Instituto Tecnológico de Estudios Superiores de Monterrey

Maestría:
Inteligencia Artificial Aplicada

Curso:
Análisis de Grandes Volúmenes de Datos

Proyecto Final:
Análisis de vuelos retrasados

Estudiantes:

Alexys Martín Coate Reyes	A01746998
Manuel Gerardo López Garza	A00821663
Diego Andres Bernal Diaz	A01795975
Annette Cristina Narváez Andrade	A00571041

Profesores:
Dr. Iván Olmos Pineda
Mtro. José Carlos Soto

Fecha de Entrega:
22 de junio del 2025

Tabla de Contenido

Resumen

El documento trata de creación un modelo de predicción de vuelos retrasados con base en el dataset *Airline Delay Analysis[1]* obtenido de kaggle. Este dataset tiene una gran cantidad de datos, por lo que se realizaron particiones para tomar una muestra del mismo. Con esta información se realizó un procesamiento y transformación del dataset para su posterior analisis con modelos como regresión logística, Random Forest, GBT y decision Trees. Se seleccionó el mejor modelo de los 4 y se procedió a utilizr un grid search, para obtener el mejor ajuste de hiperparámetros. Finalmente se obtienen las estadísticas finales obteniendo datos como matriz de confusión, curva PR, ROC, AUC para elegir el mejor modelo con los hiperparámetros más óptimos.

Enlace al código del proyecto: ***ProyectoFinal_Equipo19.ipynb***

Introducción

El análisis de demoras en vuelos es un tema de gran interés dentro del sector del transporte aéreo debido a su impacto económico, operativo y en la experiencia del pasajero. Cada año, millones de vuelos sufren retrasos por múltiples factores, desde condiciones meteorológicas hasta congestión en aeropuertos o problemas operativos. Estas demoras no solo representan una fuente de frustración para los pasajeros, sino también importantes pérdidas económicas para las aerolíneas y aeropuertos.

Con el auge del Big Data y el desarrollo de plataformas como PySpark, es posible procesar y analizar grandes volúmenes de datos históricos de vuelos para identificar patrones que permitan predecir si un vuelo se retrasará o no. Esta capacidad predictiva puede ser de gran utilidad tanto para aerolíneas como para pasajeros, ya que permite anticiparse a posibles incidencias y tomar decisiones informadas.

Este proyecto utiliza el conjunto de datos público "Airline Delay Analysis" disponible en Kaggle, el cual contiene información detallada de vuelos realizados en Estados Unidos, incluyendo horarios programados y reales, aerolíneas, aeropuertos, condiciones climáticas, entre otros atributos.

A partir de este dataset, se construyó una variable binaria denominada DELAYED, la cual toma el valor de 1 si el vuelo despegó con retraso y 0 si salió a tiempo, comparando la hora de salida efectiva (DEP_TIME) con la hora de salida programada (CRS_DEP_TIME). Para entrenar modelos supervisados robustos, se eliminaron columnas que no aportarían valor predictivo o que reflejan información posterior al vuelo.

Objetivo general

Desarrollar un modelo de aprendizaje supervisado utilizando PySpark que permita predecir si un vuelo sufrirá un retraso, a partir de variables disponibles antes del despegue.

Objetivos específicos

- Realizar una caracterización de la población y construir una muestra representativa a partir de los datos originales.
- Aplicar técnicas de muestreo y balanceo de clases según la distribución de la variable objetivo.
- Entrenar y comparar distintos modelos de clasificación supervisada sobre grandes volúmenes de datos.
- Evaluar los modelos utilizando métricas apropiadas como precisión, recall, F1-score y curva ROC.
- Visualizar los resultados obtenidos y discutir su relevancia en un contexto real.

Propuesta de solución

Descripción general

Para abordar el problema de predicción de retrasos en vuelos, se propone el desarrollo de modelos de aprendizaje automático supervisado utilizando el entorno **PySpark**, aprovechando su capacidad para el procesamiento distribuido de grandes volúmenes de datos. La idea central es construir un modelo que, a partir de características conocidas **antes del vuelo**, pueda predecir si un vuelo se retrasará o no, permitiendo una mejor planificación operativa por parte de las aerolíneas.

Se trabajó con el dataset **"Airline Delay Analysis"**, que contiene millones de registros de vuelos realizados en Estados Unidos. A partir de este conjunto de datos, se llevó a cabo un proceso riguroso de limpieza, transformación, muestreo, entrenamiento y evaluación de modelos, utilizando técnicas estándar del aprendizaje supervisado.

1. Caracterización de la población

La población original está compuesta por todos los vuelos registrados en el dataset, abarcando múltiples aerolíneas, aeropuertos y rutas. Cada registro representa un vuelo individual, con múltiples variables asociadas (fecha, horarios, aerolínea, origen/destino, etc.).

En la siguiente imagen se muestra el schema de la partición que agarramos nosotros, incorporando el tanto el nombre de la columna como el tipo de dato.

```
root
 |-- FL_DATE: date (nullable = true)
 |-- OP_CARRIER: string (nullable = true)
 |-- OP_CARRIER_FL_NUM: integer (nullable = true)
 |-- ORIGIN: string (nullable = true)
 |-- DEST: string (nullable = true)
 |-- CRS_DEP_TIME: integer (nullable = true)
 |-- DEP_TIME: double (nullable = true)
 |-- DEP_DELAY: double (nullable = true)
 |-- TAXI_OUT: double (nullable = true)
 |-- WHEELS_OFF: double (nullable = true)
 |-- WHEELS_ON: double (nullable = true)
 |-- TAXI_IN: double (nullable = true)
 |-- CRS_ARR_TIME: integer (nullable = true)
 |-- ARR_TIME: double (nullable = true)
 |-- ARR_DELAY: double (nullable = true)
 |-- CANCELLED: double (nullable = true)
 |-- DIVERTED: double (nullable = true)
 |-- CRS_ELAPSED_TIME: double (nullable = true)
 |-- ACTUAL_ELAPSED_TIME: double (nullable = true)
 |-- AIR_TIME: double (nullable = true)
 |-- DISTANCE: double (nullable = true)

Número total de registros: 18505725
```

Fig 1. printSchema() de la partición de datos utilizada, contiene el nombre y tipo de dato de la columna.

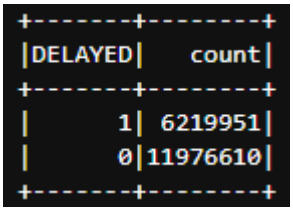
2. Recolección y descripción del dataset

Se utilizó el dataset disponible públicamente en Kaggle, titulado **"Airline Delay Analysis" [1]**, el cual presenta más de un millón de registros. Se seleccionaron únicamente aquellas columnas disponibles **antes del despegue del vuelo**, eliminando variables relacionadas a información que solo se conoce una vez que el vuelo ya ha sido operado (por ejemplo, tiempo real de arribo, demoras efectivas, etc.).

```
+----------+----------+------+----+------------+------------+--------+-------+
|   FL_DATE|OP_CARRIER|ORIGIN|DEST|CRS_DEP_TIME|CRS_ARR_TIME|DISTANCE|DELAYED|
+----------+----------+------+----+------------+------------+--------+-------+
|2016-01-01|        DL|   DTW| LAX|        1935|        2144|  1979.0|      0|
|2016-01-01|        DL|   ATL| GRR|        2125|        2321|   640.0|      1|
|2016-01-01|        DL|   LAX| ATL|        2255|         600|  1947.0|      1|
|2016-01-01|        DL|   SLC| ATL|        1656|        2229|  1590.0|      1|
|2016-01-01|        DL|   BZN| MSP|         900|        1216|   874.0|      1|
+----------+----------+------+----+------------+------------+--------+-------+
only showing top 5 rows
```

Fig. 2 Columnas seleccionadas con información antes del vuelo, a exepción de la variable de delayed.

Se analizaron la cantidad muestras que se tenían en la variable objetivo “DELAYED” y nos dimos cuenta de que estábamos lidiando con un problema de dataset desbalanceado.



DELAYED	count
1	6219951
0	11976610

Fig. 3 Cuenta de la variable objetivo.

3. Agregación de variables adicionales

Con los datos existentes de fecha se dedicó procesarlos para obtener los datos mejor divididos qudando las columnas resultantes: "DAY_OF_WEEK", "IS_WEEKEND", "MONTH", "SEASON".

Con la información de la hora de la salida se decidió realizar un análisis por hora quedando en las siguientes columnas: "DEP_HOUR" e "IS_PEAK_HOUR". (Los datos de hora pico se definieron entre los horarios de 6-9 y 15-19 que son los horarios pico estimados por nosotros).

4. Construcción de la muestra

Dado el volumen de datos, fue necesario construir una muestra manejable pero representativa para el entrenamiento y evaluación de modelos. Para ello:

- Se seleccionó un subconjunto balanceado entre vuelos "ON TIME" y "DELAYED".
- Se evitó el sesgo eliminando columnas irrelevantes o con alta correlación con la variable objetivo.
- Se aplicó transformación de variables categóricas (por ejemplo, aerolínea, aeropuerto) mediante técnicas de codificación apropiadas.

5. Estrategia de muestreo

Se aplicó una estrategia de **submuestreo estratificado**, asegurando una proporción equilibrada de clases en los conjuntos de entrenamiento y prueba. Esta técnica permite combatir el problema común de desbalance en datasets reales, donde los vuelos a tiempo suelen ser más frecuentes que los retrasados.

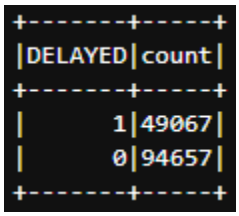


Fig. 4 Cuenta de la variable objetivo después de selección de muestra representativa. (Tamaño total de la muestra M: 143724)

6. Generación de conjuntos de entrenamiento y prueba

El conjunto de datos fue dividido en dos particiones:

- **80% para entrenamiento**
- **20% para prueba**

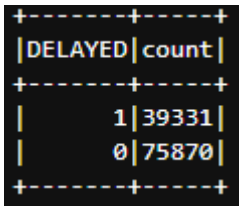


Fig. 5 Cuenta de la variable objetivo en el test data. (Total de registros: 115201)

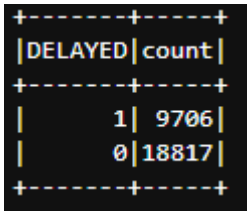
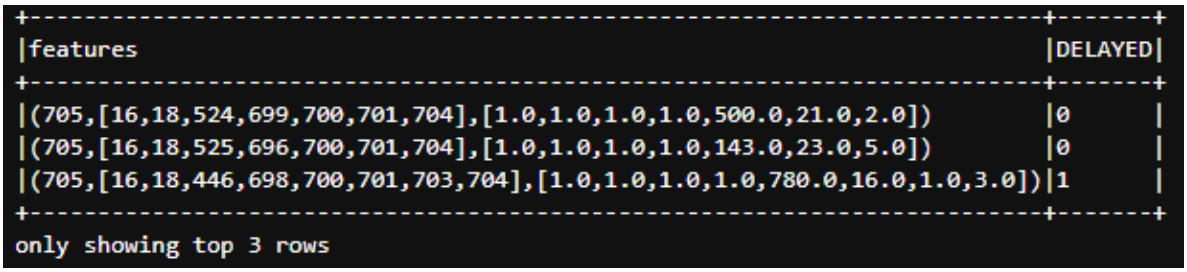


Fig. 6 Cuenta de la variable objetivo en el test data. (Total de registros: 28523)

7. Procesamiento en Pipeline

Antes de pasar los datos al modelo, los debemos preparar para que puedan ser interpretados por el modelo de mejor manera. Se estandarizaron variables numéricas, se aplicó codificación one-hot para variables categóricas, todo ello utilizando las herramientas del módulo `pyspark.ml`.

Posteriormente se realizó una vectorización de los valores no objetivo en “features” a través de un VectorAssembler().



```
+--------------------------------------------------------------------------+-------+
|features                                                                  |DELAYED|
+--------------------------------------------------------------------------+-------+
|(705,[16,18,524,699,700,701,704],[1.0,1.0,1.0,1.0,500.0,21.0,2.0])        |0      |
|(705,[16,18,525,696,700,701,704],[1.0,1.0,1.0,1.0,143.0,23.0,5.0])        |0      |
|(705,[16,18,446,698,700,701,703,704],[1.0,1.0,1.0,1.0,780.0,16.0,1.0,3.0])|1      |
+--------------------------------------------------------------------------+-------+
only showing top 3 rows
```

Fig. 7 Información codificada con VectorAssember

8. Selección de métricas

Para evaluar el rendimiento de los modelos, se utilizaron métricas clásicas de clasificación binaria:

- **Accuracy**
- **Precision**
- **Recall**
- **F1-score**
- **Área bajo la curva ROC (AUC)**

Estas métricas permiten tener una visión amplia del comportamiento de los modelos, especialmente en presencia de clases desbalanceadas.

9. Selección de algoritmos de aprendizaje

Se entrenaron y compararon varios algoritmos de aprendizaje automático, entre ellos:

- **Logistic Regression**
- **Random Forest**
- **Gradient-Boosted Trees**

Todos los modelos fueron entrenados con el módulo `pyspark.ml.classification`, y seleccionados por su robustez y rendimiento probado en tareas de clasificación binaria sobre datos tabulares.

10. Ajuste de hiperparámetros

Se aplicó un proceso de **validación cruzada k-fold** con búsqueda de hiperparámetros mediante `ParamGridBuilder` y `CrossValidator`, optimizando principalmente:

- Profundidad de árboles
- Número de árboles

- Tasa de aprendizaje (en GBT)
- Regularización (en regresión logística)

Este ajuste permitió mejorar significativamente la calidad del modelo final, evitando tanto el sobreajuste como el subajuste.

Experimentación y Resultados

Al comparar el desempeño de los diferentes tipos de obtuvimos los siguientes resultados:

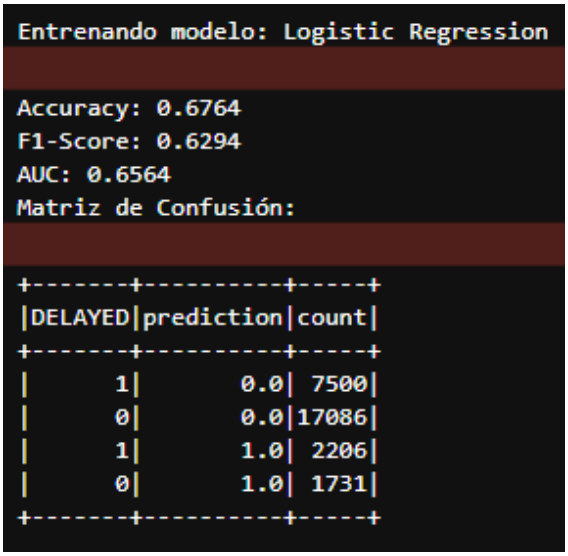


Fig. 8 Resultados del modelo de "LLogisistic Regregssion"

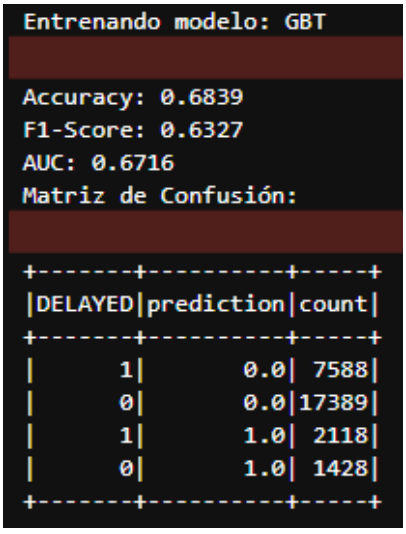


Fig. 9 Resultados del modelo de "GBT"

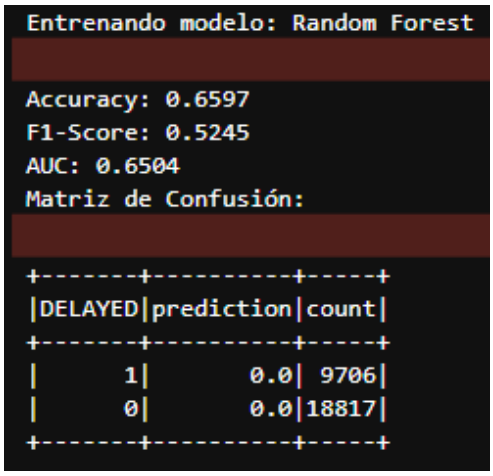


Fig. 10 Resultados del modelo de "Random Forest"

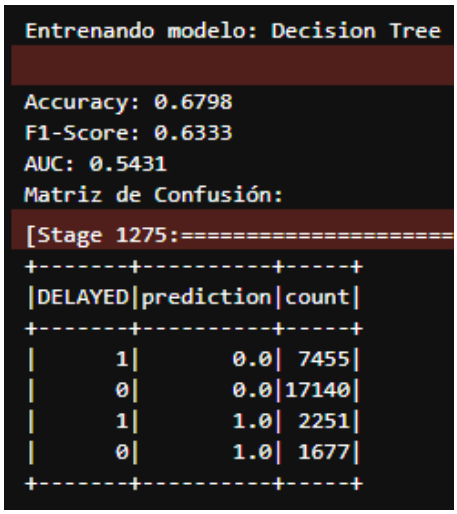


Fig. 11 Resultados del modelo de "Decision Tree"

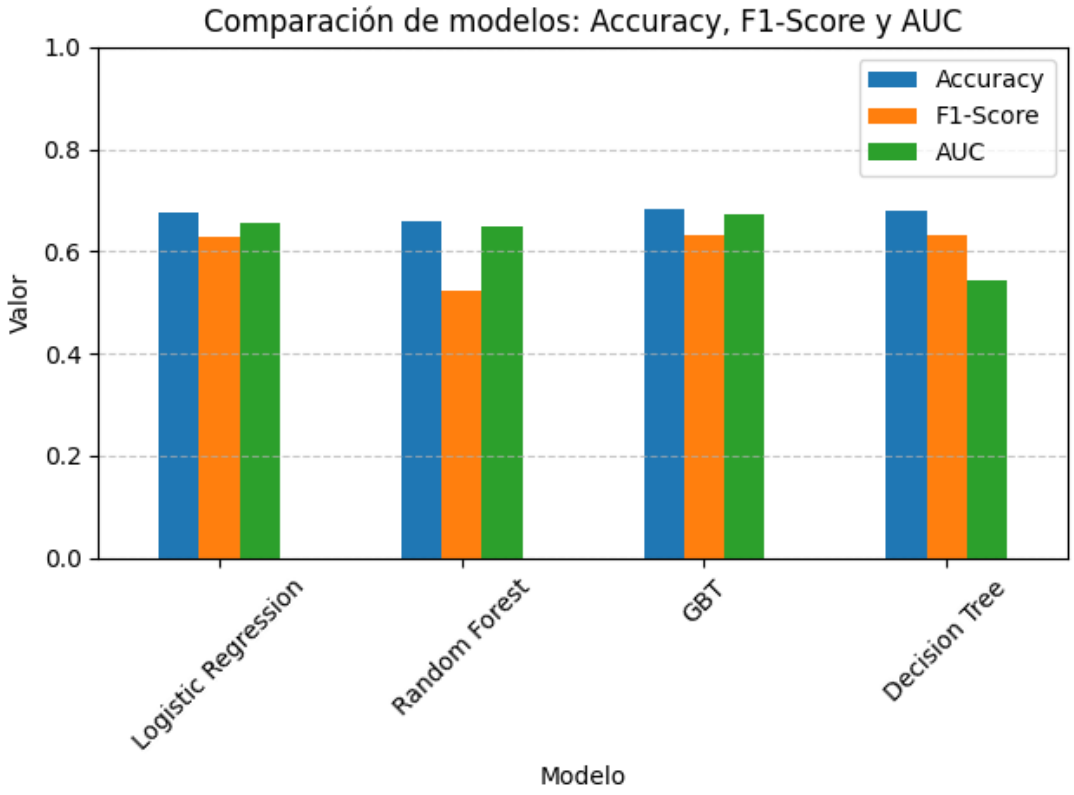


Fig. 12 Grafica comparativa de los 4 modelos, compparando las 3 métricas anteriores

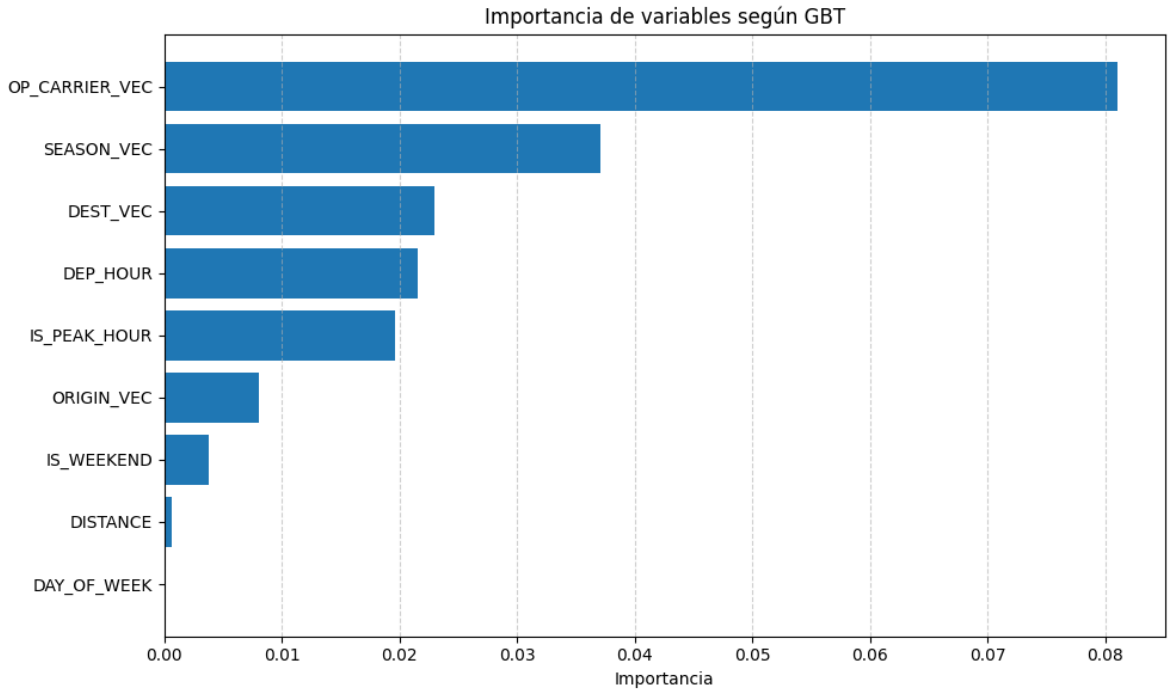


Fig. 13 Grafica del análisis de importancia de variables del GBT

Conforme a los resultados anteriores es evidente que el clasificador “GBT” fue el que tuvo mejor score, por lo que dicho clasificador se seleccionó para la etapa de ajuste de hiperparámemetros con la utilización de k-cross validation.

Al finalizar el cross validation, se analizaron los AUC de los mismos, para determinar cuales el mejor clasificador.

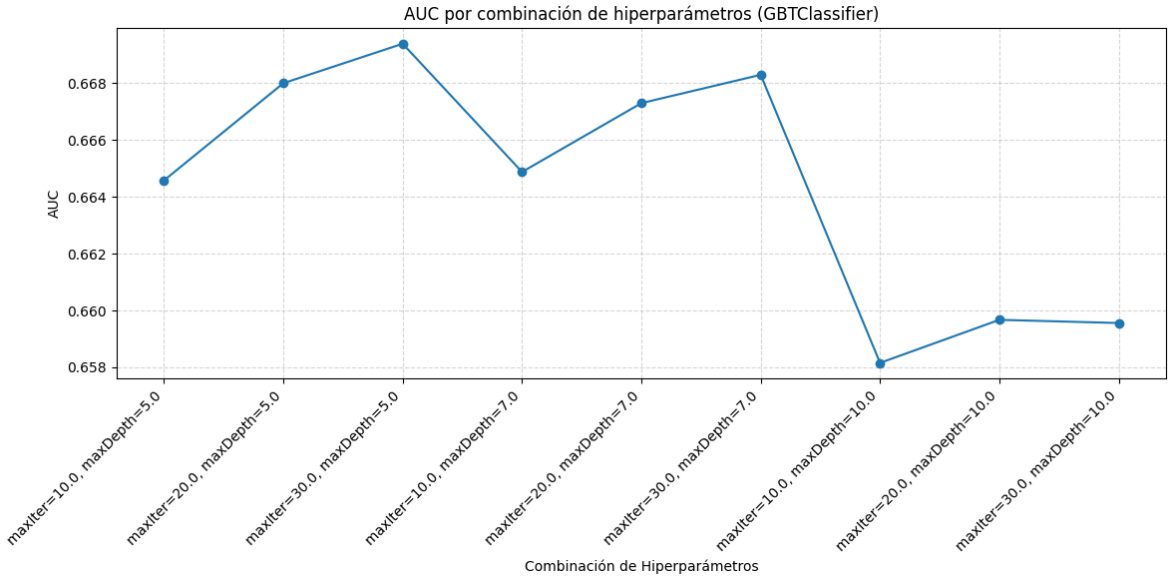


Fig. 14 Grafica análisis de AUC de los diferentes clasificadores GBT enntrenados.

Dada la imagen anterior, el clasificador con mayor puntaje fue el tercero que tiene como hiperparámetros:

- max_depth = 5
- max_iter = 30
- AUC: 0.669390

El AUC es un valor que nos indica que tan bueno es nuestro clasificador. Esta medida es el área bajo la curva de la gráfica ROC, que mientras esté más pegada a la esquina izquierda (1 en el eje de las “y”) será más cercano a un clasificador perfecto. En este caso nuestro clasificador tiene margen de mejora, pero está teniendo un mejor desempeño que un clasificador aleatorio (línea recta).

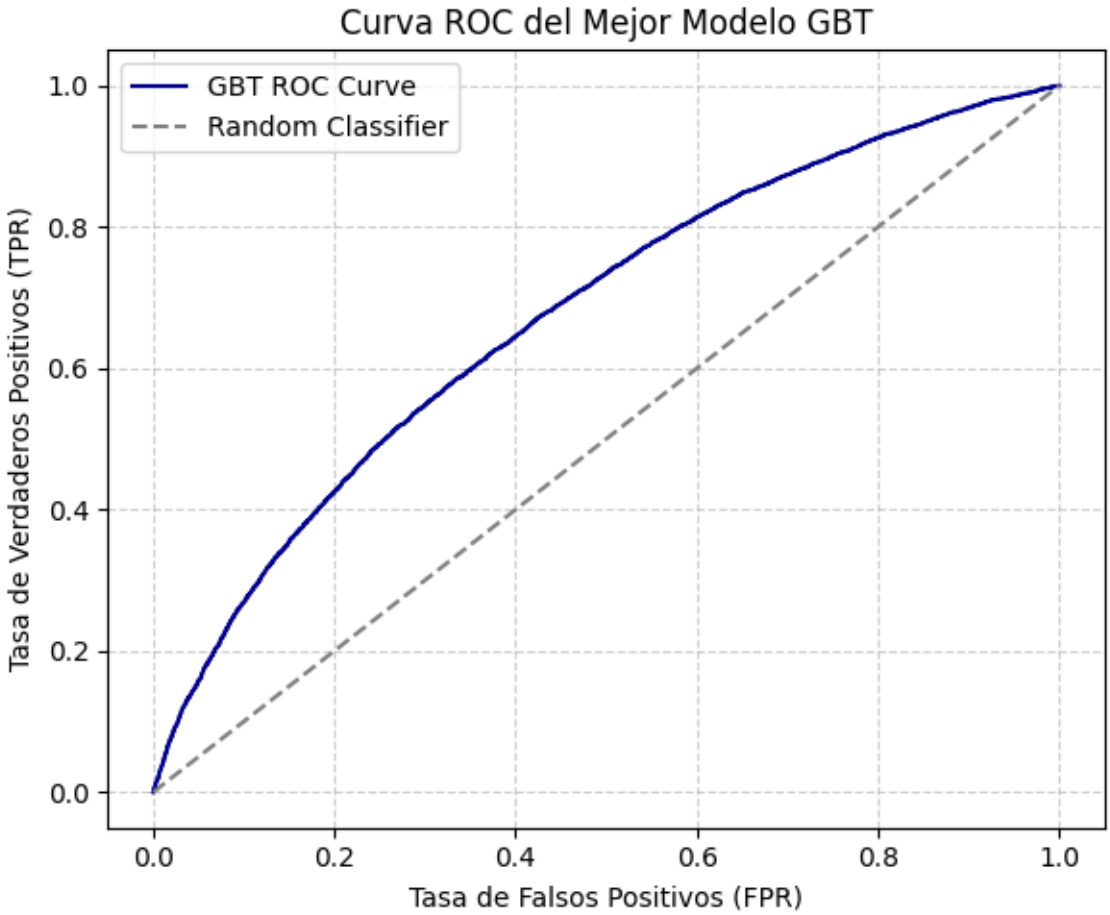


Fig. 15 Grafica ROC del mejor clasificador GBT

Así mismo se graficó la curva PR que nos muestra en el eje "x" el "Recall" contra el "Precision" en el eje "y". La grafica tiene un desempeño moderadodo cuando se tiene un treshold de alrededor de 0.7 a 0.6, posteriormente el valor baja. Para que sea un clasificador perfecto la curva debe estar más hacia la esquina derecha, por lo que no logramos el mejor resultado. Esto lo podremos comprobar en la matriz de confusión, cuyos valores son muy similares a los resultados de las métricas que obtuvimos originalmente en las pruebas de los 4 modelos.

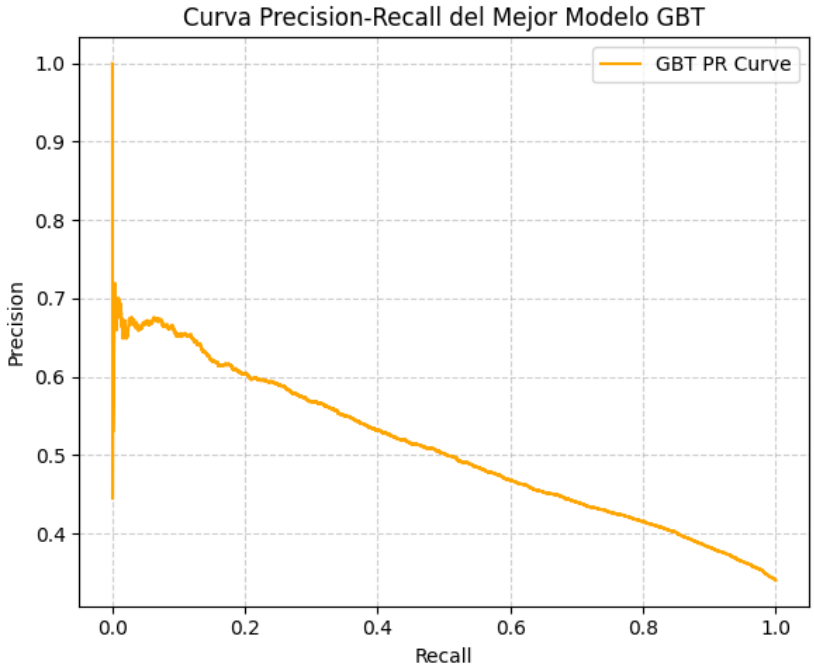


Fig. 16 Grafica PR del mejor clasificador GBT

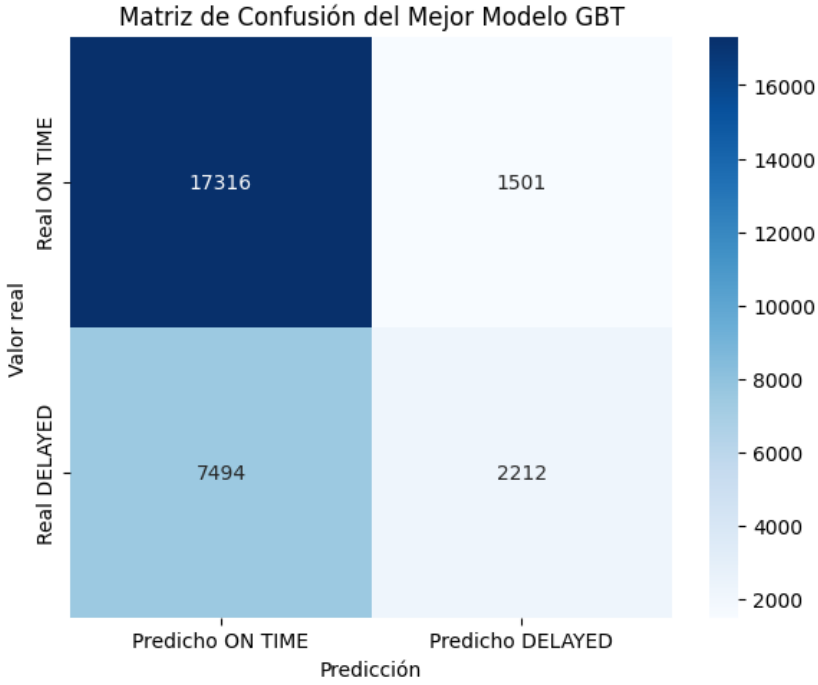


Fig. 17 Matriz de confusión del mejor clasificador GBT

Conclusiones

Se pudo obtener de manera exitosa un clasificador de vuelos. Si bien no se logró la mejor precisión, si pudimos identificar que la variable más importante para nuestro modelo es la operadora. Posteriormente sería la temporada del vuelo y el aeropuerto de destino y la hora de salida (si es hora pico o no).

Conforme a dichas variables tanto las operadoras como las logísticas de cada aeropuerto podrán tomar medidas para disminuir sus retrasos de manera conjunta. De igual manera, nuestro clasificador se podría implementar como medida preventiva, alertando tanto a la aerolínea como a la logística del aeropuerto o de destino.

Trabajo a futuro

Entre las mejoras a futuro está el incremento de las métricas de desempeño. Una posible causa de esto es el ligero desbalanceo de clase, y una posible solución es el implementar técnicas de sobremuestreo y submuestreo. Aumentando la precisión de nuestro modelo podremos lograr estimaciones más precisas, logrando así una mejor implementación de este.

Con la información de que la operadora de aviación es más importante, se podría realizar un análisis de las aerolíneas que tienen más retrasos y compararlas con las que tienen los menores retrasos para identificar las buenas prácticas e implementarlas.

De igual manera se puede optar por un enfoque por aeropuerto, ya que otro elemento importante que tiene influencia en los retrasos es el aeropuerto de destino. Dando prioridad a mejorar la logística del aeropuerto de destino, mejorando los retrasos de todas las aerolíneas.

Bibliografía

- Sherry. (2019). *Airline Delay Analysis*. Kaggle.com. https://www.kaggle.com/datasets/sherrytp/airline-delay-analysis
- Diego Andrés Bernal Díaz, (2025). *Entrega proyecto final Equipo 19*, Youtube.com. https://www.youtube.com/watch?v=9WHjxAYGt5I
- Sternberg, A., De, J., Carvalho, D., & Ogasawara, E. (2017). A Review on Flight Delay Prediction. *ResearchGate*. https://doi.org/10.48550/arXiv.1703.06118
- *Airline Flight Delay Prediction Using Machine Learning Models*. (2021). Acm.org. https://dl.acm.org/doi/fullHtml/10.1145/3497701.3497725