Assignment 3

ML Data Product

08/11/2024

Group 15

Student Last Name	Student First Name	Student ID	Group Allocation
Jeong	Taekjin	25099654	Group 15
Tsui	Raymond	10701389	Group 15
Joshi	Dhruv	25032069	Group 15

Github	Project Repo: https://github.com/raym2828/adv_mla_at3_experimentation.git Streamlit App Repo: https://github.com/raym2828/adv_mla_at3_streamlit.git

36120 - Advanced Machine Learning Application
Master of Data Science and Innovation
University of Technology of Sydney

Table of Contents

1. Executive Summary

Project Overview

This project aims to build a predictive data product to help the users in the U.S to estimate local travel airfare. Here, the users provide trip details - commonly including origin, destination, departure date and time and cabin type via the Streamlit app. These returns predicted flight fares using XGBoost. This is mainly designed to assist the travellers in making informed financial decisions by giving them an estimated forecast of the flight cost and then comparing to the actual flight cost projected.

Problem Statement and Objective

We know that travellers often face uncertainty while planning trips due to the fluctuating airfares. The existing platforms provide us with tools like historical pricing trends or simple fare comparisons, but they lack real-time predictions based on the traveller's personal needs. Hence, this project aims to tackle the problem by using historical flight data and ML model to predict airfare based on a user's unique travel details.

The main objective is to develop a Streamlit app which allows the users to input the trip details and predict the airfare using XGBoost. The final model was selected based on the evaluation of different models used on the provided dataset.

Achieved Outcomes

The dataset provided includes detailed flight and fare information, which was cleaned and pre-processed to build predictive models. Each group member trained a distinct ML model and then the model with the least RMSE and MAE score was selected i.e. XGBoost. The final app allows the users to input the trip details and then receive predicted airfare from the model. This interface enables comparison between the predicted value from the selected model and the actual airfare fetched with an API.

Relevance and Impact

By providing users with insights into potential travel costs, this tool helps reduce financial uncertainty, and also enhances travel planning. Furthermore, travel agencies and booking platforms can utilise this application and attract more users by providing improved customer satisfaction. This added value differentiates them from competitors.

2. Business Understanding

a. Business Use Cases

- Enhanced Fare Prediction for Travel Agencies and Booking Platforms
    - **Business Case** - By integrating real-time airfare prediction, travel booking platform can stand out in a competitive market as it will provide the users with personalised fare estimates based on their specific travel details. This feature will help the agency increase the platform engagement.
    - **Challenge** - Travel agencies often struggle to differentiate themselves as most platforms offer similar fare comparison tools. Furthermore, users are cautious of sudden price changes.
    - **Opportunity** - By offering accurate predictions, the platform can improve the user satisfaction and thus increase booking rates, gaining a competitive edge over others.
- Dynamic Pricing and Optimising Last-Minute Sales for Airlines
    - **Business Case** - As we know airlines often have unsold seats close to the departure date. Predictive models can help airlines forecast demand for specific routes and times, enabling dynamic pricing strategies. This can allow them to adjust prices accordingly to fill seats without undercutting revenue.
    - **Challenge** - By reducing the prices of tickets for last-minute sales, airline companies might face revenue loss, while higher prices risk leaving seats unsold.
    - **Opportunity** - With the help of the predictive model, airlines can dynamically adjust fares based on real-time demand, weather, or competitor pricing. This will ensure better utilisation of capacity and will increase the last-minute revenue opportunities.
- Cost Management for Corporate Travel
    - **Business Case** - In situations where large cooperation manages travels for their employees across various locations, this model can be useful in predicting the airfare costs and reducing the companies budget for travel. This can allow the corporation to identify best times to book flights.
    - **Challenge** - Corporate travel budgets are often exceeded due to unpredictable fare changes. Without accurate forecasting, companies may overspend on flight tickets.
    - **Opportunity** - By utilising this model, companies can reduce travel costs, enforce travel policies, and improve overall financial planning.

b. Key Objectives

The highlighted objectives of this project are as follows:

- **To Develop Accurate Airfare Prediction Models -** We aim to build a ML model which can predict airfare based on user-provided trip details (origin, destination, departure date/time, cabin type). This will provide the users with fare estimates according to personal needs.
- **To Create a User-Friendly Streamlit Application -** This application will allow the users to input trip details and view fare predictions from selected machine learning models i.e. XGBoost. This enhances the user experience by offering an interactive tool for airfare forecasting.
- **To Provide Data-Driven Insights for Stakeholders -** The use of predictive analysis and delivering insights to meet the specific needs of the stakeholders like travel agencies, airlines and corporate clients will improve customer engagement.

Stakeholders and Addressing Requirements

Some of the main stakeholders for this application are as mentioned below.

- **Travellers** - These end-users require accurate fare prediction tailored to their specific trip details. This project aims to address these needs by offering a Streamlit app which is an interactive platform for users to access predictions. ML models used for comparing the predictions to the fares obtained from API are displayed on this app.
- **Travel Agencies and Booking Platforms -** Agencies mainly need tools that can attract and retain users via enhanced fare predicting features. This project aims to address their needs by providing the users with predictions to encourage users to book flights directly through the platform, increasing revenue. The integration of airfare prediction within their platforms allows agencies to differentiate themselves and improve user trust.
- **Airlines -** We know that dynamic pricing models are used by airlines to maximise the revenue based on demand. This project considers airlines as one of the main stakeholders because the predictive algorithms provide airlines with demand insights, allowing for more strategic pricing. This model can also enable better planning for last-minute seat sales.
- **Corporate Travel Managers -** Managers of large corporate companies usually are responsible for managing the travel budgets and booking flights for their

employees across various destinations. This application with its predictive model can help these managers plan and optimise travel budgets. The forecasting tools enable them to provide employees with cost-effective booking windows.

3. Data Understanding

Airports

The dataset is flight itinerary transactions from 2022-04-16 to 2022-05-19 in CSV files. Each CSV named International Air Transport Association Code (IATA) and it includes 16 United States airports (ATL, BOS, CLT, DEN, DFW, DTW, EWR, IAD, JFA, LAX, LGA, MIA, OAK, ORD, PHL and SFO). It has 13,519,999 rows and 23 columns. When reviewing Figure 1, the number of transactions varied by airports. LAX had the highest number of transactions, while OAK had the fewest. It suggests that LAX is the busiest airport, while OAK is likely to be less busy in comparison.

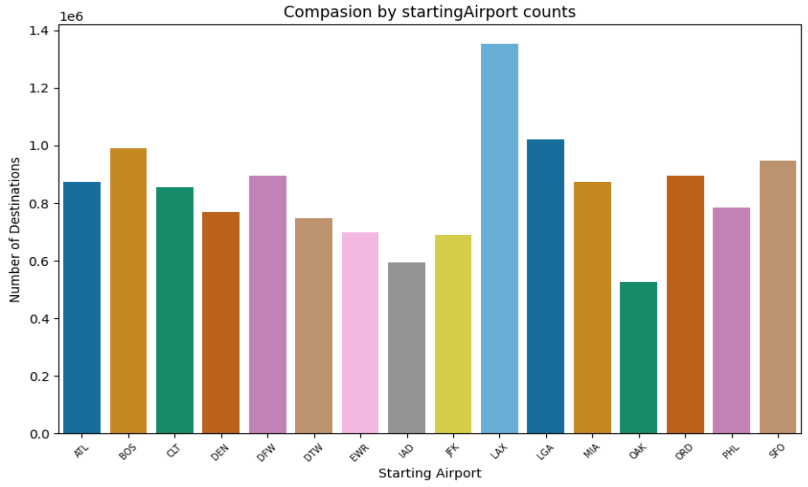


Figure 1. Comparison by StartingAirport

Furthermore, Figure 2 shows that each airport has different destination frequencies. For Denver (DEN), flights to LAX were the most frequent, while flights to OAK were the least. In contrast, for Chicago (ORD), flights to LGA were the most common, while flights to DTW were the least frequent.

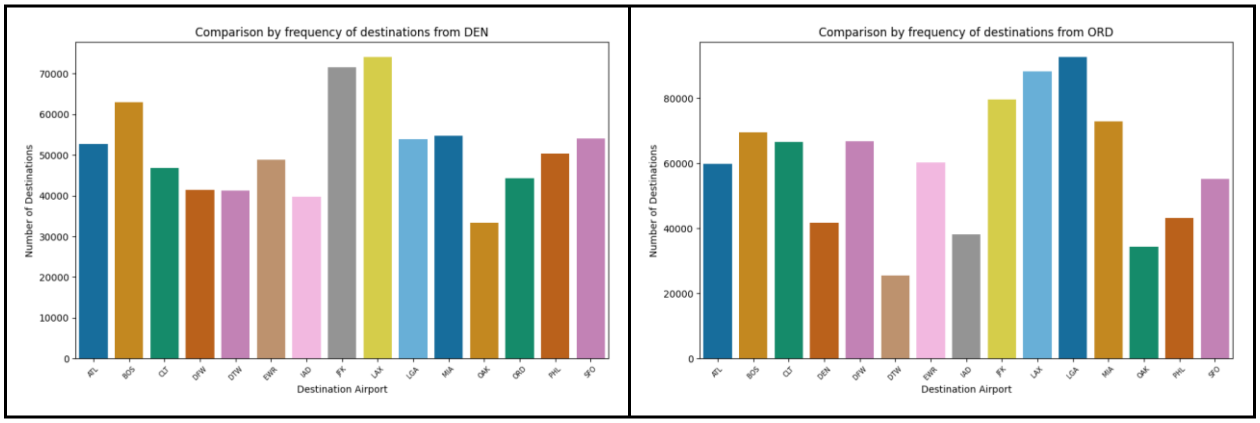


Figure 2. Comparison by frequency of destination

TotalFare (Target)

`TotalFare` is this project's target variable, and its mean is $373.75. Figure 3 indicates that most values are under $500 and there are outliers over $4,000. Minimum fare is $23.97 and maximum fare is $8260.61.

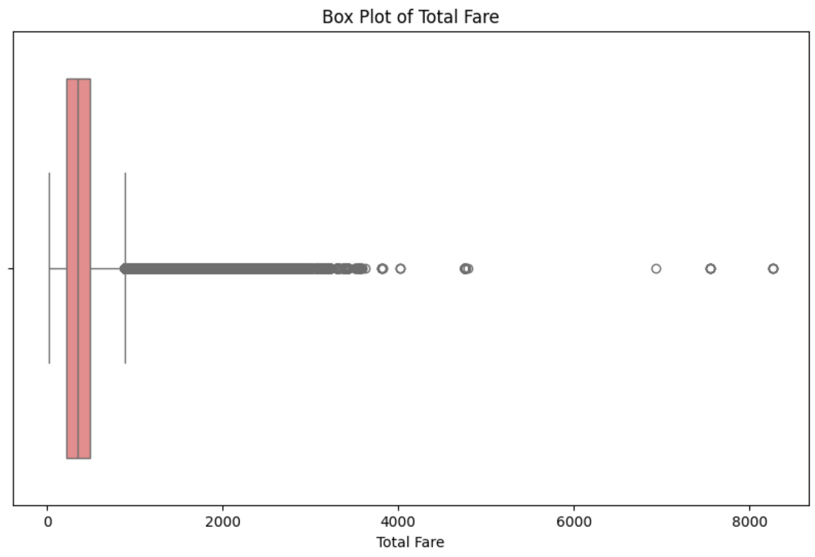


Figure 3. TotalFare box plot

Through the graph in Figure 4, we can identify the features that influence `TotalFare`. When `isBasicEconomy` and `isNonStop` are both False, `TotalFare` tends to be higher compared to when they are True. Conversely, when `inRefundable` is True, `TotalFare` is significantly higher than in other cases.

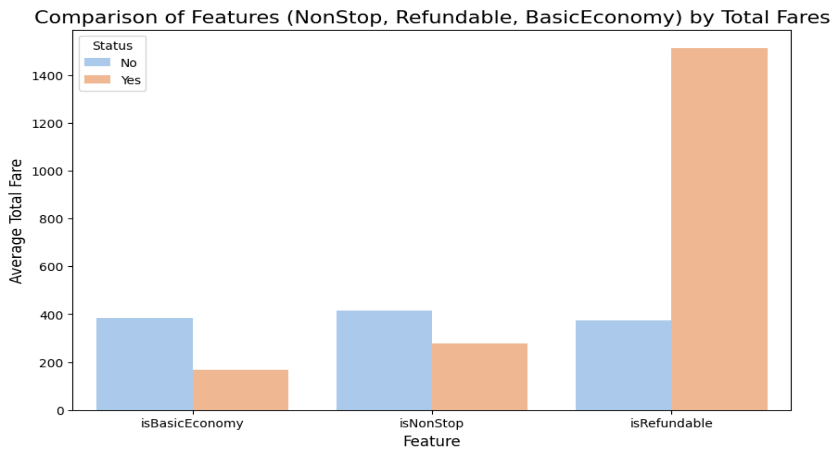


Figure 4. Comparison of Features by TotalFare

Additionally, other features are also associated with the target variable. Figure 5 shows the average price by weekday, where Sundays tend to be more expensive, while Tuesdays and Wednesdays are slightly cheaper than other days. Figure 6 suggests that the time difference between `searchDate` and `flightDate` impacts the target variable. Between days 10 and 20, the average fare is lowest, but it gradually increases thereafter. This trend may be linked to the frequency of user transactions. In Figure 7, we identify that the booking frequency is lowest between days 10 and 20 and then surges from day 20 onward. This implies that booking rates may influence `TotalFare`.

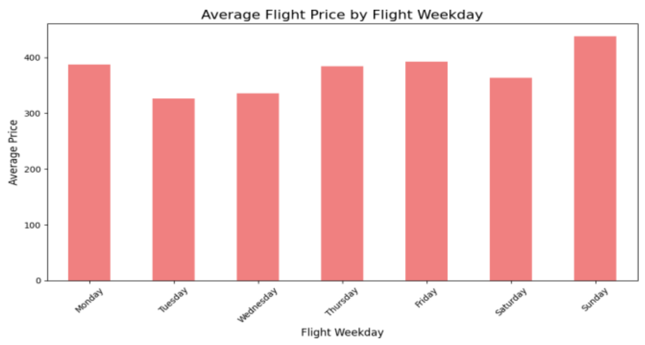


Figure 5. The average price by weekday

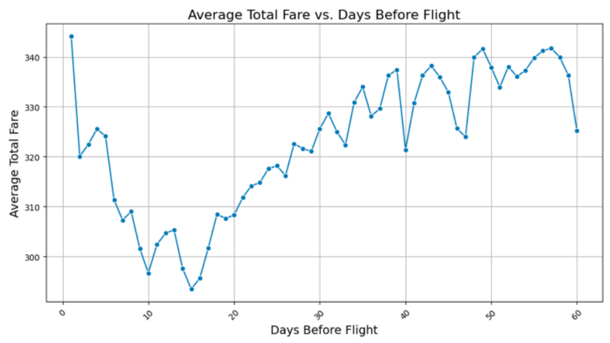


Figure 6. Average price by days differences

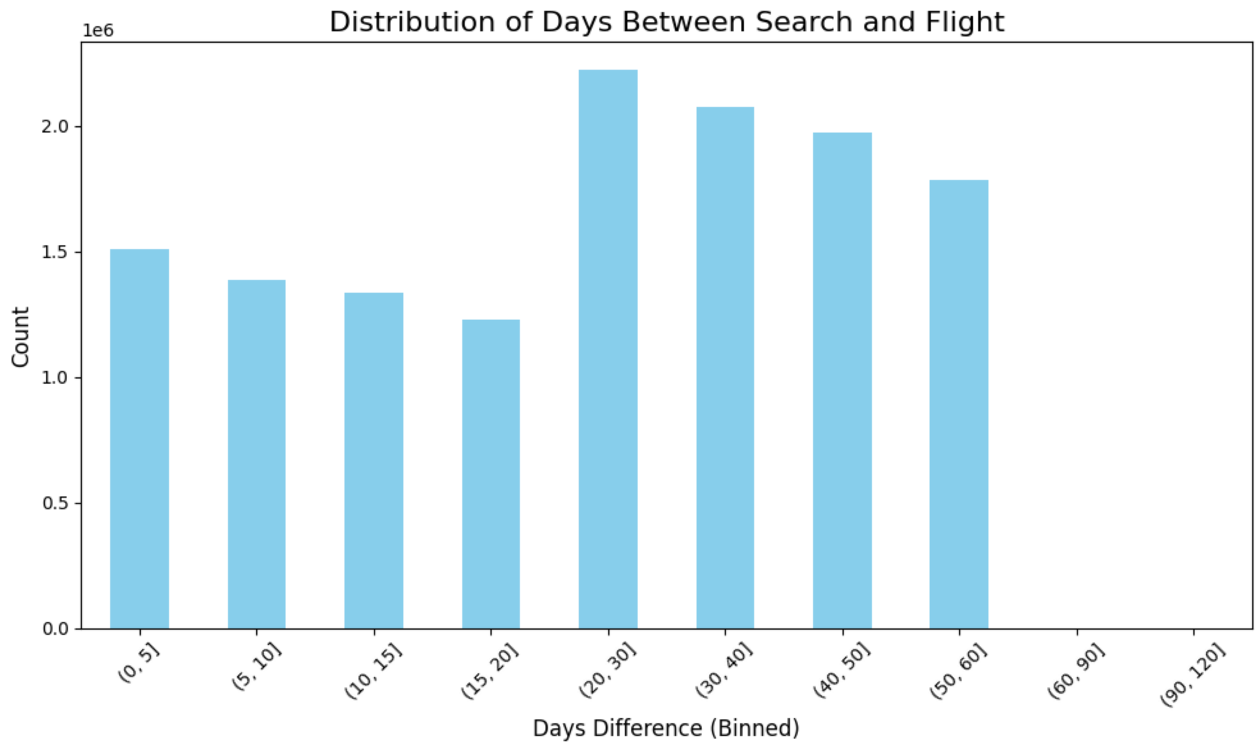


Figure 7. Distribution of days between searchdate and Flightdate

Figure 8 displays a scatter plot of the correlation between `TotalDistance` and `TotalFare`, indicating that the fare increases incrementally with distance. However, beyond 3,000 km, the fare decreases, suggesting counter inuatively that longer distances may contribute to lower `TotalFare`.

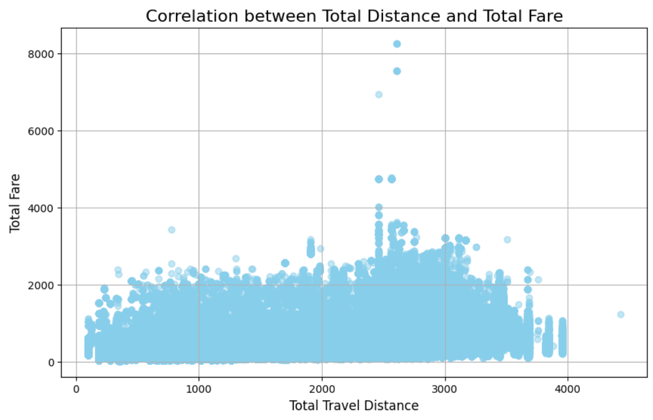


Figure 8. Correlation between TotalFare and Totaldistance

4. Data Preparation

Preprocessing

To assist with preparing the data for modelling we created a data loader function to be used by the whole team to set up the data quickly and consistently for data exploration. This involved unzipping and merging the zipped CSVs to one CSV per airport. Another function was then built to apply the transformations to each airport file and merge it into a master train and test set for the team to use.

Feature Engineering

1. `date_diff_days`: Captures the time difference between `searchDate` and `flightDate` to assess fare trends as the departure date approaches.

2. Segment-Based Features:

- Departure Time: Extracted from the first segment of `segmentsDepartureTimeRaw` to understand the impact of departure time on pricing.

- Cabin Class: Extracted from `segmentsCabinCode`, encoded using a dictionary, and averaged to represent the overall cabin class of the flight.

- Airline Category: Extracted from `segmentsAirlineName`, encoded using a dictionary, and the maximum category was assigned to the flight. This approach helps capture pricing trends within specific airline categories, identified in our EDA.

Encoding Dictionaries

Cabin Class Encoding:

```
cabin_class_encoding = {    "coach": 1,"premium coach": 2,"business": 3,"first": 4}
```

Airline to Airline Category Encoding:

```
airline_type_encoding = {'JetBlue Airways': 2,'Sun Country Airlines': 2,'United': 4,'Delta': 4,'Key Lime
Air': 3,'Boutique Air': 3,'Contour Airlines': 3,'Spirit Airlines': 1,'American Airlines': 4,'Alaska
Airlines': 4,'Southern Airways Express': 3,'Frontier Airlines': 1,'Hawaiian Airlines': 4,'Cape Air': 3}
```

```
airline_category= { 1:'Ultra Low Cost',2:'Budget', 3:'Regional', 4:'Full Service'}
```

Cleaning

After processing the `segmentsDepartureTimeRaw` we identified and removed some errors with the data. These entries had incorrect time zone formats or departure times that preceded the flight date and departing airport.

Feature Selection/Engineering

We chose features to include based on our EDA and afterward in the model experimentation stage to understand if adding additional features would add to model performance.

```
categorical_cols = ['startingAirport', 'destinationAirport']
numerical_cols = ['AirlineNameScore', 'CabinCode', 'totalTravelDistance',
'date_diff_days'  ,'DepartureTimeHour' , 'weekday'  ,'isNonStop',
'isRefundable' , 'isBasicEconomy']
```

Including

- Date_diff_days: The time difference between `SearchDate` and `FlightDate` is a key feature in predicting the target variable which is why we have created this variable.
- Airline Category: Since prices may vary depending on the airline, we created an airline category feature for scoring purposes.
- `isBasicEconomy`, `isNonStop`, and `isRefundable`: Initially, we excluded these features to maintain model simplicity. However, recognising their potential to improve model performance and their utility for users, we ultimately decided to include them.

Excluding

- `SearchDate` and `FlightDate:` As identified during EDA, this dataset includes only one month of `SearchDate` and two months of `FlightDate` information. Data from missing months or days could introduce bias in model training, so they were excluded.
- Other Segment data: It was excluded to balance model performance and complexity.

Transformations of data

To ensure consistency in our target variable, `totalFare`, we retained only the cheapest airfare for each unique set of model inputs. More than 92% flights are direct or only 1 transfer, hence we also removed flights with more than 1 stop to improve models' performance. By doing this, it also helped reduce the impact of outlier routes. This resulted in a 43% reduction in the dataset.

To optimise dataset size, we transformed data into a more efficient format. This included:

- Converting categorical features (e.g., day of the week, cabin, airline categories) to numerical representations (integers or floats).
- Optimising data types (e.g., down casting dates to datetime, and numeric features to smaller integer or float types like uint8).

These transformations significantly reduced the dataset size, decreasing the Airport CSVs from 5GB to under 250MB.

5. Modelling

a. Approach 1 - Linear Regression

- This Approach began with Linear Regression because this algorithm predicts the target as a weighted sum of the input features. Its simplicity and alignment with data help to improve our model's performance. In addition, it often serves as a good baseline model for comparison, allowing us to understand the complexity needed to improve accuracy further.

- After that, we quickly applied several models (SGDRegressor, Adaboost, RandomForest) to identify the best-performing one. However, Linear Regression delivered the most accurate results. To further enhance its performance, we incorporated PolynomialFeatures. It is a preprocessing tool that transforms the original features of the dataset into polynomial combinations, allowing linear models to capture non-linear relationships such as `'StartAirport'` and `'DestinationAirport'`, potentially improving model performance by fitting more complex patterns in the data.

b. Approach 2 - XGBOOST

- The second approach was creating a model using XGBOOST, which turned out to be the best performing model. The data was first scaled and transformed using a preprocessor function and then fit to the model using pipeline.

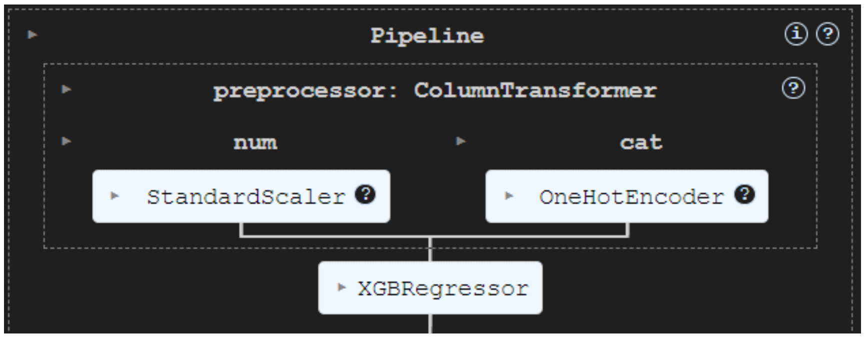
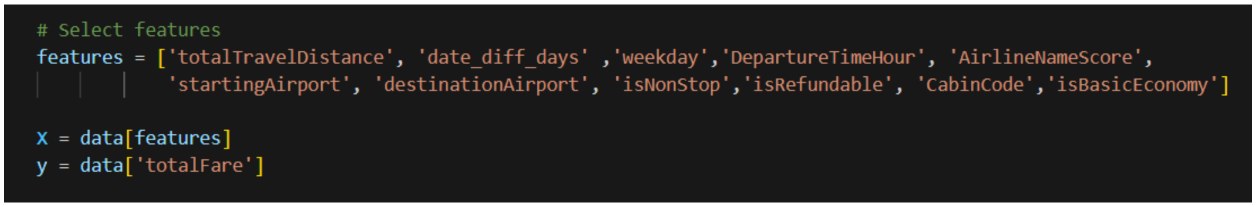


Figure 9. Pipeline process

- The features used to train the model include:



```
# Select features
features = ['totalTravelDistance', 'date_diff_days' ,'weekday','DepartureTimeHour', 'AirlineNameScore',
            'startingAirport', 'destinationAirport', 'isNonStop','isRefundable', 'CabinCode','isBasicEconomy']

X = data[features]
y = data['totalFare']
```

Figure 10. Selected features

- The RMSE got reduced from 103 to 97 after including the features 'isRefundable' and 'isBasicEconomy' indicating that these features might be important in predicting total fare.
- By leveraging XGBoost's efficiency, the platform can offer these estimates in real-time without sacrificing user experience, enhancing engagement by providing a unique and reliable feature.
- XGBoost can incorporate multiple factors (e.g., demand, time until departure, seasonal trends) into its predictions, allowing for optimised fare adjustments that consider both supply and pricing elasticity, leading to more strategic last-minute sales.

- Model performance actual vs predicted:

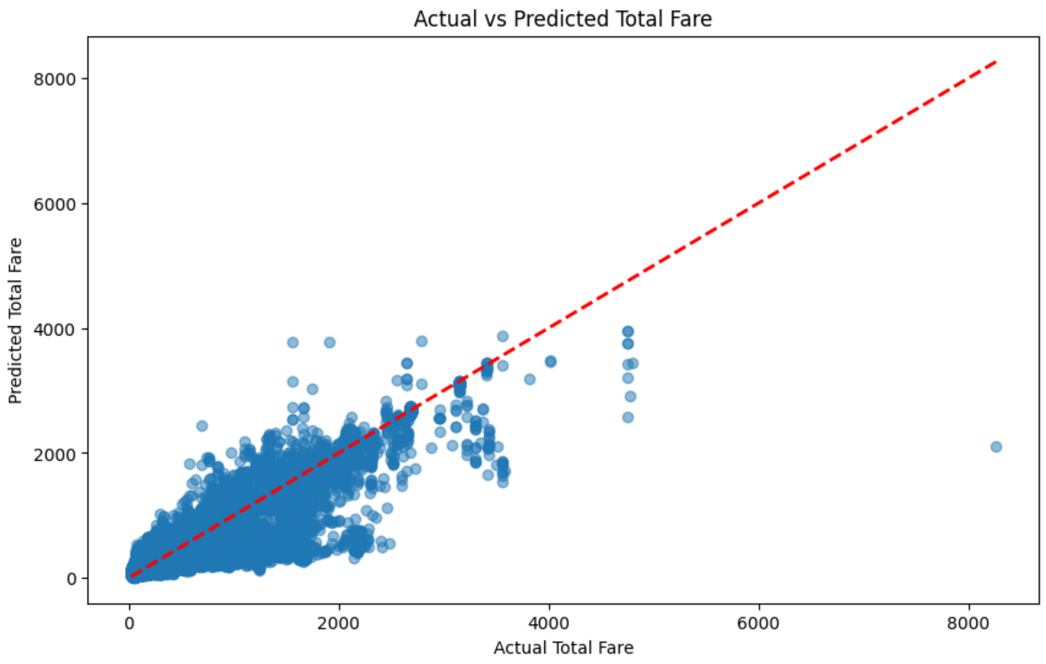


Figure 11. XGB model performance

The model performs relatively well for the lower fare range, where data density is higher, with most points clustering close to the ideal prediction line.

However, it struggles with higher fare values, showing a tendency to underpredict for more expensive fares. This could be improved with additional features or targeted tuning to better capture the dynamics of high fare predictions.

c. Approach 3- Neural Network

Neural networks are well-suited for handling complex patterns in large datasets. By passing input data through multiple layers of interconnected nodes and applying activation functions, these models can learn intricate relationships between features. We hypothesised that a neural network could effectively model the numerous factors influencing flight prices.

To optimise our model's performance, we conducted a comprehensive hyperparameter search. The optimal learning rate was determined to be 0.001, and the final network architecture is illustrated below:

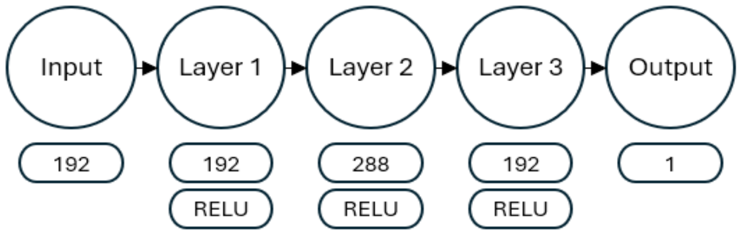


Figure 12. The final Neural network architecture

To prevent overfitting, we employed early stopping, monitoring the validation loss to terminate training when performance on the validation set began to degrade. This is particularly important in our case, as there are some rare combinations of flight features that might not generalize well to unseen data.

Training and Validation Loss per Epochs

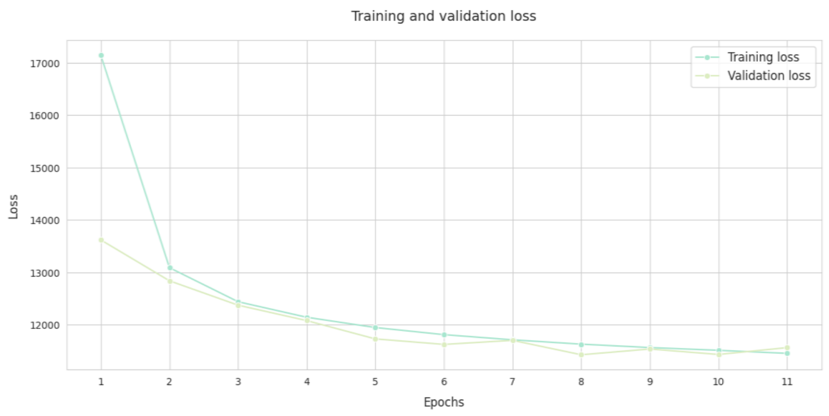


Figure 13. Training and Validation Loss per Epochs

Feature Engineering and Selection

We explored the potential benefits of incorporating additional features, such as average distance and airfare per route. However, these features did not significantly improve the model's performance. This suggests that the neural network was already capable of learning and capturing these relationships from the underlying data. For example, in the figure below you can see how the model was able to determine the average price of a route even though the average price per route feature was not included in the model.

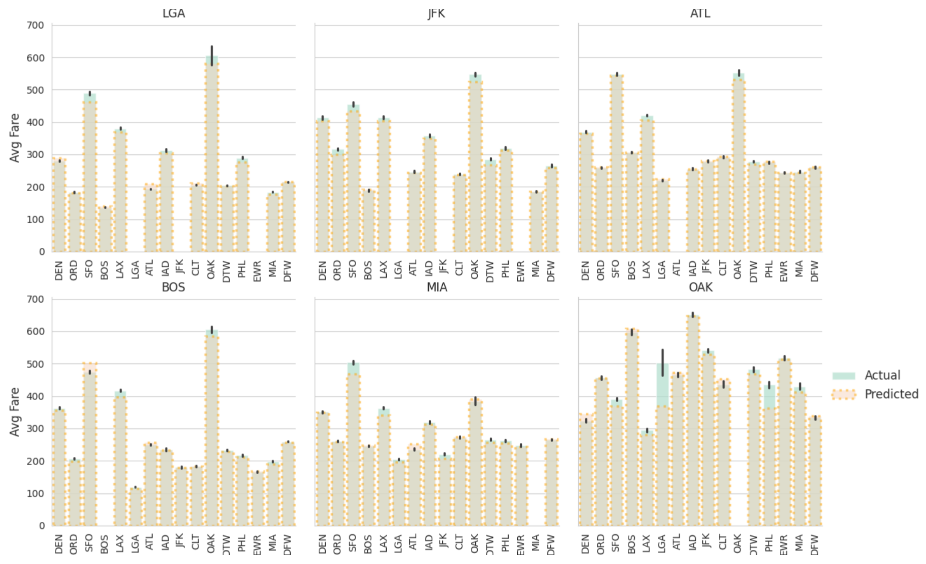


Figure 14. Average airfare price for route by Starting Airport

Categorical Embeddings

To handle high-cardinality categorical features like origin and destination airports, we utilised embedding layers. This technique reduces dimensionality and enables the model to learn relationships between similar airports for example, improving generalisation.

6. Evaluation

a. Evaluation Metrics

We used Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE) to evaluate the model performance. MAE provides an intuitive understanding of how far the prediction was away from the airfare price. RMSE penalises large errors more heavily. Given the inconsistent large price spikes in our data, which might not fully capture the features passed to the model, the RMSE might not be the most suitable metric for evaluating our model's ability to predict the cheapest flights. In such cases, MAE can offer a more reliable assessment.

b. Results and Analysis

Table 1. Results Metrics

#	Model	MAE	RMSE
1	**LR**	84.713664	121.99713
2	**XGB**	66.573627	98.770034
3	**LGBM**	74.185101	108.315537
4	**NN**	70.443315	106.791914

Our best-performing model was the XGBoost model. It achieved a Mean Absolute Error (MAE) of 66, indicating that, on average, our model's predictions were only $66 away from the actual prices. This outperformed the Neural Network (NN), which had an MAE of 70.

Additionally, the XGBoost model's Root Mean Squared Error (RMSE) of 98 was significantly lower. This suggests that it was more effective in accurately predicting a wider range of prices, especially for more complex and varied flight scenarios present in our test dataset.

One interesting insight is the smaller difference in RMSE between LGBM (108) and the NN (106) compared to the MAE. This indicates the NN is probably more sensitive to outliers or extreme airfare prices compared to the gradient boosted models.

Predicted vs Actuals Scatter Plot

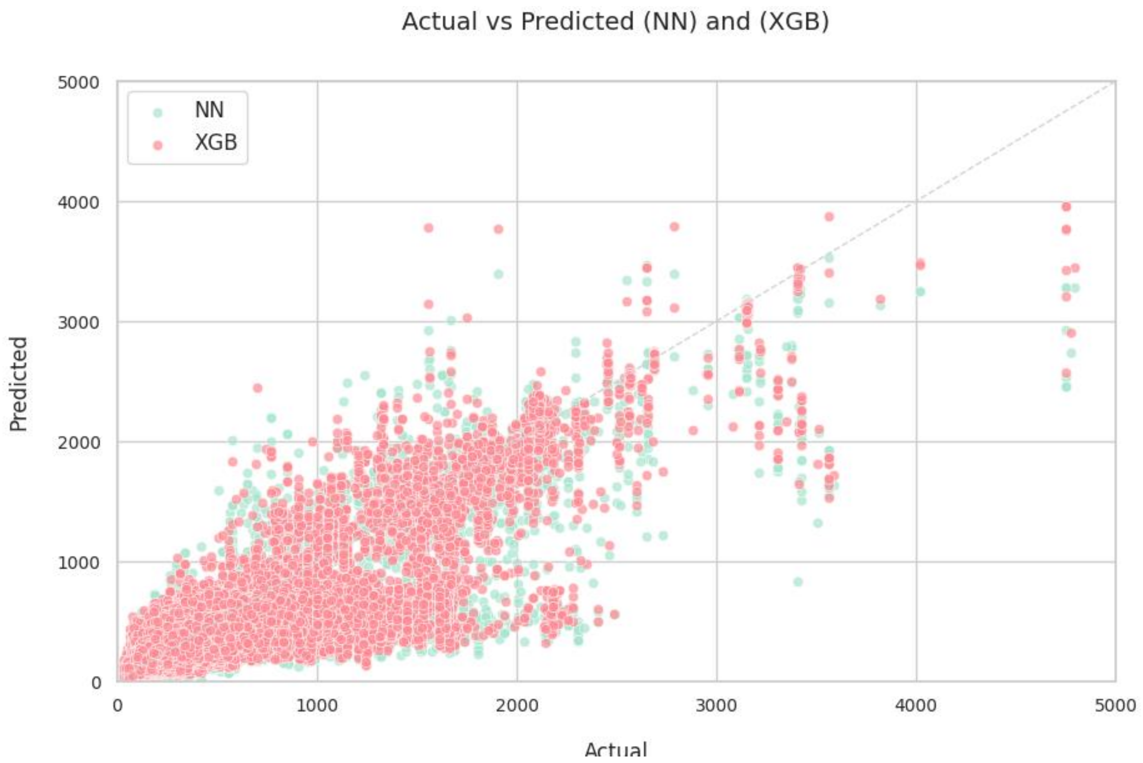


Figure 15. Predicted vs Actuals Scatter Plot

When comparing the predicted and actual values for XGBoost and the Neural Network, it becomes clear why XGBoost (red) outperformed the Neural Network (green). The XGBoost model's predictions were more closely aligned with the true values across a wide range of airfares.

This was particularly evident for a specific route where the price increased fourfold as the flight date approached. While XGBoost's prediction was still lower than the actual price, it was significantly closer than the Neural Network's prediction. This enhanced performance can be attributed to XGBoost's iterative learning process, which allows it to gradually refine its predictions, even for extreme values.

Destination Airport Average Fare by Airline Category

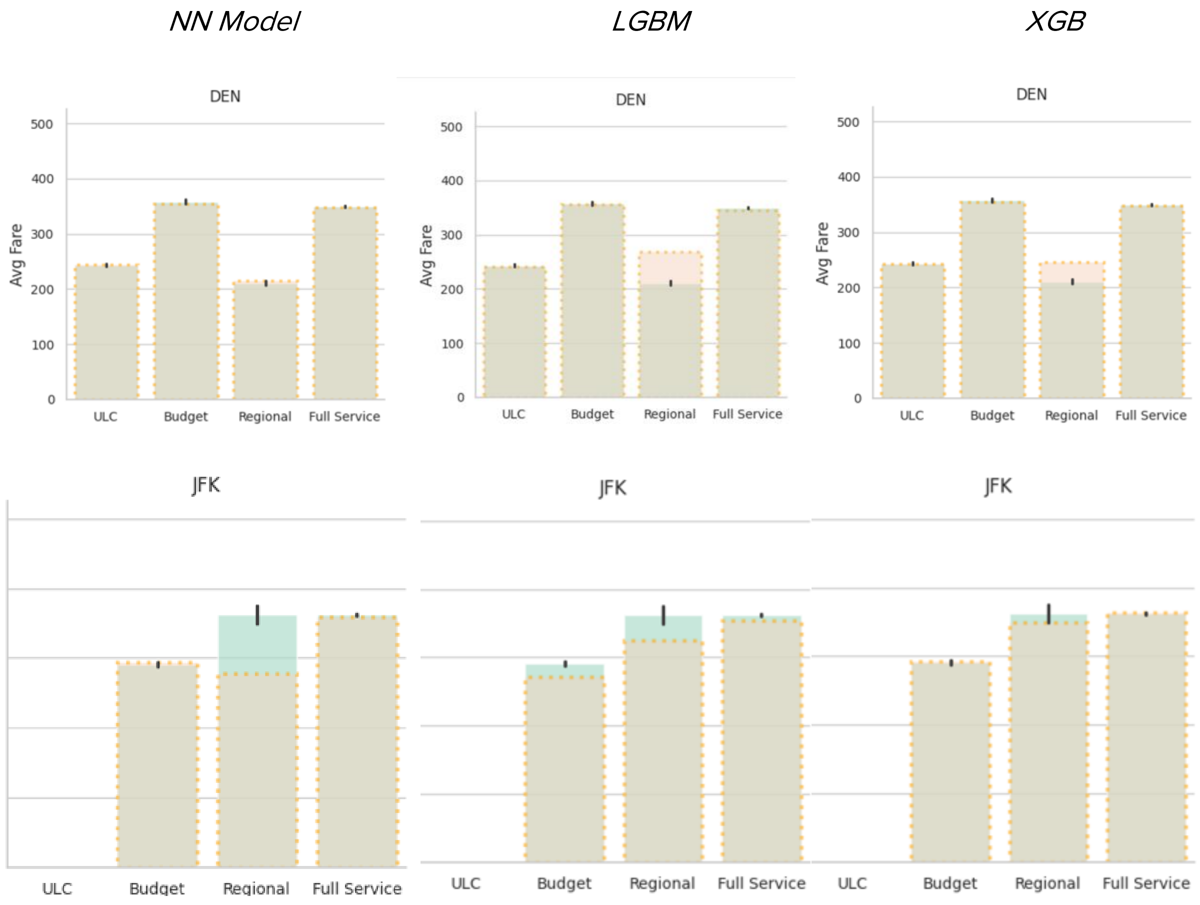


Figure 16. Destination Airport Average Fare by Airline Category

When aggregating and averaging predictions at a higher level, all models demonstrated remarkable accuracy. For instance, when analysing flights to DEN or JFK, grouped by airline category, the models generally performed well. The dotted orange bar (the predicted average airfare) was generally the same as the green bar (the actual airfares).

However, some discrepancies emerged. For regional flights to DEN, the gradient boosting models (XGBoost and LightGBM) overpredicted, while the Neural Network (NN) was more accurate. Conversely, for average airfares to JFK, the NN underpredicted compared to XGBoost for regional airlines.

This variance in model performance for regional airlines is likely due to the class imbalance in the dataset. Regional airlines might be less represented, leading to potential challenges in training and predicting their specific pricing patterns.

Airline Cabin Code

NN *XGB*

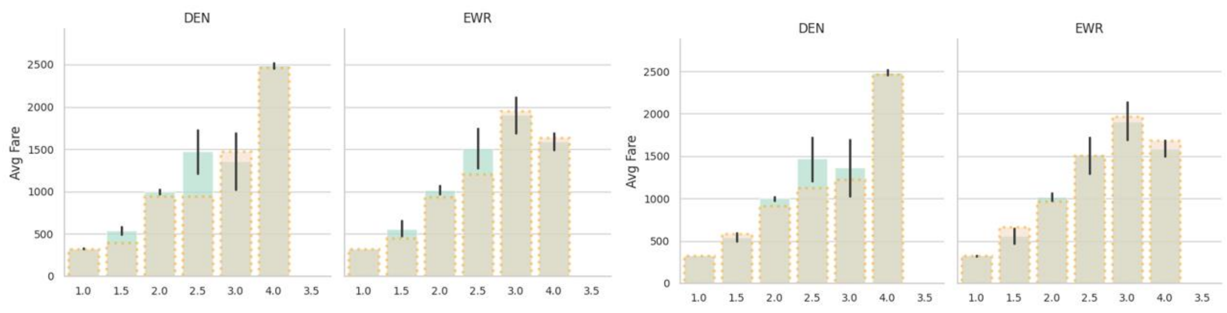


Figure 17. Airline Cabin Code

When examining the accuracy of the models based on cabin codes, XGBoost demonstrated a more pronounced edge. Notably, cabin codes with decimal values (e.g., x.5) indicate multi-segment flights with mixed cabin classes, a less common subset of the data. XGBoost proved more adept at handling these complex scenarios. For instance, it accurately predicted the 2.5 cabin code for EWR, significantly outperforming the Neural Network, which exhibited a larger error. Even in cases with high data variability, as indicated by the wide confidence intervals for DEN airport's 2.5 cabin code, the XGBoost model still provided a more accurate prediction.

Day of the Week

XGB

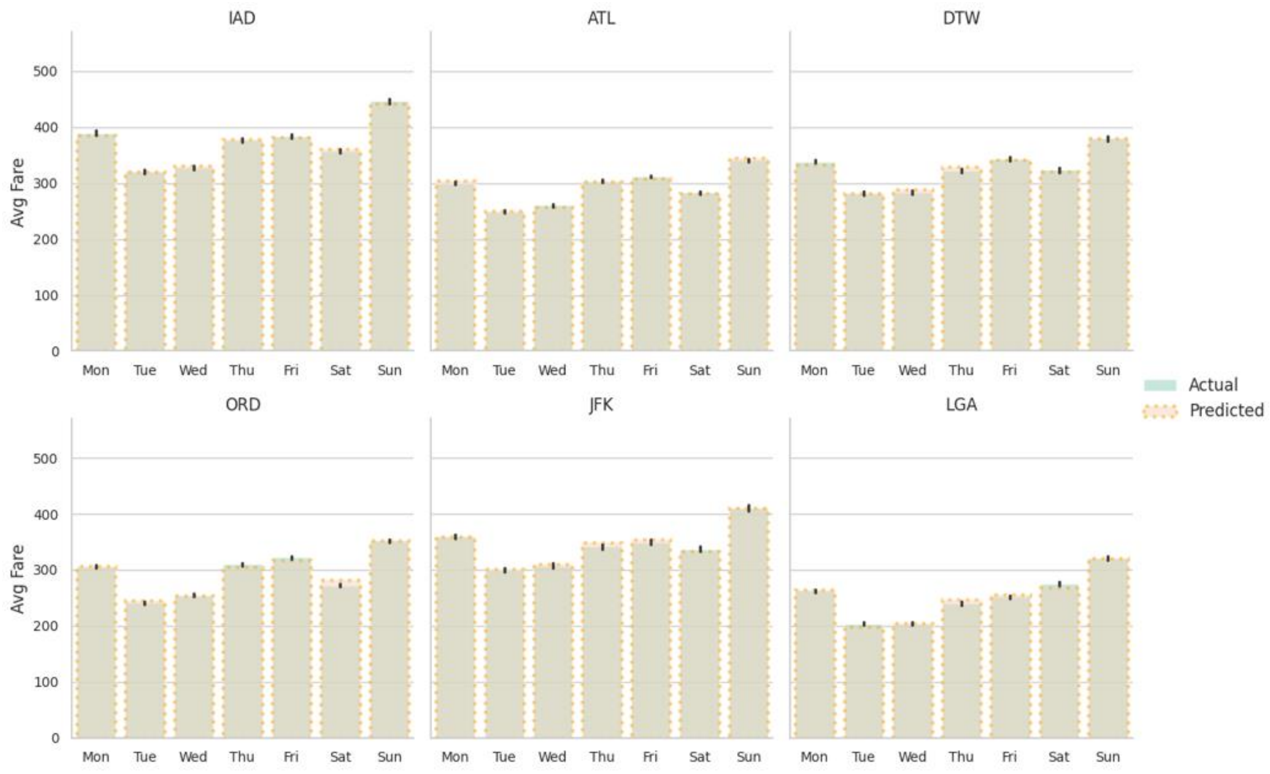


Figure 18. Days of the week

All models demonstrated generally accurate predictions for average daily prices. However, XGBoost stood out with near-perfect predictions. It effectively captured the peak demand on Mondays and Sundays, the trough on Tuesdays and Wednesdays, and the distinct patterns on Fridays and Saturdays—increasing at LGA and decreasing at DTW. This means our model can capture the weekly cycles well for our predictions.

Model Interpretation

The permutation importance (Figure) results from the relative importance of each feature in the best model based on how much the model's performance.

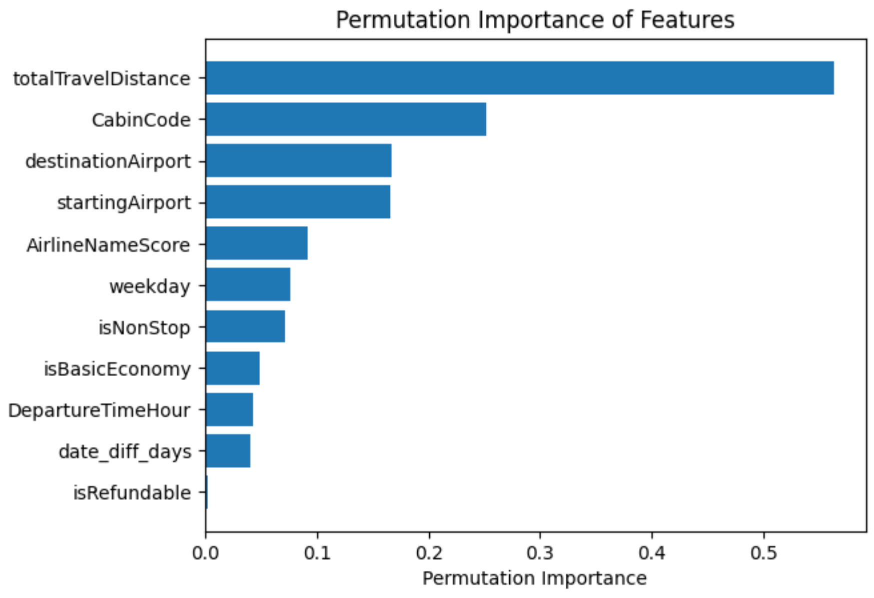


Figure 19. Permutation result

totalTravelDistance has the highest importance score with 0.56319. It plays the most significant role in predicting the target. Also, CabinCode and Start/destination Airports are valuable for models' performance. On the other hand, the difference in days between the search date and the flight date is less important. isRefundable has the lowest importance among selected features.

c. Business Impact and Benefits

Impact and Benefits of the Final Model on Business Use Cases

- **Enhanced Fare Prediction for Travel Agencies and Booking Platforms**

  Impact - The final model's ability to predict airfares accurately based on real-time travel details has a significant impact on travel agencies and booking platforms. By providing personalized fare estimates, these platforms can offer a more tailored user experience, helping them stand out from competitors who only offer basic fare comparisons.

  Benefits - Travel agencies will see increased engagement as customers are more likely to trust and return to platforms.

  Quantified Improvements - This can lead to an increase in revenue for the travel agency and improve customer retention.

- **Dynamic Pricing and Optimizing Last-Minute Sales for Airlines**

  Impact - For airlines, the predictive model allows for more effective dynamic pricing, especially in scenarios including last-minute sales. Thus, airlines can adjust the prices based on the predicted demand. This ensures they fill as many seats as possible without affecting the total revenue.

  Benefits - This model can be used by airlines to forecast the demands more accurately, leading to better utilization of their service.

  Quantified Improvements - With the improved predictions of the model, the airlines can reduce the number of unsold seats close to departure.

- **Cost Management for Corporate Travel**

  Impact - For larger scale corporate companies, the model provides a valuable insight into airfare trends, thus allowing them to forecast travel costs more accurately. This ensures they can budget more effectively and identify the best times to purchase tickets to reduce overall expenses.

  Benefits - With the improved precision of the model, the companies can optimise their travel budgets and avoid overspending.

  Quantified Improvements - This improvement contributes to better financial planning and cost management, allowing companies to allocate funds more efficiently to various departments.

Contribution to Business Challenges and Opportunities

The model solves the challenge which is possessed by price unpredictability, allowing for more strategic and cost-effective purchasing decisions. By offering personalized fare predictions, travel agencies can boost user satisfaction and increase booking rates. Furthermore, airlines can optimize seat occupancy through dynamic pricing and corporations can use the model to forecast and plan travel budgets more effectively.

d. Data Privacy and Ethical Concerns

Data Privacy Implications

This project involves working with a dataset that contains sensitive information, particularly related to users' travel details and ticket pricing. thus, this data could reveal personal identifiable information if linked to their flight bookings. Variables like startingAirport, destinationAirport, and flightDate could be combined with other available data to indirectly identify individuals or reveal personal travel patterns. This could raise privacy concerns if not handled properly.

Ethical Concerns

Several ethical concerns arise from data collection, usage, and model deployment:

- **Bias in Predictive Models** - If the model uses data from specific regions in this case USA it may not fully represent the diversity of the population, there could be a risk of biased predictions for some smaller airports and locations.
- **Transparency of Model Decisions** - XGBoost model acts as a black box and hence the users rely on predictions without understanding how they were generated. This can lead to lack of transparency and trust in the model's prediction on the estimate. We have reduced this by using model interpretation.
- **Skewed Predictions** - The model favours users from particular regions i.e. USA or airlines, which could skew airfare predictions in favour of certain demographic groups.

Steps Taken to Ensure Data Privacy and Ethical Considerations

To address data privacy and ethical concerns, several measures were taken during the project:

- **Data Aggregation:** For the purposes of model training and deployment, we aggregated travel data to avoid individual-level analysis and focused on trends and general patterns across flights and routes. This reduces the risk of exposing sensitive user information.
- **Ethical Deployment:** We ensured that the deployment of the Streamlit app and FastAPI API adhered to best practices for user privacy and security.
- **No Retention of Data:** Ensured that no user data was stored after the session ended, reducing the risk of future data misuse.
- **Data Aggregation:** Instead of focusing on specific flights or individual bookings, the dataset was aggregated by common travel routes. This allows us to observe the general trends in airfare prices across different regions.

7. Deployment

a. Model Serving

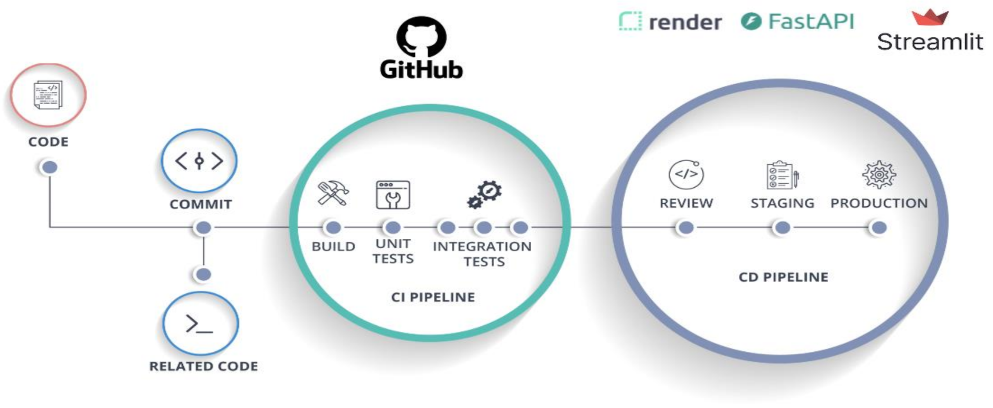
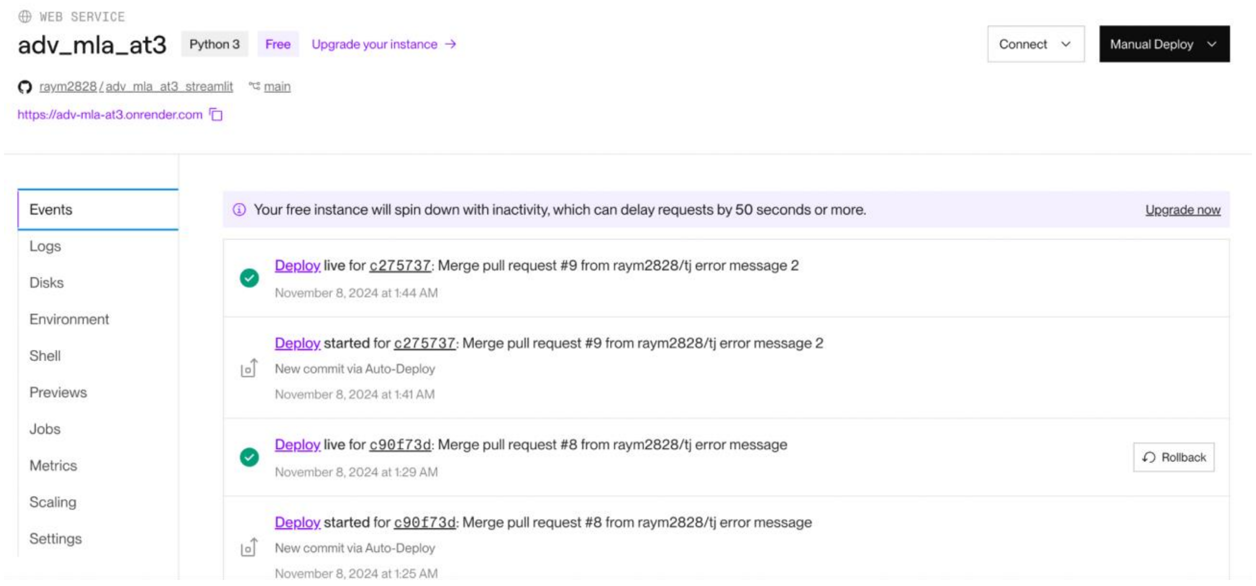


Figure 20. CI/CD pipeline

Our team focuses on CI/CD pipeline to reduce deployment time and to enable faster time-to-market Machine Learning models (Figure). To apply CI pipeline, our team divided two separate developments on Github repositories: First Github dedicated to developing the machine learning algorithms and the other focused on the web service layer. After the models' evaluation, we decided to integrate the XGB model as the best model, so it was saved as a JOBLIB file within the app repository. All code is deployed on GitHub, providing version control, and facilitating team members' collaboration. To apply the CD pipeline, we utilised two platforms (FastAPI and Streamlit Cloud).

- **FASTAPI** (URL: https://adv-mla-at3.onrender.com/): Backend API is deployed on Render, which provides a user-friendly interface to deploy web applications to the cloud. In this project, we opted not to Dockerize the FastAPI application directly. Instead, we use a start command to launch FastAPI, specifically: `uvicorn fast_api.app:app --host 0.0.0.0 --port 8000`. This approach offers several advantages, particularly in terms of flexibility and speed. By using a direct start command, we avoid the need to create a new Docker image each time a model update occurs, making the integration process faster and more agile. As a result, updates and changes to the model can be deployed without the additional step of building and managing Docker files, streamlining our workflow and reducing deployment time. This setup enables rapid iteration and ensures that our API service can adapt quickly to any model adjustments, making it ideal for a project that requires frequent updates and fast deployment cycles.


WEB SERVICE
adv_mla_at3 Python 3 Free Upgrade your instance →
raym2828 / adv_mla_at3_streamlit main
https://adv-mla-at3.onrender.com

Your free instance will spin down with inactivity, which can delay requests by 50 seconds or more.

Deploy live for c275737: Merge pull request #9 from raym2828/tj error message 2
November 8, 2024 at 1:44 AM

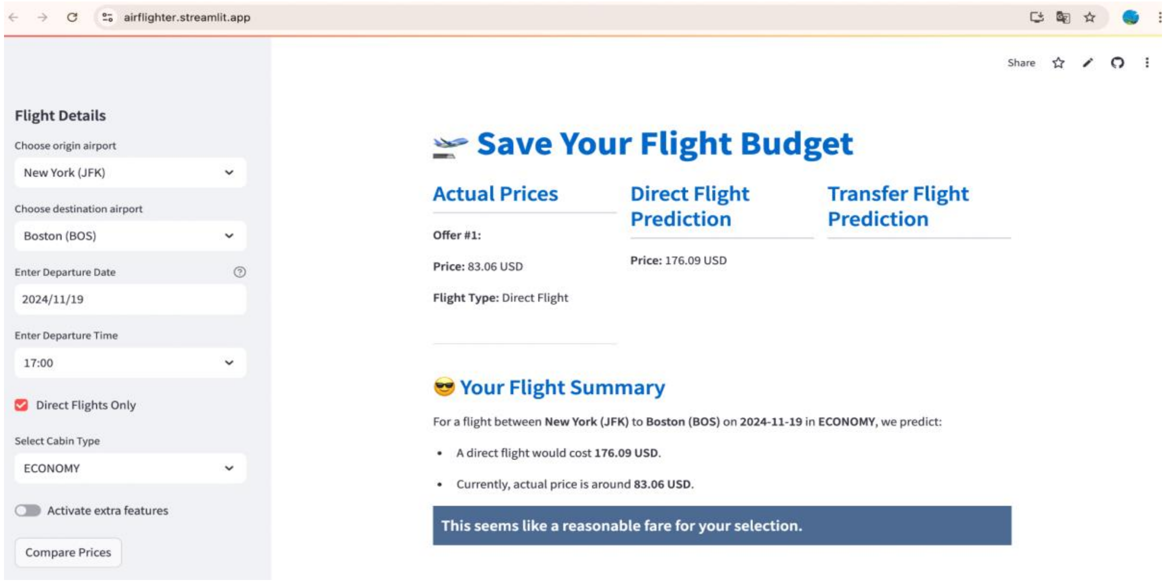
Deploy started for c275737: Merge pull request #9 from raym2828/tj error message 2
New commit via Auto-Deploy
November 8, 2024 at 1:41 AM

Deploy live for c90f73d: Merge pull request #8 from raym2828/tj error message
November 8, 2024 at 1:29 AM

Deploy started for c90f73d: Merge pull request #8 from raym2828/tj error message
New commit via Auto-Deploy
November 8, 2024 at 1:25 AM

Figure 21. Render deployment

- **STREAMLIT** (URL: https://airflighter.streamlit.app/): Our front-end is developed by Streamlit. Streamlit Cloud offers automated deployment from Github and we deployed our Web app on Streamlit Cloud. By utilising Streamlit Cloud, we could improve the robust CI/CD pipeline.


airflighter.streamlit.app

Flight Details
Choose origin airport: New York (JFK)
Choose destination airport: Boston (BOS)
Enter Departure Date: 2024/11/19
Enter Departure Time: 17:00
Direct Flights Only ✓
Select Cabin Type: ECONOMY
Activate extra features

Save Your Flight Budget

Actual Prices	Direct Flight Prediction	Transfer Flight Prediction
Offer #1: **Price:** 83.06 USD **Flight Type:** Direct Flight	**Price:** 176.09 USD	

😎 Your Flight Summary

For a flight between **New York (JFK)** to **Boston (BOS)** on **2024-11-19** in **ECONOMY**, we predict:

- A direct flight would cost **176.09 USD**.
- Currently, actual price is around **83.06 USD**.

This seems like a reasonable fare for your selection.

Figure 22. Streamlit deployment

To streamline the project, we standardised the feature selection process across team members and implemented these changes on the web interface. During model training, some team members had trained separately or only applied transformers. To address this, we developed a comprehensive pipeline for the Best Model, eliminating the need for additional data processing on the web side. Additionally, we made adjustments to handle data essential for model training that does not require direct user input.

- We matched airport names with IATA codes so users can select by name, which is then converted to the appropriate code.
- To calculate distances between airports, we created a lookup table of average distances between airports, saved as a JSON file, ensuring the necessary data is loaded before model execution.

b. Web App

Our web application's standout feature lies in its ability to guide users on whether the current price for a ticket is fair or not by comparing the actual price with the model's predicted fare. The app starts by gathering key information from the customer—such as departure and arrival airports, travel dates, flight preferences, and any additional criteria they specify. Based on this data, the model generates a predicted price that reflects expected market trends for users' inputs.

- **Flight Selection:** Users can customize their flight search using various input options. Since the training dataset only included data up to 60 days in advance, we limited the date selection to a 60-day range to ensure optimal results. By default, the search can be performed without requiring inputs for airline category and additional data, though users can toggle to add specific details if desired. Tooltips are available for fields like airline and date, offering helpful information to address potential questions. To begin the search, users simply click the "Compare Prices" button, which initiates the process.

- **Actual price API:** After clicking the "Compare Prices" button, it calls two API (Figure 23). To have actual data on the web, we build API functions using Amadeus API. The API provides search flights between two cities, perform multi-city searches for longer itineraries and access one-way combinable fares to offer the cheapest options possible (Amadeus Documentation). This API retrieves results within one hour before and after the specified input time to compare and provide outcomes across this timeframe.

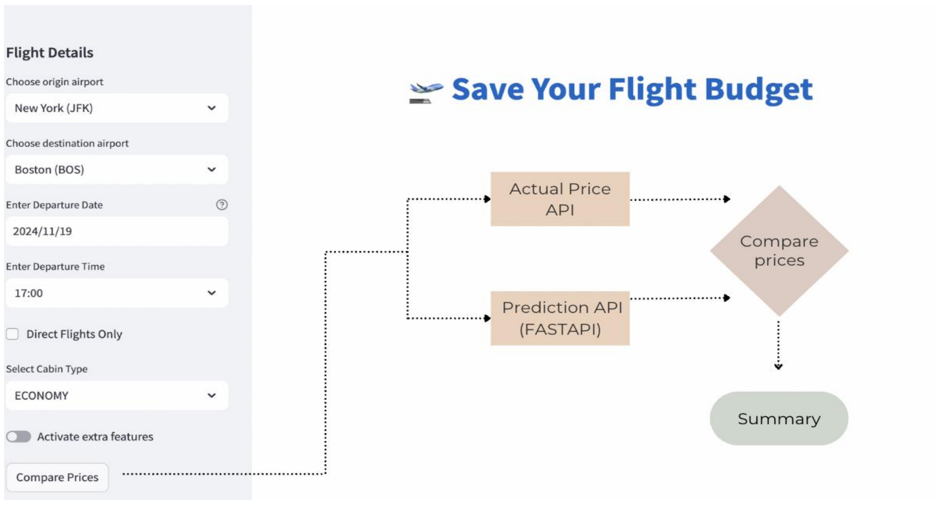
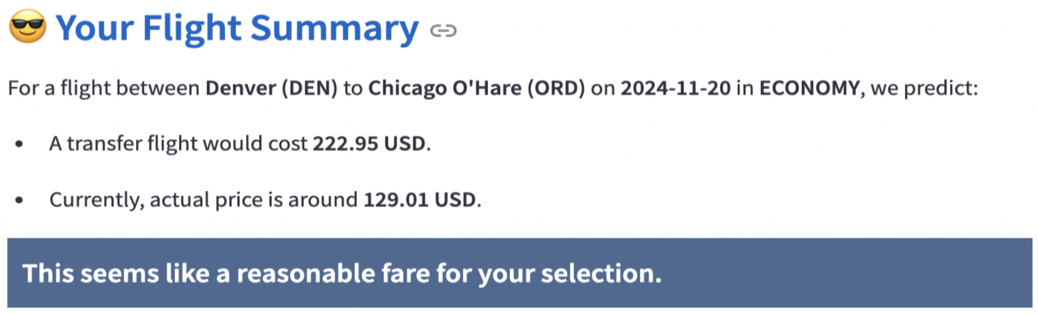


Figure 23. API Process

- **Prediction API:** This FASTAPI setup is designed to provide prediction results from the model. When the API is called, it loads the stored `Xgb_pipeline.joblib` model file. It then converts the input data into a DataFrame format for prediction processing. The API returns the predicted results.

- **Summary:** This container displays the user's input details and the resulting predictions. For direct flights, it compares the actual price to the predicted direct flight cost, and for transfer flights, it compares with the predicted transfer flight cost. This approach provides users with customised insights tailored to their selected flight type.



😎 Your Flight Summary

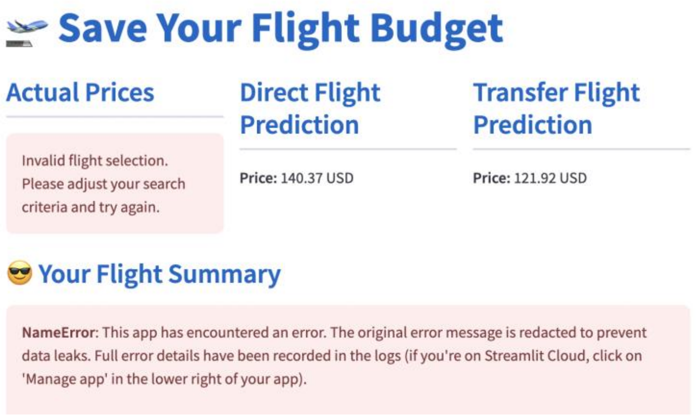
For a flight between **Denver (DEN)** to **Chicago O'Hare (ORD)** on **2024-11-20** in **ECONOMY**, we predict:

- A transfer flight would cost **222.95 USD**.
- Currently, actual price is around **129.01 USD**.

This seems like a reasonable fare for your selection.

Figure 24. Summary UI

- One minor issue we're currently facing is that, if the `actual price` value is missing, the app is unable to display the summary results. Although it indicates the lack of an `actual price` value elsewhere, the summary section displays an error instead. Resolving this limitation will improve the user experience by handling such cases more gracefully in the summary output. (Figure 25)


Save Your Flight Budget

Actual Prices

Invalid flight selection. Please adjust your search criteria and try again.

Direct Flight Prediction

Price: 140.37 USD

Transfer Flight Prediction

Price: 121.92 USD

Your Flight Summary

NameError: This app has encountered an error. The original error message is redacted to prevent data leaks. Full error details have been recorded in the logs (if you're on Streamlit Cloud, click on 'Manage app' in the lower right of your app).

Figure 25. The error in Summary output

Through this process, the app not only provides users valuable pricing context but also enhances their decision-making confidence. By offering real-time, data-driven insights into the current pricing landscape, users gain a clearer understanding of how the fare stacks up, helping them feel more informed and assured about their purchase. This approach is especially beneficial in fluctuating markets like airfare, where prices can vary dramatically day by day.

8. Collaboration

a. Individual Contributions

- *Taekjin Jeong:* Taekjin played a key role in developing several models to optimise our machine learning Algorithms. He worked to understand the dataset in depth and actively shared insights and ideas with the team. Taekjin was responsible for both front-end and back-end development, deploying the project as a web application with a strong focus on user interface and experience.

- Dhruv Joshi - I took on multiple responsibilities, which mainly consisted of developing and fine-tuning the XGBoost model, which ultimately delivered the best performance among all the models we evaluated. To improve model performance, I conducted model evaluation based on various features that seemed important to the predictive values. Metrics such as RMSE were used to ensure consistent performance across different data subsets. Furthermore, I also helped my team members in deploying the Streamlit app. Lastly, I helped the team in writing the report for the project—executive summary, business understanding, Business Impact and Benefits, Data Privacy and Ethical Concerns, my approach related to modelling the XGBoost model, and insights on team collaboration.

- Raymond Tsui: Focused on building out functions for the team to load and clean data, github creation and was responsible for training the Neural Network and LGBM models. I was also responsible for developing the functions to help the team evaluate which model had the highest performance. Furthermore, in the report I was responsible for the data preparation.
  I developed and implemented data loading and cleaning functions to ensure data consistency across the team. Established the GitHub repository for efficient collaboration and version control. Trained and optimized Neural Network and LightGBM models. For the final report i contributed to the EDA, Data preparation and visualized predictions in evaluation to derive meaningful insights and support choosing the best model.

b. Group Dynamic

Throughout the project, the team maintained an open and collaborative environment via Microsoft teams which fostered effective communication and coordination. Regular meetings were held to discuss progress, address challenges, and align on next steps. Github was used to share the code related to the project which indicated each member's progress and tasks completed. During meetings, team members frequently used screen sharing to walk through their code, demonstrate model results and troubleshoot issues collaboratively. Furthermore, each member was assigned specific roles based on their expertise, ensuring balanced workload distribution. To ensure smooth collaboration, the team also adopted key practices such as check-in meetings, ensured meetings were conducted regularly which allowed the members to share updates and make adjustments to the project. Team members also reviewed each other's code and reports to ensure quality and accuracy. We valued the diverse perspective, and decisions were made collaboratively after careful consideration of projects performance and deadlines.

c. Ways of Working Together

To manage the project effectively, the team adopted an Agile-inspired framework, emphasising flexibility, iterative progress, and continuous improvement. The deliverables were defined at the start of the project, the team used sprints to plan, execute and review the work during every meeting. This approach allowed for regular assessment of progress and quick adaptation to any challenges that arose. To ensure that everyone was aware of the progress, updates were regularly posted on Microsoft teams chat channel. Tasks were assigned a due date in the integrated planner which allowed the team to break down the project into manageable tasks, each of which were assigned to specific members. For model selection and hyperparameter tuning, team members presented their findings, and decisions were made collectively based on performance metrics and interpretability.

Specific Tools Used

Some of the tools which were used for better collaboration were as follows:

- **Microsoft Teams** - It was used for communication, file sharing and task management.
- **Microsoft loop** - To-do list was made and shared on this platform.
- **GitHub** - This was used for managing code, models and documentations. It consists of the model comparison worked on by every member of the team.
- **Planner** - Integrated planner was used to organise tasks and manage deadlines.
- **Streamlit** - For building the interactive application and integrating the team's machine learning models.

d. Issues Faced

- **Missing fields** - The dataset given to us contained missing values in some fields, thus incomplete data posed challenges during preprocessing and model training. The team made a total distance travelled table to ensure we could better preprocess the data.

- **Lack of features into flight demand** – Airfares can fluctuate significantly for the same route, even on different days. One potential factor influencing these price variations is fluctuating demand. By incorporating data from our corporate travel booking partners, we could gain valuable insights into customer demand patterns. Additionally, integrating holiday date data into our model could further enhance its predictive accuracy.
- **Limited Timeframe of Dataset** - One of the primary challenges faced was the limited timeframe of the dataset. Since the data only spanned a short period, it lacked sufficient diversity to capture longer-term or seasonal trends in airfare pricing. This was mainly handled by the team by conducting performance of the various models used for this study on simulated data points representing different time periods.
- **Time Management** - With multiple deliverables including model training, app development, report and deployment. Managing time effectively was a small hurdle faced by the group. We handled this by employing a to-do list which indicated the tasks to be completed and team members who took ownership of them.
- **Deployment of the Streamlit App and FastAP**I - We encountered difficulties in deploying on Render for the FastAPI and Streamlit applications as Render provides a single web service. To resolve this, we deployed the FastAPI on Render and Streamlit app on Streamlit Cloud.

Recommendations for Future Collaborations

- **Conduct a comprehensive risk assessment** - This can be used to identify the potential challenges early and plan mitigation strategies. Here, we can discuss data related risks, team collaboration and stakeholder and user risks.
- **Invest in More Diverse Data Sources** - It is important that we prioritise in obtaining data which spans over multiple timeframes and scenarios to improve the robustness of the models.

9. Conclusion

Throughout this project, we aimed to develop a predictive web application that accurately assesses flight prices and provides users with customised insights based on their flight details. By leveraging various machine learning models including Linear Regression, XGB Regressor, Neural Network and LightGBM, we achieved high prediction accuracy, as evidenced by RMSE reductions and model comparisons. Furthermore, we achieved a seamless deployment process for our web application by implementing a CI/CD pipeline. This allowed for efficient updates and model improvements without interrupting the user experience. Collaboration with team members played a key role in identifying and addressing potential user pain points, which helped us continuously improve the application's usability and functionality. Next step is data expansion and model enhancement. Expanding the dataset will enable more comprehensive predictions, refine model insights, and support a broader range of use cases, ultimately leading to an even better user experience.

10. References

- Amadeus Documentation (https://developers.amadeus.com/self-service/category/flights/api-doc/flight-offers-search)
- Streamlit Documentation (https://docs.streamlit.io/develop/api-reference)