Flight Ticket & Status Analysis

Xinan Wang, Zhiqing Wang, Gaohaonan He, Wenwei Li

April 20th, 2023

I. Introduction

Predicting flight prices and delays has become increasingly important in recent years, as airlines seek to optimize their operations and improve the travel experiences of their customers. By developing models that can accurately predict flight prices and delays based on various factors, such as departure time, airline, route, and historical data, airlines can adjust their pricing and scheduling strategies in real-time, allocate resources more effectively, and minimize the impact of delays on their customers.

Moreover, predicting flight prices and delays can also benefit travelers by providing them with more information about their flight options, allowing them to make more informed decisions about their travel plans and avoid unexpected expenses. Overall, the ability to predict flight prices and delays has the potential to have a significant impact on the airline industry and the travel experiences of millions of people around the world.

1. Objective of Analysis

The goal of our analysis is to develop a machine learning model that predicts flight prices and delays accurately. The purpose of this project is to assist travelers in making informed decisions about their travel plans by providing reliable predictions about the cost of air travel and potential delays. By entering information such as the desired travel dates and destinations, our application will generate predictions on flight prices and delays, enabling travelers to plan their trips more effectively.

The significance of this project lies in the fact that it can help travelers save money by predicting the best time to book a flight and avoid potential disruptions caused by flight delays. In turn, this can have a significant impact on the travel industry, which generates billions of dollars in revenue each year.

Developing a model that predicts flight prices and delays is a non-trivial task as it involves analyzing large amounts of data from multiple sources, including airline schedules, and historical flight information. It requires expertise in data science and machine learning algorithms to identify patterns in this data and make accurate predictions.

In conclusion, our project aims to provide travelers with reliable predictions on flight prices and delays, allowing them to plan their trips more effectively. By using data science and machine learning techniques, we can develop models that accurately predict flight prices and delays, benefiting both travelers and the travel industry.

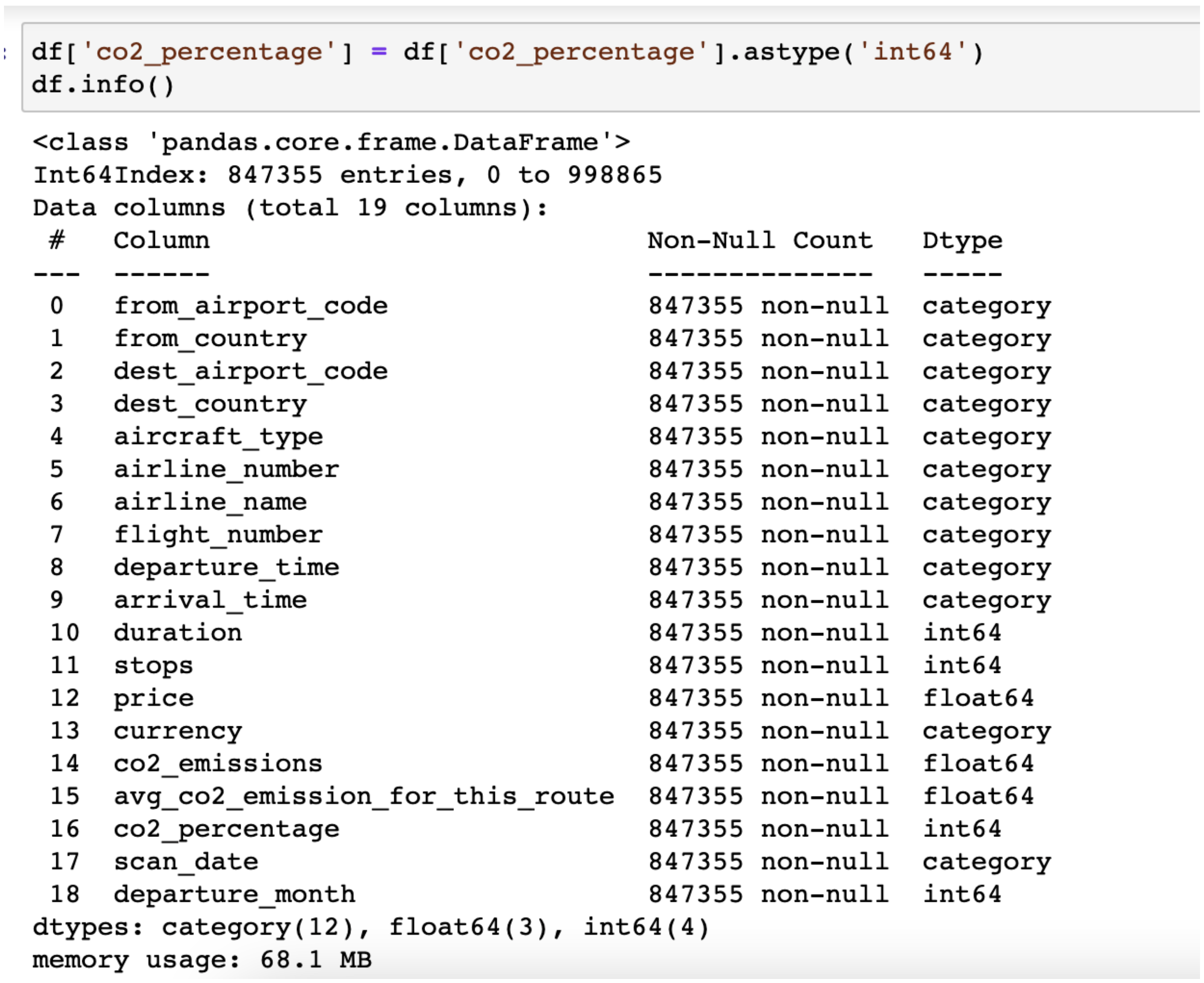
II. Dataset

For the flight price and delay prediction model, we have used two publicly available datasets. The first dataset is gathered from the TranStats data library, which contains data on US (United States) domestic flights from January 2018 to December 2019. This dataset includes information such as flight dates, origin and destination airports, scheduled and actual departure and arrival times, and delay time. The dataset also includes information on the type of carrier, whether it is a major or regional carrier, and the distance of the flight.

Another dataset is gathered from a publicly available dataset on Kaggle, which covers popular airports from Europe, Asia, America, and Africa, and contains about 1 million rows of records. The dataset includes the CO2 emissions for each flight, as well as the average CO2 emissions for each flight route. The CO2 percentage for each flight is calculated using the CO2 emission and the average CO2 emission for the respective flight route. The CO2 emissions of a flight are typically calculated based on the point-to-point distance of the flight and the estimated fuel burn for the specific aircraft category being used for the flight. This dataset is provided by BarkingData, a company specializing in web mining and web data harvesting from the world wide web, including mobile apps. They have built over 5000 datasets for researchers, analysts, scholars, retailers, and more.

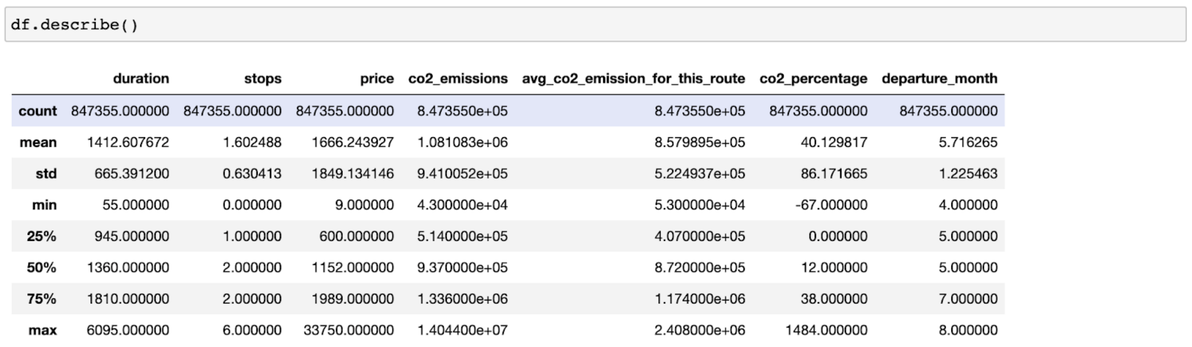
Both datasets contain a large amount of data with a range of different features that are relevant to predicting flight prices and delays. The datasets are highly comprehensive, including data on thousands of flights, carriers, airports, and routes, making them ideal for training a machine learning model. The datasets also contain both numerical and categorical features, which will allow us to build a more robust predictive model.

The following is a statistical summary of all the features present in the flight price dataset.



```
df['co2_percentage'] = df['co2_percentage'].astype('int64')
df.info()
```

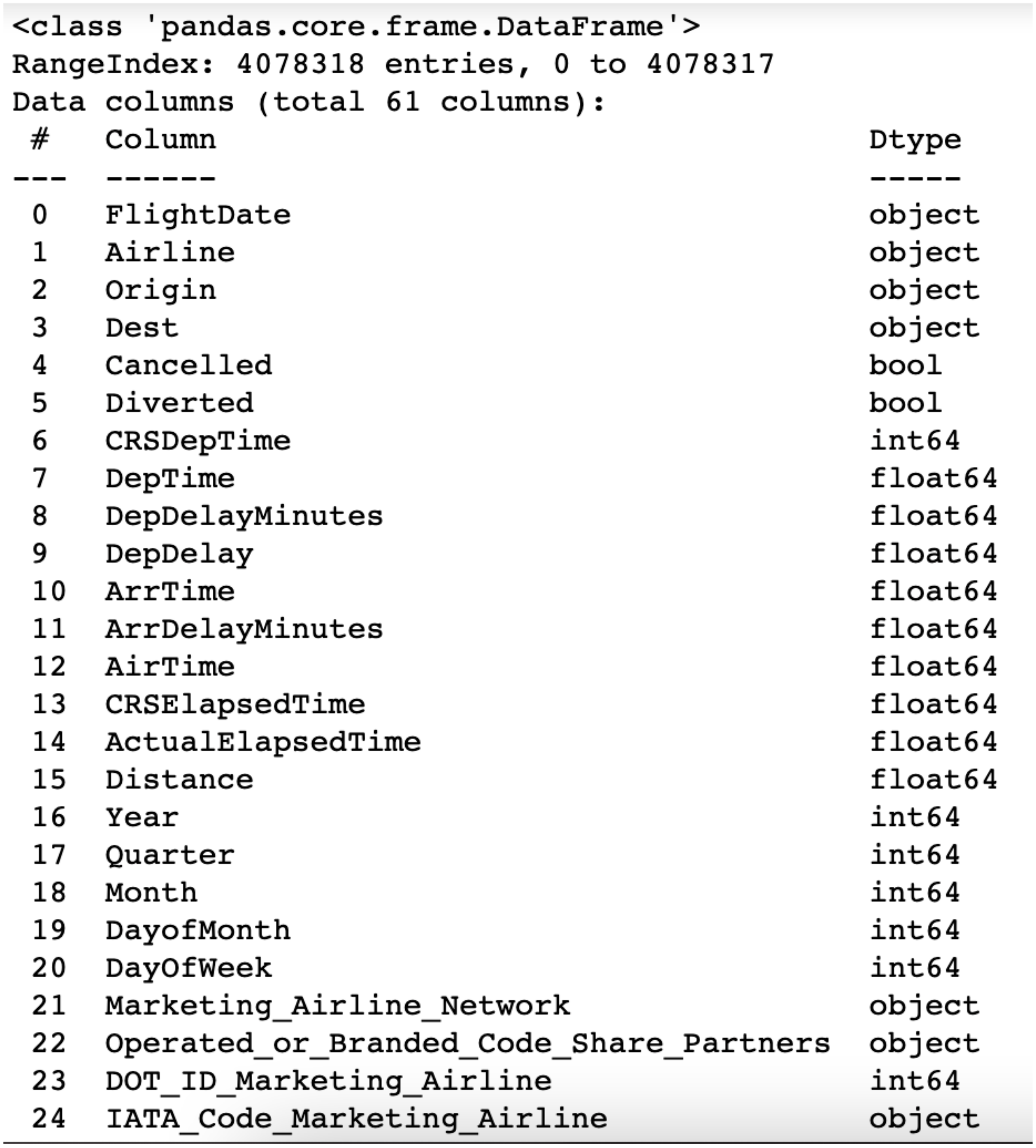
```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 847355 entries, 0 to 998865
Data columns (total 19 columns):
 #   Column                          Non-Null Count   Dtype
---  ------                          --------------   -----
 0   from_airport_code               847355 non-null  category
 1   from_country                    847355 non-null  category
 2   dest_airport_code               847355 non-null  category
 3   dest_country                    847355 non-null  category
 4   aircraft_type                   847355 non-null  category
 5   airline_number                  847355 non-null  category
 6   airline_name                    847355 non-null  category
 7   flight_number                   847355 non-null  category
 8   departure_time                  847355 non-null  category
 9   arrival_time                    847355 non-null  category
 10  duration                        847355 non-null  int64
 11  stops                           847355 non-null  int64
 12  price                           847355 non-null  float64
 13  currency                        847355 non-null  category
 14  co2_emissions                   847355 non-null  float64
 15  avg_co2_emission_for_this_route 847355 non-null  float64
 16  co2_percentage                  847355 non-null  int64
 17  scan_date                       847355 non-null  category
 18  departure_month                 847355 non-null  int64
dtypes: category(12), float64(3), int64(4)
memory usage: 68.1 MB
```

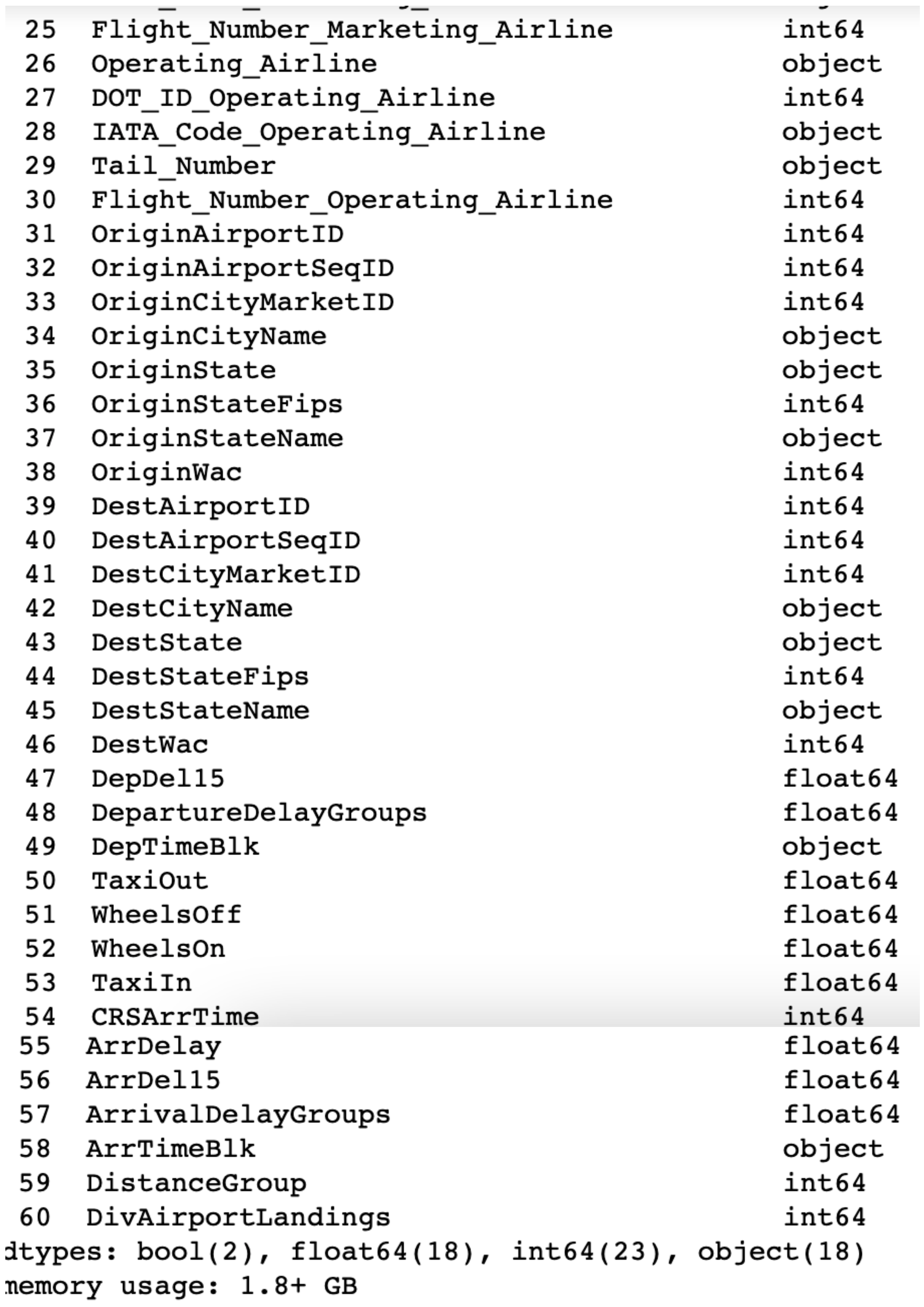


```
df.describe()
```

	duration	stops	price	co2_emissions	avg_co2_emission_for_this_route	co2_percentage	departure_month
count	847355.000000	847355.000000	847355.000000	8.473550e+05	8.473550e+05	847355.000000	847355.000000
mean	1412.607672	1.602488	1666.243927	1.081083e+06	8.579895e+05	40.129817	5.716265
std	665.391200	0.630413	1849.134146	9.410052e+05	5.224937e+05	86.171665	1.225463
min	55.000000	0.000000	9.000000	4.300000e+04	5.300000e+04	-67.000000	4.000000
25%	945.000000	1.000000	600.000000	5.140000e+05	4.070000e+05	0.000000	5.000000
50%	1360.000000	2.000000	1152.000000	9.370000e+05	8.720000e+05	12.000000	5.000000
75%	1810.000000	2.000000	1989.000000	1.336000e+06	1.174000e+06	38.000000	7.000000
max	6095.000000	6.000000	33750.000000	1.404400e+07	2.408000e+06	1484.000000	8.000000

The following is a statistical summary of all the features present in the flight delay dataset.


```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4078318 entries, 0 to 4078317
Data columns (total 61 columns):
 #   Column                                   Dtype  
---  ------                                   -----  
 0   FlightDate                               object 
 1   Airline                                  object 
 2   Origin                                   object 
 3   Dest                                     object 
 4   Cancelled                                bool   
 5   Diverted                                 bool   
 6   CRSDepTime                               int64  
 7   DepTime                                  float64
 8   DepDelayMinutes                          float64
 9   DepDelay                                 float64
 10  ArrTime                                  float64
 11  ArrDelayMinutes                          float64
 12  AirTime                                  float64
 13  CRSElapsedTime                           float64
 14  ActualElapsedTime                        float64
 15  Distance                                 float64
 16  Year                                     int64  
 17  Quarter                                  int64  
 18  Month                                    int64  
 19  DayofMonth                               int64  
 20  DayOfWeek                                int64  
 21  Marketing_Airline_Network                object 
 22  Operated_or_Branded_Code_Share_Partners  object 
 23  DOT_ID_Marketing_Airline                 int64  
 24  IATA_Code_Marketing_Airline              object 
```



```
 25  Flight_Number_Marketing_Airline         int64
 26  Operating_Airline                       object
 27  DOT_ID_Operating_Airline                int64
 28  IATA_Code_Operating_Airline             object
 29  Tail_Number                             object
 30  Flight_Number_Operating_Airline         int64
 31  OriginAirportID                         int64
 32  OriginAirportSeqID                      int64
 33  OriginCityMarketID                      int64
 34  OriginCityName                          object
 35  OriginState                             object
 36  OriginStateFips                         int64
 37  OriginStateName                         object
 38  OriginWac                               int64
 39  DestAirportID                           int64
 40  DestAirportSeqID                        int64
 41  DestCityMarketID                        int64
 42  DestCityName                            object
 43  DestState                               object
 44  DestStateFips                           int64
 45  DestStateName                           object
 46  DestWac                                 int64
 47  DepDel15                                float64
 48  DepartureDelayGroups                    float64
 49  DepTimeBlk                              object
 50  TaxiOut                                 float64
 51  WheelsOff                               float64
 52  WheelsOn                                float64
 53  TaxiIn                                  float64
 54  CRSArrTime                              int64
 55  ArrDelay                                float64
 56  ArrDel15                                float64
 57  ArrivalDelayGroups                      float64
 58  ArrTimeBlk                              object
 59  DistanceGroup                           int64
 60  DivAirportLandings                      int64
dtypes: bool(2), float64(18), int64(23), object(18)
memory usage: 1.8+ GB
```

1. Feature Descriptions

In this project, we are interested in utilizing the flight data features to perform analysis and make predictions. Each of these features describes some interesting and different aspects of a flight. The following table shows the features and their description which have been used in our flight price analysis:

Feature	Description
from_country	The country where the flight departs from
dest_country	The country where the flight is headed to
airline_name	The name of the airline company operating the flight
duration	The duration of the flight in minutes
stops	The number of stops during the flight
co2_emissions	The amount of carbon dioxide emitted during the flight in kg
departure_time	The time in which the flight departs
price	The cost of the flight in the specified currency

The following table shows the features and their description which have been used in our flight delay analysis:

Feature Name	Description
FlightDate	The date of the flight.
Airline	The name or code of the airline operating the flight.
Origin	The airport code or name of the origin airport for the flight.
Dest	The airport code or name of the destination airport for the flight.
Year	The year in which the flight was scheduled.
Month	The month in which the flight was scheduled.
DestStateName	The name of the state in which the destination airport is located.
DepDelayMinutes	The number of minutes by which the flight was delayed at the departure airport.
Canceled	A binary indicator of whether the flight was canceled (1) or not (0).
Diverted	A binary indicator of whether the flight was diverted to a different airport (1) or not (0).

2. Why data science and machine learning?

The use of data science is essential to this project because it enables us to analyze a large amount of data and identify patterns that may not be readily apparent. By leveraging these techniques, we can find the trends of the flight prices and delays based on historical data and other factors.

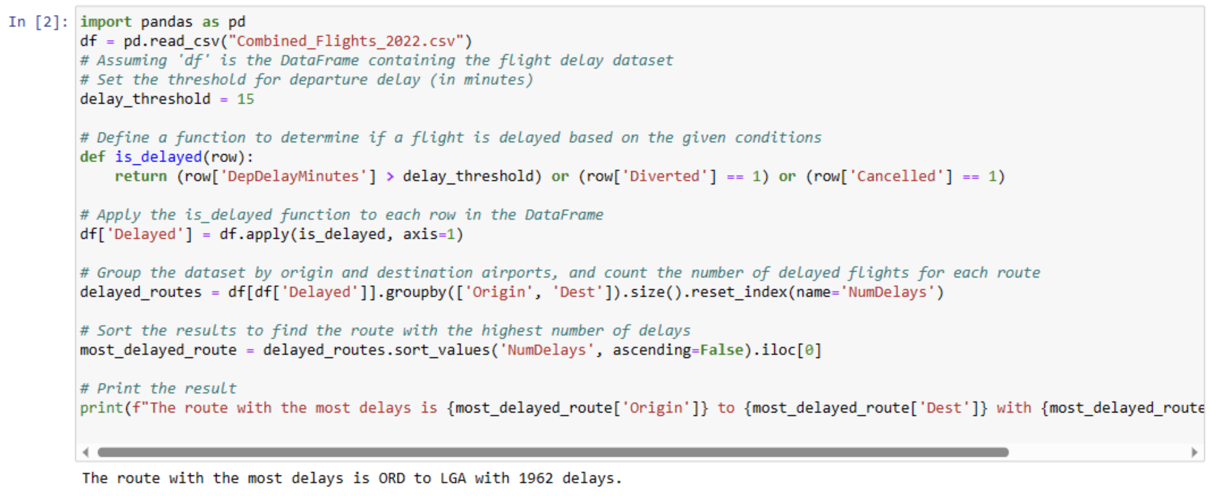
Machine learning is also very crucial. In this analysis, we leverage the power of machine learning models to accurately predict flight prices. The ability to discern patterns in ticket prices is crucial, as it enables us to offer users reliable forecasts of future airfares. Furthermore, by identifying trends in travel patterns, these models can help businesses make data-driven decisions regarding flight schedules, capacity planning, and marketing strategies. In essence, the predictive insights gained through machine learning algorithms can have a significant impact on the airline industry and the overall travel experience for consumers.

In the methodology part below, we will mention how we use data science and machine learning to conduct the analysis.

3. Interesting analysis finding

For the delay analysis, we have added more functions to find what could be the worst case delay. Some code that could potientially achieve this task are like this:

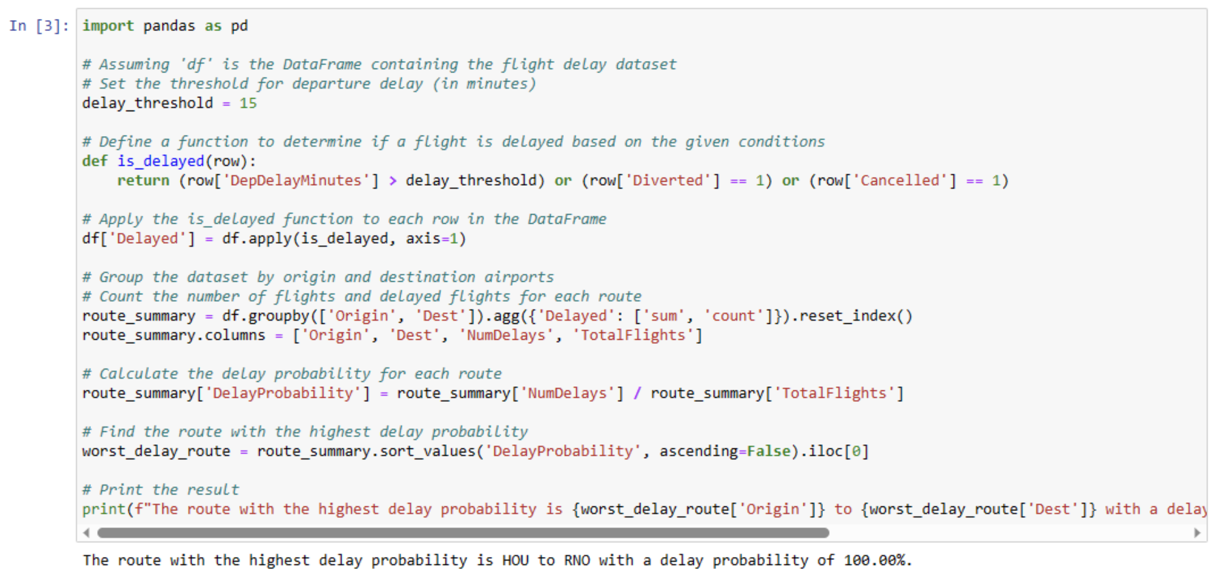
“Worst” = ”most delayed records”



```
In [2]: import pandas as pd
        df = pd.read_csv("Combined_Flights_2022.csv")
        # Assuming 'df' is the DataFrame containing the flight delay dataset
        # Set the threshold for departure delay (in minutes)
        delay_threshold = 15

        # Define a function to determine if a flight is delayed based on the given conditions
        def is_delayed(row):
            return (row['DepDelayMinutes'] > delay_threshold) or (row['Diverted'] == 1) or (row['Cancelled'] == 1)

        # Apply the is_delayed function to each row in the DataFrame
        df['Delayed'] = df.apply(is_delayed, axis=1)

        # Group the dataset by origin and destination airports, and count the number of delayed flights for each route
        delayed_routes = df[df['Delayed']].groupby(['Origin', 'Dest']).size().reset_index(name='NumDelays')

        # Sort the results to find the route with the highest number of delays
        most_delayed_route = delayed_routes.sort_values('NumDelays', ascending=False).iloc[0]

        # Print the result
        print(f"The route with the most delays is {most_delayed_route['Origin']} to {most_delayed_route['Dest']} with {most_delayed_route

        The route with the most delays is ORD to LGA with 1962 delays.
```

Based on the analysis output, the route from ORD (Chicago O'Hare airport) to LGA (New York City LaGuardia airport) has the highest frequency of flight delays. Notably, our visualization in the exploratory data analysis part below reveals that Chicago tops the list of origin cities with the highest frequency of flight delays. Furthermore, both Chicago and New York occupy the top two spots for the most common destination cities with flight delays. These findings are particularly interesting and highlight the importance of considering factors such as airport location and historical air performance when make the travel plan.

“Worst” = ”highest delay-likelyhood”

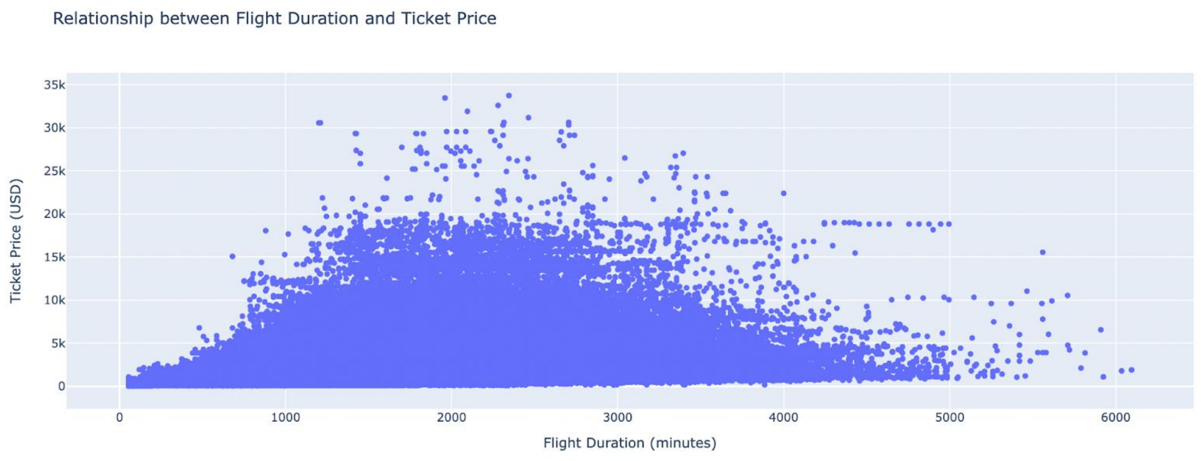


```
In [3]: import pandas as pd

        # Assuming 'df' is the DataFrame containing the flight delay dataset
        # Set the threshold for departure delay (in minutes)
        delay_threshold = 15

        # Define a function to determine if a flight is delayed based on the given conditions
        def is_delayed(row):
            return (row['DepDelayMinutes'] > delay_threshold) or (row['Diverted'] == 1) or (row['Cancelled'] == 1)

        # Apply the is_delayed function to each row in the DataFrame
        df['Delayed'] = df.apply(is_delayed, axis=1)

        # Group the dataset by origin and destination airports
        # Count the number of flights and delayed flights for each route
        route_summary = df.groupby(['Origin', 'Dest']).agg({'Delayed': ['sum', 'count']}).reset_index()
        route_summary.columns = ['Origin', 'Dest', 'NumDelays', 'TotalFlights']

        # Calculate the delay probability for each route
        route_summary['DelayProbability'] = route_summary['NumDelays'] / route_summary['TotalFlights']

        # Find the route with the highest delay probability
        worst_delay_route = route_summary.sort_values('DelayProbability', ascending=False).iloc[0]

        # Print the result
        print(f"The route with the highest delay probability is {worst_delay_route['Origin']} to {worst_delay_route['Dest']} with a delay

        The route with the highest delay probability is HOU to RNO with a delay probability of 100.00%.
```

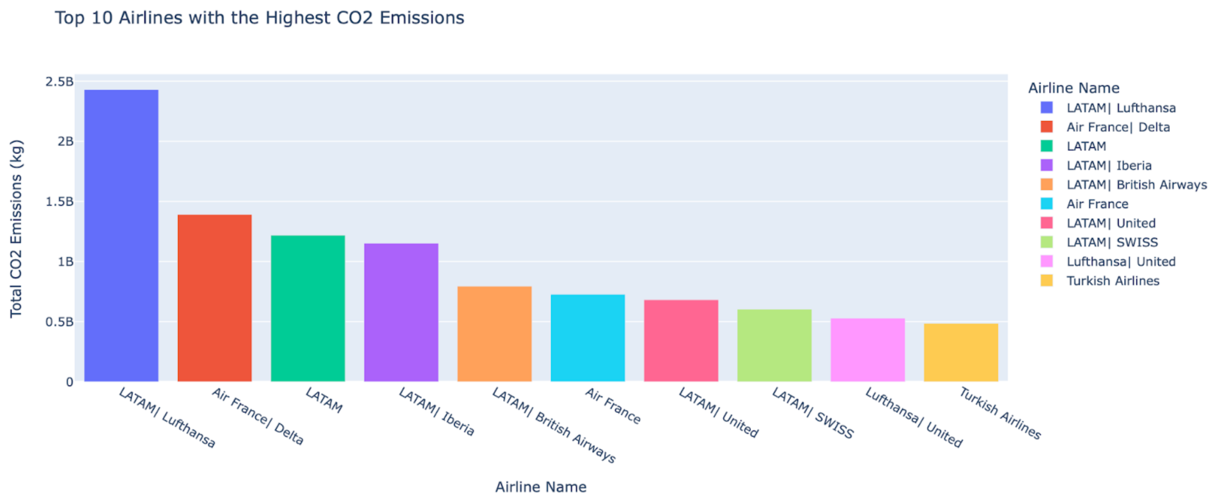
The analysis output has indicated that the route from HOU (Houston William P. Hobby Airport) to RNO (Reno-Tahoe International Airport) has the highest probability of flight delays, with a 100% likelihood of experiencing delays. This finding is particularly noteworthy and emphasizes the importance of factoring in potential delays when planning air travel on this particular route.

III. Exploratory Data Analysis

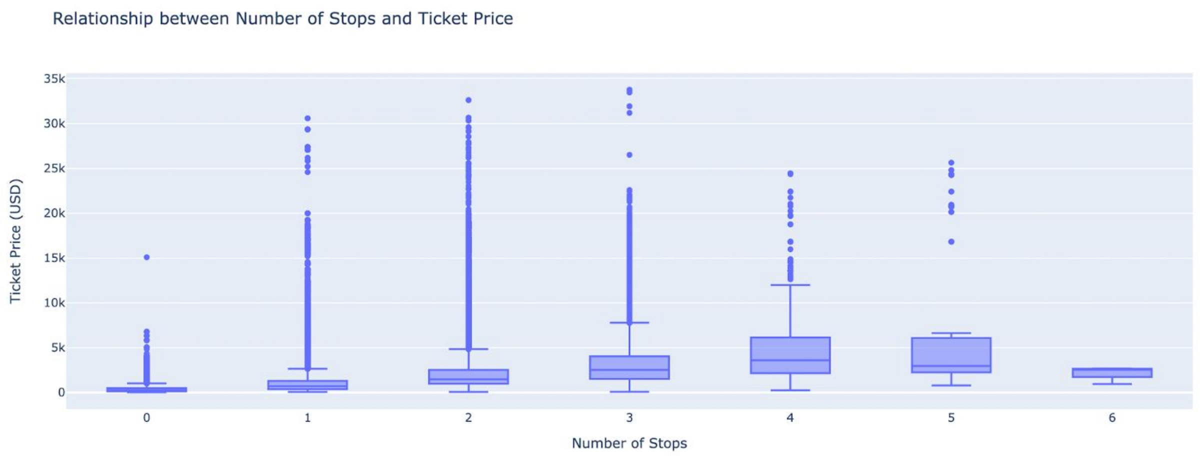
1. Price Prediction Dataset



Scatter Plot Interpretation: The scatter plot depicted above displays the relationship between flight duration (in minutes) and ticket price (in US dollars), with each data point on the graph representing a distinct row from the dataset. From the plot, it is evident that the majority of the flights have a duration between 16 to 50 hours, equivalent to 1000 to 3000 minutes. This range of flight duration aligns with our expectations since the Price Prediction dataset includes only international flights. Furthermore, we can observe that the typical flight price falls below 10,000 US dollars, which is consistent with the dataset's scope.

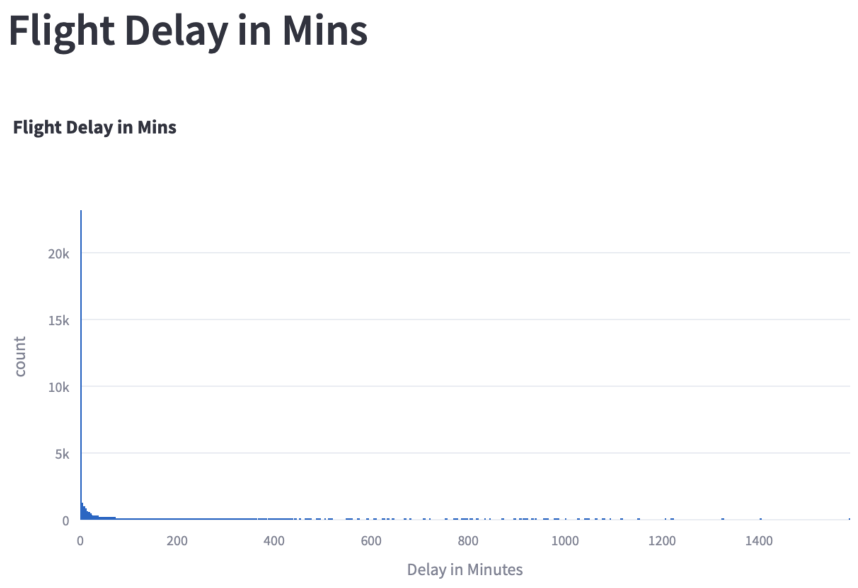


Bar Plot Interpretation: The presented bar plot illustrates the top 10 airline companies with the highest carbon dioxide emissions, with the total amount of carbon dioxide emission in kilograms (kg) on the y-axis and the respective airline companies on the x-axis. Upon inspection, the bar plot reveals that LATAM has the highest total carbon dioxide emission among the listed airline companies. The provided information empowers individuals to make informed decisions about their travel choices, considering the environmental impact of different airline companies.



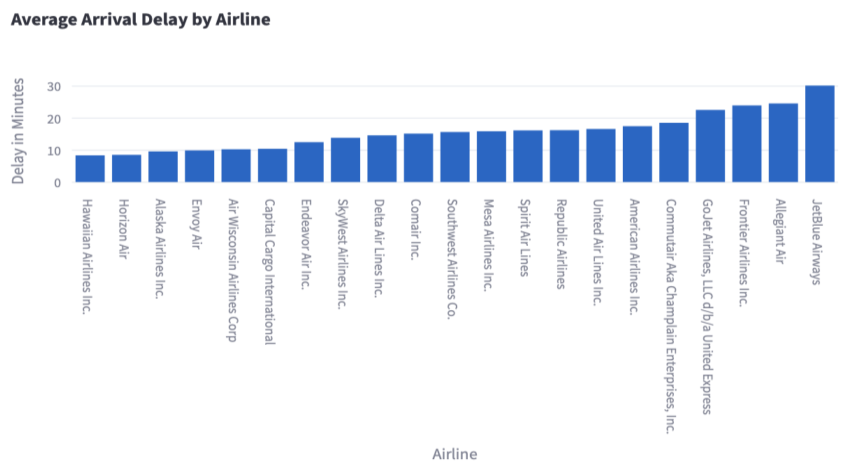
Box Plot Interpretation: The box plot presented above depicts the distribution of prices for a range of flight stops and highlights any potential outliers within each group. Upon examination, it is apparent that flight groups with two and three stops exhibit the highest number of outliers. Moreover, the width of the box provides an indication of the level of variation, which is more pronounced for groups with four and five stops. In essence, the plot suggests that the ticket prices for flights with four and five stops exhibit a more substantial variation compared to other flight groups.

2. Flight Delay Dataset



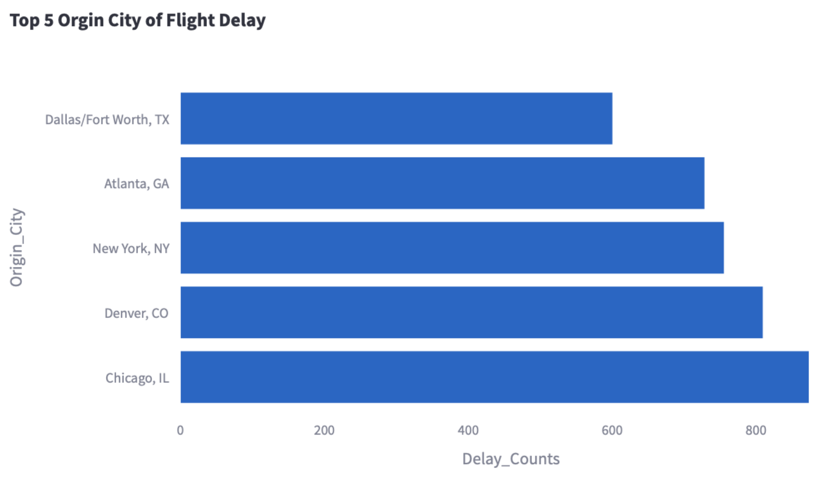
Flight Delay Distribution: The presented histogram illustrates the distribution of flight delays in minutes, with the frequency or count of flights on the y-axis. Upon inspection, it is evident that the distribution is highly left-skewed, indicating that the majority of flight delays are under 30 minutes. This observation is supported by the fact that the peak of the histogram is located towards the lower end of the x-axis. Therefore, we can conclude that the majority of flights experience minor delays, with a smaller number of flights experiencing more significant delays.

Average Arrival Delay by Airline

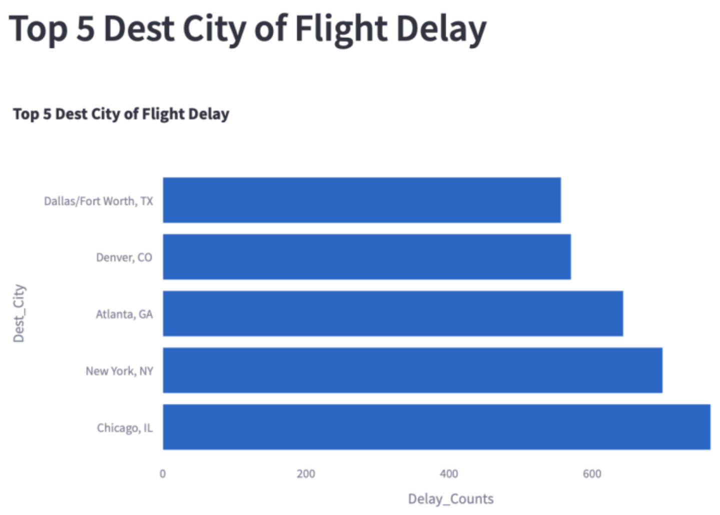


Bar Plot Interpretation: The ranking of airline companies according to their typical arrival delay is shown in the bar plot. The airline companies and accompanying delay in minutes columns within the dataset were grouped to generate this ranking, which was then obtained by taking the mean of the results. In other words, the mean delay across all flights operated by a certain airline was used to calculate the average arrival delay for each airline firm. We were able to compare and rate the airline firms using this method, giving us important information about whether airline companies had a better or poorer track record for on-time arrivals.

Top 5 Origin City of Flight Delay



Bar Plot Interpretation: The bar plot presented above depicts the top 5 origin cities with the highest count of delayed flights. The graph clearly indicates that Chicago has the highest number of flights affected by delays, suggesting that it is the city with the most significant issue regarding flight punctuality.



Bar Plot Interpretation: The bar plot displayed illustrates the top five destination cities with the highest count of delayed flights, with Chicago ranking first in delay counts. This finding is likely due to the fact that Chicago is a major city in the U.S. and experiences a high volume of flights arriving and departing on a daily basis.

IV. Methodology

The objective of our flight analysis project is threefold: to develop a machine learning model for predicting flight prices, to perform data analysis to estimate potential flight delays, and to create a user interface that allows users to input their flight details and receive both price predictions and delay estimates.

1. Machine Learning Model for Flight Price Prediction

Our strategy is to treat the flight price prediction task as a regression problem. We train a machine learning model on a dataset consisting of historical flight prices, as well as other relevant features such as departure and arrival airports, airline, travel dates, and booking details. To find the most effective model, we compare several regression algorithms, such as Linear Regression, Decision Trees, Random Forest Regressors, and KNN Regressors. Our goal is to achieve the highest possible accuracy in predicting flight prices, thereby providing users with reliable estimates to inform their booking decisions.

A. Overview of Models Used

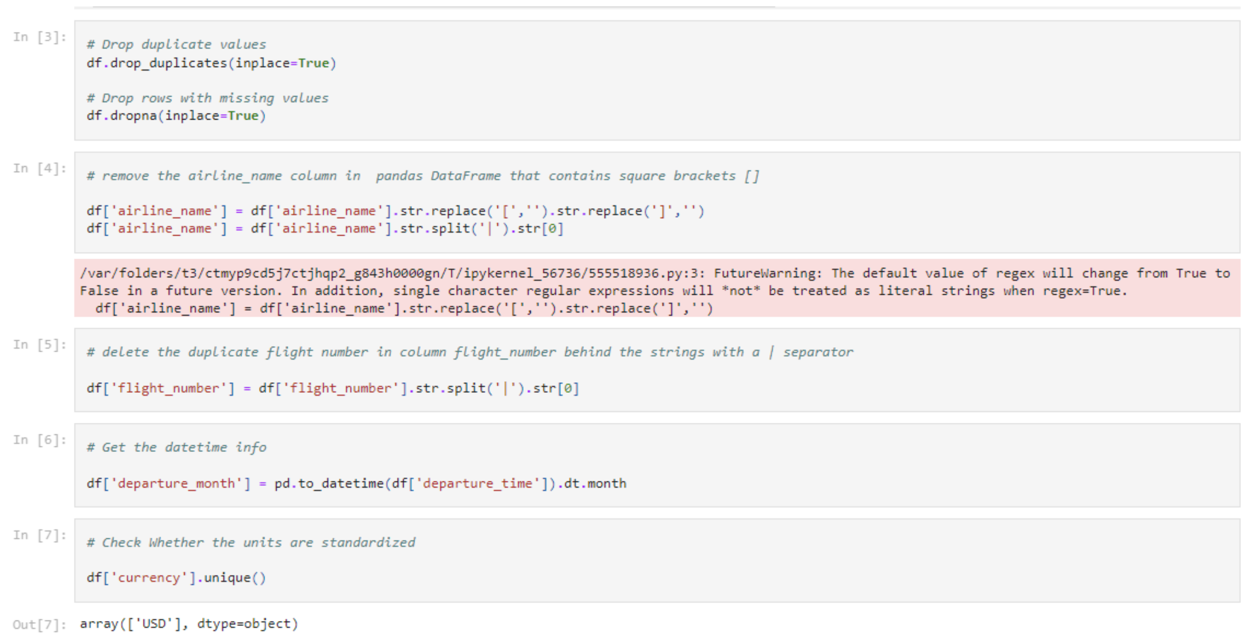
We used four different machine learning models for our analysis:

- Decision Tree Regressor
- Random Forest Regressor
- Linear Regression
- K-Nearest Neighbors (KNN) Regressor

These models were chosen due to their varying approaches and capabilities in handling regression problems. Decision Trees and Random Forests are particularly useful for non-linear relationships in the data, while Linear Regression is a simple yet powerful model for linear relationships. KNN Regressor is an instance-based learning algorithm that can be useful for problems with limited features.

B. Technical Approach and Data Preparation

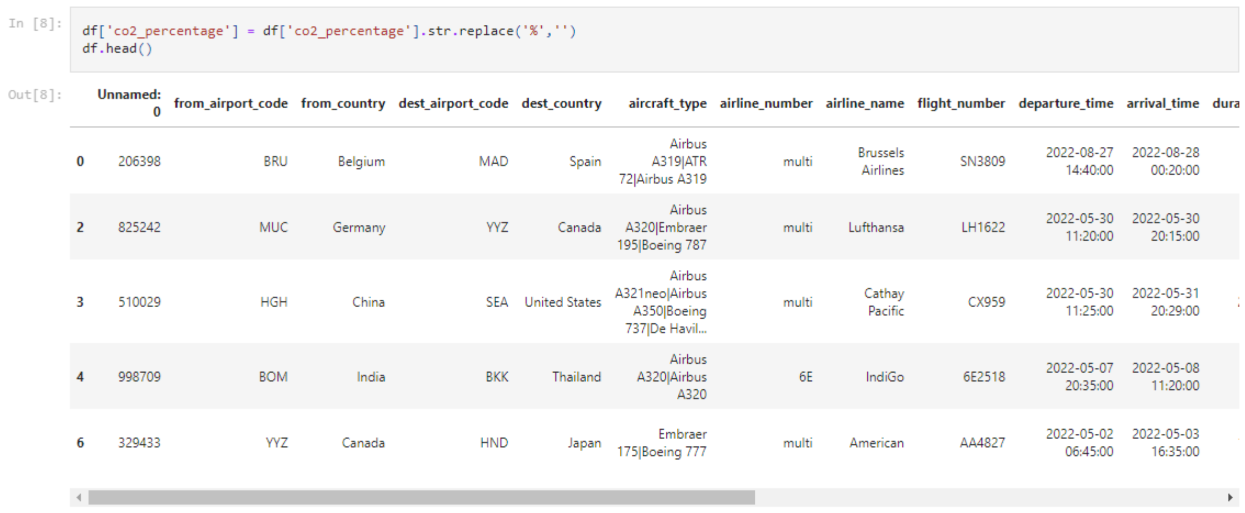
（a）Before applying the models, we needed to clean the dataset to ensure its usability. We took the following steps to clean the data:



```
In [3]:  # Drop duplicate values
         df.drop_duplicates(inplace=True)

         # Drop rows with missing values
         df.dropna(inplace=True)

In [4]:  # remove the airline_name column in  pandas DataFrame that contains square brackets []

         df['airline_name'] = df['airline_name'].str.replace('[','').str.replace(']','')
         df['airline_name'] = df['airline_name'].str.split('|').str[0]

/var/folders/t3/ctmyp9cd5j7ctjhqp2_g843h0000gn/T/ipykernel_56736/555518936.py:3: FutureWarning: The default value of regex will change from True to
False in a future version. In addition, single character regular expressions will *not* be treated as literal strings when regex=True.
  df['airline_name'] = df['airline_name'].str.replace('[','').str.replace(']','')

In [5]:  # delete the duplicate flight number in column flight_number behind the strings with a | separator

         df['flight_number'] = df['flight_number'].str.split('|').str[0]

In [6]:  # Get the datetime info

         df['departure_month'] = pd.to_datetime(df['departure_time']).dt.month

In [7]:  # Check Whether the units are standardized

         df['currency'].unique()

Out[7]: array(['USD'], dtype=object)
```

- Removed duplicate rows.

- Dropped rows with missing values.
- Removed square brackets from the 'airline_name' column and extracted the first airline name in case of multiple names.
- Extracted the first flight number from the 'flight_number' column in case of duplicates separated by a '|' character.
- Converted 'departure_time' to a datetime object and extracted the departure month as a separate column.

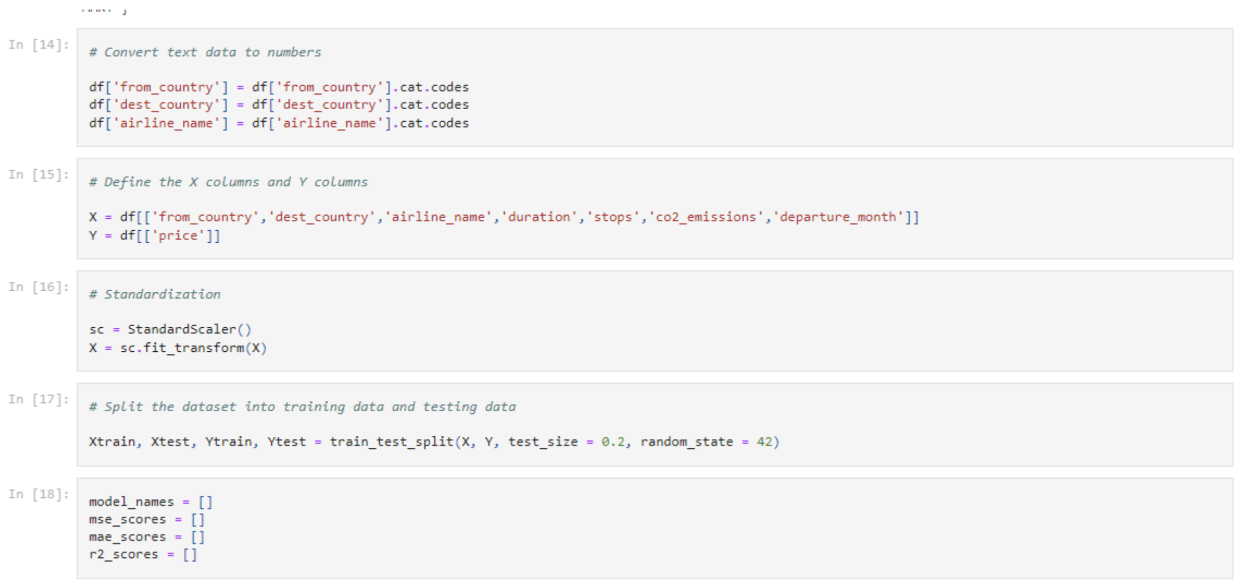


In [8]:
```
df['co2_percentage'] = df['co2_percentage'].str.replace('%','')
df.head()
```

Out[8]:

	Unnamed: 0	from_airport_code	from_country	dest_airport_code	dest_country	aircraft_type	airline_number	airline_name	flight_number	departure_time	arrival_time	dura
0	206398	BRU	Belgium	MAD	Spain	Airbus A319|ATR 72|Airbus A319	multi	Brussels Airlines	SN3809	2022-08-27 14:40:00	2022-08-28 00:20:00	
2	825242	MUC	Germany	YYZ	Canada	Airbus A320|Embraer 195|Boeing 787	multi	Lufthansa	LH1622	2022-05-30 11:20:00	2022-05-30 20:15:00	
3	510029	HGH	China	SEA	United States	Airbus A321neo|Airbus A350|Boeing 737|De Havil...	multi	Cathay Pacific	CX959	2022-05-30 11:25:00	2022-05-31 20:29:00	
4	998709	BOM	India	BKK	Thailand	Airbus A320|Airbus A320	6E	IndiGo	6E2518	2022-05-07 20:35:00	2022-05-08 11:20:00	
6	329433	YYZ	Canada	HND	Japan	Embraer 175|Boeing 777	multi	American	AA4827	2022-05-02 06:45:00	2022-05-03 16:35:00	

- Standardized units of measurement by removing the percentage sign from the 'co2_percentage' column.



In [14]:
```
# Convert text data to numbers

df['from_country'] = df['from_country'].cat.codes
df['dest_country'] = df['dest_country'].cat.codes
df['airline_name'] = df['airline_name'].cat.codes
```

In [15]:
```
# Define the X columns and Y columns

X = df[['from_country','dest_country','airline_name','duration','stops','co2_emissions','departure_month']]
Y = df[['price']]
```

In [16]:
```
# Standardization

sc = StandardScaler()
X = sc.fit_transform(X)
```

In [17]:
```
# Split the dataset into training data and testing data

Xtrain, Xtest, Ytrain, Ytest = train_test_split(X, Y, test_size = 0.2, random_state = 42)
```

In [18]:
```
model_names = []
mse_scores = []
mae_scores = []
r2_scores = []
```

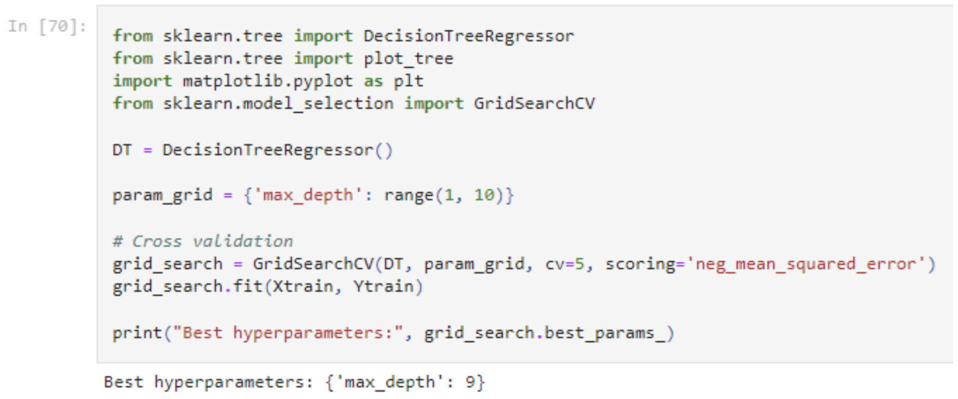
- Converted object variables to category type for memory efficiency and faster computation.
- Mapped categorical variables to numerical codes.

- Split the dataset into training and testing sets with a 80:20 ratio.

（b）Leveraging the website's data description for its domain expertise, we carefully selected relevant features to include in the training dataset. The chosen features for the training set encompass 'from_country', 'dest_country', 'airline_name', 'duration', 'stops', 'co2_emissions', and 'departure_month'. The target variable, or label, that our model aims to predict is the 'price'.

(c) Using GridSearchCV function from sklearn to perform cross-validation to find the best hyperparameters

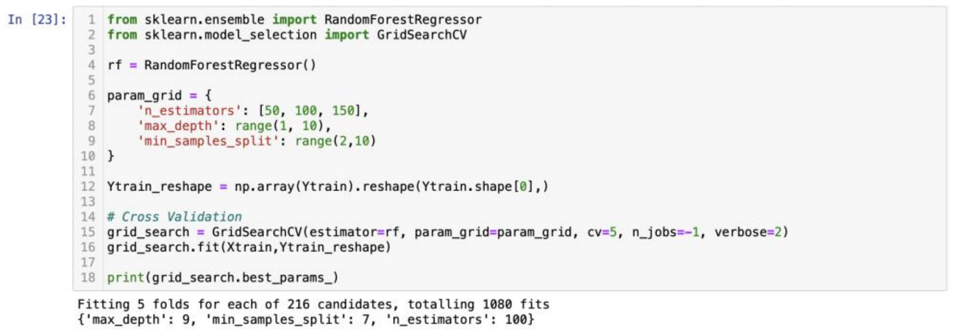
- Decision Tree Regressor



```
In [70]:
from sklearn.tree import DecisionTreeRegressor
from sklearn.tree import plot_tree
import matplotlib.pyplot as plt
from sklearn.model_selection import GridSearchCV

DT = DecisionTreeRegressor()

param_grid = {'max_depth': range(1, 10)}

# Cross validation
grid_search = GridSearchCV(DT, param_grid, cv=5, scoring='neg_mean_squared_error')
grid_search.fit(Xtrain, Ytrain)

print("Best hyperparameters:", grid_search.best_params_)
```

```
Best hyperparameters: {'max_depth': 9}
```

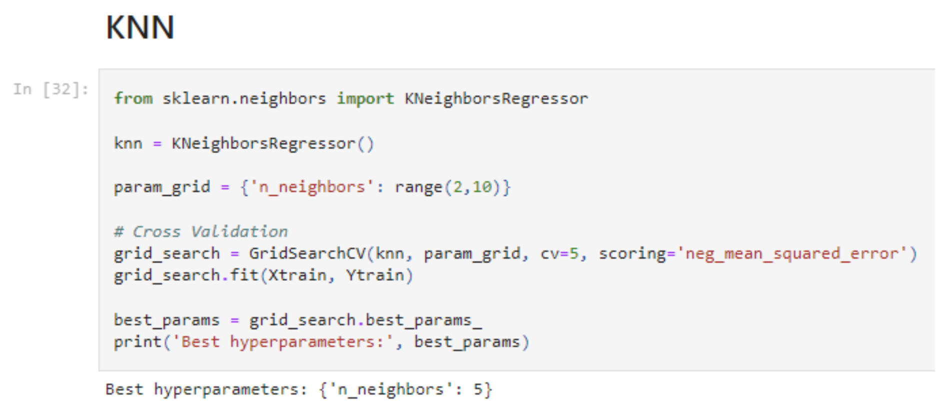
- Random Forest Regressor



```
In [23]:
from sklearn.ensemble import RandomForestRegressor
from sklearn.model_selection import GridSearchCV

rf = RandomForestRegressor()

param_grid = {
    'n_estimators': [50, 100, 150],
    'max_depth': range(1, 10),
    'min_samples_split': range(2,10)
}

Ytrain_reshape = np.array(Ytrain).reshape(Ytrain.shape[0],)

# Cross Validation
grid_search = GridSearchCV(estimator=rf, param_grid=param_grid, cv=5, n_jobs=-1, verbose=2)
grid_search.fit(Xtrain,Ytrain_reshape)

print(grid_search.best_params_)
```

```
Fitting 5 folds for each of 216 candidates, totalling 1080 fits
{'max_depth': 9, 'min_samples_split': 7, 'n_estimators': 100}
```

- K-Nearest Neighbors (KNN) Regressor



KNN

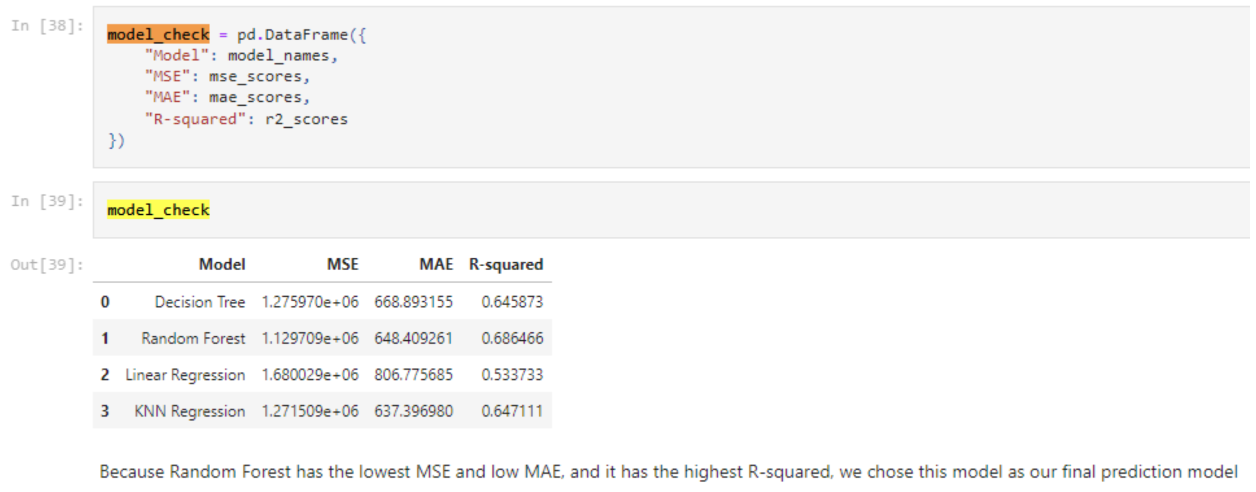
```
from sklearn.neighbors import KNeighborsRegressor

knn = KNeighborsRegressor()

param_grid = {'n_neighbors': range(2,10)}

# Cross Validation
grid_search = GridSearchCV(knn, param_grid, cv=5, scoring='neg_mean_squared_error')
grid_search.fit(Xtrain, Ytrain)

best_params = grid_search.best_params_
print('Best hyperparameters:', best_params)
```

```
Best hyperparameters: {'n_neighbors': 5}
```

C. Model Performance and Evaluation Metrics

We used three metrics to evaluate the performance of our models: Mean Squared Error (MSE), Mean Absolute Error (MAE), and R-squared. MSE measures the average squared difference between the predicted and actual values, while MAE measures the average absolute difference. R-squared provides a relative measure of how well the model explains the variance in the data, with a value closer to 1 indicating better performance. We used these metrics because they provide a comprehensive view of model performance, taking into account both the magnitude of errors and the model's ability to explain variability in the data.

Here is a summary of the performance metrics for each model:



```
model_check = pd.DataFrame({
    "Model": model_names,
    "MSE": mse_scores,
    "MAE": mae_scores,
    "R-squared": r2_scores
})
```

```
model_check
```

	Model	MSE	MAE	R-squared
0	Decision Tree	1.275970e+06	668.893155	0.645873
1	Random Forest	1.129709e+06	648.409261	0.686466
2	Linear Regression	1.680029e+06	806.775685	0.533733
3	KNN Regression	1.271509e+06	637.396980	0.647111

Because Random Forest has the lowest MSE and low MAE, and it has the highest R-squared, we chose this model as our final prediction model

- Decision Tree Regressor:
    - R-squared: 0.65
    - Mean squared error: 1,275,969.58
    - Mean absolute error: 668.89
    - Overfitting (R^2 on training set): 0.72/0.65= 1.10769
- Random Forest Regressor:

    - R-squared: 0.69
    - Mean squared error: 1,129,709.10
    - Mean absolute error: 648.41
    - Overfitting (R^2 on training set): 0.73/0.69= 1.05797
- Linear Regression:
    - R-squared: 0.53
    - Mean squared error: 1,680,029.12
    - Mean absolute error: 806.78
    - Overfitting (R^2 on training set): 0.51/0.53= 0.96226
- KNN Regression:
    - R-squared: 0.65
    - Mean squared error: 1,271,509.31
    - Mean absolute error: 637.40
    - Overfitting (R^2 on training set): 0.75/0.65= 1.15385

Based on these results, we chose the Random Forest Regressor as our final prediction model for flight price because it has

-the lowest MSE

-a low MAE

-the highest R-squared value.

-lowest overfitting

This indicates that the model is (i)better at predicting flight prices than the other models and (ii)has a stronger ability to explain the variance in the data. This model's performance can be attributed to its ability to capture non-linear relationships in the data and (iii)its robustness to outliers. Moreover, Random Forests benefit from the ensemble learning approach, which combines multiple decision trees to produce a more accurate prediction.

P.S. Although it seems the model is not the best choice considering robustness to outliers, in price prediction tasks we value closer prediction as our highest priority.

2. Data Analysis for Flight Delays

In addition to predicting flight prices, we aim to provide users with an estimate of potential flight delays. We perform data analysis on historical flight data, examining variables such as departure and arrival airports, airlines, and travel dates. By identifying patterns and trends in flight delays, we can inform users of the likelihood of experiencing a delay and help them make better-informed decisions when booking their flights

A. Data Collection and Preparation

Our dataset, "Combined_Flights.csv," contains information on various flights within the United States, including their departure and arrival airports, airlines, delay durations, and diversion or cancellation status. The dataset also includes the destination state name, which was crucial for our analysis.

First, we loaded the dataset into a pandas DataFrame and filtered the data for delayed flights. We considered flights with departure delays, diversions, or cancellations as delayed flights. We then calculated the delay ratio for each destination state, which was the number of delayed flights divided by the total number of flights for that state.

To visualize the delay ratios, we utilized Plotly Express to create an interactive choropleth map of the United States, which color-coded the states according to their delay ratios.

B. User Interface

We developed a user-friendly interface using Streamlit, which allowed users to input their destination state, destination airport, and airline. These inputs were used to filter the dataset and calculate the probability of delay for the user's specific travel plan.

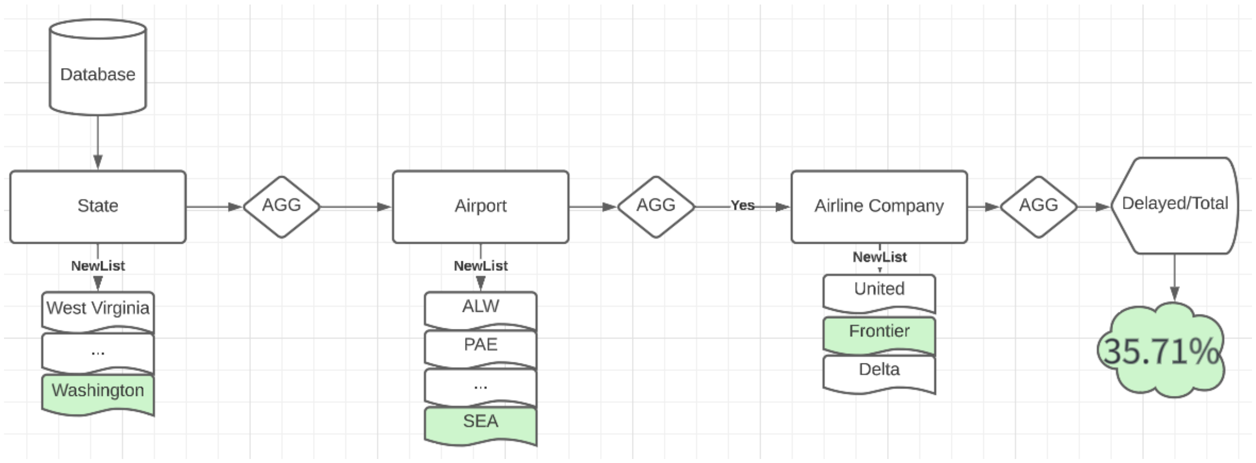
To ensure a smooth user experience, we provided dropdown menus for selecting destination states, airports, and airlines. The available options in the dropdowns were automatically populated based on the selected state and airport.

C. Data Analysis and Results

Upon receiving user inputs, the Flight Delay Probability Calculator conducted the following data analysis steps:

(a)Filtered the dataset based on the selected state, airport, and airline.

(b)Calculated the delay probability by dividing the number of delayed flights by the total number of flights for the selected travel plan.



The calculated delay probability was then displayed on the user interface, along with the interactive choropleth map of the United States. Users could explore the map to gain insights into flight delay patterns across different states.

D. Key Findings

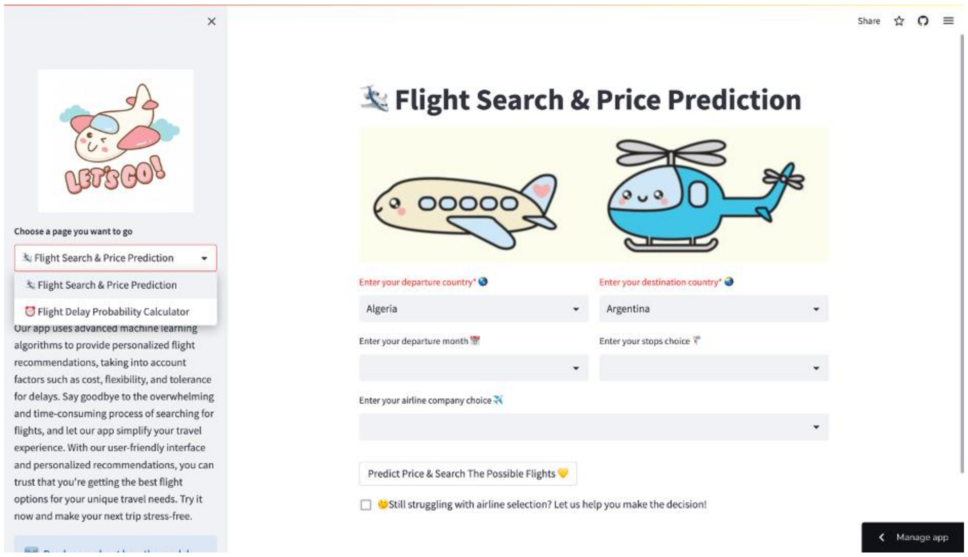
The analysis revealed that flight delay probabilities vary significantly across different states, airports, and airlines. Some states have higher overall delay ratios, which might be attributed to factors such as weather conditions, air traffic congestion, or airport infrastructure. Additionally, certain airlines may have higher delay probabilities due to operational issues or management practices.

By using the Flight Delay Probability Calculator, passengers can make more informed decisions about their travel plans and potentially minimize the risk of flight delays. They can choose to fly with airlines that have lower delay probabilities or select alternative airports in their destination state.

V. Service

Our application consists of two main parts. The first part is dedicated to flight search and price prediction. It provides flight recommendations and price prediction to the users. The second part of the application is a flight delay probability calculator that calculates the probability of flight delays for the user. This part also provides information on the delay probability for each state.

Here’s the link of our application: Flight Analysis Web API


Share
Flight Search & Price Prediction
Choose a page you want to go
Flight Search & Price Prediction
Flight Search & Price Prediction
Flight Delay Probability Calculator
Enter your departure country*
Algeria
Enter your destination country*
Argentina
Enter your departure month
Enter your stops choice
Enter your airline company choice
Predict Price & Search The Possible Flights
Manage app

Our app uses advanced machine learning algorithms to provide personalized flight recommendations, taking into account factors such as cost, flexibility, and tolerance for delays. Say goodbye to the overwhelming and time-consuming process of searching for flights, and let our app simplify your travel experience. With our user-friendly interface and personalized recommendations, you can trust that you're getting the best flight options for your unique travel needs. Try it now and make your next trip stress-free.

☐ Still struggling with airline selection? Let us help you make the decision!

1. Flight Search & Price Prediction

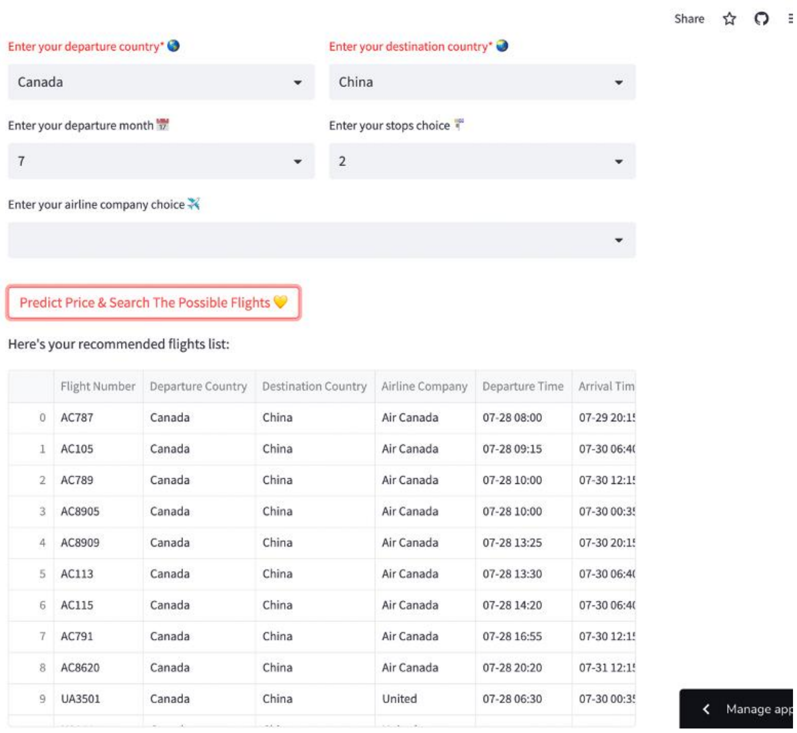
A. Flight Recommendation and Price Prediction

Our user-friendly application enables users to input their travel preferences, such as the departure and destination countries, as well as optional details such as preferred departure month, the number of stops, and the airline company of their choice. Our application will offer users the list of recommended flights, and our machine learning model which is based on random forest algorithm will generate a predicted price. Users who have not yet decided on a departure month, stops, or airline company can simply leave these fields blank and still receive tailored flight options.

Upon inputting the user's information, our application filters out the flights in our dataset that do not meet the user's requirements and shows only the ones that meet their needs. We then provide users with detailed flight information, including the flight number, departure country, destination country, airline company, departure time, arrival time, total stops, duration, and CO2 emissions.

Our machine learning model, based on the random forest regressor algorithm, uses features such as from_country, dest_country, airline_name, duration, stops, co2_emissions, departure_month as features, to predict the cost of the user's trip. The model uses the departure country input as from_country, destination country input as dest_country, airline company choice input as airline_name, stop choices input as stops, and departure month input as departure_month. If the user does not provide inputs for departure month, stops, or airline company, the application will automatically use the most frequent values for these features in the dataset in our model. Additionally, the application will take the average values of duration and co2_emissions for the remaining flights in the dataset and use them as model input.

For instance, let’s say we want to travel from Canada to China. We have decided to depart in July and have chosen to have 2 stops during our journey. However, we haven’t decided on an airline company yet, so we leave that option blank.



Enter your departure country* 🌏
Canada

Enter your destination country* 🌏
China

Enter your departure month 📅
7

Enter your stops choice 🪧
2

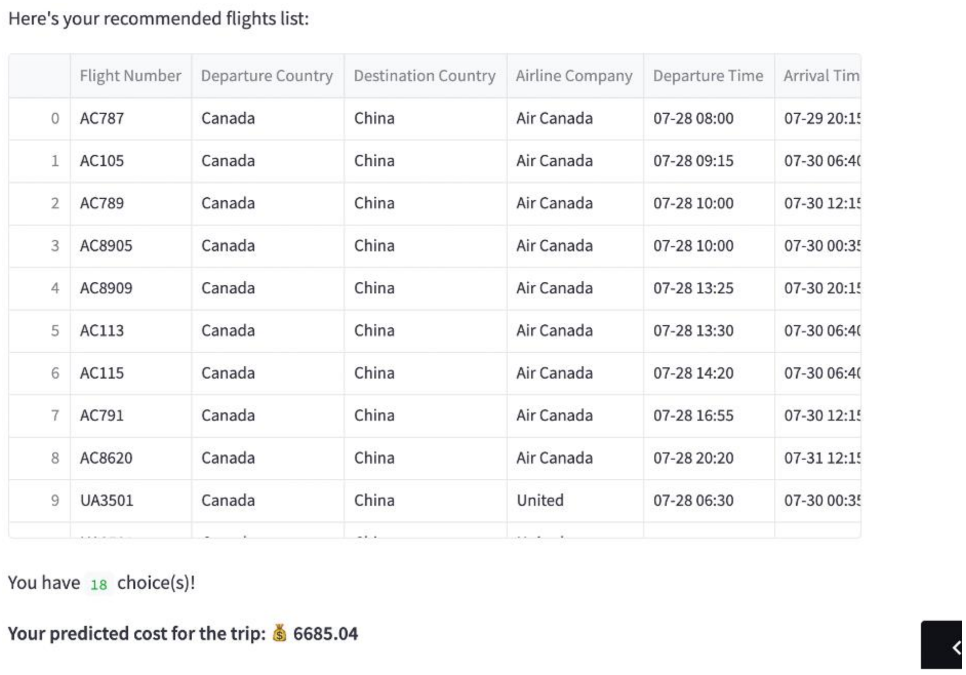
Enter your airline company choice ✈️

Predict Price & Search The Possible Flights 💛

Here's your recommended flights list:

	Flight Number	Departure Country	Destination Country	Airline Company	Departure Time	Arrival Tim
0	AC787	Canada	China	Air Canada	07-28 08:00	07-29 20:1
1	AC105	Canada	China	Air Canada	07-28 09:15	07-30 06:4
2	AC789	Canada	China	Air Canada	07-28 10:00	07-30 12:1
3	AC8905	Canada	China	Air Canada	07-28 10:00	07-30 00:3
4	AC8909	Canada	China	Air Canada	07-28 13:25	07-30 20:1
5	AC113	Canada	China	Air Canada	07-28 13:30	07-30 06:4
6	AC115	Canada	China	Air Canada	07-28 14:20	07-30 06:4
7	AC791	Canada	China	Air Canada	07-28 16:55	07-30 12:1
8	AC8620	Canada	China	Air Canada	07-28 20:20	07-31 12:1
9	UA3501	Canada	China	United	07-28 06:30	07-30 00:3

Our application displays a list of 18 flight options that meet our specified criteria, along with a predicted price for the trip which is 6685.04 USD.

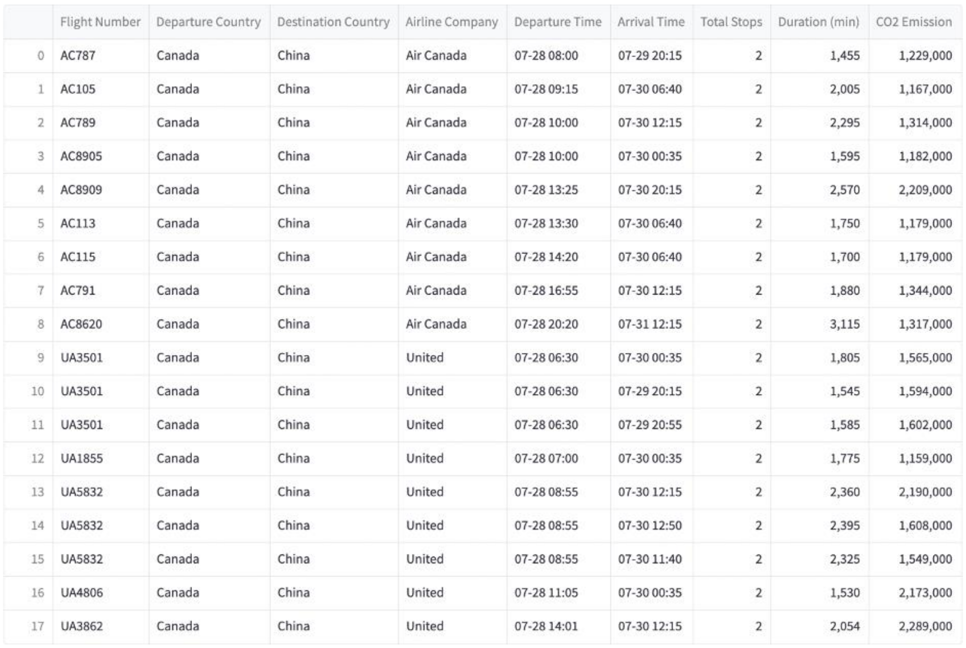


Here's your recommended flights list:

	Flight Number	Departure Country	Destination Country	Airline Company	Departure Time	Arrival Tim
0	AC787	Canada	China	Air Canada	07-28 08:00	07-29 20:1
1	AC105	Canada	China	Air Canada	07-28 09:15	07-30 06:4
2	AC789	Canada	China	Air Canada	07-28 10:00	07-30 12:1
3	AC8905	Canada	China	Air Canada	07-28 10:00	07-30 00:3
4	AC8909	Canada	China	Air Canada	07-28 13:25	07-30 20:1
5	AC113	Canada	China	Air Canada	07-28 13:30	07-30 06:4
6	AC115	Canada	China	Air Canada	07-28 14:20	07-30 06:4
7	AC791	Canada	China	Air Canada	07-28 16:55	07-30 12:1
8	AC8620	Canada	China	Air Canada	07-28 20:20	07-31 12:1
9	UA3501	Canada	China	United	07-28 06:30	07-30 00:3

You have 18 choice(s)!

Your predicted cost for the trip: 💰 6685.04



	Flight Number	Departure Country	Destination Country	Airline Company	Departure Time	Arrival Time	Total Stops	Duration (min)	CO2 Emission
0	AC787	Canada	China	Air Canada	07-28 08:00	07-29 20:15	2	1,455	1,229,000
1	AC105	Canada	China	Air Canada	07-28 09:15	07-30 06:40	2	2,005	1,167,000
2	AC789	Canada	China	Air Canada	07-28 10:00	07-30 12:15	2	2,295	1,314,000
3	AC8905	Canada	China	Air Canada	07-28 10:00	07-30 00:35	2	1,595	1,182,000
4	AC8909	Canada	China	Air Canada	07-28 13:25	07-30 20:15	2	2,570	2,209,000
5	AC113	Canada	China	Air Canada	07-28 13:30	07-30 06:40	2	1,750	1,179,000
6	AC115	Canada	China	Air Canada	07-28 14:20	07-30 06:40	2	1,700	1,179,000
7	AC791	Canada	China	Air Canada	07-28 16:55	07-30 12:15	2	1,880	1,344,000
8	AC8620	Canada	China	Air Canada	07-28 20:20	07-31 12:15	2	3,115	1,317,000
9	UA3501	Canada	China	United	07-28 06:30	07-30 00:35	2	1,805	1,565,000
10	UA3501	Canada	China	United	07-28 06:30	07-29 20:15	2	1,545	1,594,000
11	UA3501	Canada	China	United	07-28 06:30	07-29 20:55	2	1,585	1,602,000
12	UA1855	Canada	China	United	07-28 07:00	07-30 00:35	2	1,775	1,159,000
13	UA5832	Canada	China	United	07-28 08:55	07-30 12:15	2	2,360	2,190,000
14	UA5832	Canada	China	United	07-28 08:55	07-30 12:50	2	2,395	1,608,000
15	UA5832	Canada	China	United	07-28 08:55	07-30 11:40	2	2,325	1,549,000
16	UA4806	Canada	China	United	07-28 11:05	07-30 00:35	2	1,530	2,173,000
17	UA3862	Canada	China	United	07-28 14:01	07-30 12:15	2	2,054	2,289,000

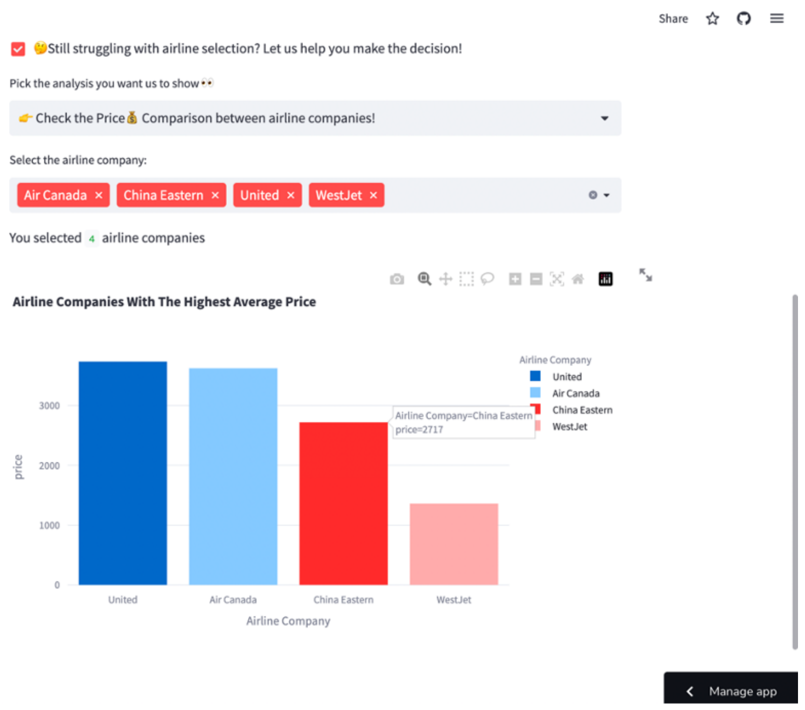
If the users are still struggling with airline selections, our application provides 3 dimensions of analysis to assist them make the decision.

B. Price Comparison

The first dimension is price comparison between airline companies.

By inputting the airline companies, our application generates a bar plot that displays the average prices of the selected airline companies. The plot is sorted from the highest average price on the left to the lowest on the right. When the user hovers over a bar, the corresponding average price of that airline company is displayed. This feature enables the user to easily compare and evaluate the pricing of different airlines and make an informed decision.

For instance, suppose we are planning to travel from Canada to China and want to compare the average prices of Air Canada, China Eastern, United, and WestJet for this route. As we have not yet decided on the departure month, stops, and airline company, we leave these options blank in the input form.



[✓] 🤔Still struggling with airline selection? Let us help you make the decision!

Pick the analysis you want us to show👀

👉Check the Price💰 Comparison between airline companies!

Select the airline company:

Air Canada × China Eastern × United × WestJet ×

You selected 4 airline companies

Our application then generates a bar plot to show the price comparison of the airline companies. From the bar plot, we can see that United average ticket price from Canada to China is the highest, and WestJet average price is the lowest. By hovering over the bars, we can view the average price of each airline company. For example, here, the average price of China Eastern flights from Canada to China is 2717 USD.

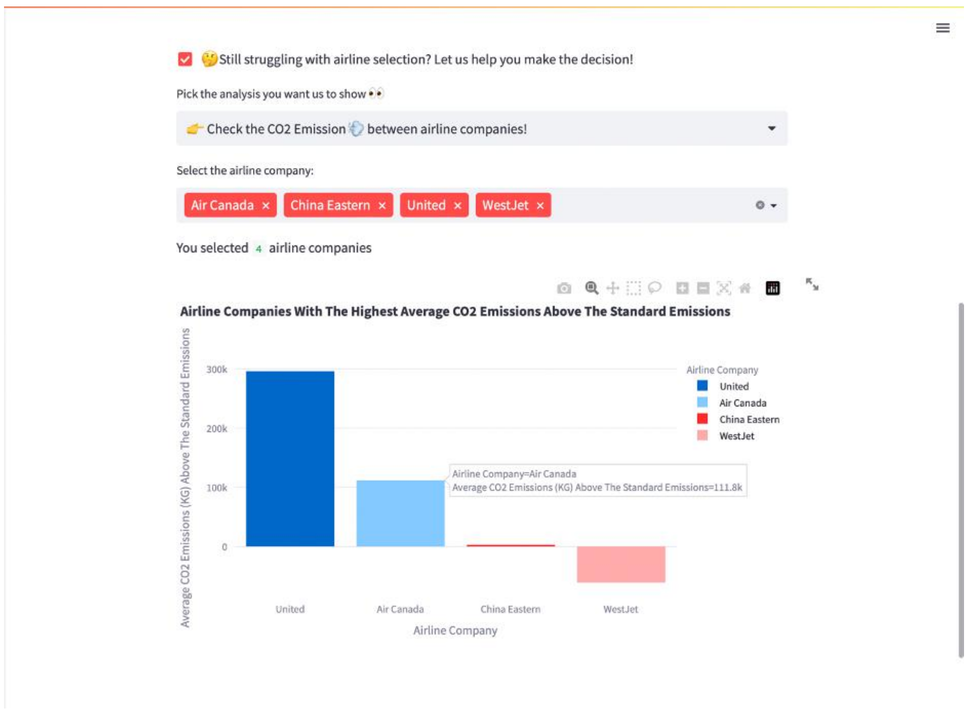
C. CO2 Emission Comparison

The second dimension is CO2 emission comparison between airline companies.

By inputting the airline companies, our application generates a bar plot that displays the average CO2 Emissions that are above the standard emissions of the selected airline companies. The plot is sorted from the highest average CO2 emissions on the left to the lowest on the right. When the user hovers over a bar, the corresponding average CO2 emissions that above the standard emissions of that airline company is displayed. This feature enables the user to easily compare and evaluate the emissions of different airlines and make an environment-friendly decision.

Please note that our calculations measure the difference between the flight CO2 emissions and the standard emissions. Therefore, it is possible for the resulting value to be negative, indicating that the average CO2 emissions of the airline company is below the standard emissions.

For instance, suppose we are planning to travel from Canada to China and want to compare the average prices of Air Canada, China Eastern, United, and WestJet for this route. As we have not yet decided on the departure month, stops, and airline company, we leave these options blank in the input form.



🤔Still struggling with airline selection? Let us help you make the decision!

Pick the analysis you want us to show👀

👉 Check the CO2 Emission between airline companies!

Select the airline company:

Air Canada × China Eastern × United × WestJet ×

You selected 4 airline companies

Airline Companies With The Highest Average CO2 Emissions Above The Standard Emissions
Average CO2 Emissions (KG) Above The Standard Emissions
300k
200k
100k
0
United
Air Canada
China Eastern
WestJet
Airline Company
Airline Company
United
Air Canada
China Eastern
WestJet
Airline Company=Air Canada
Average CO2 Emissions (KG) Above The Standard Emissions=111.8k

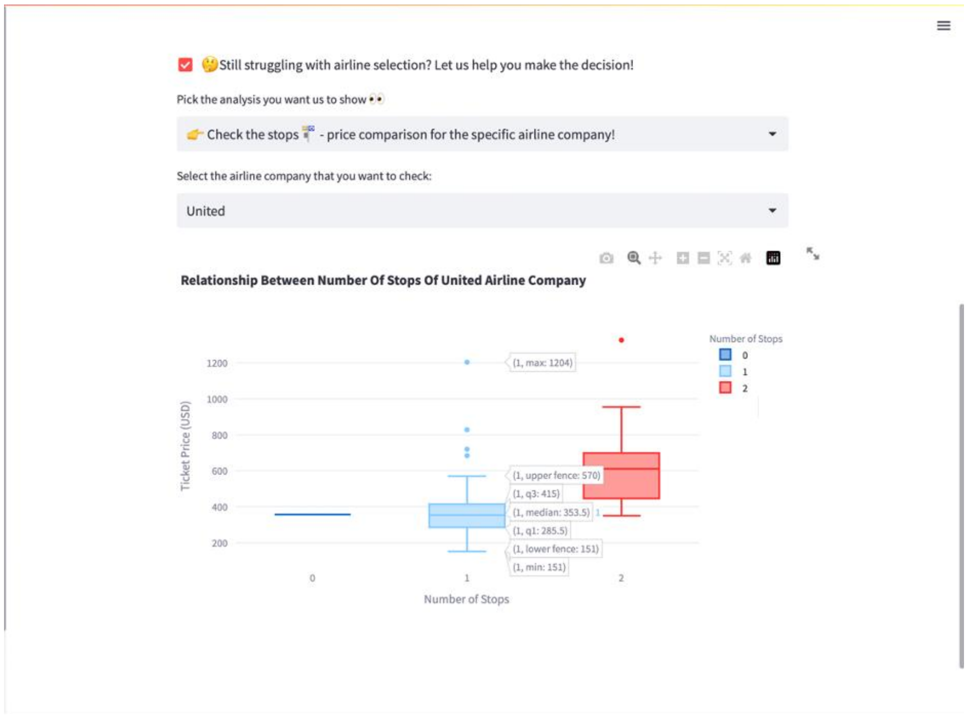
Our application then generates a bar plot to show the CO2 emission comparison of the airline companies. From the bar plot, we can see that United average CO2 emission from Canada to China is the highest, and WestJet average price is the lowest. And WestJet average CO2 emissions is negative, which means the average CO2 emissions of WestJet flights from Canada to China is below the standard emissions. By hovering over the bars, we can view the average CO2 emissions of each airline company that are above the standard emissions. For example, here, the average CO2 emission of Air Canada flights from Canada to China that above the standard emissions is 111800 KG.

D. Stops Price Comparison within the specific airline company

The final dimension is stops - price comparison within the specific airline company.

By inputting one specific airline company, our application generates a box plot that displays the relationship between the number of stops and the ticket price of the selected airline company. The plot is sorted from the less stops (usually begin with 0) on the left to the more stops on the right. When the user hovers over a bar, the corresponding IQR of the stop is displayed. This feature enables the user that have decided on their airline company but haven’t decided their total stops of the trip to easily compare and evaluate the stops and make an informed decision.

For instance, suppose we are planning to travel from Canada to United States and want to compare the number of stops for United airline company. As we have not yet decided on the departure month, we leave this option blank in the input form.



☑ 😲Still struggling with airline selection? Let us help you make the decision!

Pick the analysis you want us to show 👀

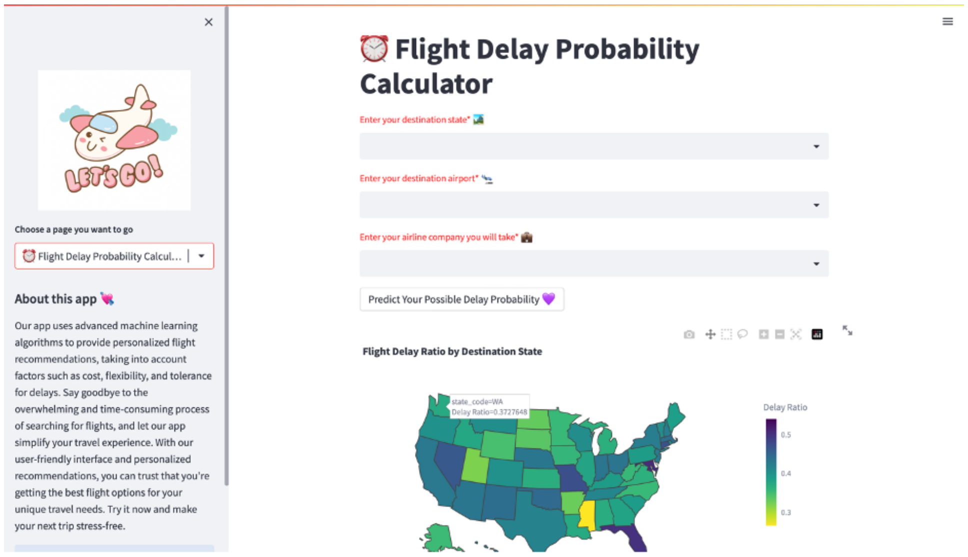
👉 Check the stops 🚏 - price comparison for the specific airline company!

Select the airline company that you want to check:

United

Our application then generates a box plot to show the relationship between the number of stops and the ticket price of United Airlines. From the box plot, we can see that 2 stops tend to have higher price, and the median of 2 stops is around 600 USD. 0 stops and 1 stop tends to have similar cheaper price and their median are around 300 - 400 USD. By hovering over the boxes, we can IQR of the stop. For example, here, the median of United Airlines flights from Canada to China for 1 stop is 353.5 USD. The highest price is 1204 USD, and the lowest price is 151 USD.

2. Flight Delay Probability Calculator


LET'S GO!
Choose a page you want to go
⏰ Flight Delay Probability Calcul...
Flight Delay Ratio by Destination State
state_code=WA
Delay Ratio=0.3727648
Delay Ratio
0.5
0.4
0.3

⏰ Flight Delay Probability Calculator

Enter your destination state*

Enter your destination airport*

Enter your airline company you will take*

Predict Your Possible Delay Probability 💜

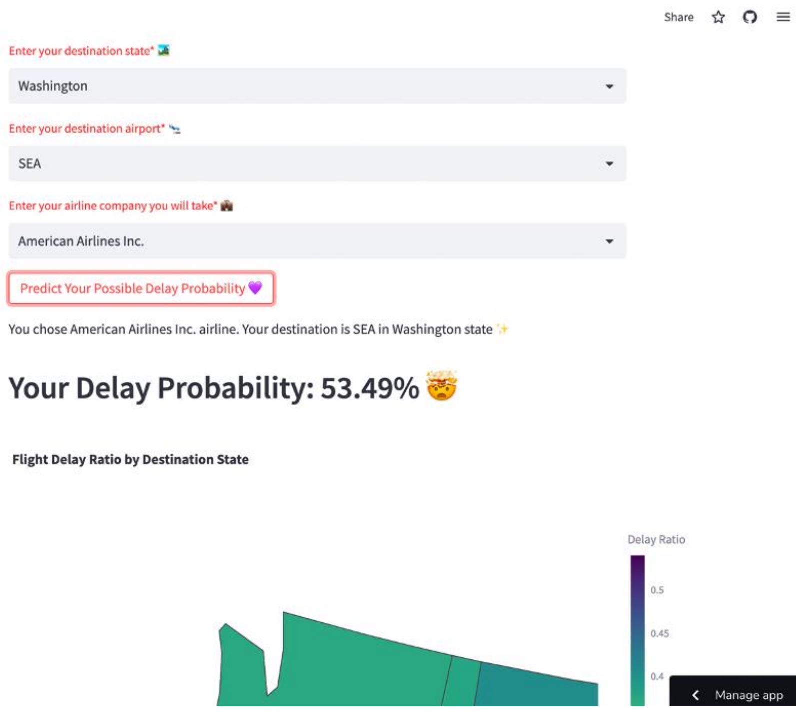
About this app 💘

Our app uses advanced machine learning algorithms to provide personalized flight recommendations, taking into account factors such as cost, flexibility, and tolerance for delays. Say goodbye to the overwhelming and time-consuming process of searching for flights, and let our app simplify your travel experience. With our user-friendly interface and personalized recommendations, you can trust that you're getting the best flight options for your unique travel needs. Try it now and make your next trip stress-free.

The flight delay probability calculator provides a map that displays the delay ratio for each destination state, allowing users to easily identify potential delays. Hovering over a state on the map provides the delay ratio for that particular state. For instance, the overall delay ratio for Washington state is roughly 37.28%.

Furthermore, our application enables users to enter their destination state, destination airport, and the airline that they plan to travel with, providing them with the likelihood of a delay based on previous records. Once users input this information, the application will filter the data to only include matches with the user's preference from the dataset. Subsequently, it calculates the delay percentage based on the remaining dataset and presents it to the user. Additionally, the map automatically zooms in to the user's state of destination.

For instance, suppose we are planning to travel to Washington state Seattle airport, and we'll take American Airlines flights.



Enter your destination state* 🏞️

Washington

Enter your destination airport* 🛬

SEA

Enter your airline company you will take* 💼

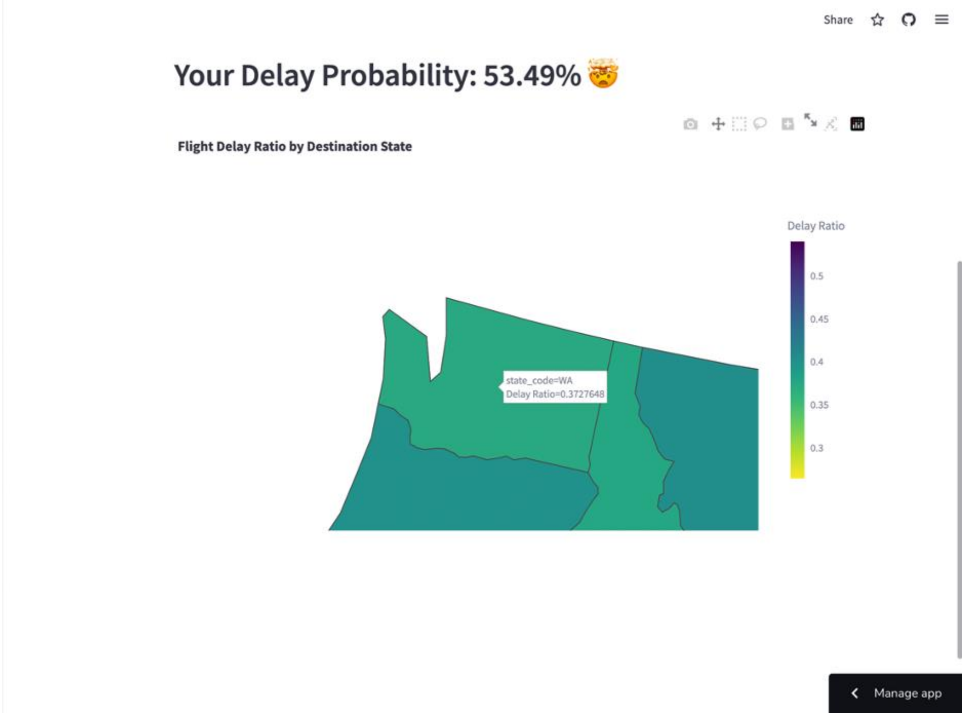
American Airlines Inc.

Predict Your Possible Delay Probability 💜

You chose American Airlines Inc. airline. Your destination is SEA in Washington state ✨

Your Delay Probability: 53.49% 🤯

Flight Delay Ratio by Destination State



Your Delay Probability: 53.49% 🤯

Flight Delay Ratio by Destination State

state_code=WA
Delay Ratio=0.3727648

The application then tells us that the delay probability for our trip will be 53.49%.

VI. Conclusion & Future Scope

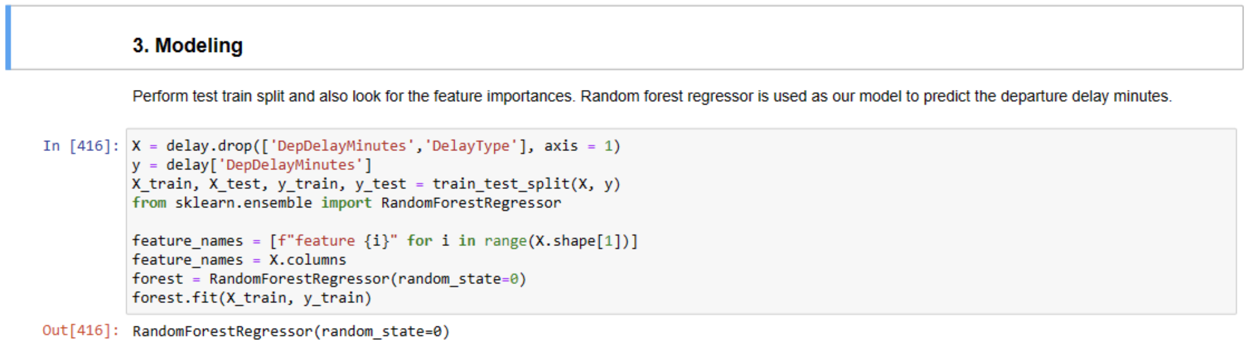
A. Limitations

In this report, we have presented our approach to predicting flight prices using machine learning models. However, there are some limitations we encountered.

(a) It is essential to acknowledge that some of the flight price predictions made, particularly those at the country level, may not be as actionable as desired. This is because (i) they tend to generalize across various factors that may significantly impact flight prices. For instance, flight prices can vary depending on specific airports, routes, airlines, and time of the year. Predictions made at the country level may fail to account for these nuances, leading to less accurate and less useful predictions. (ii)The delay analysis was able to go deep to not only which state the flight took place, but also which airport. Apparently, it would be harder for the two tasks to colaborate with each other when there is a difference in their required user inputs.

(b) Outliers, as we previously mentioned, for the price prediction dataset we did not drop potential outliers and the prices in the dataset did not take into account that ticket prices for different classes could diverge dramatically, for example a first-class ticket price can eaily surpass the price of an ecnomic ticket, and thus these ”outliers” cripples the initiitiveness of the prediction.

(c) Originally, we tried to implement machine learning to the Delay dataset as well, but failed to come up with something meaningful enough. Consisting of 61 different features, and over 1.3GB of records, the dataset is too large for our kernel to handle, we tried many different approaches to address this issue, including for-loop chunking, google drive mount, and random sub-samples. We finally came up with a random forest model for the delay analysis like this:



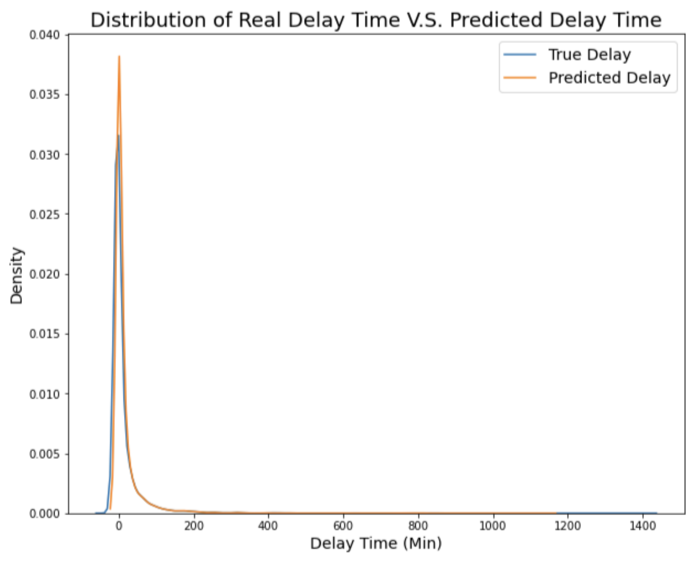
3. Modeling

Perform test train split and also look for the feature importances. Random forest regressor is used as our model to predict the departure delay minutes.

```
In [416]: X = delay.drop(['DepDelayMinutes','DelayType'], axis = 1)
          y = delay['DepDelayMinutes']
          X_train, X_test, y_train, y_test = train_test_split(X, y)
          from sklearn.ensemble import RandomForestRegressor

          feature_names = [f"feature {i}" for i in range(X.shape[1])]
          feature_names = X.columns
          forest = RandomForestRegressor(random_state=0)
          forest.fit(X_train, y_train)

Out[416]: RandomForestRegressor(random_state=0)
```

With a sky-high accuracy which looks like this:

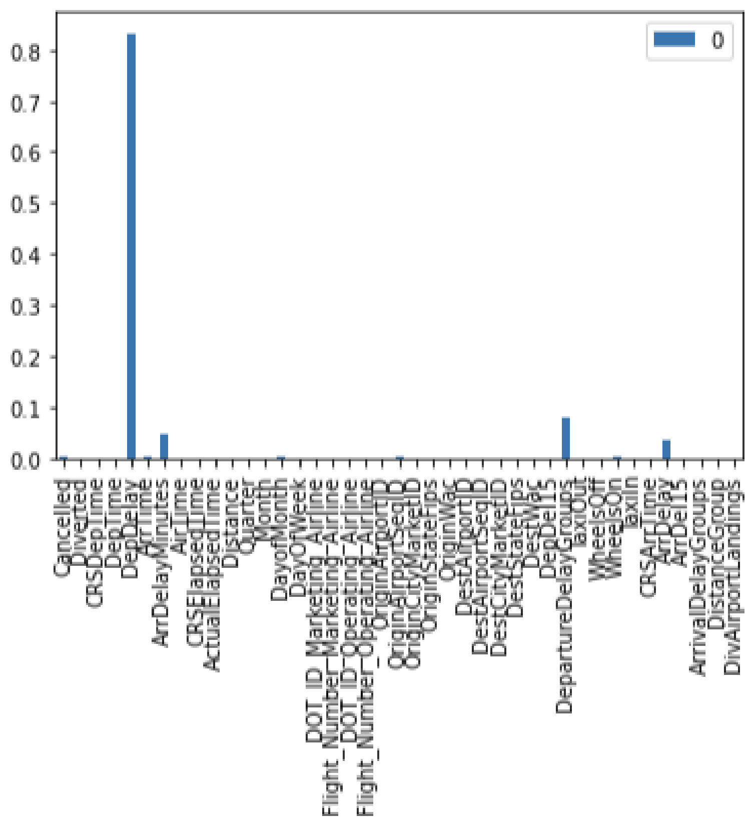
```
In [407]: plt.figure(figsize=(10,8))
          sns.kdeplot(X_test_2['DepDelay'], label='True Delay')
          sns.kdeplot(X_test_2['PredictedDelayMin'], label='Predicted Delay')
          plt.title("Distributions of Real Delay Time V.S. Predicted Delay Time", fontsize=18)
          plt.xlabel("Delay Time (Min)", fontsize=14)
          plt.ylabel("Density", fontsize=14)
          plt.legend(fontsize=14)
          plt.show();
```



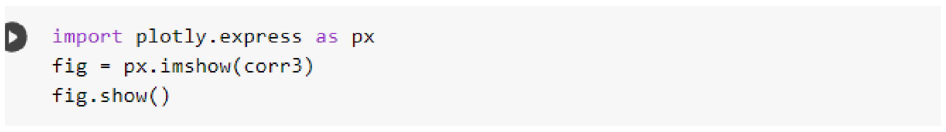
And that is when we sense something is a bit off. We performed feature importance analysis for the model and correlation analysis for the delay-dataset, and we it turns out that out of the 61 features included in the dataset, only a handful of them actually “matters”:

```
In [418]: 
          forest_importances = pd.DataFrame(importances, feature_names)
          print(forest_importances.sort_values(by=0, ascending=False))


          #forest_importances.plot.bar(yerr=std, ax=ax)
          forest_importances.plot.bar()
          #ax.set_title("Feature importances using MDI")
          #ax.set_ylabel("Mean decrease in impurity")
```

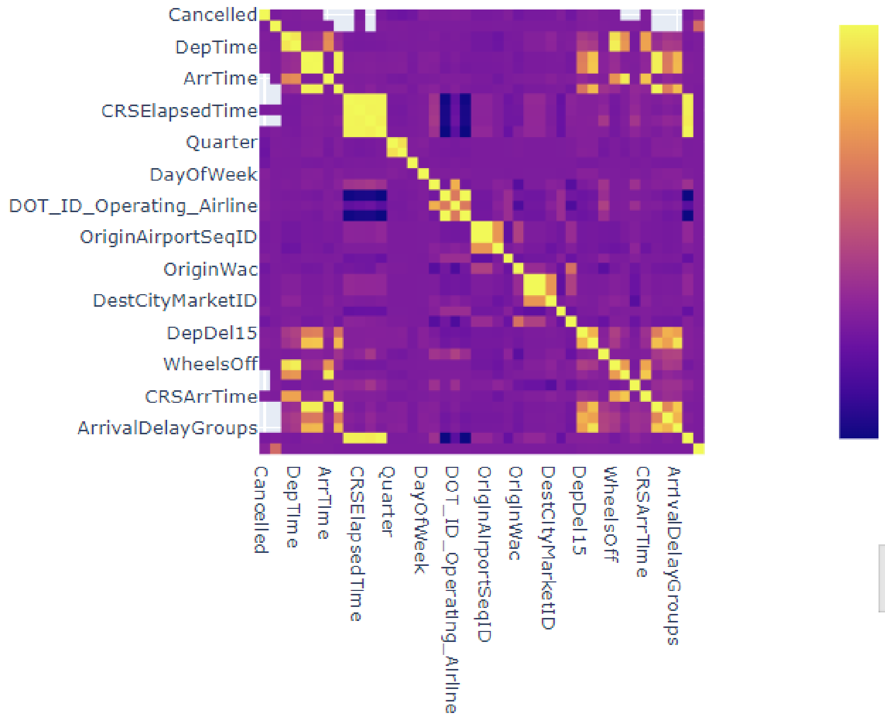
```
Out[418]: <AxesSubplot:>
```



0
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
Cancelled
Diverted
CRSDepTime
DepTime
DepDelay
ArrTime
ArrDelayMinutes
AirTime
CRSElapsedTime
ActualElapsedTime
Distance
Quarter
Month
DayofMonth
DayOfWeek
DOT_ID_Marketing_Airline
Flight_Number_Marketing_Airline
DOT_ID_Operating_Airline
Flight_Number_Operating_Airline
OriginAirportID
OriginAirportSeqID
OriginCityMarketID
OriginStateFips
OriginWac
DestAirportID
DestAirportSeqID
DestCityMarketID
DestStateFips
DestWac
DepDel15
DepartureDelayGroups
TaxiOut
WheelsOff
WheelsOn
TaxiIn
CRSArrTime
ArrDelay
ArrDel15
ArrivalDelayGroups
DistanceGroup
DivAirportLandings



```
import plotly.express as px
fig = px.imshow(corr3)
fig.show()
```



These features' relationship towards the label feature "DepDelayMinutes" is way too strong to a point that it is almost a linear sum, as these features represents meanings such as "departure delay" and "arriving delay". Considering the already-long waiting time for our streamlit app, we decided to make a more intuitive and rational approach which is calculate the delay probability through a data analysis method.

B. Future Works

To improve the model's performance and make its predictions more actionable, we recommend exploring additional features and refining the model. Potential future steps include incorporating more granular information about airports, routes, and airlines, as well as examining the impact of external factors such as weather, holidays, and major events. We could also consider using more advanced machine learning techniques or ensemble models to further enhance the model's predictive capabilities.

In addition, for the price prediction task we could really use a dataset that discriminates the differences among the flightclasses to make separate predictions for each flight class.

C. Summary

In conclusion, our analysis demonstrates that the Random Forest Regressor is the most effective model for predicting flight prices based on the available data. However, there is potential for improvement through feature engineering, hyperparameter tuning, model stacking, and exploring alternative models. By implementing these recommendations, we can enhance the accuracy of our predictions and provide more reliable estimates for users seeking to make informed decisions about their flight bookings.

Reference

1. Flight Status Prediction Dataset
2. Flight Data with 1 Million or More Records
3. Web API Reference
4. Github Link
5. Streamlit Gallery
6. Sklearn.neighbors.KNeighborsRegressor Scikit-learn1.0.2 documentation
7. Cute Flight Picture 1
8. Cute Flight Picture 2