible, el cual tiene un nivel de incertidumbre asociado. Si bien hoy existe un auge en cuanto al estudio de la chatarra espacial Kessler y Cour-Palais (Kessler y Cour-Palais, 1978) ya lo habían advertido casi 50 años atrás. La Agencia Espacial Europea (ESA) tiene un departamento específico para el estudio de esta problemática, vale aclarar que no es solo importante para evitar colisiones entre naves espaciales, satélites y chatarra, sino que, también es vital desde el punto de vista ambiental.

La hipótesis propuesta es conseguir ajustar un modelo de regresión para predecir la probabilidad de colisión del próximo reporte de evento de conjunción (CDM).

Este informe comienza con un breve repaso del marco conceptual, antecedentes y estado del arte en el apartado estado de la cuestión. Luego se determina el problema de estudio y, se define el alcance y limitaciones de esta investigación. Después se plantea la hipótesis, se detallan las variables involucradas, y se proponen los objetivos. Se explica la metodología a utilizar durante el proyecto. Se desarrolla la solución, validación de resultados, discusión y conclusiones. Al finalizar se incluye el apartado correspondiente de bibliografía.

Sección 2

Estado de la cuestión

1. Marco conceptual

Desde hace décadas que la humanidad comenzó a realizar vuelos espaciales, más precisamente en 1957 con el primer satélite ruso conocido como Sputnik-1. A lo largo de todos estos años, se ha realizado un gran avance pero esto también viene acompañado de la generación de desechos o chatarra espacial. Este problema, ya había sido detectado por Kessler (Kessler y Cour-Palais, 1978) quien introdujo conceptos como cinturón de chatarra (debris belt) y probabilidad de colisión entre objetos. Kessler identificó tres tipos de chatarra: satélites inoperativos, motores cohete, y otros restos de material asociados con el lanzamiento y desprendimiento de las diferentes etapas. Además concluyó, ya en aquel momento, que las colisiones entre satélites serían una fuente adicional de chatarra antes del 2000, e incluso realizó proyecciones para el 2020.

En la actualidad con la reducción de los costos de los lanzamientos y con el acceso a la tecnología, el espacio se vuelve una posibilidad real para empresas privadas. En la Figura 2.1 publicada en el sitio oficial de la ESA dedicado a la seguridad espacial (ESA, 2021), puede verse el cambio en la tendencia y el gran aumento en la cantidad de objetos lanzados a órbitas terrestres a partir del 2010. Por este motivo, toma importancia estudiar las posibles colisiones entre ellos. Los eventos de colisión son monitoreados y se emiten reportes conocidos bajo las siglas CDM (conjunction data messages), para un mismo evento se generan diferentes mensajes a lo largo del tiempo cada uno con datos más precisos que el anterior. En los CDM se destacan los siguientes parámetros:

- Identificador de los objetos
- Probabilidad de colisión
- Distancia entre los cuerpos en el TCA (time of closest approach) que es el instante en el que los cuerpos se encuentran más próximos entre sí.
- Incertidumbre asociada al evento expresado a través de covarianzas.

Los CDM junto a cada uno de sus atributos se verán en detalle en 9.2 Análisis de los datos.

Para mitigar el riesgo de colisión se realiza una maniobra disuasiva de propulsión siempre y cuando sea posible. A nivel operativo es importante resaltar que propulsar insume recursos y supone también que el satélite que realiza la maniobra esté fuera de servicio, es decir, durante este intervalo de tiempo no agrega valor al negocio. Por lo general, las maniobras se definen por el equipo de Operaciones de la empresa operadora del satélite en función de un protocolo. En este trabajo de investigación tomaremos el criterio propuesto por Merz en la 7ma Conferencia Europea de Chatarra Espacial (Merz y col., 2017). Merz propone que si se toma la decisión 24 horas antes del TCA la disminución del riesgo de colisión es considerable, se entrará en detalles en el 9.2.1 Conjunto de datos de entrada.

Entonces, resulta de interés estudiar los mensajes de alerta de colisión para poder optimizar los tiempos y planificación de las maniobras.

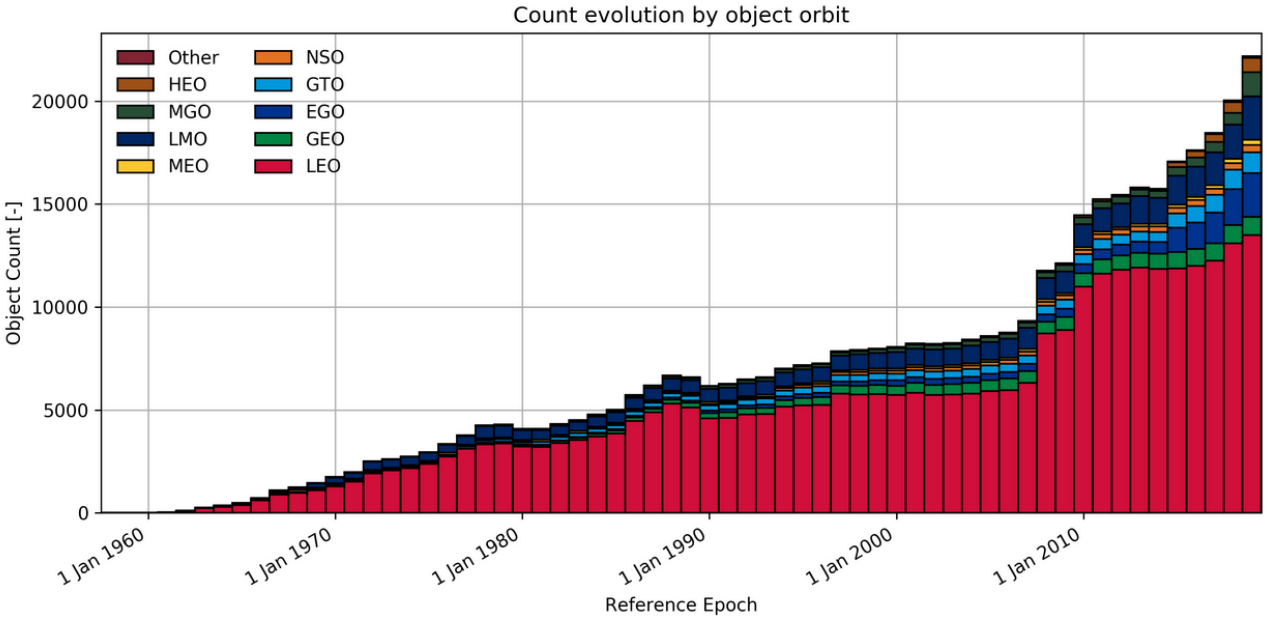

Figura 2.1: *Cantidad de objetos en la órbita terrestre a lo largo de los años, clasificados según el nivel de altura. (ESA, 2021)*

2. Antecedentes

El método oficial para calcular la probabilidad de colisión que utiliza la ESA es a través de un software denominado CORAM que cuenta con dos módulos (Cobo y col., 2014).

- *CORCOS:* (*COllision Risk COmputation Software*) este módulo se utiliza para propagar la órbita de los diferentes objetos espaciales, buscar posibles encuentros cercanos y evaluar el riesgo de colisión, es decir, calcular su probabilidad de ocurrencia. Además cuenta con

todas las librerías necesarias para las transformaciones entre los sistemas de referencia. Tiene modelos válidos tanto para objetos esféricos o de geometrías complejas, y para encuentros a alta y baja velocidad.

Asumiendo que los objetos puedan considerarse esféricos, los algoritmos que incluye este programa para hacer el cómputo son los siguientes:

- Akella: simplifica el problema de encuentro pasándolo de tres dimensiones a dos dimensiones. En este análisis toda la incertidumbre en cuanto a la posición de los cuerpos queda contenida en el plano donde se produce el encuentro. Entiéndase por incertidumbre, la diferencia entre la posición real y estimada del cuerpo (Akella y Alfriend, 2000).
- Patera: calcula la probabilidad de colisión a través de una integral de camino cerrado cuya expresión matemática permite mayor velocidad de cómputo con respecto al algoritmo anterior (Patera, 2001).
- Máxima probabilidad de acuerdo a Chan (Chan, 2008).
- Máxima probabilidad de acuerdo a Klinkrad (Klinkrad, 2006).
- Covarianza escalada: en este método se escala la matriz de covarianzas de los dos objetos, preservando la forma y orientación de cada cuerpo.

- *CAMOS:* (*Collision Avoidance Maneuver Optimization Software*) con este módulo se evalúan diferentes estrategias para mitigar el riesgo de colisión, es decir, propone una maniobra de propulsión disuasiva.

La estrategia que se utiliza es recopilar los CDM en una base de datos, luego analizar todas las posibles colisiones utilizando el módulo CORCOS. Este programa calcula como dato de salida la probabilidad de colisión. Una vez hecho esto, se pasa al módulo CAMOS y se proyectan las posibles maniobras de propulsión. Finalmente, se vuelve a ejecutar CORCOS pero ahora considerando los resultados obtenidos en CAMOS para verificar que la maniobra propuesta disminuya el riesgo de colisión efectivamente.

3. Estado del arte

Hasta hace pocos tiempo, estos problemas de encuentro eran solamente abordados analíticamente de acuerdo a las investigaciones y los algoritmos nombrados anteriormente. Pero con el auge del aprendizaje automático los investigadores están proponiendo nuevas técnicas para realizar predicciones acerca del evento de colisión. Se centran en estimar si los parámetros del CDM convergen en una colisión o no. Esto permitiría, optimizar los tiempos y no tener que esperar

hasta último momento la aparición de un nuevo CDM. Se aborda este tema principalmente con dos tipos de algoritmos: Redes Neuronales y Árboles de Decisión.

Metz junto a otros investigadores trabajan con ambos algoritmos y se enfocan en predecir la incertidumbre en la posición del objeto en la órbita en el TCA (Metz, Simon y Letizia, 2021). Expresado a través de las covarianzas posicionales y así obtener una probabilidad de colisión asociada. La ventaja que presenta encarar el modelo de esta manera es que aprovecha el conjunto de datos (dataset) de CDM completo, ya que todos los eventos tienen varianzas posicionales asociadas y no solamente aquellos que corresponden a un evento de colisión de alto riesgo. Esto se destaca porque si bien las colisiones existen, son eventos escasos dentro de los datos de entrada. De esta manera estos investigadores omiten este problema. Otro supuesto que asumen es definir como cuerpo objetivo (target) al satélite de la operadora, mientras que denomina cuerpo perseguidor (chaser) a aquella pieza de chatarra espacial. Propone esta división, ya que del satélite operativo se tiene mayor precisión de su vector de estado (posición y velocidad) dado que es el objeto de interés monitoreado por la empresa y, por el contrario, del perseguidor, o chatarra espacial tiene una mayor incertidumbre. Vale volver a aclarar, que puede darse el caso de un evento de encuentro entre dos satélites operativos, pero este investigador no lo trata de esa manera en su estudio. Entonces, la propuesta de estos científicos es predecir la incertidumbre posicional del perseguidor en el TCA. Estimar las varianzas posicionales en el sistema de referencia RTN: ${\sigma_{\mathrm{R}}}^2$, ${\sigma_{\mathrm{T}}}^2$, ${\sigma_{\mathrm{N}}}^2$ (Figura 1 del Anexo A); y comparar la performance de los algoritmos de Bosques Aleatorios de Regresión y de Redes Neuronales Recurrentes, en especial la técnica *Long Short Term Memory* LSTM. Para terminar, utilizan las covarianzas predichas como parámetro de entrada del algoritmo de Akella para computar la probabilidad de colisión (Akella y Alfriend, 2000). Se describe brevemente el sistema RTN en el Anexo A, (Vallado y McClain, 2013).

En este contexto, Acciarini (Acciarini y col., 2021) presentan un paquete de código abierto denominado *Kessler* que se desarrolló especialmente para el estudio de las posibles colisiones entre satélites (Acciarini y col., 2021). La librería incluye modelos de inferencia bayesiana y programación probabilística. En cuanto al módulo de aprendizaje automático, se observa que está alineado con el trabajo mencionado anteriormente, ya que también utiliza la técnica de LSTM. La ventaja de utilizar la inferencia bayesiana con las redes neuronales, es que permite diseñar modelos que en vez de tener una estimación puntual, generan como resultado una distribución de probabilidad y esto permite medir la incertidumbre de estas predicciones a través de la varianza. Además el paquete incorpora características adicionales para importar o exportar CDM, y otras opciones de visualización del evento a lo largo del tiempo.

Cabe destacar que muchas de estas investigaciones están patrocinadas y son de interés actual para la Agencia Espacial Europea, universidades y organismos internacionales del ámbito espacial.

Sección 3

Definición del problema

Por lo general las maniobras de propulsión se definen horas antes del TCA con la información del último CDM emitido hasta ese momento y en función de un protocolo definido por la empresa operadora del satélite. El principal problema es que al momento de autorizar la maniobra éste puede tener cierta antigüedad por lo que suele esperarse uno nuevo con parámetros con menor incertidumbre. Sin embargo, en ocasiones, ya no es posible seguir dilatando la decisión sólo por aguardar un nuevo CDM. Por estos motivos, es conveniente estudiar alternativas para reducir la incertidumbre en el cálculo de la probabilidad de colisión. Con el objeto de tomar decisiones a tiempo, mejor planificadas y lograr independizar el protocolo de maniobras evasivas de colisión de la espera de un nuevo CDM en las horas más cercanas al TCA.

Sección 4

Justificación del estudio

Esta investigación se realiza debido al gran desarrollo de la industria espacial privada conocido actualmente con el término New Space. El espacio se hace accesible a empresas privadas debido a la reducción en costos de lanzamiento. Hoy en día, el modelo de negocios está orientado a ubicar constelaciones de satélites en la región conocida como LEO (Low Earth Orbit), que comprende el rango 200 km - 1000 km, (ESA, 2020). Entonces, se está produciendo una transformación del negocio espacial, antes se buscaba ubicar un sólo satélite de alto costo en una órbita de gran altitud, como por ejemplo, los geoestacionarios a 35000 km de altura aproximadamente. Por el contrario, ahora se utiliza grupos de naves espaciales de menores dimensiones y bajo costo de fabricación, en órbitas más bajas, se ubican en un mismo plano orbital pero con cierto ángulo de desfasaje entre cada de ellos de manera de garantizar la cobertura del área de interés. Para mayores detalles sobre Dinámica Órbital se recomienda el libro de Vallado y McClain (Vallado y McClain, 2013). Las constelaciones de satélites se puede pensar como si se viera en el cielo un tren de satélites, justamente es el modelo de negocio implementado por la empresa Starlink (Starlink, 2022) para sus satélites de comunicaciones. Debido a este incremento de interesados en la industria espacial, es importante considerar la seguridad de los propios satélites, que son activos de estas empresas y también la seguridad de otras misiones espaciales. La Agencia Espacial Europea tiene una oficina de chatarra espacial (Space Debris Office) especializada en el tema. En los últimos años, ha hecho foco en automatizar las maniobras propulsivas para mitigar el riesgo de la colisión. Por estas razones tener algoritmos válidos para realizar este tipo de predicciones es de interés para las agencias espaciales y las empresas privadas. Todo esto permitiría reducir la carga laboral de los operadores y optimizar los protocolos existentes para mitigar el riesgo de colisión.

Sección 5

Alcance del trabajo y limitaciones

El estudio que se presenta aborda la predicción de la probabilidad de colisión. Se centra en estudiar los algoritmos de aprendizaje automático que se encuentran en el estado del arte de esta área y el alcance del mismo está restringido a:

- Optimización de hiperparámetros.
- Creación de nuevas variables que aporten valor al algoritmo. (Feature Engineering).
- En caso de no encontrar una oportunidad de mejora, proponer otro tipo de algoritmo de aprendizaje automático.

Está totalmente fuera del alcance de este trabajo calcular o proyectar la maniobra disuasiva necesaria para cada evento de conjunción en particular.

Sección 6

Hipótesis

Se podría ajustar un modelo de regresión para predecir la probabilidad de colisión del próximo reporte de evento de conjunción (Conjunction Data Message).

Sección 7

Objetivos

- **General**
  - Desarrollar un modelo de regresión que permita predecir la probabilidad de colisión del próximo reporte de encuentro (CDM).
- **Específicos**
  - Proponer un modelo de regresión lineal múltiple
  - Desarrollar un modelo no lineal utilizando el algoritmo ADAboost (Adaptative Boosting)
  - Desarrollar un modelo no lineal utilizando el algoritmo Light GBM (Light Gradient Boosting Machines)
  - Validar los modelos.
  - Comparar modelos de regresión.
  - Seleccionar el modelo que se implementaría en un entorno productivo.
  - Integrar el programa para que ingresados los CDM se obtenga el resultado directamente.

Sección 8

Metodología

1. Técnicas

Para llevar adelante este proyecto se utilizará una metodología similar a la propuesta por el modelo CRISP utilizado normalmente en proyectos de ciencia de datos (Wijaya, 2021). Esta técnica incluye las siguientes etapas:

- Comprensión de la problemática
- Análisis de los datos
- Preparación de los datos
- Modelado
- Evaluación
- Despliegue o puesta en producción

En primer lugar se debe comprender el contexto en el que existe esta problemática desde el marco de la seguridad operacional espacial (space safety) y las maniobras disuasivas de colisión (automatic collision avoidance maneuvers). En cuanto al análisis de los datos, la recolección inicial es directa ya que se obtiene de datos publicados por la ESA anonimizados. Se hará hincapié en las etapas de descripción, exploración y verificación de la calidad. Luego se llevará adelante la fase de preparación de datos y se evaluará si es conveniente o necesario incorporar una nueva fuente de datos.

El foco de esta investigación está puesto en el modelado y evaluación. Es aquí donde se intentará incorporar valor agregado y en torno al cual se desarrollará el trabajo. Finalmente se armará una herramienta para que el modelo diseñado pueda ser implementado por otros usuarios o investigadores en el futuro.

2. Herramientas

En este proyecto se utilizará *Python* (Python, 2021) como lenguaje de programación y fundamentalmente herramientas *open-source*. Se usarán los paquetes y librerías siguientes:

- Pandas (McKinney, 2010)
- Numpy (Harris y col., 2020)
- Sci-kit learn (Pedregosa y col., 2011)
- LightGBM (LightGBM, 2021)
- Bayesian-optimization (Python, 2021)
- Jupyter (Jupyter, 2021)
- matplotlib: visualización (Hunter, 2007)
- Plotly: visualización (Plotly, 2021)
- Bokeh: visualización (Bokeh Development Team, 2018)
- Seaborn: visualización (Seaborn, 2021)
- Black: formato de código en python (Black, 2021)

3. Definiciones

3.1. Modelo Lineal

Aquel modelo que puede estimar la variable a predecir Y a través de una combinaciones lineal de las variables predictoras X_i. Una combinación lineal es simplemente la sumatoria de las variables predictoras pudiendo cada una de ellas estar multiplicada por un factor o no. (Hastie y Tibshirani, 2017)

3.2. Modelo de Regresión Lineal Simple

$$Y = \beta_0 + \sum_{j=1}^{p} X_j \beta_j \tag{8.1}$$

Siendo:

β_i coeficientes de ajuste del modelo (pendiente)

β_0 término independiente

X_i variables de entrada del modelo

Y variable dependiente u objetivo

En los modelos de Regresión Lineal Simple o múltiple se busca calcular los coeficientes β_i que ajustan el modelo minimizando algúna función de optimización. Por lo general la más utilizada es la sumatoria de los residuos al cuadrado RSS. (Hastie y Tibshirani, 2017)

$$RSS = \sum_{i=1}^{n} (y_i - \widehat{y})^2 \tag{8.2}$$

$$RSS = \sum_{i=1}^{n} \left(y_i - \beta_i - \sum_{j=1}^{p} x_{ij} \beta_j \right)^2 \tag{8.3}$$

3.3. Modelo No Lineal

Aquel modelo cuya relación entre las variables predictora y a predecir no puede ser representada a través de una relación lineal.

3.4. Modelos de Regresión

Aquel modelo donde la variable a predecir es un número real.

3.5. Modelos de Clasificación

Aquel modelo donde la variable a predecir es una clase o categoría.

3.6. Aprendizaje Supervisado

Técnica de Aprendizaje Automático donde la variable a predecir es conocida para cada una de las observaciones en el conjunto entrenamiento.

3.7. Aprendizaje No Supervisado

Técnica de Aprendizaje Automático donde no existe la variable a predecir.

3.8. Árboles de Decisión

Es un modelo no lineal de aprendizaje supervisado que consiste en dividir el espacio de las variables predictoras X_i en diferentes regiones R_j utilizando reglas para esta división. Los Árboles de Decisión se pueden aplicar tanto a modelos de regresión como de clasificación. Por la manera en la que se van realizando las divisiones de los registros del conjunto de datos, es que este modelo puede visualizarse y su forma es semejante a la de un árbol con sus ramificaciones y hojas. Conceptualmente, se parte del conjunto de datos completo (nodo inicial), se elige una variable para hacer la división y se utiliza una regla o criterio diferente. De esta manera, pasamos a tener dos nuevos nodos los que se pueden seguir dividiendo sucesivamente hasta llegar a los nodos terminales u hojas. Los nodos terminales u hojas son aquellos que no sufrieron una nueva división y se corresponden con las regiones R_j del espacio de variables predictoras. Es importante destacar que estas regiones no pueden solaparse entre si. (James y col., 2013)

A continuación se explican las diferencias básicas entre un árbol de regresión y clasificación.

3.8.1. Árboles de Decisión para Regresión

En los árboles de regresión, se destaca:

- Dividir el espacio de variables predictoras X_i en J regiones no solapadas R_j.
- Cada una de las regiones R_j tienen dentro de ellas un conjunto de observaciones de las variables predictoras X_i y la variable a predecir Y_i. El valor que toma esa región es simplemente de la media de todas las variables a predecir que caen dentro de esta región.

Cabe desarrollar el primer ítem, las divisiones para generar cada una de las regiones tienen como objetivo minimizar la suma del cuadrado de los residuos RSS (residual sum of squares).

$$RSS = \sum_{j=1}^{J} \sum_{i \epsilon R_j}^{n} \left(y_i - \widehat{y}_{R_j} \right)^2 \tag{8.4}$$

Para lograr esto se utiliza un mecanismo llamado división binaria recursiva, básicamente consiste en dividir el conjunto de datos desde arriba hacia abajo. En la cima del árbol el

espacio de variables predictoras está en una misma región y luego con las sucesivas divisiones se va alcanzando en los nodos terminales las regiones finales. Este mecanismo considera la división que minimiza el RSS en el nodo que se está dividiendo en esa instancia, es decir, no es un algoritmo que considere cuál es la mejor división en la instancia actual para optimizar la minimización del RSS en divisiones futuras, por tal motivo se lo suele llamar en la bibliografía algoritmo avaro (greedy algorithm) (James y col., 2013).

3.8.2. Árboles de Decisión para Clasificación

La diferencia principal es que en los árboles de clasificación es que el espacio de variables predictoras también queda divido en J regiones R_i, pero en este caso se predice una clase. Esta clase vendrá dada por la correspondiente a la mayoría de las observaciones que se encuentren en ese nodo terminal; es decir, si en dicha región hay 10 observaciones de la clase A y 2 de la clase B, esta región predecirá las futuras observaciones que caigan aquí serán de la clase A.

Para realizar las divisiones en modelos de clasificación no se puede recurrir al RSS por lo tanto se suele utilizar el índice de Gini y la entropía. No se entrará en posteriores detalles porque este trabajo está orientado a modelos de regresión.

3.9. Aprendizaje Conjunto (Ensemble learning)

Un modelo predictivo conformado por un solo Árbol de Decisión por lo general podría tener menor exactitud en las predicciones que otros modelos predictivos. Además la forma final de un Árbol de Decisión suele verse afectada por pequeños cambios en el conjunto de datos. Sin embargo, cuando en vez de utilizar un solo Árbol de Decisión, se realiza un Aprendizaje Conjunto de muchos Árboles de Decisión, la exactitud en las predicciones mejora en gran medida (James y col., 2013).

El Aprendizaje Conjunto (Ensemble Learning) es una técnica estadística en la que se construye un modelo predictivo combinando un conjunto de modelos predictivos más simples (Hastie y Tibshirani, 2017). En el caso particular de Árboles de Decisión esto se aplica a través de dos métodos principalmente:

- Bagging
- Boosting

3.9.1. Bagging

Bagging o Agregación de Bootstrap es un método estadístico que busca reducir la varianza de los modelos de aprendizaje estadístico. Los Árboles de Decisión tienen gran varianza. Esto se refleja en que si tomamos dos partes diferentes del mismo conjunto de datos y ajustamos un Árbol de Decisión a cada una, los resultados que se obtienen pueden ser muy diferentes. La Agregación de Bootstrap se basa en el fundamento estadístico de que si promediamos un

conjunto de n observaciones Z_1,...,Z_n cada una con una varianza σ^2, la varianza de la media $\overline{Z}$ será σ^2/n, esto es, promediando un conjunto de observaciones, disminuye la varianza (James y col., 2013).

Llevado esto al campo de los Árboles de Decisión constaría de los siguientes pasos:

- Extraer del conjunto de datos de entrenamiento, una cantidad n de subconjuntos de datos diferentes.
- Para cada uno de estos n subconjuntos entrenar un Árbol de Decisión
- El modelo de predicción final consta del Aprendizaje Conjunto (Ensemble Learning) de los n Árboles de Decisión desarrollados.
- Al momento de realizar la predicción en el caso de regresión se toma el promedio del resultado de los n árboles y en el caso de clasificación se vota por la clase que ocurre más veces en el conjunto de árboles.

Matemáticamente, el método Bagging y la función de aprendizaje final del ensamble puede expresarse:

$$f_{bag}(x) = \frac{1}{B}\sum_{b=1}^{B} f^{*b}(x) \tag{8.5}$$

Siendo B la cantidad total de Árboles de Decisión del modelo y $f^{*b}(x)$ cada una de las funciones que representa el modelo de Árbol de Decisión individual del conjunto de datos al que se le realizó el bootstrap o bagging.

En la Figura 8.1 puede verse de manera visual el flujo de trabajo de la técnica Bagging. En especial, observar la división del conjunto de entrenamiento en subconjuntos y la generación de un algoritmo de aprendizaje para cada uno de ellos, para luego tomar una decisión (López, 2021).

3.9.1.1. Bosques Aleatorios (Random Forest) El algoritmo de Bosques Aleatorios propone utilizar árboles descorrelacionados, introduciendo un leve cambio al momento de dividir los nodos en el método de Bagging. En Bagging cada nodo puede dividirse por cualquiera de las variables predictoras, esto puede hacer que la mayoría de los árboles comiencen sus divisiones por las variables más importantes del modelo. Por el contrario, en el algoritmo de Bosques Aleatorios la división en cada uno de los nodos puede realizarse solamente a través de un subconjunto de variables predictoras. Esto significa que dentro de este subconjunto no neceseriamente se encuentran las variables de mayor importancia para el modelo, y en consecuencia permite generar un modelo de aprendizaje en conjunto con árboles más descorrelacionados entre sí. Por lo tanto, Bosques Aleatorios debería tener resultados con menor varianza que los propuestos a través del Bagging.

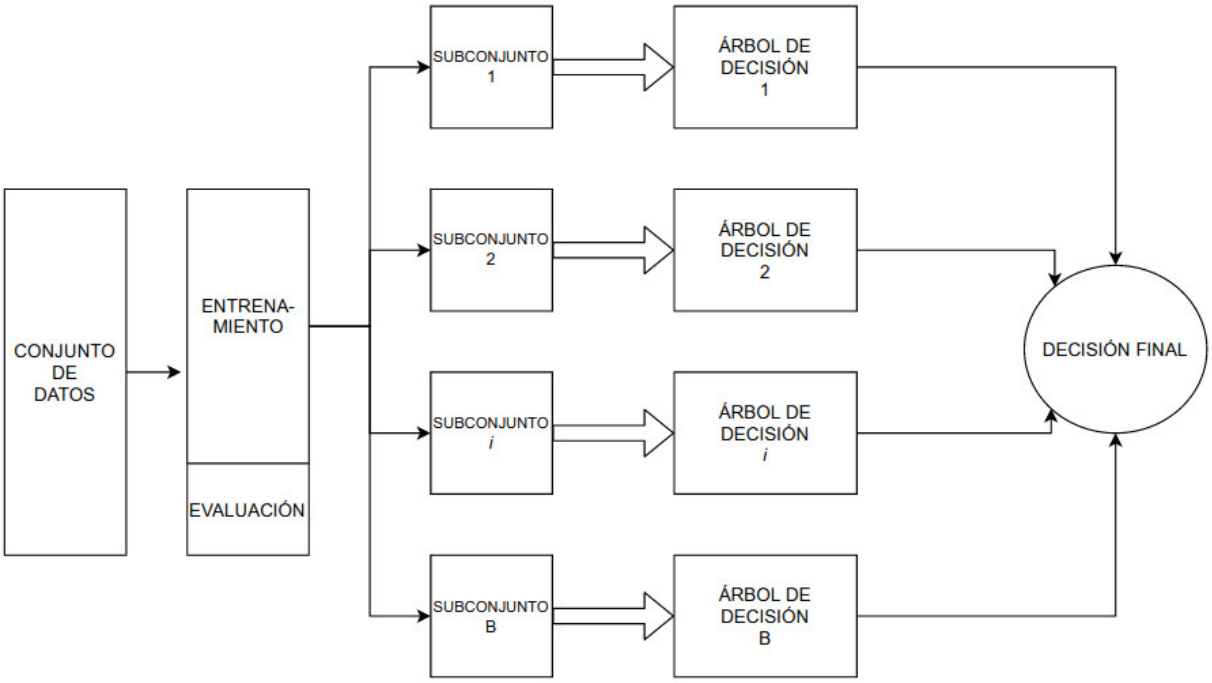

Figura 8.1: *Diagrama de método de bagging (López, 2021)*

3.9.2. Boosting

La diferencia principal entre Bagging y Boosting es que en el primero se crean árboles en paralelo a partir de subconjuntos de datos de entrenamiento diferentes, en cambio en boosting se generan Árboles de Decisión de modo secuencial. Es decir el modelo tiene un crecimiento secuencial, en el que cada nuevo árbol que se genera, está basado en los árboles desarrollados previamente. Boosting no implica muestrear el conjunto de datos haciendo Bootstrap, sino que cada nuevo árbol es generado siempre con el mismo conjunto de datos original.

Los pasos que se siguen en un algoritmo de Boosting son los siguientes:

- A partir de un conjunto de datos se crea un modelo de Árbol de Decisión.
- Se calculan los residuos del árbol anterior y el nuevo árbol se ajusta en función de los errores del anterior, esta es la clave en Boosting.
- Luego con este nuevo árbol se vuelven a calcular o actualizar los residuos y así sucesivamente.

Matemáticamente el modelo de Boosting puede expesarse:

$$f_{boost}(x) = \sum_{b=1}^{B} \lambda f^{boost}(x) \tag{8.6}$$

La función de predicción final de boosting $f_{boost}(x)$ implica una sumatoria de los modelos de cada árbol individual, el parámetro λ es justamente el que controla la tasa de crecimiento del árbol.

La diferencia entre los algoritmos de Boosting radica en la manera en la que minimizan el error en cada nueva iteración. En algunos casos se busca minimizar los residuos a través de alguna función y en otros se penaliza los errores a través de la asignación de pesos haciendo una ponderación (López, 2021). En este trabajo se analizará dos técnicas de boosting AdaBoost (Adaptative Boost) y LGBM (Light Gradient Boosting Machines).

3.9.2.1. Adaptative Boosting ADAboost Es una técnica de Boosting en la que se aplica secuencialmente una ponderación a cada registro de las variables de entrada. Esto implica que aquellos registros que fueron mal clasificados reciban un peso mayor que aquellos que estuvieron bien clasificados, de tal modo que en la próxima iteración en la que se genera un nuevo Árbol de Decisión el algoritmo le de más importancia a clasificar éstos correctamente. Cabe destacar que en el primer Árbol de Decisión todos los registros del conjunto reciben el mismo peso, y recién en el segundo árbol se actualizan los pesos. En la Figura 8.2 se puede ver un diagrama representando los pasos de este algoritmo. Recordar que esto es una técnica de Boosting por lo que el crecimiento de árboles es secuencial y cada uno nuevo que se construye está basado en el anterior. El diagrama es una versión modificada del original presentado por López (López, 2021).

En el párrafo anterior, se dijo que los registros mal clasificados reciben mayor peso en la próxima iteración, esto es válido tanto para modelos de clasificación como de regresión. Ahora bien, para aplicar esto en modelos de regresión y poder determinar si un registro u observación estuvo bien o mal clasificada se calcula el error porcentual absoluto y se toma la decisión a través de un valor umbral. El error porcentual absoluto es simplemente el porcentaje de error que existe entre la predicción y el valor real. Entonces, si un registro tiene determinado valor de error porcentual relativo y éste está por debajo del valor umbral utilizado, entonces está bien clasificado. De esta manera el algoritmo de ADAboost para regresión puede aplicar los pesos ponderados al conjunto de datos para la próxima iteración, tal como lo hace el algoritmo ADAboost para regresión (Kummer y Najjaran, 2014).

3.9.2.2. Light Gradient Boosting Machines LGBM En este algoritmo no se implementa un vector de pesos para ponderar los registros del conjunto de datos. En el método de Boosting a través de Gradiente (Gradient Boosting) lo que se busca es minimizar los residuos de cada árbol del modelo a través del cálculo del gradiente de la función de optimización (loss function). El gradiente conceptualmente indica la dirección en la que la tasa de cambio de la función de optimización es mayor, por tal motivo indica la dirección en la que debe ser generado el próximo árbol para que los residuos del próximo sean menores. Este proceso se repite hasta que los residuos de la función de optimización son nulos o hasta alcanzar determinada cantidad

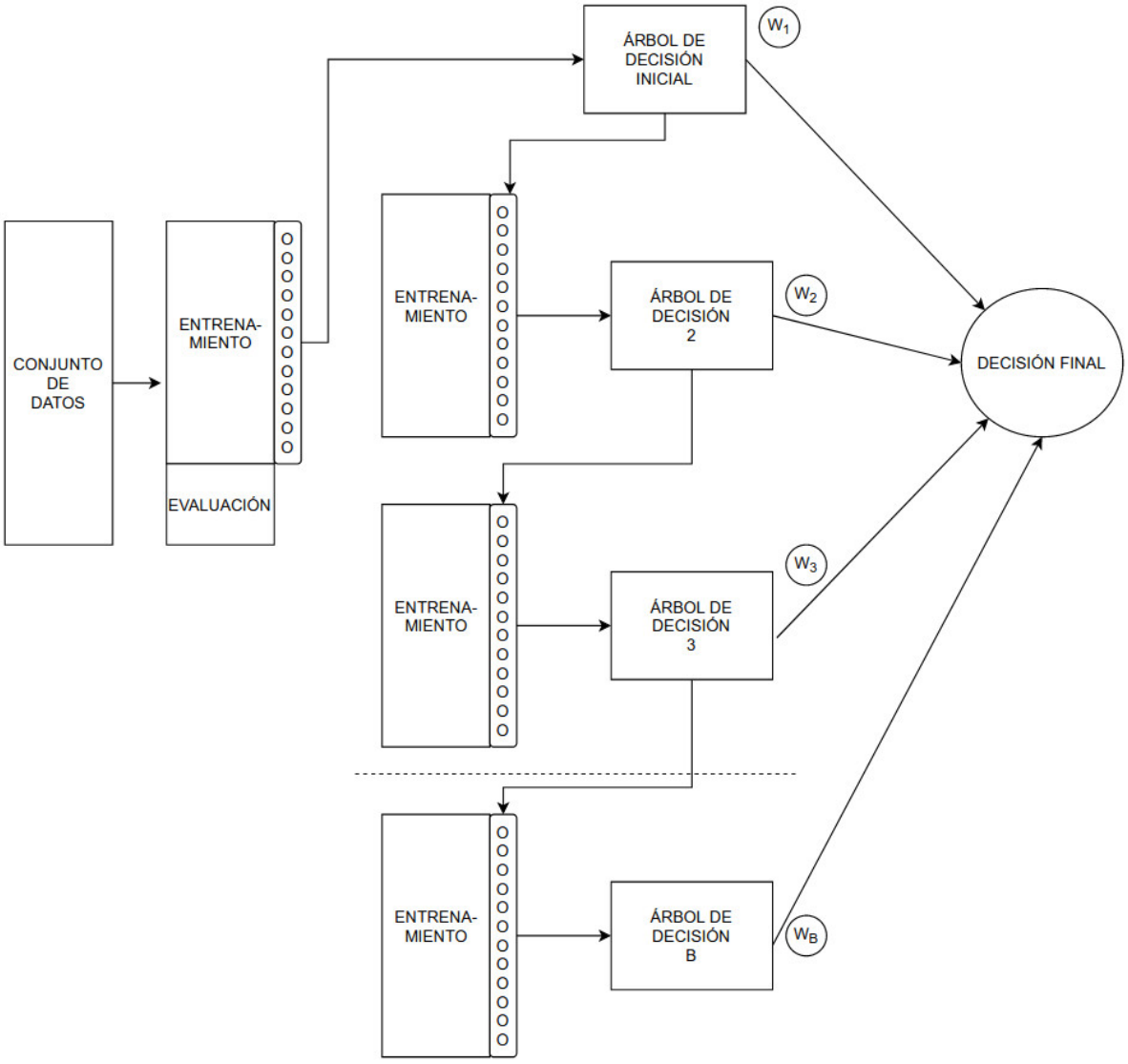

Figura 8.2: *Diagrama de algoritmo ADAboost (López, 2021)*

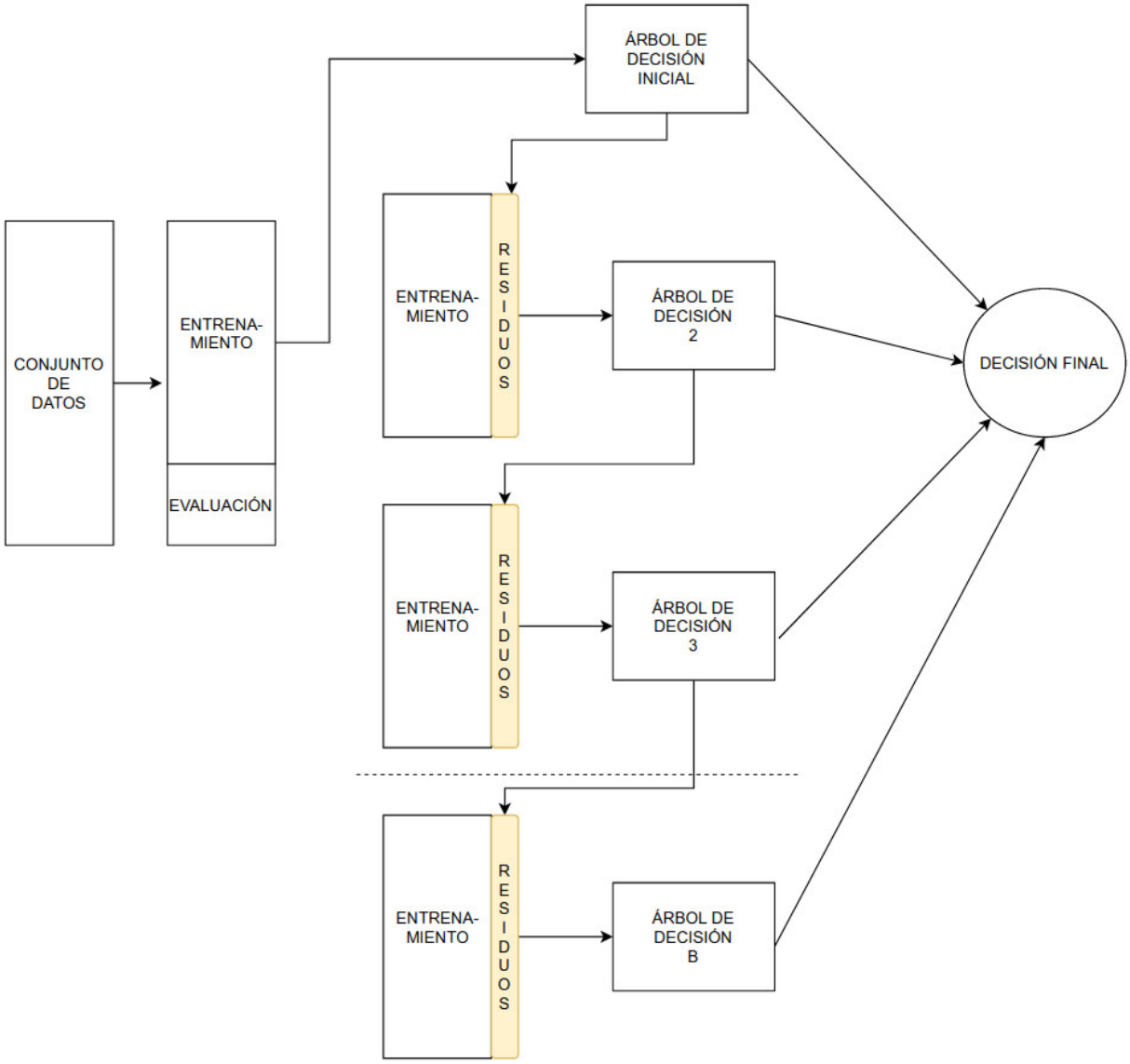

Figura 8.3: *Diagrama de algoritmo Light GBM (López, 2021)*

de árboles definida arbitrariamente. En la Figura 8.3 se puede ver un diagrama esquemático del algoritmo LGBM que intenta poner en evidencia el tratamiento de los residuos en este algoritmo. Esta figura también está basada en el presentado por López (López, 2021).

Sección 9

Experimentación

1. Conjunto de datos de entrada

El conjunto de datos es público y provisto por la Agencia Espacial Europea (ESA). Estos datos son accesibles al público en general ya que la ESA los pone a disposición en el marco de competiciones abiertas para estudiantes e investigadores a nivel mundial. En particular, este set de datos se encuentra en la página web Kelvin destinada a estos concursos en la sección "Collision Avoidance Challenge" (ESA, 2019).

El conjunto de datos de la ESA tiene 162634 registros y 103 atributos. Sin embargo, este conjunto de datos no se usará directamente para generar el modelo. Esto se debe a que la ESA publicó el dataset en el marco de un concurso o competición de Machine Learning modificando el formato de ciertos atributos o brindando valores precalculados por ellos. Uno de los subobjetivos del trabajo de investigación es generar un algoritmo que pueda utilizar datos de entrada en formato estándar. Por tal motivo como primer paso inicial y con la idea de simular una situación real en la que los reportes CDM vienen en formato estándar, este conjunto de datos de la ESA será transformado de acuerdo al estándar aceptado internacionalmente *Recommended Standard Conjunction Data Message* del Instituto CCSDS (CCSDS, 2013). Esto implica que el conjunto de datos de entrada tendrá menos columnas, ya que sólo se utilizarán las disponibles en un reporte estándar y ninguna con datos precalculados; y además algunas de ellas cambiarán la nomeclantura por la indicada en el estándar.

Entonces, luego de esta primera transformación inicial pasamos a tener la misma cantidad de registros pero 83 columnas. Los atributos que no se encuentran en un reporte de CDM y por lo tanto son eliminados, son los siguientes:

- *mission_id*
- *max_risk_estimate*

- *max_risk_scaling*
- *geocentric_latitude*
- *azimuth*
- *elevation*
- *F10*
- *AP*
- *F3M*
- *SSN*
- *x_position_covariance_det**
- *x_ecc**
- *x_j2k_sma**
- *x_span**
- *x_rcs_estimate**

Los campos marcados con un asterisco (*) indican que existen dos columnas para dicha variable una para cada uno de los objetos. De esta manera, en total se omiten 20 variables las cuales no están presentes en un reporte de CDM estándar.

Es menester mencionar que en el conjunto de datos de la ESA se presentan los coeficientes de correlación de la posición y velocidad del objeto ya precalculados. Esto también debió considerarse en el algoritmo ya que los CDM en formato estándar presentan los valores como covarianzas. Así que también se realizó dicha transformación, pero ésta no introduce modificaciones en la cantidad de atributos.

Por último, una de las transformaciones más importantes fue la variable *risk* que la Agencia Espacial Europea (ESA) la presenta en escala logarítmica. En este trabajo se ha transformado a número decimal que es la manera en la que vienen expresados los CDM.

2. Análisis de los datos

2.1. Análisis de las variables del CDM (Conjunction Data Messages)

De los parámetros incluidos en cada CDM se extraen las variables de este estudio. Las primeras a destacar son las que expresan directamente el riesgo de colisión:

- Riesgo o probabilidad de colisión *PC*

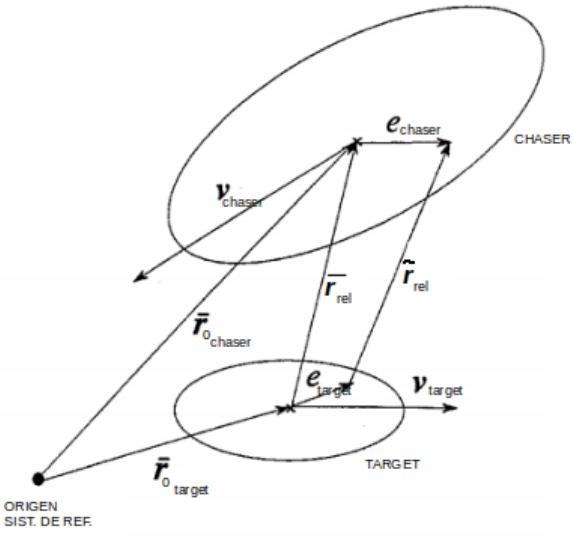

Figura 9.1: *Posición relativa entre el cuerpo objetivo y perseguidor.(Akella y Alfriend, 2000)*

- Tiempo en días remanente para el instante en que los cuerpos se encuentran más próximos entre sí (TCA).
- Identificador único del evento de colisión

Luego se tiene aquellas relacionadas con la dinámica orbital. La trayectoria de estos objetos se modela utilizando un vector posición $\overline{r}$ y un vector velocidad $\overline{v}$. Se suele unificar ambos vectores en uno solo conocido como vector de estado $\overline{y}$, expresado en un sistema de referencia geocéntrico X, Y, Z, sería:

$$\overline{y} = \begin{bmatrix} r_X \\ r_Y \\ r_Z \\ v_X \\ v_Y \\ v_Z \end{bmatrix} \tag{9.1}$$

Ambos cuerpos tanto el objetivo como el perseguidor tienen un vector de estado asociado. Es de particular interés la posición y velocidad relativa entre ambos en el TCA. La posición y velocidad relativa es simplemente la diferencia entre los vectores. Esto se puede apreciar en la Figura 9.1 que es una adaptación del presentado por Akella en su publicación donde se modificaron los subíndices para ser consistente con la nomenclatura utilizada en este trabajo (Akella y Alfriend, 2000).

En el conjunto de datos los parámetros están presentados en el sistema de referencia centrado en el satélite, conocido como R, T, N, para lo cual es necesario transformar el vector anterior.

Luego, se tiene tres componentes del vector posición relativa (en inglés, vector miss distance) y otras tres componentes de velocidad relativa entre ambos cuerpos en el TCA, junto al módulo de estos vectores.

$$\overline{r_{rel}} = \begin{bmatrix} r_r \\ r_t \\ r_n \end{bmatrix}_t - \begin{bmatrix} r_r \\ r_t \\ r_n \end{bmatrix}_c \tag{9.2}$$

$$\overline{v_{rel}} = \begin{bmatrix} v_r \\ v_t \\ v_n \end{bmatrix}_t - \begin{bmatrix} v_r \\ v_t \\ v_n \end{bmatrix}_c \tag{9.3}$$

Se agregan seis variables más por cada objeto que son las componentes de la diagonal de la matriz de covarianzas, es decir, las varianzas. A continuación se presenta el vector de estado genérico válido tanto para cuerpo objetivo o perseguidor y la matriz de covarianzas.

$$\overline{y} = \begin{bmatrix} r_r \\ r_t \\ r_n \\ v_r \\ v_t \\ v_n \end{bmatrix} \tag{9.4}$$

$$\overline{\overline{COV}} = \begin{bmatrix} {\sigma_{r_r}}^2 & & & & & \\ \sigma_{r_t r_r} & {\sigma_{r_t}}^2 & & & & \\ \sigma_{r_n r_r} & \sigma_{r_n r_t} & {\sigma_{r_n}}^2 & & & \\ \sigma_{v_r r_r} & \sigma_{v_r r_t} & \sigma_{v_r r_n} & {\sigma_{v_r}}^2 & & \\ \sigma_{v_t r_r} & \sigma_{v_t r_t} & \sigma_{v_t r_n} & \sigma_{v_t v_r} & {\sigma_{v_t}}^2 & \\ \sigma_{v_n r_r} & \sigma_{v_n r_t} & \sigma_{v_n r_n} & \sigma_{v_n v_r} & \sigma_{v_n v_r} & {\sigma_{v_n}}^2 \end{bmatrix} \tag{9.5}$$

El CDM incluye la matriz de covarianzas que es simétrica. En el modelo se utilizarán los coeficientes de correlación ρ calculados a partir de los elementos de dicha matriz. Dado que es una matriz simétrica se tiene 15 nuevas variables por cada objeto para nuestro caso de estudio.

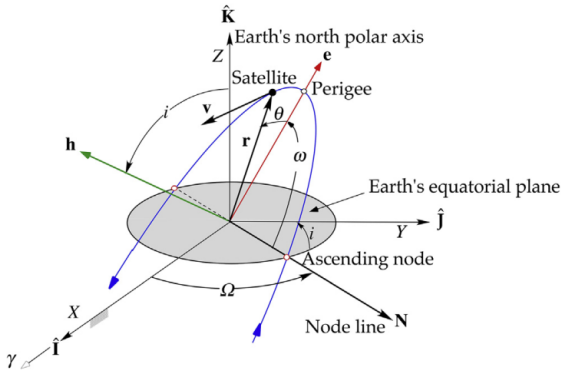

Figura 9.2: *Sistema de referencia centrado en la Tierra y con los parámetros que determinan la órbita.(Curtis, 2014)*

$$\overline{\overline{CORR}} = \begin{bmatrix} 1 & & & & \\ \rho_{r_t r_r} & 1 & & & \\ \rho_{r_n r_r} & \rho_{r_n r_t} & 1 & & \\ \rho_{v_r r_r} & \rho_{v_r r_t} & \rho_{v_r r_n} & 1 & \\ \rho_{v_t r_r} & \rho_{v_t r_t} & \rho_{v_t r_n} & \rho_{v_t v_r} & 1 \\ \rho_{v_n r_r} & \rho_{v_n r_t} & \rho_{v_n r_n} & \rho_{v_n v_r} & \rho_{v_n v_r} \quad 1 \end{bmatrix} \tag{9.6}$$

Entre probabilidad de colisión, tiempo al TCA, vectores de estado, varianzas y coeficientes de correlación suman hasta aquí 53 variables entre ambos cuerpos. Adicionalmente se incluyen estos parámetros por cada uno de los objetos:

- Asociado a la órbita, ver Figura 9.2 (Curtis, 2014).
  - Apogeo
  - Perigeo
  - Inclinación i
  - Energía asociada
- Coeficiente balístico
- Coeficiente de radiación solar
- Variables relacionadas con la cantidad de observaciones y propagación de la órbita.

2.2. Variables más destacadas del CDM: PC, MD, tiempo al TCA

Para el análisis de eventos de colisión se destacan tres variables:

- $COLLISION\_PROBABILITY$ (PC): probabilidad de colisión
- $MISS\_DISTANCE(MD)$: distancia entre ambos cuerpos en el TCA
- $time\_to\_tca$: tiempo en días al TCA

La primera de ellas determina si es necesario realizar la maniobra, de acuerdo al siguiente criterio:

$$\begin{cases} PC >= 10^{-4} & => \text{Área de Maniobra} \\ 10^{-4} < PC <= 10^{-6} & => \text{Área de Monitoreo o Alerta} \\ PC < 10^{-6} & => \text{Fuera de Protocolo} \end{cases} \tag{9.7}$$

El criterio para definir el área de maniobra en este trabajo de investigación se presenta en la Ecuación 9.7 y es el utilizado por Merz junto a otros investigadores de la Agencia Espacial Europea, publicado en la 7^{ma} Conferencia Europea de Chatarra Espacial llevada a cabo en Alemania en el 2017, (Merz y col., 2017). En la sección 1.3 de dicha publicación tratan el umbral de riesgo y los criterios de maniobra. Allí indican que utilizando un valor umbral de la probabilidad de colisión de 10^{-4} un día antes del TCA del evento de colisión, el riesgo de impacto se reduce en un 90 %. Además, con fines académicos, se adopta una probabilidad de corte de 10^{-6} como umbral de monitoreo, y se asume que eventos con probabilidad de colisión mayores a ésta quedan fuera del protocolo de maniobra. Este valor de corte será luego utilizado en este misma sección 9.3.5 Filtrado de registros. En la Figura 9.3 se grafica los criterios de ejecución de maniobra junto a la proporción de registros que tiene el conjunto de datos. Se observa que la zona de maniobra y de alerta tiene escasos CDM con respecto a los que se encuentran fuera de protocolo. Para mayor detalle y para cuantificar esta situación se incluye la Tabla 9.1 de aquí se deduce que solo el 2 % de las tuplas corresponden a eventos con probabilidad mayor a 10^{-4}, es decir, aquellos donde efectivamente se tuvo que realizar una maniobra de acuerdo al protocolo de seguridad operacional. Extendiendo la probabilidad hasta la zona de monitoreo (o alerta) aparecen alrededor de 11 mil mensajes de conjunción adicionales para analizar. Entonces, el 13 % del total de los CDM tienen una PC menor a 10^{-6}.

En cuanto $MISS\_DISTANCE$ cada empresa operadora de satélites define un valor umbral para que un CDM entre en área de maniobra o de monitoreo (o alerta). A fines académicos y por simplicidad en este trabajo se analizará el nivel de riesgo de un CDM en función de la probabilidad de colisión. El $time\_to\_tca$ indica el momento del evento de conjunción.

En la Tabla 9.2 se presentan valores estadísticos de estas tres variables, se debe tener en cuenta que esta tabla se realizó sin tener en cuenta los registros con valores nulos. Todo lo referente al procesamiento de los datos se detalla en 9.3.3.1 ETL. Esto permite visualizar que los registros están desbalanceados en cuanto a su probabilidad de colisión. Es decir, los CDM que refieren a eventos de colisión real son escasos en el conjunto de datos, como también puede observarse en el histograma de la variable PC Figura 9.4. Resulta necesario abordar esto para poder generar un modelo que permita predecir la probabilidad correctamente justamente en situaciones en la zona de alarma o monitoreo.

Aclaración sobre la Tabla 9.2, la Figura 9.4, la Figura 9.3 y la Tabla 9.1 en todas ellas los valores de PC corresponden al $\log_{10}(PC)$, esta transformación se explicará en detalle en la sección siguiente, 9.3 Preparación de los datos.

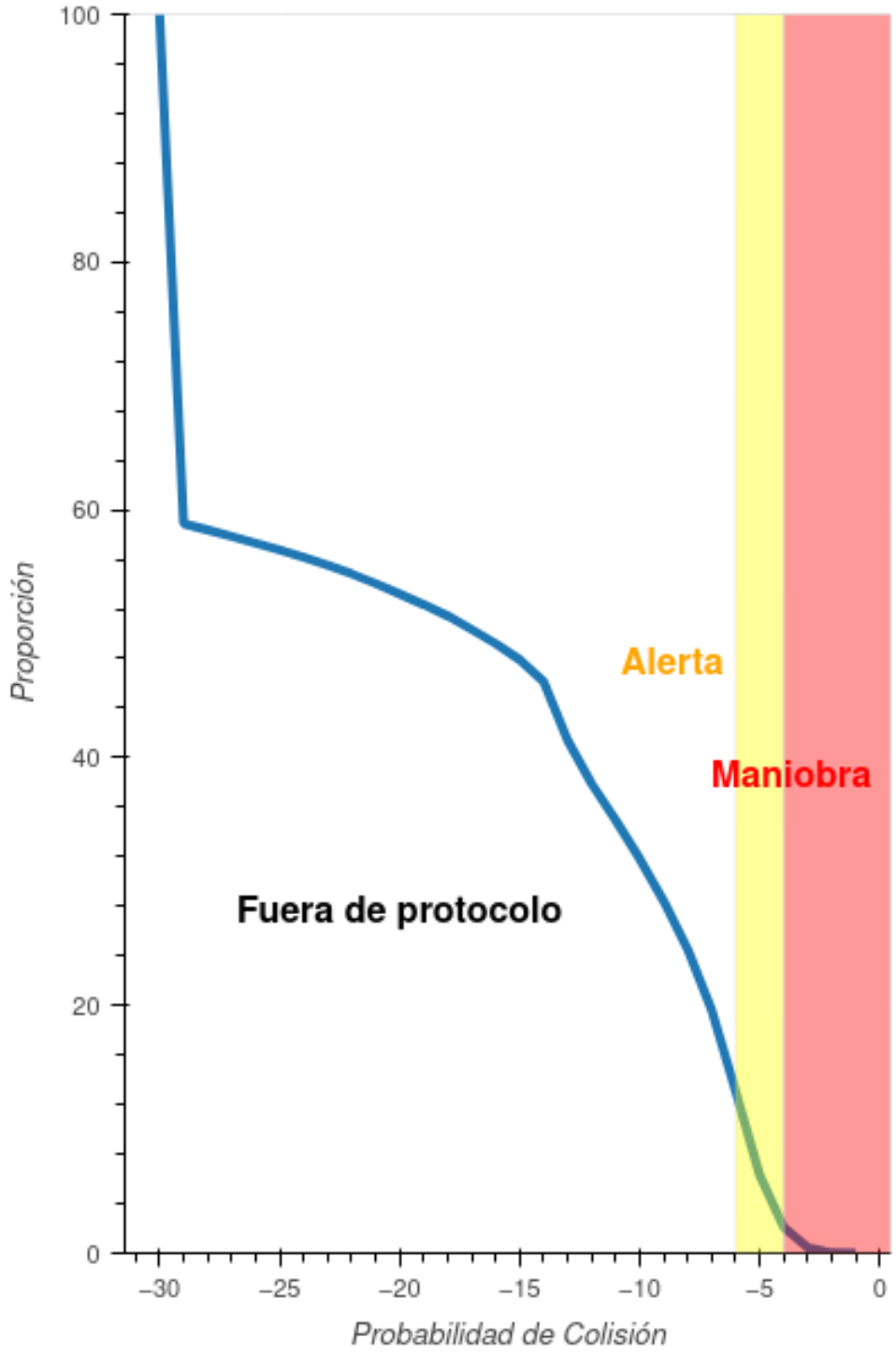

Figura 9.3: *Relación de los CDM según riesgo de la variable COLLISION_PROBABILITY con respecto al total de registros.* $PC > -4 \Rightarrow$ Área de maniobra; $-4 < PC < -6 \Rightarrow$ Área de Monitoreo; $PC < -6 \Rightarrow$ Fuera de protocolo

Tabla 9.1: Valores de probabilidad de colisión y su proporción en el conjunto de datos

Rango de PC	**Cantidad**	**Proporción**	**Acumulado**
$-1 \leq x < 0$	8	0.00005	0.00005
$-2 \leq x < -1$	42	0.00027	0.00033
$-3 \leq x < -2$	647	0.00422	0.00454
$-4 \leq x < -3$	2525	0.01646	0.02100
$-5 \leq x < -4$	6450	0.04205	0.06305
$-6 \leq x < -5$	10380	0.06767	0.13072
$-7 \leq x < -6$	10065	0.06562	0.19634
$-8 \leq x < -7$	7517	0.04900	0.24534
$-9 \leq x < -8$	5971	0.03893	0.28427
$-10 \leq x < -9$	5251	0.03423	0.31850
$-11 \leq x < -10$	4765	0.03106	0.34957
$-12 \leq x < -11$	4497	0.02932	0.37888
$-13 \leq x < -12$	5391	0.03515	0.41403
$-14 \leq x < -13$	7250	0.04726	0.46129
$-15 \leq x < -14$	2712	0.01768	0.47897
$-16 \leq x < -15$	2005	0.01307	0.49204
$-17 \leq x < -16$	1743	0.01136	0.50341
$-18 \leq x < -17$	1683	0.01097	0.51438
$-19 \leq x < -18$	1421	0.00926	0.52364
$-20 \leq x < -19$	1297	0.00846	0.53210
$-21 \leq x < -20$	1285	0.00838	0.54047
$-22 \leq x < -21$	1195	0.00779	0.54826
$-23 \leq x < -22$	1066	0.00695	0.55521
$-24 \leq x < -23$	983	0.00641	0.56162
$-25 \leq x < -24$	927	0.00604	0.56767
$-26 \leq x < -25$	857	0.00559	0.57325
$-27 \leq x < -26$	813	0.00530	0.57855
$-28 \leq x < -27$	802	0.00523	0.58378
$-29 \leq x < -28$	786	0.00512	0.58891
$-30 \leq x < -29$	63059	0.41109	1.00000

Tabla 9.2: Métricas estadísticas descriptivas de las tres principales variables del conjunto de datos

Métrica	**COLLISION_PROBABILITY PC**	**MISS_DISTANCE**	**time_to_tca**
n	153393.00	153393.00	153393.00
$\overline{x}$	-19.28	16209.89	3.36
s	10.02	14047.74	2.01
MIN	-30.00	9.00	-0.15
$Q25$	-30.00	4589.00	1.59
$Q50$	-17.71	12138.00	3.31
$Q75$	-9.11	24768.00	5.09
MAX	-1.44	67373.00	6.99

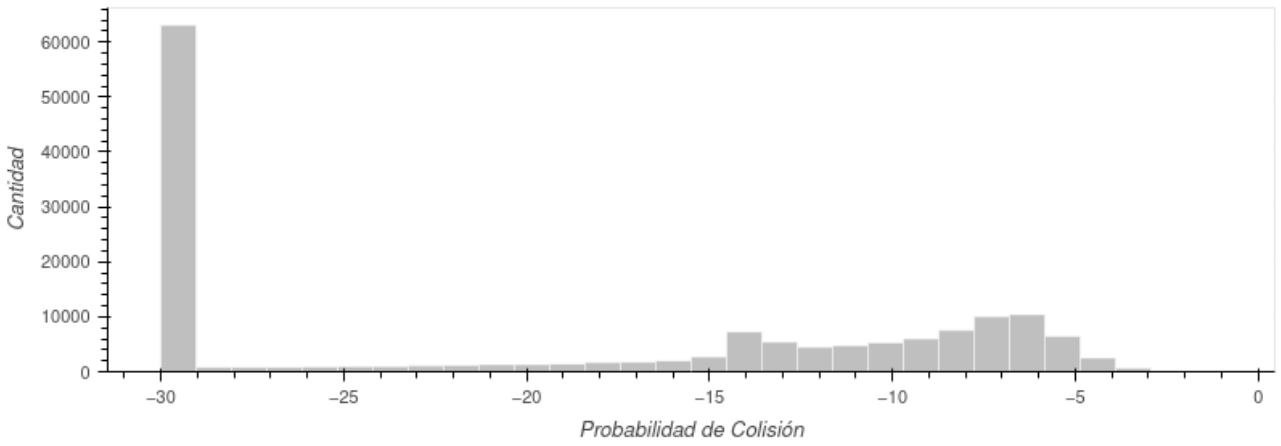

Figura 9.4: Histograma de la variable probabilidad de colisión *COLLISION_PROBABILITY*

3. Preparación de los datos

Esta sección se divide en tres partes: ETL (extracción, transformación y carga) de los datos del CDM propiamente dichos, la ingeniería de variables (feature engineering) para el modelo de aprendizaje automático y el balanceo del conjunto de datos.

3.1. ETL

Para el entrenamiento del modelo se utilizó el conjunto de datos provisto por la ESA mencionado anteriormente en 9.1 Conjunto de datos de entrada. Se ha renombrado las variables en función de los estándares internacionales utilizados en los CDM (CCSDS, 2013). El objetivo es que el código pueda ser adaptado y cualquier usuario del código pueda obtener los reportes CDM a través de una API directamente del ente internacional de monitoreo que utiliza el formato estandarizado. Las acciones tomadas en este proceso se pueden resumir de la siguiente manera:

- Extracción de los datos desde un archivo de texto del dataset público (ESA, 2019).
- Adaptación de los registros de CDM provistos por la ESA al formato estándar internacional como se detalló en este misma sección 9.1 Conjunto de datos de entrada.
    - Se transformó los nombres de las variables a los nombres indicados en el estándar internacional.
    - Se omitió del conjunto de datos aquellas variables precalculadas para el concurso organizado por la ESA.
    - Se omitió variables que no están presentes en los CDM con formato estándar.
    - Se transformó los componentes no diagonales de la matriz de correlación a sus equivalentes covarianzas. Los valores de correlación presentados en los CDM son entre el vector posición y velocidad del objeto en cuestión.
    - Se transformó la variable *risk* en escala logarítmica a número decimal con el nombre *COLLISION_PROBABILITY*.

Hasta aquí todas las transformaciones fueron realizadas para simular un entorno de trabajo real con datos de entrada estándares de manera de que el código resultante del trabajo pueda ser portable y utilizado en otras situaciones más allá del dataset de la ESA. Luego de esta conversión del conjunto de datos al estándar internacional se tiene 162634 registros con 83 variables.

Los ítems siguientes son transformaciones requeridas para generar el Modelo de Machine Learning propiamente dicho.

- Transformación de las variables de fecha hora de tipo texto (string) a variables de fecha.

- Transformación de las variables de tipo fecha a variables de rango de tiempo. Esto significa que se generó la variable *time_to_tca* como la diferencia entre la fecha de creación del CDM y la fecha del evento de colisión (TCA). De modo tal que donde antes teníamos dos variables, se pasó a tener una sola.

- Transformación de covarianzas de la posición y velocidad del objeto a coeficientes de correlación.

- Transformación de las probabilidades de colisión a logaritmo en base 10. Esto significa que una probabilidad de colisión de 10^{-5} en el CDM corresponde directamente con el número entero -5 en el desarrollo de este trabajo. Esta transformación es de vital importancia para poder mejor la construcción y predicción del modelo de predicción.

- Eliminación de atributos no numéricos como el tipo de objeto *OBJECT2_OBJECT_TYPE*.

- Eliminación de las siguientes columnas por presentar datos inconsistentes:

  - *OBJECT1_TIME_LASTOB_START*
  - *OBJECT2_TIME_LASTOB_START*
  - *OBJECT1_TIME_LASTOB_END*
  - *OBJECT2_TIME_LASTOB_END*

- Eliminación de todos los registros donde existía algún valor nulo en al menos uno de sus campos.

- Carga de los datos en un dataframe para generar el modelo de aprendizaje automático posteriormente.

Después de las transformaciones listados en los ítem anteriores se tiene un conjunto de datos con 110337 registros y 77 variables. Esto ocurre porque teníamos 83 variables, pero se eliminaron 5 de ellas por los motivos explicados en el 5^{to} y 6^{to} ítem del último listado. Y de acuerdo a lo dicho en el 2^{do}, donde antes existían dos variables, ahora hay una sola que expresa justamente la diferencia entre ellas.

3.2. Estructura del conjunto de datos

Antes de seguir avanzando con el ETL y la Ingeniería de Variables, resulta conveniente explicar brevemente la estructura del dataset de la ESA. En la página oficial del concurso se puede encontrar una aclaración similar a la que se hace en este apartado (ESA, 2019).

De los 110337 registros que tiene el dataset, se tiene 9229 eventos distintos de colisión. Es decir, en promedio se tiene casi 12 reportes CDM por evento de colisión. Se aclara que ese valor indica la media, pero hay casos en los que habrá menos reportes y otros en los que habrá más. Se intenta ilustrar esto esquemáticamente en la Figura 9.5, esto es, los valores allí mostrados no son los reales del dataset. Aquí se puede ver que registros consecutivos corresponden a un mismo evento de colisión y que la variable *time_to_tca* va decreciendo. En la gráfica se encuentran coloreados del mismo tono las filas (o registros) que corresponden al mismo evento de colisión. El conjunto de datos fue ordenada con dicha estructura, para que sea más fácil identificar cuando se produce un cambio de evento de colisión. Ya que existe una fila correspondiente a un evento de colisión dado cuya consecutiva corresponderá a otro evento de colisión, esto también está contemplado en el algoritmo como se verá en la sección siguiente.

event_id	time_to_tca	PC	MD	...	Y
1	92 horas	PC(1,1)	MD(1,1)	...	Y(1,1)
1	72 horas	PC(1,2)	MD(1,2)	...	Y(1,2)
1	48 horas	PC(1,3)	MD(1,3)	...	Y(1,3)
1	32 horas	PC(1,4)	MD(1,4)	...	Y(1,4)
1	24 horas	PC(1,5)	MD(1,5)	...	Y(1,5)
2	64 horas	PC(2,1)	MD(2,1)	...	Y(2,1)
2	48 horas	PC(2,2)	MD(2,2)	...	Y(2,2)
2	12 horas	PC(2,3)	MD(2,3)	...	Y(2,3)
3	72 horas	PC(3,1)	MD(3,1)	...	Y(3,1)
3	48 horas	PC(3,2)	MD(3,2)	...	Y(3,2)
3	30 horas	PC(3,3)	MD(3,3)	...	Y(3,3)
3	20 horas	PC(3,4)	MD(3,4)	...	Y(3,4)
.	.	.	.	...	.
.	.	.	.	...	.
.	.	.	.	...	.
n	X + ΔX horas	PC(n,1)	MD(n,1)	...	Y(n,1)
n	X horas	PC(n,2)	MD(n,2)	...	Y(n,2)

Figura 9.5: Ilustración esquemática de la estructura del conjunto de datos presentado por la ESA (ESA, 2019)

3.3. Ingeniería de variables

Dado que reportes CDM de un mismo evento se encuentran ordenados de manera consecutiva, para relacionar los reportes de un mismo evento se toma la siguiente estrategia. A partir de $COLLISION\_PROBABILITY$ PC , $MISS\_DISTANCE$ MD y $time\_to\_tca$ se construyen las variables tendencia y gradiente con el objeto de vincular el CDM i-ésimo con los anteriores. El registro i-ésimo se relaciona con su inmediato anterior (i-1) y con tres anteriores (i-3) y se define matemáticamente cada una de las nuevas variables:

$$PC_{tendencia_1} = PC_i - PC_{i-1} \tag{9.8}$$

$$PC_{tendencia_3} = PC_i - PC_{i-3} \tag{9.9}$$

$$PC_{gradiente_1} = \frac{PC_i - PC_{i-1}}{|time\_to\_tca_i - time\_to\_tca_{i-1}|} \tag{9.10}$$

$$PC_{gradiente_3} = \frac{PC_i - PC_{i-1}}{|time\_to\_tca_i - time\_to\_tca_{i-3}|} \tag{9.11}$$

$$MD_{tendencia_1} = MD_i - MD_{i-1} \tag{9.12}$$

$$MD_{tendencia_3} = MD_i - MD_{i-3} \tag{9.13}$$

$$MD_{gradiente_1} = \frac{MD_i - MD_{i-1}}{|time\_to\_tca_i - time\_to\_tca_{i-1}|} \tag{9.14}$$

$$MD_{gradiente_3} = \frac{MD_i - MD_{i-1}}{|time\_to\_tca_i - time\_to\_tca_{i-3}|} \tag{9.15}$$

A cada uno de los reportes CDM es necesario agregarles estas ocho variables para vincularlos con los reportes anteriores y correspondientes a un mismo evento. Vale aclarar que el algoritmo sólo vincula reportes de un mismo evento de colisión, se ha tenido especial cuidado en evitar vincular erróneamente reportes de eventos distintos.

Para finalizar, una vez que se tiene reorganizado el conjunto de datos se le agrega la variable a predecir que es justamente la probabilidad de colisión PC del reporte de colisión siguiente. Para esto a la fila i-1 se le agrega la PC correspondiente a la fila i. A la variable a predecir dentro del conjunto de datos se la denomina $TARGET\_PC$. El dataset fue preparado de esta manera con la idea de predecir el reporte siguiente, es decir, un valor en el futuro.

En cuanto al conteo nuevas variables, se tiene 8 definidas matemáticamente más 1 que es la variable a predecir. En total 9 nuevas variables, sin embargo, como ya se ha vinculado las

filas de un mismo evento de colisión, es momento de eliminar la variable *event_id*. Antes de la Ingeniería de Variables (o Atributos) se tenía 77 campos, ahora se tiene 85 variables que son las que quedan finalmente en el dataset que se introducirá en el modelo de Machine Learning.

3.4. Resumen de variables

El resumen completo de las 85 variables se encuentra en la Tabla 9.3, Tabla 9.4, Tabla 9.5 y Tabla 9.6 que es el conjunto de datos final luego del ETL y basado en el conjunto de datos original provisto por la Agencia Espacial Europea (ESA, 2019). En particular, la Tabla 9.4 y la Tabla 9.5 representan las mismas variables pero una por cada objeto: principal y perseguidor. Por este motivo, en el conjunto de datos, se verán reflejadas con el sufijo correspondiente t o c por sus siglas en inglés. (t: cuerpo principal u objetivo - target - y c: cuerpo perseguidor - chaser -)

Tabla 9.3: Variables del CDM comunes a ambos objetos

N°	Variable	Símbolo	Unidad	Descripción
1	time_to_tca	—	—	Intervalo de tiempo desde la creación del CDM al TCA
2	MISS_DISTANCE	$\|r_{rel}\|$	m	Distancia entre objetivo y perseguidor en el TCA
3	RELATIVE_SPEED	$\|v_{rel}\|$	m/s	Velocidad relativa entre objetivo y perseguidor en el TCA
4	RELATIVE_POSITION_N	r_{relr}	m	Distancia entre objetivo y perseguidor: normal
5	RELATIVE_POSITION_R	r_{relt}	m	Distancia entre objetivo y perseguidor: radial
6	RELATIVE_POSITION_T	r_{reln}	m	Distancia entre objetivo y perseguidor: transverse
7	RELATIVE_VELOCITY_N	v_{reln}	m/s	Velocidad relativa entre objetivo y perseguidor: normal
8	RELATIVE_VELOCITY_R	v_{relr}	m/s	Velocidad relativa entre objetivo y perseguidor: radial
9	RELATIVE_VELOCITY_T	v_{relt}	m/s	Velocidad relativa entre objetivo y perseguidor: transverse
10	COLLISION_PROBABILITY	PC	—	Probabilidad de colisión computada en el epoch del CDM

Las unidades mostradas en la Tabla 9.3, Tabla 9.4 y Tabla 9.5 también se encuentran en la página oficial del Concurso Kelvin *Collision Avoidance* de la Agencia Espacial Europea (ESA, 2019). Las unidades presentadas en la Tabla 9.6 se deducen de la expresiones matemáticas detalladas en Subsección 3.3.

Tabla 9.4: Variables del CDM correspondiente al objeto principal (target)

N°	**Variable**	**Símbolo**	**Unidad**	**Descripción**
11	t_CRDOT_RDOT	${\sigma_{v_r}}^2$	m^2/s^2	Varianza de la velocidad radial
12	t_CN_N	${\sigma_{r_n}}^2$	m^2	Varianza de la componente normal de posición
13	t_CORR_CN_R	$Corr(r_n, r_r)$	—	Correlación de la posición normal y posición radial
14	t_CORR_CN_T	$Corr(r_n, r_t)$	—	Correlación de la posición normal y posición transversal
15	t_CORR_CNDOT_T	$Corr(v_n, r_n)$	—	Correlación de la velocidad normal y la posición normal
16	t_CNDOT_NDOT	${\sigma_{v_n}}^2$	m^2/s^2	Varianza de la componente normal de velocidad
17	t_CORR_CNDOT_R	$Corr(v_n, r_r))$	—	Correlación de la velocidad normal y la posición radial
18	t_CORR_CNDOT_RDOT	$Corr(v_n, v_r)$	—	Correlación de la velocidad normal y de la velocidad radial
19	t_CORR_CNDOT_T	$Corr(v_n, t)$	—	Correlación de la velodidad normal y de la posición transversal
20	t_CORR_CNDOT_TDOT	$Corr(v_n, v_t)$	—	Correlación de la velocidad normal y de la velocidad transversal
21	t_CR_R	σ_r^2	m^2	Varianza de la componente radial de la posición
22	t_CORR_CT_R	$Corr(t, r)$	—	Correlación de la posición transversal y posición radial
23	t_CT_T	σ_t^2	m^2	Varianza de la componente transversal de la posición
24	t_CORR_CTDOT_N	$Corr(v_t, n)$	—	Correlación de la velocidad transversal y la posición normal
25	t_CORR_CRDOT_N	$Corr(v_r, n)$	—	Correlación de la velocidad radial y posición normal
26	t_CORR_CRDOT_T	$Corr(v_r, t)$	—	Correlación de la velocidad radial y posición transversal
27	t_CORR_CRDOT_R	$Corr(v_r, r)$	—	Correlación de la velocidad radial y posición radial
28	t_CORR_CTDOT_R	$Corr(v_t, r)$	—	Correlación de la velocidad transversal y posición radial
29	t_CORR_CTDOT_RDOT	$Corr(v_t, v_r)$	—	Correlación de la velocidad transversal y velocidad radial
30	t_CORR_CTDOT_T	$Corr(v_t, t)$	—	Correlación de la velocidad transversal y posición transversal
31	t_CTDOT_TDOT	$\sigma_{v_t}^2$	m^2/s^2	Varianza de la velocidad transversal
32	t_CD_AREA_OVER_MASS	—	m^2/kg	Coeficiente balístico
33	t_CR_AREA_OVER_MASS	—	—	Coeficiente de radiación solar (Coeficiente balístico equivalente)
34	t_APOGEE_ALTITUDE	—	km	Apogeo ($-R_{tierra}$)
35	t_PERIGEE_ALTITUDE	—	km	Perigeo ($-R_{tierra}$)
36	t_INCLINATION	i	deg	Inclinación
37	t_SEDR	—	W/kg	Tasa de disipación de energía
38	t_ACTUAL_OD_SPAN	—	días	Intervalo de actualización para determinación de órbita
39	t_OBS_AVAILABLE	—	obs/CDM	Cantidad de observaciones disponibles para determinar la órbita
40	t_OBS_USED	—	obs/CDM	Cantidad de observaciones usadas para determinar la órbita
41	t_RECOMMENDED_OD_SPAN	—	días	Intervalo recomendado para actualizar la órbita
42	t_RESIDUALS_ACCEPTED	—	—	Residuos de la determinación de órbita
43	t_WEIGHTED_RMS	—	—	Raíz cuadrada del método mínimos cuadrados para calc. órbita

Tabla 9.5: Variables del CDM correspondiente al segundo objeto o perseguidor (chaser)

N°	Variable	Símbolo	Unidad	Descripción
44	c_CRDOT_RDOT	${\sigma_{v_r}}^2$	m^2/s^2	Varianza de la velocidad radial
45	c_CN_N	${\sigma_{r_n}}^2$	m^2	Varianza de la componente normal de posición
46	c_CORR_CN_R	$Corr(r_n, r_r)$	—	Correlación de la posición normal y posición radial
47	c_CORR_CN_T	$Corr(r_n, r_t)$	—	Correlación de la posición normal y posición transversal
48	c_CORR_CNDOT_T	$Corr(v_n, r_n)$	—	Correlación de la velocidad normal y la posición normal
49	c_CNDOT_NDOT	${\sigma_{v_n}}^2$	m^2/s^2	Varianza de la componente normal de velocidad
50	c_CORR_CNDOT_R	$Corr(v_n, r_r))$	—	Correlación de la velocidad normal y la posición radial
51	c_CORR_CNDOT_RDOT	$Corr(v_n, v_r)$	—	Correlación de la velocidad normal y de la velocidad radial
52	c_CORR_CNDOT_T	$Corr(v_n, t)$	—	Correlación de la velodidad normal y de la posición transversal
53	c_CORR_CNDOT_TDOT	$Corr(v_n, v_t)$	—	Correlación de la velocidad normal y de la velocidad transversal
54	c_CR_R	σ_r^2	m^2	Varianza de la componente radial de la posición
55	c_CORR_CT_R	$Corr(t, r)$	—	Correlación de la posición transversal y posición radial
56	c_CT_T	σ_t^2	m^2	Varianza de la componente transversal de la posición
57	c_CORR_CTDOT_N	$Corr(v_t, n)$	—	Correlación de la velocidad transversal y la posición normal
58	c_CORR_CRDOT_N	$Corr(v_r, n)$	—	Correlación de la velocidad radial y posición normal
59	c_CORR_CRDOT_T	$Corr(v_r, t)$	—	Correlación de la velocidad radial y posición transversal
60	c_CORR_CRDOT_R	$Corr(v_r, r)$	—	Correlación de la velocidad radial y posición radial
61	c_CORR_CTDOT_R	$Corr(v_t, r)$	—	Correlación de la velocidad transversal y posición radial
62	c_CORR_CTDOT_RDOT	$Corr(v_t, v_r)$	—	Correlación de la velocidad transversal y velocidad radial
63	c_CORR_CTDOT_T	$Corr(v_t, t)$	—	Correlación de la velocidad transversal y posición transversal
64	c_CTDOT_TDOT	$\sigma_{v_t}^2$	m^2/s^2	Varianza de la velocidad transversal
65	c_CD_AREA_OVER_MASS	—	m^2/kg	Coeficiente balístico
66	c_CR_AREA_OVER_MASS	—	—	Coeficiente de radiación solar (Coeficiente balístico equivalente)
67	c_APOGEE_ALTITUDE	—	km	Apogeo ($-R_{tierra}$)
68	c_PERIGEE_ALTITUDE	—	km	Perigeo ($-R_{tierra}$)
69	c_INCLINATION	i	deg	Inclinación
70	c_SEDR	—	W/kg	Tasa de disipación de energía
71	c_ACTUAL_OD_SPAN	—	días	Intervalo de actualización para determinación de órbita
72	c_OBS_AVAILABLE	—	obs/CDM	Cantidad de observaciones disponibles para determinar la órbita
73	c_OBS_USED	—	obs/CDM	Cantidad de observaciones usadas para determinar la órbita
74	c_RECOMMENDED_OD_SPAN	—	días	Intervalo recomendado para actualizar la órbita
75	c_RESIDUALS_ACCEPTED	—	—	Residuos de la determinación de órbita
76	c_WEIGHTED_RMS	—	—	Raíz cuadrada del método mínimos cuadrados para calc. órbita

Tabla 9.6: Atributos creados a partir de la ingeniería de variables y variable a predecir

N°	**Variable**	**Símbolo**	**Unidad**	**Descripción**
77	PC_trend_1	—	—	Tendencia de PC con respecto a un reporte anterior
78	PC_trend_3	—	—	Tendencia de PC con respecto a tres reportes anteriores
79	PC_gradient_1	—	$1/días$	Gradiente de PC con respecto a un reporte anteriores
80	PC_gradient_3	—	$1/días$	Gradiente de PC con respecto a tres reportes anteriores
81	MD_trend_1	—	m	Tendencia de MD con respecto a un reporte anterior
82	MD_trend_3	—	m	Tendencia de MD con respecto a tres reportes anteriores
83	MD_gradient_1	—	$m/días$	Gradiente de MD con respecto a un reporte anterior
84	MD_gradient_3	—	$m/días$	Gradiente de MD con respecto a tres reportes anteriores
85	TARGET_PC	—	—	Probabilidad de colisión del reporte de CDM siguiente

3.5. Filtrado de registros

Como se ha visto en la sección 2 el conjunto de datos está desbalanceado siendo los eventos de colisión escasos. Resulta de utilidad aplicar una herramienta de aprendizaje automático en aquellas situaciones en las que el operador de satélites tiene incertidumbre acerca de dentro de cuál de las dos franjas de mayor riesgo cae el evento: área de maniobra o área de monitoreo. Carece de sentido aplicar algún procedimiento cuando el evento se encuentra totalmente fuera del protocolo de acción. Por ejemplo, si el CDM indica una probabilidad de colisión de -15, entonces se puede asegurar que no se necesitará realizar una maniobra de disuasiva de colisión y, por lo tanto, carece de sentido aplicar cualquier tipo de análisis adicional. Sin embargo, cuando el CDM presenta una PC entre -6 y -4, es aquí donde es conveniente tener un algoritmo que permita obtener información adicional de la secuencia de mensajes emitida por el ente de monitoreo. Lo aquí dicho es consistente con los criterios señalados en 9.2.2.1 Análisis de las variables del CDM (Conjunction Data Messages) en esta misma sección.

Debido a esto, se decidió trabajar con dos conjuntos de datos y observar si es posible que el modelo generado con ellos devuelva mejores resultados:

- No filtrado: comprende todas las probabilidades de colisión del dominio del problema desde -30 a 0
- Filtrado: comprende las probabilidades de colisión mayores o iguales a -6 solamente.

3.6. Conjunto de entrenamiento y evaluación

Para todos los modelos desarrollados y analizados en este trabajo de investigación se dividió al conjunto de datos de la siguiente manera: 70 % para entrenamiento y 30 % para la validación. Se ha utilizado siempre el mismo valor semilla y verificado que los conjuntos de datos se mantienen idénticos cada vez que se ha construido un modelo. Para esto se ha utilizado la función *train_test_split* de la librería Sci-kit learn (Pedregosa y col., 2011). En resumen, en cuanto al conjunto de datos, se tiene:

- Conjunto de datos no filtrado:
  - Conjunto de entrenamiento (77235 registros)
  - Conjunto de validación (33102 registros)
- Conjunto de datos filtrado:
  - Conjunto de entrenamiento (3775 registros)
  - Conjunto de validación (1616 registros)

En todos los conjuntos de datos siempre se tiene las mismas 85 variables ya detalladas anteriormente en 9.3.3.4 Resumen de variables.

4. Modelado

4.1. Selección de Modelos de Machine Learning

Los modelos de Machine Learning elegidos para este trabajo fueron los siguientes:

- Regresión Lineal Múltiple
- AdaBoostRegressor
- LightGBM

Se comienza el Trabajo de Investigación construyendo un Modelo de Regresión Lineal Múltiple. Este modelo es útil para el Aprendizaje Supervisado. Requiere bajo poder de cómputo y permite obtener resultados rápidamente para tener una primera impresión de la resolución del problema. En particular, el Modelo de Regresión Lineal es de gran utilidad cuando la variable a predecir es de tipo cuantitativo, como es justamente en esta problemática (James y col., 2013). Además, en el caso de existir una relación lineal entre las variables predictoras y la variable a predecir, es un modelo que sería suficiente para realizar predicciones. Si con este modelo, se obtiene buenas predicciones es probable que no sea necesario utilizar modelos no lineales. Por todo lo dicho, la Regresión Lineal suele ser un buen punto de partida y es un método que se usa ampliamente en Estadística. En cuanto a la implementación del algoritmo en código computacional, se puede decir que es sencilla porque está incluido en la librería Sci-Kit Learn (Pedregosa y col., 2011).

En una segunda instancia se avanzó con algoritmos de Aprendizaje Conjunto (Ensemble Learning) que están basados en Árboles de Decisión. Los Árboles de Decisión pueden ser graficados e interpretados, siendo esto una ventaja frente a otros algoritmos. Las técnicas de Aprendizaje Conjunto permiten generar un modelo predictivo a partir de modelos predictivos más simples, en este caso, se utiliza una gran cantidad de Árboles de Decisión. Por este motivo, aplicar estos modelos suele dar buenos resultados. En especial, suele usarse para problemas de Clasificación, pero también puede usarse para problemas de Regresión. El Trabajo Final de Investigación se presentó como una oportunidad interesante para explorar esta técnica de Ensamble de Árboles de Decisión para problemas de Regresión y evaluar sus resultados.

Se eligió los modelos de AdaBoostRegressor y Light Gradient Boosting Machines porque son algoritmos de Boosting cuyo concepto básico es ir perfeccionando o mejorando los Árboles de Decisión con las sucesivas iteraciones. A su vez, en cuanto a su aplicación computacional ambos cuentan con una sólida documentación: AdaBoostRegressor se puede aplicar utilizando también la librería Sci-Kit Learn (Pedregosa y col., 2011) y Light Gradient Boosting Machines con la librería que lleva su mismo nombre LightGBM (LightGBM, 2021). Por ser algoritmos muy utilizados en los últimos años, se puede encontrar soporte en foros y artículos relacionados con el tema fácilmente en Internet.

Finalmente, cabe mencionar que los tres modelos enunciados en esta sección se aplicarán tanto al conjunto de datos filtrado y al no filtrado.

4.2. Modelo Lineal: Regresión Lineal Múltiple

En primer lugar se esboza una solución al problema utilizando un modelo de regresión lineal múltiple. Múltiple indica que existen dos o más variables de entrada, en este problema en particular, son las que figuran en la Tabla 9.3 más las variables creadas en la sección Ingeniería de variables. A excepción de la variable probabilidad de colisión $TARGET\_PC$ que es la variable dependiente que el modelo busca predecir.

Para generar el modelo de regresión lineal se ha utilizado el modelo *LinearRegression* de la librería Sci-kit learn (Pedregosa y col., 2011). Previo a ajustar los parámetros del modelo con el método fit, se debe generar dos arreglos de datos el conjunto X y el conjunto Y. El conjunto X se constituye de todas las variables predictoras del conjunto de datos y el Y consiste en la variable a predecir $TARGET\_PC$. Una vez ajustado el modelo se obtiene los parámetros de pendiente y ordenada al origen que se presentan en los bloques de código que se presentan a continuación 9.1 y 9.2.

Luego con los modelos ya generados se predice los valores utilizando el conjunto de datos de validación y las funciones *r2_score*, *root_mean_squared_error*, *mean_absolute_error* también de la librería Sci-kit learn [28]. La comparativa de ambos modelos junto a sus métricas de evaluación se presenta en la Tabla 9.7.

```

intercept: -9.807337806180318
slope: [ 1.38614063e+00 -6.09367686e-05  6.71907841e-05  1.08827030e-04
  8.54783784e-06 -3.92714006e-06  5.58618068e-04  1.38466528e-04
  4.47941851e-07  4.85449867e-01 -4.28666570e-05 -5.71793055e-08
  4.31275204e-05  4.18873695e-02 -8.72734078e-02 -8.72009253e-02
  1.29017650e-01 -9.44738849e-02 -1.66344091e-04  1.17478095e-04
  2.19028171e-02  5.34976859e-03  5.30266218e-05 -6.38158530e-06
 -1.19604804e-03  1.89562077e-02 -2.04312511e-02 -3.23141889e-02
  1.20212521e-09 -7.08828829e-13  1.06509096e-08  6.79756249e-07
 -9.32026560e-03  1.60438570e-03  1.10063733e-01  9.84867269e-02
 -4.90277373e-03  1.67966463e-03 -1.07028176e-02  1.31485757e-02
  1.25465701e-03 -5.11044643e-02  4.21948626e-02 -2.27362736e-05
 -3.38558293e-03 -1.85212683e-02 -2.90244569e-02  2.02167334e-02
 -1.22741597e-02  2.52653345e-02 -4.21001651e-03  8.79620116e-03
  6.83108933e-03  3.15405712e-02 -1.88764461e-02 -2.76391677e-02
 -4.34707977e-02 -2.22364666e-03 -1.30944479e-02  1.10775696e-03
  4.30511843e-02  3.05781432e-02 -9.97506888e-03  2.61926525e-02
 -2.66549574e-02 -2.12681840e-03 -2.76916398e-02  2.39621484e-01
 -3.72370283e-02 -1.97522387e-02  3.33805720e-02 -7.63360425e-03
 -6.63545325e-02  5.21382961e-02  6.72343839e-02 -6.74599824e-03
  9.20984942e-03  2.77938539e-02  8.30492146e-02  7.91650784e-02
  5.68656794e-05  1.12555974e-05 -3.48839978e-06 -7.46638959e-06]
```

Código 9.1: *Parámetros del modelo de Regresión Lineal Múltiple para conjunto de datos no filtrado*

```
intercept: -6.691519910046921
slope: [-3.09807240e-02 -2.40573478e-05  2.21683144e-05 -1.27823011e-04
 -9.04643425e-07  1.09994147e-06 -7.81919511e-05  2.07537346e-05
  7.41529079e-06  6.95000870e-02 -1.13900275e-05  5.24349733e-08
  1.13990724e-05 -3.93768153e-02  5.81538045e-04 -1.22469205e-03
 -1.95866318e-03 -4.86581015e-02  5.25653218e-05 -1.80809871e-04
  5.70890674e-02 -4.49226534e-01 -1.42906931e-04  5.65960245e-03
  1.17844494e-02 -2.39975738e-03  2.91106941e-03 -2.29866319e-03
 -2.76644279e-10 -7.36342237e-12 -1.18791883e-08  8.25713592e-06
  7.53419317e-02 -6.74326295e-02 -5.75477192e-03  3.93084932e-03
  1.88957973e-03  2.79901236e-05 -2.21388914e-02 -4.60738601e-02
 -1.31411562e-02  2.99400396e-03 -5.69729333e-02 -3.01783591e-05
 -1.87855482e-04 -1.04128432e-03 -5.31053780e-03 -2.42122739e-02
 -1.60499823e-01  3.25797933e-02 -1.12223948e-03  1.79476159e-01
  1.33810208e-01 -5.50937628e-02  5.29296979e-02  2.46568234e-02
  1.07937822e-02 -1.28809130e-01 -5.13881986e-02  1.36411232e-01
 -1.38606719e-02 -4.49306422e-02 -1.11806294e-01 -2.77724554e-02
  5.72852118e-02  2.52199518e-03  1.04411174e-02  8.69113366e-02
 -1.69761030e-02 -1.19658832e-01  4.41380420e-03  7.28581749e-02
  1.65064124e-03 -3.64844231e-02 -1.77654122e-02  5.09788818e-02
  1.05788694e-02 -2.25683353e-02 -9.23092652e-03  1.56282570e-02
  2.75127641e-05 -4.61405853e-06 -6.17996966e-06  7.66374556e-06]
```

Código 9.2: *Parámetros del modelo de regresión lineal múltiple para conjunto de datos filtrado*

Tabla 9.7: Métricas de evaluación del modelo de regresión lineal múltiple

Parámetro	**Dataset no filtrado**	**Dataset filtrado**
R^2	0.69901	0.20680
MSE	28.64101	0.36999
$RMSE$	5.35173	0.60827
MAE	4.06549	0.47539

En la Tabla 9.7 se observa un valor R^2 aceptable para el modelo de regresión lineal múltiple para el conjunto de datos no filtrado. Y si bien, el R^2 para el modelo generado con el conjunto de datos filtrado es bajo, también son muy superiores las métricas $RMSE$ y MAE para el segundo modelo.

4.3. Modelos No Lineales

4.3.1. Modelo No Lineal: AdaBoost

Para construir el modelo de regresión AdaBoost, lo primero que se hace es instanciar la clase *AdaBoostRegressor* del módulo *ensemble* de la librería Sci-kit learn (Pedregosa y col., 2011). El algoritmo implementado por esta librería es el AdaBoost.R2 presentado por Drucker (Drucker, 1997). Se aplica la técnica de validación cruzada (cross validation) a través de la clase *RepeatedKFold* del módulo *model_selection* también de la librería Sci-kit learn (Pedregosa y col., 2011). Al instanciar la clase *RepeatedKFold* se cuenta con dos parámetros *n_splits* y *n_repeats*. El primero de ellos indica la cantidad de divisiones que se hace en el conjunto de entrenamiento para la validación cruzada, por defecto se ha fijado arbitrariamente en un valor igual a 5 (es el mismo valor propuesto en la documentación oficial de la librería). El segundo parámetro *n_repeats* indica la cantidad de veces que se repite la validación cruzada, este valor se ha fijado en un valor igual a 3, para disminuir el tiempo de cómputo del algoritmo; cuanto mayor este número más tardará en generarse el modelo. Cabe destacar que al instanciar esta clase *RepeatedKFold* se seleccionó un valor semilla fijo con el objeto que siempre las divisiones del dataset sean constantes.

La técnica de validación cruzada se aplica solamente sobre el conjunto de entrenamiento para generar el modelo. Esta técnica implica subdividir el set de entrenamiento. Mientras se optimiza el modelo, lo valida contra uno de los subconjuntos previamente divididos del mismo set de entrenamiento. Si bien ocurre una validación intrínsecamente al utilizar la validación cruzada, en este trabajo de investigación se decidió realizar la validación final utilizando el conjunto de evaluación detallado en 9.3.3.6 Conjunto de entrenamiento y evaluación. De esta manera, se realiza la evaluación contra un conjunto que no ha sido utilizado para generar el modelo y es totalmente independiente del conjunto de entrenamiento. Esta misma manera de trabajar se aplicó en el Modelo AdaBoost y en el LightGBM que se verá en la sección siguiente.

Los hiperparámetros disponibles para ajustar en el modelo *AdaBoostRegressor* son *n_estimators*, *learning_rate* y *loss*. El primer parámetro *n_estimators* indica la cantidad máxima de Árboles de Decisión en la cual la técnica de boosting finaliza, en el caso de un ajuste perfecto este número de Árboles de Decisión no se alcanza y se dice que el algoritmo terminó temprano (*early stopping*). El segundo parámetro es *learning_rate*, tasa de aprendizaje, es una medida del peso y la importancia que se le aplica a cada Árbol de Decisión. Por último, el parámetro *loss* es la función de pérdida utilizada para actualizar los pesos luego de cada iteración, es este caso se ha definido utilizar la curva exponencial. En este trabajo se decidió utilizar la técnica de búsqueda en grilla (*Grid Search*) para el ajuste de los primeros hiperparámetros. Para esto la librería Sci-kit learn (Pedregosa y col., 2011) cuenta con el módulo *model_selection* y la clase *GridSearchCV*. Al inicializar la clase *GridSearchCV* se definen los siguientes parámetros:

- *estimator*: es el modelo sobre el que se hace el *GridSearch*, en este caso se elige la instancia

creada del *AdaBoostRegressor*.

- *param_grid*: es un diccionario al cual se le pasa dos listas de valores una para *n_estimators* (cantidad de árboles) y la otra para *learning_rate* (tasa de aprendizaje entre cada iteración). La lista de valores pasadas en este trabajo se presentan en 9.3.

- *scoring*: indica la estrategia utilizada para validar la performance del modelo de cross-validation. En este caso se ha utilizado la función que ofrece el módulo *metrics* de la librería Sci-kit learn (Pedregosa y col., 2011), en particular R^2 *r2_score*.

- *cv*: en este parámetro se seleccionó la clase *RepeatedKFold* instanciada anteriormente.

- *n_jobs*: indica la cantidad de núcleos del procesador (CPU) que se pueden utilizar para el cómputo. En este trabajo por defecto se ha utilizado un valor de -1 que se traduce en todos los núcleos disponibles.

- *verbose*: indica el nivel de expresividad que se imprime por la consola mientras el algoritmo *GridSearch* está corriendo.

Para ajustar los hiperparámetros a través del *GridSearch* esta clase posee el método *fit* a la cual se le pasa las variables predictoras del conjunto entrenamiento X y la variable a predecir Y.

```
1 n_estimators_v = [100,300,500,800,1000,1500,2000,3000,4000,5000]
2 learning_rate_v= [0.001,0.002,0.003,0.005,0.01,0.03,0.05]
```

Código 9.3: *Listado de valores para los parámetros n_estimators y learning_rate para el GridSearch*

Todo lo mencionado en este apartado fue integrado en una única función incluida en el repositorio *grid_search_optimization* a la cual es posible configurarle todos los parámetros. En esta función también se ha incorporado la manera de ir registrando los resultados. Se graban dos archivos, uno que contiene los hiperparámetros óptimos del modelo y el otro con la corrida completa iteración por iteración.

Se presentan una muestra reducida de los mejores resultados obtenidos para el dataset sin filtrar en la Tabla 9.8 y para el dataset filtrado en la 9.9.

Para el primer caso, en la Tabla 9.8 se coloreó en amarillo la fila correspondiente al modelo que devuelve mejores predicciones. El mínimo error cuadrático medio MSE es de 16.64689 y se obtiene con 100 árboles y una tasa de aprendizaje de 0.001, La métrica de error medio absoluto MAE es de 2.10868 y también es mínima para esta configuración de hiperparámetros. Cabe mencionar que para decidir qué modelo realiza mejores predicciones se tomará como principal métrica el MSE. Se observa un valor asintótico del R^2 de aproximadamente 0.82 para diferentes valores de *n_estimators* y de *learning_rate*

En cuanto al modelo generado a partir del conjunto de datos filtrado y los resultados presentados en la Tabla 9.9, se observa que el valor óptimo de de MSE es 0.1075 y se logra con 100 árboles (*n_estimators*) y una tasa de aprendizaje de 0.005 (*learning_rate*). Esto es consistente con las métricas $RMSE$ 0.3278 y MAE 0.2069. A su vez, también presenta el mayor valor de R^2

Al comparar las métricas de evaluación del modelo construido a partir del conjunto de datos completo y del filtrado resalta en primer lugar que el R^2 del primero (0.82506) es mayor al obtenido a través del dataset filtrado (0.7696). No obstante, se debe tener en cuenta que el valor de MSE en el primer caso es de 16.64689 mientras que para el segundo caso es 0.10748 . Esto indefectiblemente indica que las predicciones realizadas por el modelo generado a partir de los datos del conjunto filtrado, tendrán un desvío mucho menor con respecto al valor real que en el caso del modelo generado a partir del dataset completo (sin filtrar).

Tabla 9.8: Métricas de evaluación de los modelos AdaBoost generados con el dataset no filtrado

id	**n_estimators**	**learning_rate**	R^2	MSE	$RMSE$	MAE
1	100	0.001	0.82506	16.64689	4.08006	2.10868
2	300	0.001	0.82456	16.69432	4.08587	2.16359
3	500	0.001	0.82457	16.69285	4.08569	2.21788
4	800	0.001	0.82436	16.71327	4.08819	2.28693
5	100	0.002	0.82478	16.67307	4.08327	2.13508
6	300	0.002	0.82451	16.6989	4.08643	2.23696
7	100	0.003	0.82464	16.68617	4.08487	2.16068
8	300	0.003	0.82398	16.74976	4.09265	2.31679

Tabla 9.9: Métricas de evaluación de los modelos AdaBoost generados con el dataset filtrado

id	**n_estimators**	**learning_rate**	R^2	MSE	$RMSE$	MAE
1	300	0.001	0.76751	0.10845	0.32932	0.20847
2	500	0.001	0.7682	0.10813	0.32883	0.20756
3	800	0.001	0.76813	0.10817	0.32888	0.20815
4	300	0.002	0.76894	0.10779	0.32831	0.20783
5	500	0.002	0.76799	0.10823	0.32898	0.20947
6	100	0.003	0.76822	0.10812	0.32882	0.20725
7	300	0.003	0.76759	0.10842	0.32927	0.20959
8	100	0.005	0.7696	0.10748	0.32784	0.20688

4.3.2. Modelo No Lineal: LightGBM

Para implementar el modelo LGBM (Light Gradient Boosting Machines) se utiliza la librería LightGBM (LightGBM, 2021). El primer requisito para generar un modelo con esta librería es instanciar una clase para el dataset *lgbm.Dataset()* la cual consta de dos parámetros, el primero de ellos es *data* en el cual se indica la parte del dataframe que corresponde con las variables predictoras y el parámetro *label* que corresponde con la variable a predecir. Luego, dado que la librería LightGBM (LightGBM, 2021) no incluye las métricas de evaluación utilizadas en este estudio, se tuvo que definir funciones para el R^2, $RMSE$, MAE utilizando la librería Scikit-learn (Pedregosa y col., 2011). En este modelo también se aplicó la técnica de validación cruzada (*cross validation*), pero en este caso se utiliza el método correspondiente a esta librería (LightGBM, 2021), en este caso *lgbm.cv()*. Éste cuenta con los siguientes parámetros:

- *params*: es un diccionario en el cual se indican los hiperparámetros del modelo a generar. Se hará especial hincapié en este punto en el próximo listado.
- *train_set*: se debe indicar la instancia generada para el dataset (mencionada en el párrafo anterior) en la cual se detallan las variables predictoras y a predecir.
- *nfolds*: indica la cantidad de divisiones del conjunto para aplicar la técnica de validación cruzada. En este caso, se ha dejado arbitrariamente 5 divisiones, para ser consistentes con las ya utilizadas en el modelo anterior y hacer una comparación pareja.
- *seed*: se utiliza un valor semilla para que los cálculos del procesador siempre comiencen desde el mismo punto. Esto permite por ejemplo que la división del dataset indicadas en el *nfolds* siempre sean las mismas.
- *stratified*: este parámetro se lo dejó en *False*. Esto significa que se decidió no aplicar la técnica *stratified sampling* en este caso.
- *feval*: indica la función que se utilizará como métrica de evaluación. Como se ha mencionado en el párrafo anterior, se definieron tres funciones para esta métrica, en este caso para ser consistente con ambos modelos también se ha elegido como función a optimizar R^2.

Como se dijo en la enumeración anterior, es menester detallar el ítem *params* que se corresponde con los hiperparámetros del modelo LightGBM.

- *application*: indica el tipo de modelo a aplicar. En este caso es un modelo de regresión, se debe indicar *regression*
- *Cantidad de árboles (num_iterations)*: cantidad de árboles a generar en el modelo

- *Detención temprano (early_stopping_round)*: si el modelo converge en una cantidad determinada de árboles, no seguirá computando nuevos árboles. Se ha definido por defecto, un valor de 50 árboles para este parámetro.
- *Número de hojas (num_leaves)*: cantidad de hojas de cada árbol.
- *Fracción de predictores (feature_fraction)*: cantidad de atributos del conjunto de datos utilizada como predictores en cada iteración.
- *Tasa de aprendizaje (learning_rate)*: indica en cuánto deben mejorar los residuos de un árbol al siguiente.
- *Fracción de bagging (bagging_fraction)*: cantidad de registros que deben ser tomados como subconjunto de datos del conjunto original en cada iteración. Aquí vale hacer una aclaración, si bien la librería permite ajustar este hiperparámetro, en este caso se decidió dejarlo en un valor de 1. Esto significa que no se aplicará la técnica de bagging con el objetivo de ser consistentes con la teoría del modelo LGBM en la cual en su sentido más puro no se utiliza el bagging.
- *Máxima profundidad (max_depth)*: máxima cantidad de niveles del árbol.
- *Mínima ganancia por división (min_split_gain)*: mínima ganancia aceptada para considerar una división de una hoja como valida mientras se contruye el árbol.
- *Mínimo peso de las hojas hijas (min_child_weight)*: mínima cantidad de elementos dentro de una hoja terminal del árbol.

Una vez definido el modelo y explicado cómo se aplica al algoritmo la validación cruzada es necesario realizar el ajuste de hiperparámetros. Para esto, se decidió aplicar la técnica de optimización bayesiana utilizando la librería (Nogueira, 2021). Se utiliza la función *BayesianOptimization*, a la cual se le pasa como argumentos la función que genera el modelo de LGBM y una estructura similar a la de un diccionario la cual indica entre que rango de valores se ajustarán los hiperparámetros. En este caso se denomina *pds* y se presentan en el siguiente bloque de código 9.4. Cada uno de los pares de valores entre paréntesis indica el valor mínimo y máximo utilizado.

```
  pds = {
    "num_leaves": (2, 120),
    "feature_fraction": (0.1, 0.9),
    "bagging_fraction": (1, 1),
    "max_depth": (7, 15),
    "learning_rate": (0.001, 0.05),
    "min_split_gain": (0.001, 0.1),
    "min_child_weight": (10, 35)
}
```

Código 9.4: *Rango utilizado para evaluar los hiperparámetros para el modelo LGBM a través de la optimización bayesiana*Rango utilizado para evaluar los hiperparámetros para el modelo LGBM a través de la optimización bayesiana

Todo esto se ha integrado en una función que en el repositorio se ha denominado *create_and_validate_model()* la cual genera el modelo y calcula los hiperparámetros óptimos para el mismo. A su vez realiza un registro de los valores obtenidos en archivos de texto plano para su posterior utilización y visualización. La función *create_and_validate_model()* acepta como parámetro la cantidad de árboles con la que se tiene que generar el modelo. Por lo tanto, este algoritmo que incluye la optimización bayesian de hiperparámetros se corrió para diferentes cantidades de árboles de decisión, estos se pueden apreciar a continuación en Código 9.5 .

```
num_iteration = [300,500,800,1000,1500,2000,3000,4000,5000]
```

Código 9.5: *Cantidad de Árboles de Decisión con la que se realizó diferentes corridas de la optimización bayesiana*

Los resultados obtenidos de las diferentes corridas se presentan en la Tabla 9.10 y en la Tabla 9.11.

Tabla 9.10: Resultados obtenidos de la optimización bayesiana para modelos generados con distinto valor de árboles para el conjunto de datos sin filtrar

trees	f_fraction	l_rate	max_depth	min_child_weight	min_split_gain	num_leaves	R^2	MSE	$RMSE$	MAE
100.0	0.7737	0.0443	10.7859	31.9295	0.0521	76.5098	0.8481	14.4497	3.8013	1.9366
300.0	0.8738	0.0428	13.1633	22.3756	0.0794	112.4888	0.8535	13.937	3.7332	1.8017
500.0	0.5505	0.0336	11.1567	23.6016	0.0441	117.0283	0.853	13.9885	3.7401	1.8394
800.0	0.5123	0.0275	14.2923	22.5871	0.0088	89.2244	0.8524	14.0466	3.7479	1.8573
1000.0	0.5961	0.0234	11.0664	18.8842	0.0222	77.6732	0.8513	14.1496	3.7616	1.8587
1500.0	0.635	0.0168	14.9115	15.4894	0.0722	66.4412	0.8489	14.3766	3.7916	1.9297
2000.0	0.5959	0.0219	14.9682	26.6986	0.0156	45.2324	0.8497	14.3042	3.7821	1.8891
3000.0	0.5742	0.0154	14.9185	14.7716	0.0399	51.1713	0.8474	14.5203	3.8105	1.9826
4000.0	0.5746	0.0078	11.5114	15.2462	0.0324	51.1223	0.8294	16.2351	4.0293	2.6822
5000.0	0.6155	0.0061	14.6074	18.5323	0.0086	62.4166	0.8121	17.8796	4.2284	3.1376

Tabla 9.11: Resultados obtenidos de la optimización bayesiana para modelos generados con distinto valor de árboles para el conjunto de datos filtrado

trees	f_fraction	l_rate	max_depth	min_child_weight	min_split_gain	num_leaves	R^2	MSE	$RMSE$	MAE
300.0	0.1898	0.0285	13.4736	10.7263	0.0468	43.2314	0.7982	0.0941	0.3068	0.1857
500.0	0.1879	0.0182	14.8973	16.3655	0.0835	48.939	0.7945	0.0959	0.3096	0.1902
800.0	0.1932	0.0331	9.1966	11.9574	0.0999	11.4661	0.8046	0.0911	0.3019	0.1812
1000.0	0.1932	0.0331	9.1966	11.9574	0.0999	11.4661	0.8046	0.0911	0.3019	0.1812
1500.0	0.1879	0.0182	14.8973	16.3655	0.0835	48.939	0.7945	0.0959	0.3096	0.1902
2000.0	0.1936	0.0286	9.7955	18.8954	0.0685	75.1235	0.7959	0.0952	0.3086	0.1867
3000.0	0.1861	0.0059	14.9599	27.0837	0.0279	16.4312	0.684	0.1474	0.3839	0.2858
4000.0	0.1932	0.0331	9.1966	11.9574	0.0999	11.4661	0.8046	0.0911	0.3019	0.1812
5000.0	0.1932	0.0331	9.1966	11.9574	0.0999	11.4661	0.8046	0.0911	0.3019	0.1812

De estas tablas se desprende que, el valor óptimo de árboles en cada caso es el siguiente:

- Caso dataset completo
  - $trees = 300$
  - Métricas de evaluación: se obtiene el mínimo MSE 13.937 y un MAE de 1.8017. El R^2 de 0.853 también es el máximo.
- Caso dataset filtrado
  - $trees = 800$
  - Métricas de evaluación: se obtiene el mínimo MSE 0.0911 y un MAE de 0.1812. El R^2 de 0.853 también es el máximo
  - Se observa que el modelo converge ya que para una cantidad de árboles de 1000, 4000, 5000 la optimización bayesiana ha obtenido los mismos valores.

Comparando nuevamente los resultados para el modelo generado a partir del conjunto de datos completo y filtrado, se llega a una conclusión similar que la obtenida para el modelo AdaBoost. En este modelo LGBM, ocurre lo mismo los mejores valores de R^2 se obtienen para el caso del dataset completo. Mientras que para el caso del dataset no filtrado el R^2 es levemente inferior, pero el MSE es solo de 0.0911 lo que permite predicciones con mucho menor desvío con respecto a la realidad. Recordar que la media del error cuadrático MSE es la que se tomará para tomar las decisiones finales al evaluar los modelos.

Sección 10

Resultados

En este capítulo se resume lo realizado en la parte experimental de este Trabajo de Investigación.

En primer lugar, se obtuvo un conjunto de datos público de la Agencia Espacial Europea. En particular, éste se encuentra en la página web Kelvin destinada a estos concursos en la sección "Collision Avoidance Challenge" (ESA, 2019). El conjunto de datos de la ESA tiene 162634 registros y 103 atributos. Este dataset no se usó directamente, sino que se convirtió al formato estándar de reportes de CDM con el objeto de que el algoritmo desarrollado en este trabajado pueda ser extensivo a otros datos de entrada y no sólo a los brindados en el concurso "Collision Avoidance Challenge" de la ESA. Para convertirlo al formato estándar se tomó como referencia al *Recommended Standard Conjunction Data Message* del Instituto CCSDS (CCSDS, 2013). Luego de este proceso el dataset quedó con la misma cantidad de registros 162634, pero con 83 variables. En su mayoría se debe a variables que no están presentes en un reporte CDM estandarizado y fueron exclusivamente precalculadas por la ESA para el concurso antes mencionado.

Luego, se realizó un análisis de los datos presentes en el dataset. Las variables más destacas son: probabilidad de colisión (PC), MISS_DISTANCE (MD),y time_to_tca. Conceptualmente representan la distancia a la que pasará un objeto con respecto al otro (MD) en el momento del evento de conjunción cuando se encuentren más próximo uno del otro (TCA). La variable time_to_tca computa la diferencia entre la fecha de creación del reporte y el TCA del evento de conjunción, y la variable probabilidad de colisión indica la probabilidad de ocurrencia del evento de colisión.

La distribución de la variable PC se representó en Figura 9.4.Allí se observó que existían un gran número de reportes de colisión con probabilidades bajas que no era necesario tener en cuenta para realizar una predicción, porque estaban completamente fuera de cualquier protocolo de maniobra. Se definió para este trabajo de investigación un valor de umbral de 10^{-4} para que un reporte entrara en protocolo de maniobra, se utilizó el mismo valor propuesto por Merz y col. (Merz y col., 2017). Sólo a fines académicos con el objeto de tener una probabilidad de

corte para el análisis de los CDM, se tomó como valor 10^{-6}. Aquellos CDM cuya PC está entre 10^{-6} y 10^{-4} se los consideró como dentro de la zona de alerta o monitoreo.

El proceso de ETL para preparar el conjunto de datos se detalló exhaustivamente en 9.3.3.1. Lo más destacado fue convertir las PC a escala logarítmica base 10, eliminar todos los registros nulos, eliminar variables que tenían inconsistencias en sus valores y variables no numéricas. Otra conversión realizada fue pasar los elementos de la diagonal de la matriz de covarianzas de la velocidad y la posición del objeto a coeficientes de correlación. Luego de estas transformaciones el conjunto de datos quedó constituido por 110337 registros y 77 variables.

También se explicó la estructura general del dataset presentado por la ESA, esto es importante para entender el trabajo porque de los 110337 registros que tiene el dataset, se tiene 9229 eventos distintos de colisión. Es decir, en promedio se tiene casi 12 reportes CDM por evento de colisión. Luego se realizó Feature Engineering agregando 9 variables más al conjunto de datos. La variable a predecir se denominó TARGET_PC, y corresponde con la probabilidad de colisión del reporte CDM siguiente (considerando siempre un mismo evento de colisión).

En síntesis, quedaron 110337 registros y 85 variables en el conjunto de datos. Un resumen detallado de ellas se puede encontrar en 9.3.3.4.

Por último, se trabajó con dos datasets uno que contempló todas las PC hasta 10^{-30} llamado dataset no filtrado; y otro llamado dataset filtrado que solo tiene PC mayores a 10^{-6}. A estos dataset de los dividió en un conjunto de entrenamiento y otro de validación o evaluación. Todos ellos con la misma cantidad de variables (columnas), quedando de la siguiente manera:

- Conjunto de datos no filtrado (110337 filas):
  - Conjunto de entrenamiento (77235 registros)
  - Conjunto de validación (33102 registros)
- Conjunto de datos filtrado (5391 filas):
  - Conjunto de entrenamiento (3775 registros)
  - Conjunto de validación (1616 registros)

Luego se seleccionó los Modelos de Machine Learning a utilizar en este Trabajo Final de Investigación: Regresión Lineal Múltiple, AdaBoostRegressor y LightGBM (Light Gradient Boosting Machines).

El primer modelo propuesto resolvió el problema a través de una regresión lineal múltiple. Se computó los parámetros de este modelo, calculando el término independiente y ajustando los valores de las pendientes y se obtuvo un valor de MSE de 28.61101 y R^2 de 0.699 para el dataset completo. Para el dataset filtrado se obtuvo MSE de 0.36999 y de R^2 0.2068. Estas métricas de evaluación están muy por debajo de las obtenidas con los modelos no lineales.

En cuanto a los modelos no lineales, se trabajó con algoritmos AdaBoost y LightGBM. En ambos casos para la creación del modelo se trabajó con la técnica de validación cruzada sobre el conjunto de entrenamiento. Y, luego como el último paso de evaluación se contrastó el modelo construido contra el conjunto de evaluación correspondiente, que estuvo separado y es totalmente independiente del subconjunto utilizado para generar el modelo.

Para definir qué configuración de hiperparámetros realiza mejores predicciones se toma como referencia la métrica MSE. Esto es consistente con el tratamiento que realiza James, Witten, Hastie y Tibshirani en la sección de Laboratorio de Árboles de Decisión 8.3 de su libro (James y col., 2013), y es extensible a modelos no lineales en general.

Los resultados obtenidos para ambos modelos se observan en la Tabla 10.1. En lo que respecta a los modelos generados con el dataset completo, se alcanzó mejor resultados de MSE utilizando el algoritmo LightGBM con respecto al AdaBoost. Por el otro lado, cuando se trabajó con el dataset filtrado (utilizando un rango acotado de las probabilidades de colisión para el entrenamiento del modelo), se obtuvo mejor resultado de R^2 con el algoritmo AdaBoost. Sin embargo en las demás métricas MSE y MAE el modelo de LightGBM obtiene levemente mejores prestaciones.

Tabla 10.1: Métricas de evaluación de los modelos construidos

Algoritmo	**Dataset**	**trees**	**learning_rate**	**R^2**	**MSE**	**RMSE**	**MAE**
Reg. Lineal	Completo	—	—	0.69901	28.64101	5.35173	4.06549
Reg. Lineal	Filtrado	—	—	0.20680	0.36999	0.60827	0.47539
AdaBoost	Completo	100	0.001	0.82506	16.64689	4.08006	2.10868
AdaBoost	Filtrado	100	0.005	0.7696	0.10748	0.32784	0.20688
LightGBM	Completo	300	0.0428	0.8535	13.937	3.7332	1.8017
LightGBM	Filtrado	800	0.0331	0.8046	0.0911	0.3019	0.1812

En ambos modelos se ha utilizado la técnica de Ensamble de Árboles de Decisión. Se observa que utilizando AdaBoost con una cantidad de 100 árboles ya se alcanzó los resultados óptimos. Mientras que con el algoritmo LightGBM se requirió en un caso de 300 y en el otro de 800 Árboles de Decisión.

Sección 11

Conclusiones y trabajos futuros

El problema a resolver consistió en predecir la probabilidad de colisión entre dos objetos o cuerpos que orbitan alrededor de la Tierra (satélites) y se encuentran en una trayectoria de conjunción o encuentro. En síntesís, se trata de predecir la probabilidad de que dos satélites o un satélite y chatarra choquen entre sí. Se presentó a Kessler y Cour-Palais quien fue uno de los pioneros del área de la Chatarra Espacial (Kessler y Cour-Palais, 1978) y se mencionó algunos de sus conceptos más interesantes como Cinturón de Chatarra (Debris Belt).

El problema actual que tienen las empresas que operan satélites es que reciben numerosos reportes de alerta de colisión CDM. Algunos de estos reportes requieren realizar una maniobra disuasiva de colisión y otros no, eso lo define cada empresa en función de sus propios protocolos. Es importante resaltar que para un mismo evento de colisión se reciben más de un CDM, siendo siempre el último aquel cuyos valores de probabilidad de colisión y distancia mínima en el TCA (time of closest approach) tiene menos incertidumbre. Esto significa que siempre el último CDM es el que contará con la información más precisa y la secuencia de CDM de un mismo evento a lo largo de los días permite entender si el evento se está haciendo más riesgoso o no. Entonces, el problema que se aborda en este trabajo consiste en predecir la probabilidad de colisión del próximo CDM (aquel que el Ente de Monitoreo aún no ha publicado) con el objeto de tomar decisiones basadas en datos y agregándole valor a través del Machine Learning. El alcance del TFI, se limita esctrictamente a todo lo referente a los Modelos de Aprendizaje Automático. Quedó totalmente fuera del alcance el diseño de la maniobra disuasiva de colisión propiamente dicha, ya que eso corresponde al campo de la Dinámica Orbital y no de la Ciencia de Datos.

La Sección 9 Experimentación es la que permite observar la aplicación de las Técnicas de Machine Learning al dominio del problema y es una de las más extensas de este TFI. En esta sección primero se presenta el conjunto de datos de entrada el cual fue publicado por la ESA. Estos datos son accesibles al público en general ya que la ESA los puso a disposición en el marco de competiciones abiertas para estudiantes e investigadores a nivel mundial. En particular, este set de datos se obtuvo de la página web Kelvin destinada a estos concursos en la sección “Collision Avoidance Challenge” (ESA, 2019). Después de presentar el dataset con el

que se trabajó, se realizó una análisis de datos exhaustivo, en el cual se analizó cada una de las variables que se encuentran en un CDM. Particularmente, resultaron de interés la probabilidad de colisión (PC), la distancia entre ambos cuerpos en el TCA (MD) y el tiempo restante al TCA (time_to_tca). Aquí también se presentó el criterio utilizado para definir si una probabilidad de colisión requiere realizar una maniobra. En la Subsección 3 Preparación de los datos lo primero que se detalló fue el procesado de Extracción, Transformación y Carga de Datos (ETL) realizado. Para destacar en esta Sección de conclusiones se puede decir que la variable PC de un reporte CDM estándar está en formato decimal y resulta conveniente transformarla a escala logarítmica como se hizo en este TFI. En esta Subsección también se crearon nuevas variables a partir de las más sobresalientes mencionadas en el párrafo anterior. Básicamente las nuevas variables generadas en el Feature Engineering tuvieron como objetivo vincular reportes CDM de un mismo evento de colisión. Luego se enumeró cada una de las variables que fueron utilizadas en el modelo como parámetros de entrada y se indicó que la variable a predecir que fue denominada $TARGET\_PC$, que hace referencia a la probabilidad de colisión del próximo CDM. Se decidió abordar el TFI con dos subconjuntos de datos. Al primero se lo llamó dataset completo y comprende todos los registros que quedaron luego del ETL. Al segundo, se lo denominó dataset filtrado y comprende todos los registros cuya probabilidad de colisión es mayor o igual a -6. Para concluir con la Subsección 3 Preparación de los datos se detalló que tanto el dataset completo como el filtrado fueron divididos un 70 % para entrenamiento y el 30 % restante quedó para la evaluación del modelo.

En la Subsección 4 Modelado se seleccionó las Técnicas de Aprendizaje Automático que se utilizaron en el TFI:

- Regresión Lineal Múltiple
- AdaBoostRegressor
- Light Gradient Boosting Machines (LightGBM)

Se aplicó ajustó y aplicó cada uno de los Modelos tanto al conjunto de datos completo como al filtrado.

En primer lugar, se aplicó el Modelo de Regresión Lineal Múltiple, para ello se ajustaron los parámetros de pendiente y término independiente. De la Tabla 10.1 se concluyó que la métrica de error cuadrático medio MSE está alrededor de un 72 % por encima del MSE obtenido aplicando el modelo AdaBoost al dataset completo. Para el caso del dataset filtrado la comparativa a través del MSE es aún peor, la Regresión Lineal Múltiple tiene un MSE 3.45 veces mayor que el MSE logrado con AdaBoost. Si en vez, de compararlo con AdaBoost, se lo compara con LightGBM, se obtienen mayores diferencias en estas métricas. El R^2 de la Regresión Lineal ajustada para el dataset completo es de 0.70 y para el conjunto filtrado 0.20. Contrastando el MSE obtenido a través de la Regresión Lineal Múltiple contra los Modelos No Lineales, se

puede deducir que en este TFI se trató un problema no lineal. De lo contrario, la Regresión Lineal Múltiple hubiese sido suficiente para predecir las probabilidades de colisión con un error aceptable. Aquí se comprobó que el MSE es alto y no es útil este Modelo para realizar predicciones.

También se abordó el problema con Modelos No Lineales. En particular se trabajó con Árboles de Decisión dentro del Área de Aprendizaje Conjunto (Ensemble Learning) aplicando la Técnica de Boosting a través de dos algoritmos en voga: AdaBoostRegressor y Light Gradient Boosting Machines (LightGBM).

Ambos algoritmos obtuvieron resultados aceptables de MSE como se presentó en la Tabla 10.1. Aplicando los Modelos al dataset completo se obtuvo para AdaBoost un MSE de 16.65 y para LightGBM de 13.94. Se observa que LightGBM es levemente superior en cuanto a esta métrica, aproximadamente un 20 %. Al aplicar los Modelos al dataset filtrado se obtuvo un MSE de 0.107 para AdaBoost y de 0.091 para LightGBM, es decir una diferencia del 18 % a favor del LightGBM.

En base a los datos señalados en el párrafo anterior, se puede concluir que ambos Modelos tanto AdaBoost como LightGBM aplicados al dataset filtrado permiten predecir la probabilidad de colisión del próximo CDM con un MSE menor a 0.11. Este valor de MSE permitiría realizar predicciones con un error aceptable para la toma de decisiones. Por lo tanto, sería factible incorporar alguno de estos dos Modelos como una herramienta para la evalución de eventos de conjunción a través de CDM. Aplicar estos Modelos No Lineales al dataset completo, tiene como resultados predicciones con un MSE de al menos 13.94, y en ese caso no sería conveniente tomar esas predicciones para tomar decisiones. Cabe destacar también que entre ambos Modelos, las métricas de LightGBM son levemente superiores. En el caso de tener que decidir entre alguno de estos dos algoritmos para esta problemática en particular, y basado en los resultados de este trabajo, sería conveniente optar por el Modelo LightGBM.

Como conclusión final de este TFI, se puede decir que los Modelos AdaBoost o LightGBM entrenados con un dataset con probabilidades de colisión acotadas al rango de interés (cercano al Área de Maniobra) son útiles para predecir probabilidades de colisión de futuros CDM.

Por último en lo que respecta a Trabajos Futuros, existe un Área del Machine Learning que sería interesante aplicar y comparar con los resultados obtenidos en este TFI: Redes Neuronales Recurrentes, en particular el tipo Long Short Term Memory (LSTM). La Técnica LSTM es aplicada por Acciarini y col. en su publicación (Acciarini y col., 2021) y resulta de particular interés en este dominio de predicción de eventos de colisión. Sería propicio investigar oportunidades de mejora dentro de la aplicación de esa técnica o con algún otro tipo de red neuronal que se considere conveniente.

Sección 12

Bibliografía

[1] Acciarini, Giacomo, Pinto, Francesco, Letizia, Francesca, Martínez-Heras, Jose, Merz, Klaus y Bridges, C.
"Kessler: a Machine Learning Library for Spacecraft Collision Avoidance".
En: *8th European Conference on Space Debris*. Abr. de 2021.
URL: `https://github.com/kesslerlib/kessler`.

[2] Akella, Maruthi R. y Alfriend, Kyle T.
"Probability of Collision Between Space Objects".
En: *Journal of Guidance Control Dynamics* 23.5 (sep. de 2000), págs. 769-772.
DOI: `10.2514/2.4611`.

[3] Black. *Black: the uncompromising Python code formatter Repositorio Oficial*. 2021.
URL: `https://github.com/psf/black`.

[4] Bokeh Development Team. *Bokeh: Python library for interactive visualization*. 2018.
URL: `https://bokeh.pydata.org/en/latest/`.

[5] CCSDS. *Conjunction Data Message*. Recommended Standard.
Washington DC, USA: Consultative Committee for Space Data Systems, 2013.

[6] Chan, F. K. "Spacecraft Collision Probability". En: *Aerospace Press*. 2008.

[7] Cobo, Juan, Sánchez-Ortiz, Noelia, Grande, Ignacio y Merz, Klaus. "CORAM: ESA's Collision Risk Assesment and Avoidance Manoeuvres Computation Tool".
En: *2nd IAA Conference on Dynamics and Control of Space Systems*. Mar. de 2014.

[8] Curtis, Howard D. *Orbital Mechanics for Engineering Students (Third Edition)*.
Third Edition. Boston: Butterworth-Heinemann, 2014. ISBN: 978-0-08-097747-8.
DOI: `https://doi.org/10.1016/B978-0-08-097747-8.05001-5`. URL:
`https://www.sciencedirect.com/science/article/pii/B9780080977478050015`.

[9] Drucker, Harris. "Improving regressors using boosting techniques". En: *ICML*. Vol. 97.
Citeseer. 1997, págs. 107-115.

[10] ESA. *ESA: Types of orbits*. 2020. URL: https://www.esa.int/Enabling_Support/Space_Transportation/Types_of_orbits.

[11] ESA. *Kelvin: Collision Avoidance Challenge*. 2019. URL: https://kelvins.esa.int/collision-avoidance-challenge/data/.

[12] ESA. *Safety and Security: Analysis and prediction*. 2021. URL: https://www.esa.int/Safety_Security/Space_Debris/Analysis_and_prediction.

[13] Harris, Charles R., Millman, K. Jarrod, Walt, Stéfan J. van der, Gommers, Ralf, Virtanen, Pauli, Cournapeau, David, Wieser, Eric, Taylor, Julian, Berg, Sebastian, Smith, Nathaniel J., Kern, Robert, Picus, Matti, Hoyer, Stephan, Kerkwijk, Marten H. van, Brett, Matthew, Haldane, Allan, Río, Jaime Fernández del, Wiebe, Mark, Peterson, Pearu, Gérard-Marchant, Pierre, Sheppard, Kevin, Reddy, Tyler, Weckesser, Warren, Abbasi, Hameer, Gohlke, Christoph y Oliphant, Travis E. "Array programming with NumPy". En: *Nature* 585.7825 (sep. de 2020), págs. 357-362. DOI: 10.1038/s41586-020-2649-2. URL: https://doi.org/10.1038/s41586-020-2649-2.

[14] Hastie, Trevor y Tibshirani, Robert. *The Elements of Statistical Learning*. 2°. New York, NY, USA: Springer, 2017.

[15] Hunter, J. D. "Matplotlib: A 2D graphics environment". En: *Computing in Science & Engineering* 9.3 (2007), págs. 90-95. DOI: 10.1109/MCSE.2007.55.

[16] James, Gareth, Witten, Daniela, Hastie, Trevor y Tibshirani, Robert. *An Introduction to Statistical Learning: with Applications in R*. Springer, 2013. URL: https://faculty.marshall.usc.edu/gareth-james/ISL/.

[17] Jupyter. *Proyecto Jupyter Página Oficial*. 2021. URL: https://jupyter.org/.

[18] Kessler, Donald J. y Cour-Palais, Burton G. "Collision frequency of artificial satellites: The creation of a debris belt". En: *Journal of Geophysical Research: Space Physics* 83.A6 (1978), págs. 2637-2646. DOI: https://doi.org/10.1029/JA083iA06p02637.

[19] Klinkrad, H. *Space Debris: Models and Risk Analysis*. Springer, ene. de 2006. ISBN: 978-3-540-25448-5. DOI: 10.1007/3-540-37674-7.

[20] Kummer, Nikolai y Najjaran, H. "Adaboost.MRT: Boosting regression for multivariate estimation". En: *Artificial Intelligence Research* 3 (oct. de 2014). DOI: 10.5430/air.v3n4p64.

[21] LightGBM. *LightGBM Página Oficial*. 2021. URL: https://lightgbm.readthedocs.io.

[22] López, Fernando. *Ensemble Learning: Bagging and Boosting*. 2021. URL: https://towardsdatascience.com/ensemble-learning-bagging-boosting-3098079e5422.

[23] McKinney, Wes. "Data Structures for Statistical Computing in Python".
En: *Proceedings of the 9th Python in Science Conference*.
Ed. por Stéfan van der Walt y Jarrod Millman. 2010, págs. 56-61.
DOI: 10.25080/Majora-92bf1922-00a.

[24] Merz, Klaus, Bastida Virgili, Benjamin, Braun, Vitali, Flohrer, Tim, Funke, Quirin, Krag, Holger y Lemmens, Stijn.
"Current Collision Avoidance service by ESA's Space Debris Office".
En: *7th European Conference on Space Debris*. ESA Space Debris Office, 2017.

[25] Metz, Sascha, Simon, Henrik y Letizia, Francesca. "Implementation and comparison of data-based methods for collision avoidance in satellite operations".
En: *8th European Conference on Space Debris*. ESA Space Debris Office, abr. de 2021.

[26] Nogueira, Fernando. *Bayesian Optimization Repositorio Oficial*. 2021.
URL: https://github.com/fmfn/BayesianOptimization.

[27] Patera, Russell. "General Method for Calculating Satellite Collision Probability".
En: *Journal of Guidance Control and Dynamics - J GUID CONTROL DYNAM* 24 (jul. de 2001), págs. 716-722. DOI: 10.2514/2.4771.

[28] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M. y Duchesnay, E.
"Scikit-learn: Machine Learning in Python".
En: *Journal of Machine Learning Research* 12 (2011), págs. 2825-2830.

[29] Plotly. *Plotly Página Oficial*. 2021. URL: https://plotly.com/.

[30] Python. *Python Página Oficial*. 2021. URL: https://www.python.org/.

[31] Seaborn. *Seaborn: statistical data visualization Página Oficial*. 2021.
URL: https://seaborn.pydata.org/.

[32] Starlink. *Starlink: Página Oficial*. 2022. URL: https://www.starlink.com/.

[33] Vallado, David A. y McClain, Wayne D.
Fundamentals of astrodynamics and applications.
4th ed. Vol. The space technology library. Microcosm Press, 2013.

[34] Wijaya, Yudha. *CRISP-DM Methodology For Your First Data Science Project*. 2021.
URL: https://towardsdatascience.com/crisp-dm-methodology-for-your-first-data-science-project-769f35e0346c.

[35] Wu, Songhao. *3 Best metrics to evaluate Regression Model*. 2020.
URL: https://towardsdatascience.com/crisp-dm-methodology-for-your-first-data-science-project-769f35e0346c.

Las páginas web fueron visitadas en la fecha 2022-03-01.

Anexos

1. Anexo A - Sistema de referencia espacial RTN

El marco de referencia espacial conocido bajo las siglas RTN (radial, tangencial y normal) es un sistema cuyo origen está centrado en la nave espacial (o satélite), siendo los versores del mismo:

- Radial: dirección formada por el satélite y el centro de la Tierra, cuyo sentido apunta hacia la nave.
- Tangencial (o Along-track): en el mismo plano de la órbita, perpendicular a la radial y en el sentido de avance de la trayectoria del cuerpo.
- Normal (o Cross-track): perpendicular al plano obsculador de la órbita del satélite alrededor de la Tierra y de acuerdo a terna derecha.

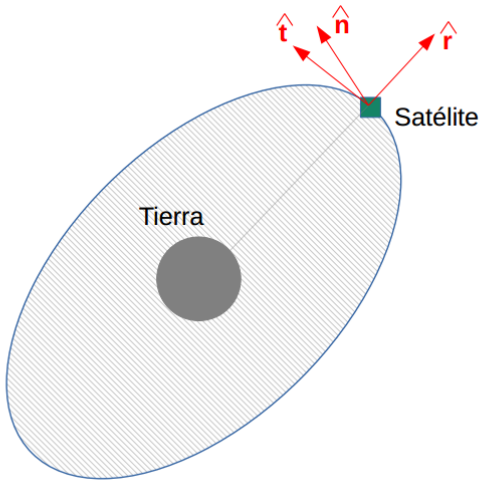

Figura 1: *Sistema de referencia RTN. Adaptación simplificada del diagrama presentado en Vallado y McClain (Vallado y McClain, 2013)*

Para más detalle sobre este sistema de referencia y su aplicación en la industria espacial se recomienda consultar Vallado y McClain (Vallado y McClain, 2013).

2. Anexo B - Métricas de evaluación

A continuación se define las métricas de evaluación utilizadas en los modelos de regresión a lo largo de este trabajo de investigación (Wu, 2020).

- Coeficiente de determinación R^2

$$R^2 = 1 - \frac{\sum_{i=1}^{n} (y_i - \hat{y}_i)^2}{\sum_{i=1}^{n} (y_i - \overline{y})^2} \tag{1}$$

- Error cuadrático medio MSE

$$MSE = \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 \tag{2}$$

- Raíz cuadrada del error cuadrático medio RSME

$$RMSE = \left(\frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 \right)^{\frac{1}{2}} \tag{3}$$

- Media del error absoluto MAE

$$MAE = \frac{1}{n} \sum_{i=1}^{n} |y_i - \hat{y}_i| \tag{4}$$

Siendo:

n cantidad total de registros del conjunto de datos usado en la iteración