Tecnológico de Monterrey

Campus:
Monterrey

Inteligencia Artificial Avanzada para la Ciencia de Datos (Gpo 102)

Curso:
TC3006C.102

Predicción de Supervivencia en el Desastre del Titanic: Enfoque y Evaluación de Modelos

Equipo 2:

Rafhael Eduardo Chavez Ramirez	A00832228
Jose David de la Garza Salas	A00834760
Pablo Andrés Martínez Sánchez	A01252489
Andre Sebastian Galindo Posadas	A00833376
Daniel Sánchez Villarreal	A01197699

Lugar y Fecha:
Monterrey, Nuevo León
31 de Agosto del 2024

Índice

Abstract

The tragic sinking of the RMS Titanic, resulting in the loss of thousands of lives, remains one of the most infamous maritime disasters in history. A significant factor contributing to this high fatality rate was the insufficient number of lifeboats available for all passengers and crew. Interestingly, survival rates varied among different groups, with women and children being given priority during the evacuation. This project utilizes the Titanic dataset to explore predictive modeling techniques for estimating passenger survival. The primary objective is to analyze the dataset, implement data cleaning, and perform feature engineering to prepare it for effective model training. Various machine learning models, including logistic regression, decision trees, and Random Forest, are employed to classify passenger survival outcomes. The models are evaluated using performance metrics such as accuracy, precision, recall, and F1 Score to determine their effectiveness. Through this analysis, the project provides valuable insights into the application of machine learning techniques for solving binary classification problems, emphasizing the critical role of feature selection and algorithm choice in predictive accuracy.

Keywords: Titanic dataset, survival prediction, machine learning, logistic regression, decision trees, Random Forest, feature engineering, model optimization, binary classification.

1. Introducción

El hundimiento del RMS Titanic es uno de los desastres marítimos más famosos de la historia, ocurrido en 1912 durante su viaje inaugural. El viaje inició desde Southampton, Inglaterra, y el destino final era Nueva York, Estados Unidos, pero en la noche del 14 de abril, el Titanic colisionó con un iceberg en el Atlántico Norte, lo que resultó en su hundimiento en las primeras horas de la mañana del 15 de abril. De los aproximadamente 2,224 pasajeros y tripulantes a bordo, más de 1,500 personas perdieron la vida, lo que convierte a este evento en uno de los desastres marítimos más mortales en tiempos de paz.

En el ámbito del análisis de datos y el aprendizaje automático, el desastre del Titanic presenta un caso de estudio desafiante. Al trabajar con los datos disponibles, es posible explorar la relación entre diversas características personales de los pasajeros y las circunstancias del viaje con la probabilidad de supervivencia, y estas características pueden ser utilizadas para entrenar modelos predictivos que determinen la probabilidad de que un pasajero determinado haya sobrevivido a la tragedia.

2. Definición del problema

El objetivo principal de este reto es desarrollar y construir un modelo predictivo altamente eficaz que sea capaz de determinar con precisión si un pasajero específico sobrevivió o no al hundimiento del Titanic. Este desafío está fundamentado en el análisis de un conjunto de datos históricos, que contiene información de los pasajeros a bordo del Titanic durante su viaje inaugural en abril de 1912. Estos datos incluyen una variedad de características y atributos relevantes de los pasajeros, tales como su edad, género, clase en la que viajaban, estado civil, número de familiares a bordo, tarifa pagada por el billete, puerto de embarque, etc.

Para este problema, se utilizará un enfoque de aprendizaje automático, lo que implica la creación de un modelo que pueda aprender patrones relevantes a partir de nuestros datos disponibles y, por lo tanto, predecir la supervivencia de los pasajeros. Esto se logrará entrenando el modelo con un subconjunto de datos de entrenamiento, que incluirá ejemplos de pasajeros tanto sobrevivientes como no sobrevivientes, permitiendo que el modelo pueda identificar características y combinaciones de características que estén asociadas con una mayor o menor probabilidad de supervivencia. Posteriormente, el modelo será evaluado utilizando un conjunto de datos de prueba para validar su capacidad de generalización y precisión.

3. Origen del Dataset

El dataset utilizado en este proyecto proviene de una competencia de la página Kaggle: https://www.kaggle.com/competitions/titanic y está basado en registros reales del Titanic. Este conjunto de datos es un estándar en la comunidad de ciencia

de datos para practicar técnicas de limpieza de datos, análisis exploratorio y modelado predictivo.

El dataset se divide en dos archivos:

- train.csv: Contiene los datos de entrenamiento con características y la variable objetivo (Survived).
- test.csv: Contiene características similares pero sin la variable objetivo, utilizado para probar el modelo.

4. Exploración del Dataset crudo

Contenido del Dataset

Nombre	Descripción	Tipo	Valores
PassengerId	Identificador único de cada pasajero.	No categórica	1, 2, 3, …, 891
Survived	Indica si el pasajero.	Categórica	sobrevivió (1) no sobrevivió (0)
Pclass	Clase del boleto del pasajero	Categórica	1 = 1ra clase, 2 = 2da clase 3 = 3ra clase
Name	Nombre completo del pasajero.	Textual	"Braund, Mr. Owen Harris", "Heikkinen, Miss. Laina" …
Sex	Sexo del pasajero.	Categórica	"male", "female"
Age	Edad del pasajero	No categórica	22, 38, 26, nan (valores faltantes) …
SibSp	Número de hermanos o cónyuges a bordo.	No categórica	0, 1, 2, 3
Parch	Número de padres o hijos a bordo.	No categórica	0, 1, 2, 3
Ticket	Número del boleto.	Categórica	"A/5 21171", "PC 17599" …

Fare	Tarifa pagada por el boleto.	No categórica	7.25, 71.2833, 8.05 …
Cabin	Número de cabina.	Categórica	"C85", "B28", nan (valores faltantes), …
Embarked	Puerto de embarque	Categórica	C = Cherburgo, Q = Queenstown, S = Southampton).

Tabla 1: En esto se muestran la información que presenta cada una de las columnas

PassengerId	Survived	Pclass	Name	Sex	Age	SipSp	Parch	Ticket	Fare	Cabin	Embarked
1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
2	1	1	Cumings, Mrs. John Bradley (Florence Briggs)	female	38.0	1	0	PC 17599	71.2833	C85	C
3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S

Tabla 2: En esta es la tabla del archivo train.csv y se enseña por cuantas columnas y filas está hecho

PassengerId	Pclass	Name	Sex	Age	Sip Sp	Parch	Ticket	Fare	Cabin	Embarked
892	3	Kelly, Mr. James	male	34.5	0	0	330911	7.8292	NaN	Q
893	3	Wilkes, Mrs. James (Ellen Needs)	female	47.0	1	0	363272	7.0000	NaN	S
894	2	Myles, Mr. Thomas Francis	male	62.0	0	0	240276	9.6875	NaN	Q
895	3	Wirz, Mr. Albert	male	27.0	0	0	315154	8.6625	NaN	S
896	3	Hirvonen, Mrs. Alexander (Helga E Lindqvist)	female	22.0	1	1	3101298	12.2875	NaN	S

Tabla 3: En esta es la tabla del archivo test.csv y se enseña por cuantas columnas y filas está hecha

- **Survived**: Indica si el pasajero sobrevivió (No se usa como característica, es el objetivo a predecir).

- **Pclass**: Clase de boleto del pasajero (Útil, refleja el estatus socioeconómico y acceso a recursos).

- **Nombre:** Pasajero (No útil directamente, no aporta información relevante para la predicción).

- **Sex**: Género del pasajero (Útil, influye en las probabilidades de supervivencia, dado que mujeres y niños tuvieron prioridad).

- **Age**: Edad del pasajero (Útil, la edad puede afectar las probabilidades de supervivencia).

- **SibSp**: Número de hermanos/cónyuges a bordo (Útil, indica si el pasajero viajaba en familia, lo que podría influir en el rescate).

- **Parch**: Número de padres/hijos a bordo (Útil, similar a **SibSp**, puede influir en la ayuda recibida durante el rescate).

- **Ticket**: Número del boleto (No útil directamente, no aporta información relevante para la predicción).

- **Fare**: Tarifa pagada por el boleto (Útil, correlacionada con la clase social y, por ende, con la probabilidad de supervivencia).

- **Cabin**: Número de cabina (Depende, podría ser útil para conocer la ubicación en el barco, pero suele estar incompleta).

- **Embarked**: Puerto de embarque (Útil, podría estar relacionado con la clase social o nacionalidad, lo que influye en la supervivencia).

Valores crudos:

- En el dataset de entrenamiento (train.csv), se observaron valores faltantes en las columnas Age y Cabin.
- En la columna Age, hay 177 valores faltantes (19.86% de los datos). En Cabin, faltan datos en un 77.1% de los registros.

5. Limpieza y transformación

El primer paso en la limpieza de los datos consistió en identificar las columnas con mayor cantidad de valores ausentes (NaN), esto se muestra en la **tabla 4**.

Nombre	Descripción
PassengerId	0
Survived	0
Pclass	0
Name	0
Sex	0
Age	177
SibSp	0
Parch	0
Ticket	0
Fare	0
Cabin	687
Embarked	2

Tabla 4: Esta tabla presenta la cantidad de datos faltantes (NaN) en cada columna del archivo train.csv.

En la tabla se observó que las columnas "Cabin" y "Age" son las que contienen más valores faltantes. Específicamente, la columna "Cabin" presenta 687 datos faltantes de un total de 891 registros, lo que representa aproximadamente el 77% de los datos. Debido a la alta proporción de valores ausentes, se decidió eliminar esta columna del análisis.

Por otro lado, la columna "Age" tiene 177 valores faltantes sobre un total de 891, lo que equivale al 19.86% de los datos. Para abordar este problema, se consideraron dos opciones: la imputación de datos o la eliminación de la columna. Tras evaluar el

rendimiento del modelo en ambos escenarios, se seleccionará la opción que ofrezca una mayor precisión. Para la imputación de los datos faltantes de la columna de edad, se utilizó el método de imputación de K-Nearest Neighbors (KNN) utilizando el imputador de KNN de la librería scikit-learn. Este método imputa valores faltantes basándose en la similitud con otros registros y se considera especialmente útil cuando hay patrones complejos en los datos. En este dataset hay una menor proporción de datos faltantes en la columna de edad comparado con el hecho de que están presentes la gran mayoría (prácticamente el 80%), implicando con ello justamente el hecho de que puedan existir patrones complejos en los datos que sí se registraron en esa columna. Es por esto que este método de imputación con KNN se consideró apropiado para llenar los valores faltantes de esta columna. Tras aplicar este método pudimos ver una mejora en la configuración, aplicación y desempeño de los modelos.

Adicionalmente, se identificaron columnas con información irrelevante para el modelo, como "PassengerID", que únicamente indica el número de identificación de cada pasajero en la base de datos, sin aportar información relevante para la predicción. Otras columnas eliminadas son "Ticket", la cual no ofrece información sobre la supervivencia de los pasajeros; y la columna "Fare", la cual tampoco contribuye significativamente a la predicción de la supervivencia de los viajeros en el accidente del Titanic. La **tabla 5** muestra el resultado de esto.

Survived	**PClass**	**Name**	**Sex**	**Age**	**SibSp**	**Parch**	**Embarked**
0	3	Braund, Mr. Owen Harris	male	22.0	1	0	S
1	1	Cumings, Mrs. John Bradley (Florence Briggs	female	38.0	1	0	C
1	3	Heikkinen , Miss. Laina	female	26.0	0	0	S
1	1	Futrelle, Mrs. Jacques Heath	female	35.0	1	0	S

		(Lily May Peel)					
0	3	Allen, Mr. William Henry	male	35.0	0	0	S

Tabla 5: Esta tabla muestra los datos resultantes tras la limpieza en los archivos train.csv y test.csv.

Una modificación adicional realizada fue la transformación de la columna "Sex" en valores booleanos para simplificar el análisis del modelo. Se reasignaron los valores 0 para representar al sexo masculino y 1 para el sexo femenino, eliminando la columna original "Sex". De esta manera, la mayoría de los datos analizados por el sistema son numéricos. Después de todas las modificaciones, la tabla final se presenta de la siguiente forma en la **tabla 6**.

Survived	**PClass**	**Name**	**Sex**	**SibSp**	**Parch**	**Embarked**	**Sex Bool**
0	3	Braund, Mr. Owen Harris	male	1	0	S	0
1	1	Cumings, Mrs. John Bradley (Florence Briggs	female	1	0	C	1
1	3	Heikkine n, Miss. Laina	female	0	0	S	1
1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	1	0	S	1
0	3	Allen, Mr. William Henry	male	0	0	S	0

Tabla 6: Esta tabla muestra los datos finales sin elementos redundantes en los archivos train.csv y test.csv.

5.1 Procesamiento de datos

Dado que muchos algoritmos de aprendizaje automático no pueden trabajar directamente con variables categóricas, ya que esperan entradas numéricas, se utilizó el método de One-Hot Encoding para convertir estas variables en un formato adecuado. A diferencia de la asignación de números enteros a las categorías (por ejemplo, "S" = 1, "Q" = 2, "C" = 3), el One-Hot Encoding evita que el modelo asuma un orden o jerarquía entre las categorías. En nuestro caso, la columna "Embarked" clasifica las instancias en tres categorías según el puerto de embarque: S, C o Q. Decidimos utilizar One-Hot Encoding en este feature debido a su practicidad y facilidad para el procesamiento de datos. En la gráfica vemos el resultado de nuestro dataset despues de aplicar el método de one hot encoding en el feature de Embarked:

Survived	**PClass**	**Name**	**Sex**	**SibSp**	**Parch**	**Sex Bool**	**Emb_C**	**Emb_Q**	**Emb_S**
0	3	Braund, Mr. Owen Harris	male	1	0	0	0	0	1
1	1	Cumings, Mrs. John Bradley (Florence Briggs	female	1	0	1	1	0	0
1	3	Heikkinen, Miss. Laina	female	0	0	1	0	0	1
1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	1	0	1	0	0	1
0	3	Allen, Mr. William Henry	male	0	0	0	0	0	1

Tabla 7: Esta tabla muestra los datos finales sin elementos redundantes en los archivos train.csv y test.csv.

6. Estructura final

Survived	PClass	Name	Sex	SibSp	Parch	Sex Bool	Emb_C	Emb_Q	Emb_S
0	3	Braund, Mr. Owen Harris	male	1	0	0	0	0	1
1	1	Cumings, Mrs. John Bradley (Florence Briggs	female	1	0	1	1	0	0
1	3	Heikkinen, Miss. Laina	female	0	0	1	0	0	1
1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	1	0	1	0	0	1
0	3	Allen, Mr. William Henry	male	0	0	0	0	0	1

7. Selección, configuración y entrenamiento

Cada modelo de aprendizaje funciona de manera diferente, esto implica que cada modelo es capaz de identificar patrones, aprender de los datos y hacer predicciones de manera distinta, en base este concepto es por lo que usualmente antes de escoger un modelo para trabajar se hace una comparación entre los modelos más certeros, conforme su funcionalidad con nuestros datos.

Los modelos tienen como métricas:

- Score de exactitud: Proporción de predicciones correctas sobre el total de predicciones. Es útil cuando las clases están balanceadas, ya que da una idea general de qué tan bien está funcionando el modelo en general. Sin embargo, puede no ser suficiente por sí solo si tienes un desequilibrio en las clases.

- Score de precisión: Proporción de verdaderos positivos entre todos los casos clasificados como positivos. Es relevante porque estamos más interesados en minimizar el número de falsos positivos, es decir, las predicciones de supervivencia que en realidad no sobrevivieron. Puede ser importante si el costo de una predicción incorrecta es alto.

- Score f1: Media armónica de precisión y recall. Dado que el problema del Titanic es una clasificación binaria y puede tener un desbalance en las clases (más pasajeros no sobrevivieron que sobrevivieron), el F1 Score ayuda a equilibrar la precisión y el recall, dándote una métrica que considera ambos tipos de errores.

- Score de recall: Proporción de verdaderos positivos entre todos los casos realmente positivos. Es importante para asegurarnos de identificar la mayor cantidad posible de pasajeros que realmente sobrevivieron. En algunos contextos, es crucial no pasar por alto a los pasajeros que sobrevivieron.

- Matriz de confusión: Tabla que muestra el conteo de verdaderos positivos, falsos positivos, verdaderos negativos y falsos negativos. Nos da una visión clara de los errores específicos que está cometiendo el modelo. Podemos ver cuántos verdaderos positivos, falsos positivos, verdaderos negativos y falsos negativos hay, lo cual es útil para entender mejor los resultados y ajustar tu modelo.

7.1 Selección de modelos

7.1.1 **Modelo de regresión logística**

La regresión logística es un modelo ampliamente utilizado para problemas de clasificación binaria, cómo predecir si un pasajero del Titanic sobrevivió o no. Su principal ventaja radica en su capacidad para estimar probabilidades de pertenencia a una clase, en este caso, la probabilidad de que un pasajero haya sobrevivido. Este enfoque permite no solo clasificar, sino también obtener una medida de confianza en cada predicción, lo cual es útil en situaciones donde la probabilidad de un resultado es tan importante como la clasificación misma.

Este modelo asume una relación lineal entre las características del dataset y el logaritmo de las probabilidades de la clase objetivo. Aunque la regresión logística requiere que las características sean numéricas, se pueden transformar variables categóricas como el sexo o la clase del pasajero en variables dummy, permitiendo al modelo capturar cómo influyen estas características en la probabilidad de supervivencia. Además, la regresión logística es menos propensa al sobreajuste en comparación con modelos más complejos, especialmente si se implementa con regularización para penalizar los coeficientes excesivamente grandes.

Otro punto a favor de la regresión logística es su simplicidad y rapidez de entrenamiento, lo que la convierte en una buena opción cuando se necesita obtener resultados rápidos y efectivos. También es fácil de interpretar, ya que los coeficientes resultantes indican la dirección y magnitud del impacto de cada característica en la probabilidad de supervivencia. Por ejemplo, un coeficiente positivo asociado con el sexo femenino indicaría que ser mujer aumenta la probabilidad de sobrevivir.

Sin embargo, la regresión logística puede verse limitada en situaciones donde las relaciones entre las características y la variable objetivo no son lineales, o donde hay interacciones complejas entre múltiples características. A pesar de estas limitaciones, sigue siendo una opción sólida para problemas de clasificación binaria cuando se busca un modelo interpretable, rápido de entrenar y que proporcione estimaciones de probabilidad fiables.

En el modelo de regresión logística que hemos definido, se están utilizando los siguientes hiper parámetros clave:

- **max_iter:** Este hiperparámetro establece el número máximo de iteraciones que el algoritmo de optimización (el método de descenso de gradiente) puede realizar para converger a una solución. Valor seleccionado: 2000

- **penalty:** Por defecto, la regresión logística en scikit-learn utiliza regularización L2, también conocida como "Ridge". Esta regularización penaliza los coeficientes grandes para prevenir el sobreajuste.Valor seleccionado: L2

- **solver:** El algoritmo por defecto utilizado para la optimización es lbfgs, un algoritmo que es adecuado para problemas pequeños a medianos y soporta la regularización L2. Valor seleccionado: 'lbfgs'

- **C:** Este parámetro controla la fuerza de la regularización. Un valor más alto de C reduce la regularización, mientras que un valor más bajo aumenta la regularización. Valor seleccionado: 1.528

7.1.2 Modelo de árboles de decisión

Un árbol de decisión es un modelo de aprendizaje supervisado que se utiliza tanto para tareas de clasificación como de regresión. La estructura del árbol se asemeja a un diagrama de flujo, donde cada nodo interno representa una condición o "decisión" basada en el valor de un atributo específico, cada rama representa el resultado de la condición, y cada hoja representa una etiqueta de clase o un valor final.

Un árbol de decisión es una de las opciones que elegimos para el problema de predicción de supervivencia en el Titanic debido a su facilidad de interpretación. Este modelo permite visualizar y entender claramente cómo se toman las decisiones para clasificar a los pasajeros. Por ejemplo, podemos ver cómo características como

la clase del pasajero, el sexo y la edad influyen en la predicción de si alguien sobrevivió o no.

Además, los árboles de decisión manejan bien tanto datos numéricos como categóricos, lo que es útil en un dataset como el del Titanic que contiene ambos tipos de características. Estos modelos son capaces de capturar interacciones complejas entre las características sin necesidad de que estas sean especificadas de antemano, lo que aumenta su flexibilidad en la modelización.

Otra ventaja significativa de los árboles de decisión es su robustez frente a outliers y datos faltantes. Estos modelos son menos sensibles a valores extremos y pueden manejar la ausencia de datos sin mayores problemas, lo que los hace adecuados para datasets que podrían no estar completamente limpios. Además, los árboles no asumen ninguna relación lineal entre las características y la variable objetivo, lo que los hace más adaptables en situaciones donde las relaciones entre los datos son no lineales.

Los árboles de decisión permiten un buen control sobre la complejidad del modelo, lo que ayuda a evitar el sobreajuste. Aunque pueden volverse inestables con muchos datos irrelevantes o ruido, son rápidos de implementar y ajustables para encontrar un equilibrio entre precisión y generalización.

Hiperparámetros Clave para el modelo:

- **criterion:** Define la función utilizada para medir la calidad de una división.el índice gini se usa para medir la pureza de los nodos. Valor seleccionado: gini

- **max_depth:** Especifica la profundidad máxima del árbol. Limitar la profundidad ayuda a evitar el sobreajuste, especialmente si el árbol tiende a volverse demasiado complejo. Valor seleccionado: 5

- **min_samples_split:** Determina el número mínimo de muestras necesarias para dividir un nodo. Valor seleccionado: 2

- **min_samples_leaf:** Define el número mínimo de muestras que debe tener un nodo hoja. Valor seleccionado: 5

7.1.3 Modelo de random forest

Seleccionar un modelo de Random Forest para la predicción de supervivencia de los pasajeros del Titanic es una opción que consideramos debido a su robustez y estabilidad. Random Forest combina los resultados de múltiples árboles de decisión, promediando sus predicciones o votando por la clase más frecuente. Esta combinación ayuda a reducir la varianza y la sensibilidad a fluctuaciones en los datos, proporcionando un modelo más robusto y menos propenso al sobreajuste en comparación con un solo árbol de decisión. Al promediar las predicciones de varios árboles, se obtiene un modelo que generaliza mejor en datos no vistos.

Además, Random Forest es eficaz para manejar datos complejos y no lineales. En el contexto del Titanic, esto significa que el modelo puede capturar las interacciones entre múltiples características, como la edad, la clase de pasaje y el sexo, que pueden influir en la supervivencia de los pasajeros. Los árboles individuales en el bosque pueden identificar patrones complejos que un solo árbol de decisión podría no captar adecuadamente.

Otra ventaja importante de Random Forest es su capacidad para proporcionar una medida de la importancia de cada característica en la predicción. Esto permite identificar qué factores son más relevantes para la supervivencia, facilitando la interpretación del modelo y la selección de características importantes. Esta información es valiosa para comprender mejor los factores que afectan la supervivencia y para realizar ajustes en el modelo si es necesario.

Random Forest también es robusto frente a datos faltantes y desbalance en las clases. La presencia de datos faltantes no afecta significativamente su rendimiento, ya que los árboles individuales manejan estos casos de manera independiente durante la construcción. Además, si el dataset presenta un desbalance en las clases, Random Forest puede ajustarse para manejarlo mediante el uso de pesos de clase o técnicas similares.

Hiperparámetros:

- **Profundidad del árbol (max_depth):** Limitar la profundidad ayuda a evitar árboles demasiado complejos, lo que reduce el riesgo de sobreajuste y mejora la generalización. Valor seleccionado: 10

- **Número de características (max_features):** Usar la raíz cuadrada del número total de características es un enfoque estándar que ayuda a mantener la diversidad entre los árboles del bosque. Valor seleccionado: 'sqrt'

- **Número mínimo de muestras por hoja (min_samples_leaf):** Asegura que cada hoja tenga suficientes muestras para hacer predicciones confiables. Valor seleccionado: 4

- **Número mínimo de muestras para dividir (min_samples_split):** Evita divisiones en nodos con muy pocas muestras, lo que también ayuda a reducir el sobreajuste. Valor seleccionado: 10

- **Número de árboles (n_estimators):** Un número razonable de árboles proporciona un buen equilibrio entre rendimiento y tiempo de entrenamiento. Valor seleccionado: 100

7. 2 Configuración de los modelos

7.2.1 Configuración de Regresión Logística

La regresión logística es un modelo lineal utilizado para tareas de clasificación binaria. La elección de hiperparámetros en la regresión logística afecta la rapidez de convergencia y la regularización del modelo, lo cual es muy importante para controlar el sobreajuste y mejorar la precisión de las predicciones. A continuación, se detalla cómo se seleccionaron estos hiperparámetros:

- **Número máximo de iteraciones (max_iter = 2000):** La regresión logística utiliza un algoritmo iterativo para ajustar sus parámetros. El valor de max_iter se fijó en 2000 para asegurar que el modelo tuviera suficiente tiempo para converger a una solución óptima. Durante la experimentación, se observó que valores más bajos (por ejemplo, 1000) a veces no permitían que el algoritmo convergiera, especialmente en casos donde los datos eran más complejos o presentaban relaciones no lineales. Al aumentar el número de iteraciones, se dio al modelo más oportunidad de encontrar los coeficientes óptimos, mejorando la precisión.

- **Penalización (penalty = 'l2'):** La penalización L2, se eligió para evitar que los coeficientes del modelo se volvieran excesivamente grandes, lo que podría indicar un ajuste excesivo a los datos de entrenamiento. La regularización L2 agrega una penalización basada en la suma de los cuadrados de los coeficientes, ayudando a mantener los valores de los parámetros bajo control. En tareas de clasificación binaria, L2 es comúnmente utilizada debido a su efectividad y eficiencia en la reducción del sobreajuste.

- **Algoritmo de optimización (solver = 'lbfgs'):** El solver 'lbfgs' (Limited-memory Broyden–Fletcher–Goldfarb–Shanno) es un algoritmo de optimización de segundo orden que es eficiente para problemas de regresión logística de tamaño medio a grande. Es conocido por su rápida convergencia y compatibilidad con la regularización L2. La elección de 'lbfgs' se basó en su capacidad para manejar de manera efectiva los datos del proyecto y su compatibilidad con los otros hiperparámetros seleccionados.

- **Fuerza de regularización (C = 1.528):** El parámetro C controla la fuerza de la regularización inversamente proporcional; es decir, valores más pequeños de C implican una regularización más fuerte. En nuestro caso, se utilizó una búsqueda en cuadrícula (Grid Search) junto con validación cruzada para encontrar el valor óptimo de C. Se evaluaron múltiples valores de C, y se encontró que 1.528 proporcionaba el mejor equilibrio entre el ajuste adecuado a los datos de entrenamiento y la generalización a nuevos datos.

Este valor de C permitió al modelo aprender las relaciones significativas sin ajustarse en exceso a los ruidos de los datos.

7.2.2 Configuraciones de Árboles de Decisión

Los árboles de decisión son modelos predictivos que dividen repetidamente los datos en subconjuntos basados en valores de características. Esta división se realiza mediante criterios que maximizan la ganancia de información, como el índice Gini o la entropía. La configuración de los hiperparámetros en un árbol de decisión es muy importante para encontrar un equilibrio entre el ajuste a los datos de entrenamiento y la capacidad de generalización. A continuación, se detallan los hiperparámetros seleccionados y el razonamiento detrás de cada uno:

- **Criterio (criterion = 'gini'):** El índice Gini fue seleccionado para medir la calidad de las divisiones. Este índice evalúa la impureza de un nodo y es efectivo para problemas de clasificación. En nuestro caso, se optó por Gini debido a su simplicidad y eficacia comprobada en tareas similares de clasificación. Comparado con la entropía, Gini suele ser menos costoso computacionalmente y, en muchos casos, ofrece resultados similares.

- **Profundidad Máxima (max_depth = 5):** Limitar la profundidad máxima del árbol a 5 nodos asegura que el modelo no se sobreajuste a los datos de entrenamiento. El sobreajuste ocurre cuando un modelo es demasiado complejo y empieza a capturar ruido en lugar de patrones significativos. Durante la experimentación, se probaron diferentes profundidades (por ejemplo, de 3 a 10). Se observó que una profundidad mayor a 5 no mejoraba significativamente la precisión del modelo en un conjunto de validación, pero aumentaba la complejidad y el riesgo de sobreajuste. Por lo tanto, max_depth=5 se consideró un compromiso adecuado entre precisión y generalización.

- **Número mínimo de muestras para dividir (min_samples_split = 2):** Este hiperparámetro controla el número mínimo de muestras necesarias para que un nodo se divida. Mantenerlo en el valor por defecto de 2 permite que el árbol crezca y divida nodos siempre que haya al menos dos muestras presentes. Esto es útil para capturar divisiones importantes, aunque en modelos con un número muy bajo de muestras, podría llevar a divisiones basadas en ruido. En nuestro caso, se encontró que mantener este valor estándar ayudaba a capturar más información relevante sin introducir demasiado ruido.

- **Número mínimo de muestras en una hoja (min_samples_leaf = 5):** Este parámetro define el número mínimo de muestras que un nodo hoja debe contener. Configurar min_samples_leaf=5 ayuda a suavizar el modelo, asegurando que cada nodo hoja represente una cantidad suficiente de datos para ser significativo. Valores más bajos llevarían a hojas con muy pocos datos, lo que podría hacer al modelo susceptible al sobreajuste. A través de pruebas, se encontró que un valor de 5 proporciona un buen equilibrio entre mantener nodos significativos y prevenir el sobreajuste.

Estos parámetros fueron ajustados utilizando validación cruzada, en la que se entrenó y evaluó el modelo con diferentes combinaciones de valores. La combinación seleccionada resultó en el mejor rendimiento promedio en los conjuntos de validación, balanceando adecuadamente la precisión y la generalización del modelo.

7.2.3 Configuración de Random Forest

El modelo de Random Forest utiliza una combinación de múltiples árboles de decisión para mejorar la precisión y la robustez de las predicciones. A través de la agregación de los resultados de múltiples árboles, Random Forest reduce la varianza del modelo, lo que lo hace menos susceptible al sobreajuste comparado con un solo árbol de decisión. Los siguientes hiperparámetros fueron seleccionados para optimizar el rendimiento del modelo:

- **Número de árboles en el bosque (n_estimators = 100):** El hiperparámetro ‘n_estimators’ define cuántos árboles de decisión se construyen en el modelo de Random Forest. Se seleccionó un valor de 100 árboles después de realizar experimentación con diferentes cantidades, como 10, 50, 100, y 200. Se observó que, a partir de 100 árboles, el rendimiento del modelo en términos de precisión se estabilizaba, y añadir más árboles no resultaba en mejoras significativas. Este número proporciona un buen equilibrio entre rendimiento y tiempo de entrenamiento.

- **Profundidad máxima de los árboles (max_depth = 10):** Establecer una profundidad máxima ayuda a limitar la complejidad de cada árbol individual en el bosque. A través de la validación cruzada, se determinó que una profundidad de 10 proporcionaba un buen compromiso entre capturar patrones significativos y evitar que los árboles se volvieran demasiado complejos y específicos del conjunto de entrenamiento. Valores más altos tendían a sobreajustar, mientras que valores más bajos no capturaban adecuadamente las relaciones en los datos.

- **Número máximo de características a considerar para la mejor división (max_features = 'sqrt'):** Configurar max_features como ‘sqrt’ implica que en cada nodo de cada árbol, sólo se considerará un subconjunto aleatorio de características igual a la raíz cuadrada del número total de características. Esta configuración introduce diversidad entre los árboles y reduce la correlación entre ellos, lo cual es esencial para mejorar la precisión del conjunto. Este enfoque es estándar y ha demostrado ser efectivo en múltiples aplicaciones.

- **Número mínimo de muestras en una hoja (min_samples_leaf = 4):** Este parámetro especifica el número mínimo de muestras que debe contener una hoja terminal en un árbol. Un valor de 4 ayuda a asegurar que las hojas no

sean excesivamente pequeñas, lo cual podría llevar a decisiones basadas en datos insuficientes. A través de la experimentación, se observó que este valor mejoraba la estabilidad de las predicciones, especialmente en casos donde los datos de entrada contenían ruido.

- **Número mínimo de muestras para dividir un nodo (min_samples_split = 10):** Este parámetro define el número mínimo de muestras necesarias para considerar la división de un nodo. Un valor de 10 evita que los nodos se dividan basándose en un número muy pequeño de muestras, lo que podría conducir a divisiones no significativas basadas en ruido. Se probó con valores más bajos y más altos, y 10 fue seleccionado por proporcionar un buen equilibrio entre la capacidad del modelo para capturar relaciones en los datos y la robustez frente al ruido.

Estos hiperparámetros fueron afinados utilizando una combinación de experiencia previa, pruebas empíricas y técnicas de búsqueda de hiperparámetros automatizadas, como la búsqueda en cuadrícula y la búsqueda aleatoria, en conjunto con la validación cruzada. El objetivo final era maximizar la precisión del modelo en los datos de prueba, asegurando al mismo tiempo que los modelos se generalizaran bien a nuevos datos.

7.3 Entrenamiento

Resultados de cada modelo y las métricas con las que comparamos en una tabla de donde se va a escoger un modelo.

Modelo/Métricas	Exactitud	Presición	Puntaje F1	Recall	Matriz de Confusión
Regresión Logistica (Quitando los NaN de Edad)	0.8391608392	0.8253968254	0.8188976378	0.8125	[68 11] [12 52]
Árboles de desiciones (Quitando los NaN de Edad)	0.7902097902	0.8148148148	0.7457627119	0.6875	[69 10] [20 44]
Random Forest (Quitando los NaN de Edad)	0.7902097902	0.8035714286	0.75	0.703125	[68 11] [19 45]
Regresión Logistica (Imputación por vecino más cercano)	0.7892376682	0.7011494253	0.7218934911	0.743902439	[115 26] [21 61]
Árboles de desiciones (Imputación por vecino más cercano	0.8071748879	0.8305084746	0.695035461	0.5975609756	[131 10] [33 49]
Random Forest (Imputación por vecino más cercano)	0.8324022346	0.8545454545	0.7580645161	0.6811594203	[102 8] [22 47]

El modelo de Regresión Logística aplicado tras la eliminación de los valores faltantes en la variable de edad ha demostrado ser el más consistente en términos de rendimiento general entre las opciones evaluadas. Este modelo no solo obtuvo la mayor exactitud (0.8392), sino que también mantuvo un equilibrio notable entre precisión (0.8254), recall (0.8125) y puntaje F1 (0.8189).

Este equilibrio en las métricas sugiere que el modelo logra un buen desempeño tanto en la identificación de verdaderos positivos como en la minimización de falsos positivos y negativos. La robustez y estabilidad de estos resultados indican que es el modelo con el que podemos trabajar para este reto.