ST2195 - Programming for Data Science

(Coursework Report – Python and R)

UOL Student ID Number - **210475649**

Page count – 10 (excluding cover page and table of contents)

Contents

Introduction

This report is based on a subset of the 2009 ASA Statistical Computing and Graphics Data Expo, which includes flight arrival and departure details for major carriers within the USA from October 1987 to April 2008. The datasets used in this project are from the years 2006 and 2007, and additional CSV files on airports and plane data were also utilized for specific questions.

The report is divided into five sections:

- Finding the optimal time of day, day of the week, and time of year for reducing flight delays.
- Investigating if there is a correlation between the age of planes and delays.
- Examining the trends in the number of passengers flying between different locations over time.
- Analyzing if there are cascading effects resulting from delays at one airport that impact delays in other airports.
- Building a model that can predict delays

To achieve these goals, the dataset was first cleaned to ensure that it was meaningful and ready for analysis. After cleaning, various visualizations and statistical analyses were performed to answer the five questions.

Data Cleaning Process

In order to ensure that the datasets are useful for analysis, we need to perform cleaning procedures to remove any irrelevant or incorrect information. This will allow us to properly manipulate and visualize the data. To achieve this, the datasets for both years (2006 and 2007) were merged and then checked for missing values and duplicate rows were removed. Missing values in the combined dataset and the other datasets (plane data and airport data) were removed where necessary when doing the questions. The column “CancellationCode” seemed to have meaningless values (either 0 or NA), therefore it was removed entirely. To ensure data integrity, only flights with departure and arrival times below 2400 (midnight) are included in the analysis. A new column called “Total_Delay” was created in the dataset by adding “ArrDelay” and “DepDelay” columns. The cleaned dataset is then exported to a CSV file named **cleaned_dataset.csv** to answer the questions.

Question 1: When is the best time of day, day of the week, and time of year to fly to minimize delays?

The columns Year, Month, DayOfWeek, DepTime, ArrTime, ArrDelay, DepDelay, and Total_Delay from the cleaned_dataset is used to create an array called delays. In this question, the delay was considered as total delay of the planes (sum of arrival delay and departure delays). Before answering this question, missing values in the Arrival, Departure, and Total Delays were eliminated because they could contribute to an incorrect final conclusion.

When is the best time of day to fly

Departure time of the flight was divided into 4 time slots, each with a 6-hour time difference, beginning from 00:00 to 06:00, 06:00 to 12:00, 12:00 to 18:00, and 18:00 to 24:00, in order to determine the best time of the day to fly. The names given to each bin are "Night" for the first slot (0:00-6:00), "Morning" for the second slot (6:00-12:00), "Afternoon" for the third slot (12:00-18:00), and "Evening" for the fourth slot. (18:00-24:00).

To find the time slot with the smallest delay time, the data were grouped by time slots, the average delay time for each slot was computed, and results were shown in a bar graph.

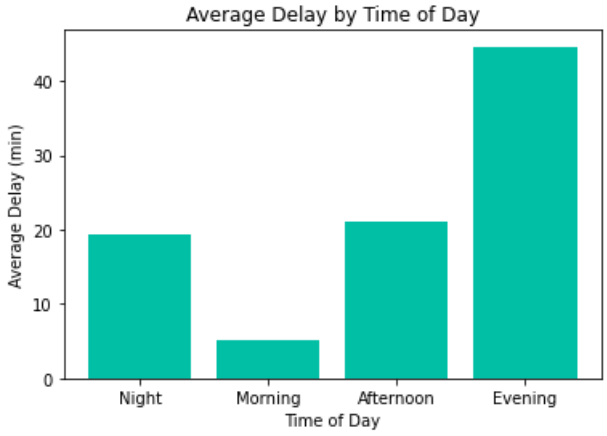

Figure 1 : Average Delay by Time of Day-python

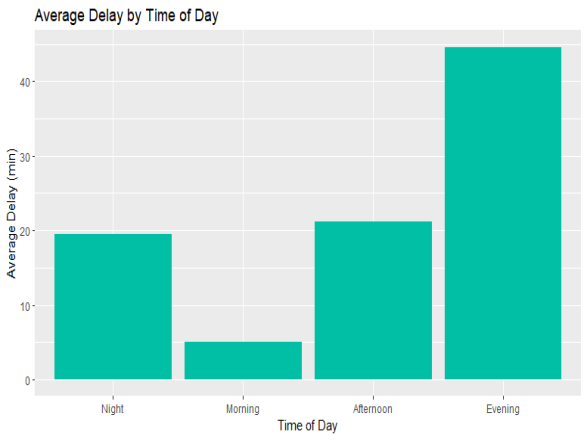

Figure 2 : Average Delay by Time of Day-R

It is visible from the bar graph above that flying in the morning is optimal as it has the smallest average delay time, whereas flying in the evening experiences the largest average delays.

When is the best time of the week to fly

A similar approach was used to identify the best day of the week that comprises the minimum delay. The average delay is found based on the day of the week ('DayOfWeek') using an aggregate function.

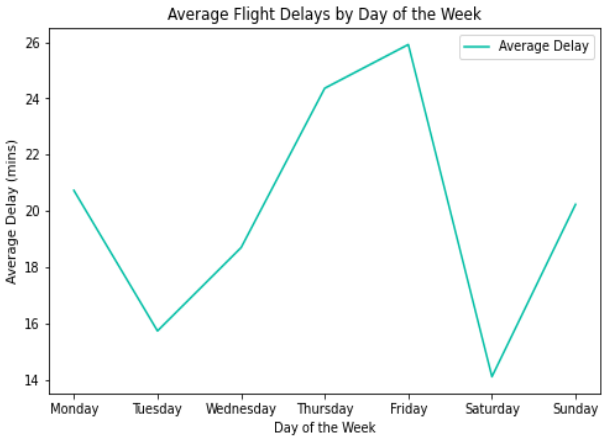

Figure 3 : Average Flight Delays by Day of the Week – python

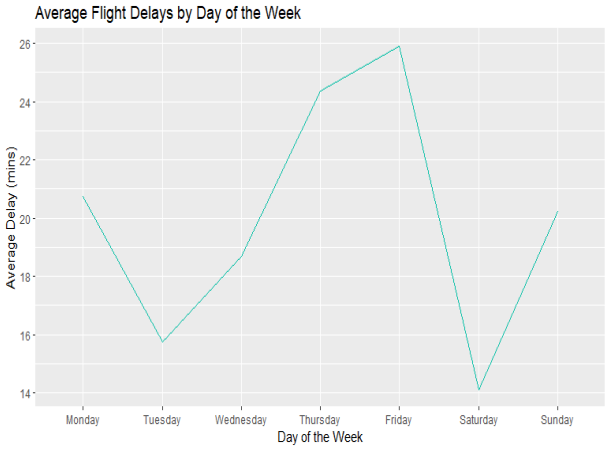

Figure 4 : Average Flight Delays by Day of the Week – R

The analysis shows that Saturday had the lowest average delay time, making it the best day to travel. Additionally, Tuesday was found to be a better day to fly than the other weekdays. On the other hand, Friday had the highest average delay time, making it the worst day to travel.

When is the best time of the year to fly

A line plot was created to show the relationship between the average delay and the month of the year in order to further analyze the delay pattern. This allowed for a better understanding of how the delay varied throughout the year.

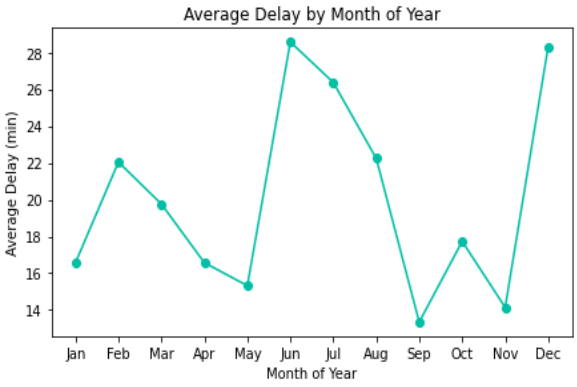

Figure 5 : Average Delay by Month of Year – python

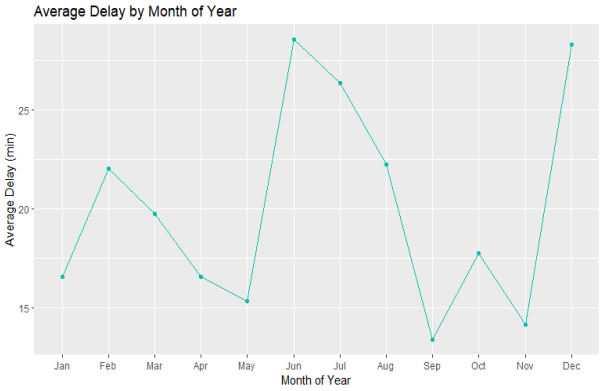

Figure 6 : Average Delay by Month of Year - R

According to the graph, September had the lowest average delay, with November and May following closely behind. On the other hand, June had the highest average delay, with December being the next highest.

The results of this analysis could help travelers in avoiding flight delays and having a more pleasant trip. By adding more flights during less busy periods, airlines could increase their on-time performance, but they must balance this with costs and passenger demand.

Question 2: Do older planes suffer more delays?

To do this question the tailnum and year columns were taken from the supplementary dataset 'plane-data' and cleaned the data by removing NaN values. To create a common column for combining the dataset with the entire set that uses the same column name, the tail number column is changed to "TailNum." After that, a refined data set is created by combining the two planes' data columns with the previously cleaned dataset. The year column was changed to "YearOfManufacture" to avoid any confusion.

Plane age was calculated by subtracting the "YearOfManufacture" from the "Year" column. The resulting data was grouped by plane age, and the average of the total delay was calculated for each age range (0-10, 10-20, 20-30, 30-40, 40-50, 50-60). A bar graph was plotted to visualize the relationship between average delay and plane age.

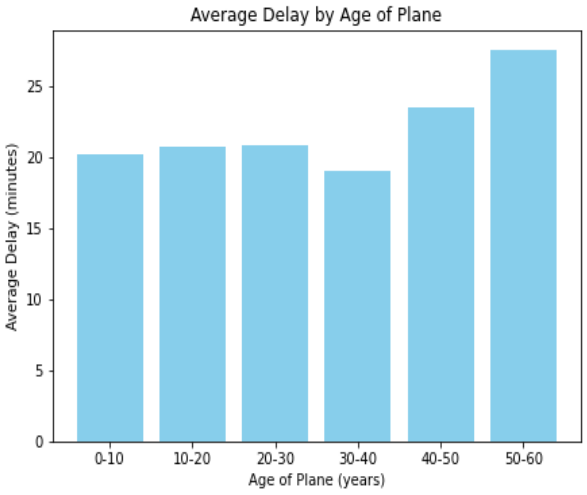

Figure 7 : Average Delay by Age of Plane – python

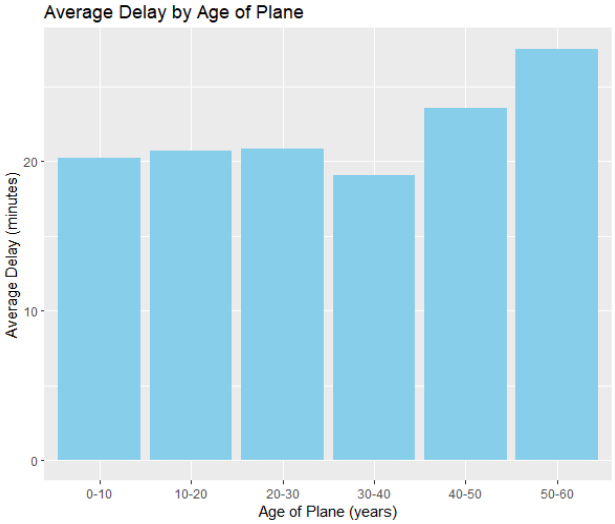

Figure 8 : Average Delay by Age of Plane – R

The above bar plot shows us that the age range of 50-60 had the highest average delay, while the age range of 30-40 had the lowest average delay but the age ranges of 0-10, 10-20, and 20-30 had higher average delays than the 30-40 age range therefore to get a better understanding a regression plot of average Total delay by plane age was then plotted and the correlation was calculated using "corr" method.

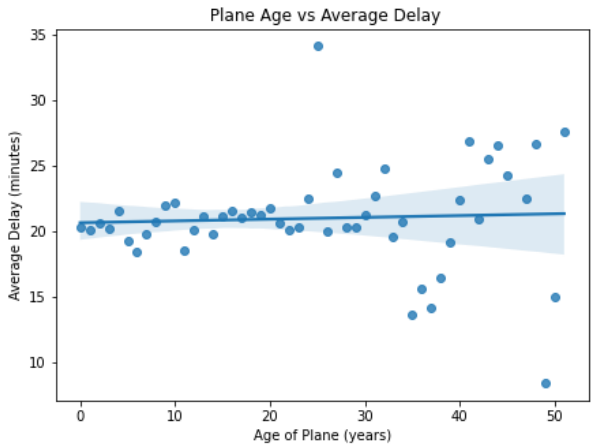

Figure 9 : Plane Age vs Average Delay – python

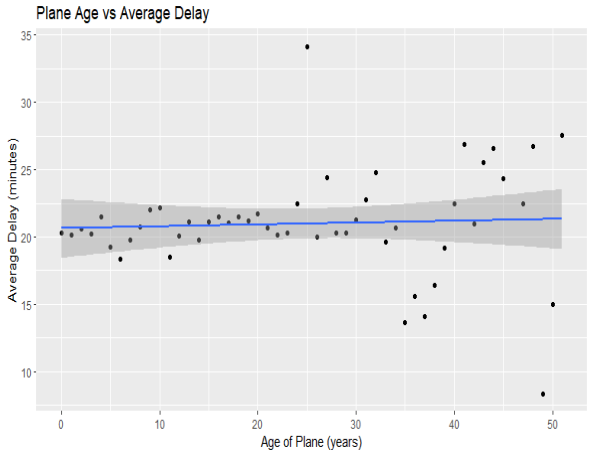

Figure 10 : Plane Age vs Average Delay – R

We can draw the conclusion that there is a weak positive correlation between a plane's age and the average delay it encounters based on the calculated correlation coefficient of 0.05248885. This suggests that there is a slight tendency for a plane to suffer longer delays as it ages, but this relationship is not very strong. As a result, factors other than plane age, such as weather conditions, air traffic volume, and maintenance problems, may have a greater impact on flight delays. To improve their on-time performance and give their customers a better experience, airlines must routinely assess the age of their planes and retire old planes.

Question 3: How does the number of people flying between different locations change over time?

In order to analyze the flying patterns of people between different destinations, the number of passengers traveling to each destination would ideally be required. Since this information is not available with the given dataset, the number of flights will be used as a proxy with the assumption that there is a direct relationship between the two.

The number of flights over the years 2006 and 2007 are illustrated as a time series graph below,

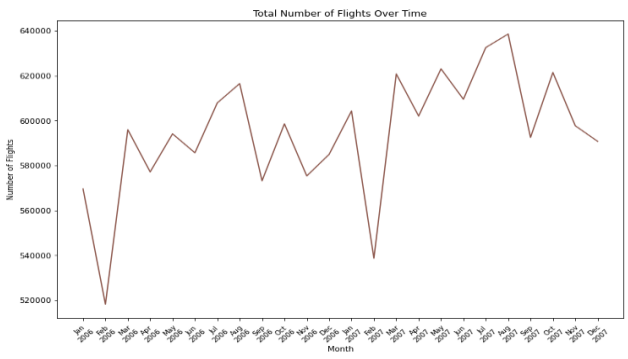

Figure 11 : Total Number of Flights Over Time – python

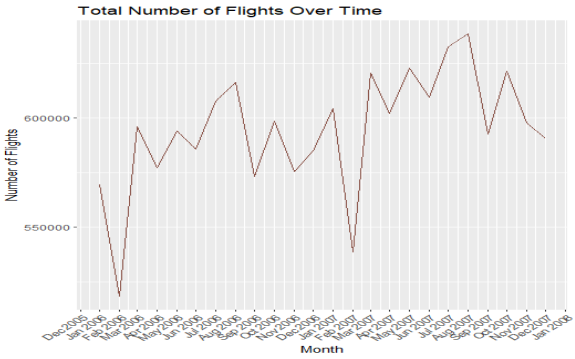

Figure 12 : Total Number of Flights Over Time – R

In the absence of passenger data, the number of flights serves as a useful indicator to identify patterns and trends in air travel. A quick overview of the above time series graphs shows that there has been a general upward trend in the number of flights over time with notable fluctuations in the number of flights in certain months. This growth can be characterized by progressively higher peaks and higher lows during each year for seasons.

A seasonal trend can be observed in both the years where the Number of flights tend to peak at months such as January, March, May, August, and October. This distinct pattern likely corresponds to popular travel periods such as holidays, school breaks and seasonal events.

To gain further insights into how the number of people flying between different locations changes over time, the origin locations of the flights have been factored in. Due to the extensive nature of the data, the analysis focuses on the top 5 airports based on the number of flights, showcasing the seasonal trend for each. The seasonal trend identified in the below graphs also aligns with the trend observed in the time series analysis, with higher numbers of flights occurring in January, March, May, August, and October. This correlation confirms the existence of a consistent seasonal pattern in the number of flights taken over time with "ATL" airport being the busiest airport throughout the months as it has the highest number of flights.

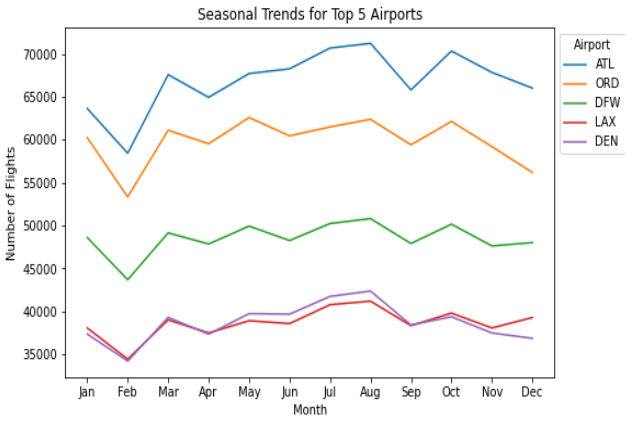

Figure 13 : Seasonal Trends for Top 5 Airports – python

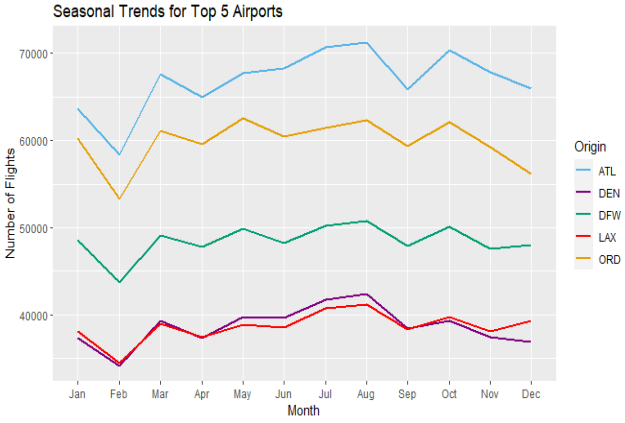

Figure 14 : Seasonal Trends for Top 5 Airports – R

Question 4: Can you detect cascading failures as delays in one airport create delays in others?

The dataset was first cleaned by filtering out the canceled flights and filling any missing values in the Total Delay column with zeros. Rows with missing values in either the TailNum or CRSDepTime columns were then dropped to ensure that the dataset is complete. The data was grouped by unique tail numbers and the scheduled departure times of the flights were then sorted in ascending order, creating a continuous timeline that would allow us to track the delay of each flight over time.

To determine the delay of each plane at the previous airport, the total delay column was shifted down by one row within each group of tail numbers. This was achieved by grouping the flights by tail number and using the shift method to shift the total delay column down by one row.

To ensure that the data is clean and ready for analysis, any rows with missing values in the PrevAirportDelay column were dropped. The delays in the current airport were taken to be equal to the total delay of each plane.

A scatter plot was then plotted to show the relationship between the delays in the current airport and the delays in the previous airport of the planes.

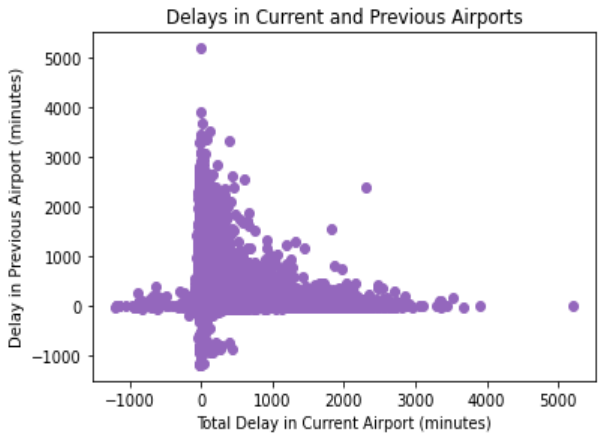

Figure 15 : Delays in Current and Previous Airports – python

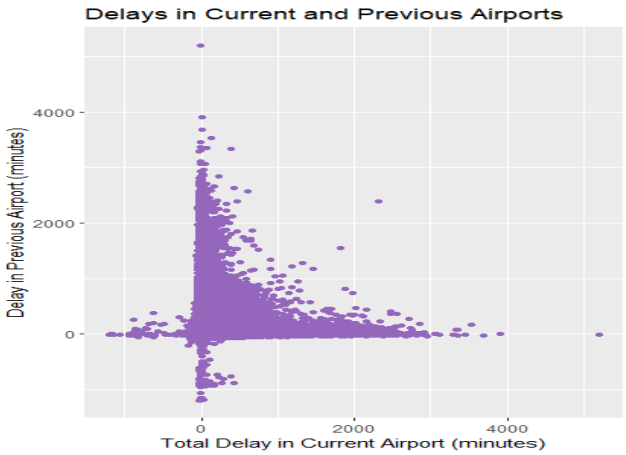

Figure 16 : Delays in Current and Previous Airports - R

It can be seen that the scatter plot does not clearly show the relationship between the delays in the current airport and the delays in the previous airport of the planes due to the large number of overlapping data points.

To explore more about whether the cascading delays in one airport create delays in others, a series of tests were conducted on the dataset. Firstly, a Pearson correlation test was performed to determine the relationship between the delays at the current airport and those at the previous airport for each flight. The resulting correlation coefficient of 0.06 indicates a weak positive correlation between the two variables, with a statistically significant p-value of 0. This suggests that there is evidence of a correlation between the delays at the two airports, although the strength of the relationship is relatively weak. To further test the hypothesis, a second hypothesis test was carried out.

Below, we present the hypothesis test conducted and its corresponding results.

Defining the hypothesis:

- H0: There is no correlation between the delays in the current and previous airports
- H1: There is a correlation between the delays in the current and previous airports

Results:

- Testing at a 5% significance level
- Test Statistic > Critical value

- The null hypothesis (H0) is rejected - There is a significant correlation between the delays in the current and previous airports.

By setting a significance level of 0.05 and calculating the degrees of freedom based on the sample size, a t-statistic was computed using the Pearson correlation coefficient. The critical value was then calculated based on the significance level and degrees of freedom. The absolute value of the t-statistic was compared to the critical value to determine whether to reject or fail to reject the null hypothesis. Based on the output, the null hypothesis was rejected, indicating a significant correlation between the delays in the current and previous airports. Hence, we can conclude that there is evidence from the hypothesis test that cascading delays in one airport create delays in other airports.

Question 5: Use the available variables to construct a model that predicts delays.

The dataset was preprocessed by dropping constant columns and unimportant columns and then dropping rows with missing values. The correlation matrix was calculated, and a mask was created for the highly correlated features. The threshold for the correlation coefficient was set to 0.8, and the highly correlated features were removed. The correlation between the other available variables is found using the below correlation plot.

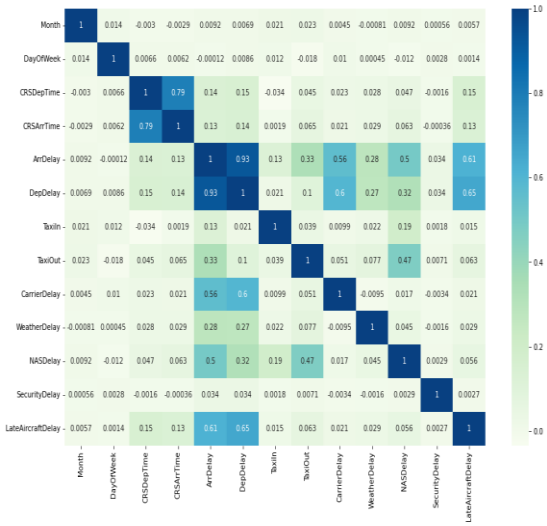

Figure 17 : correlation heatmap – python

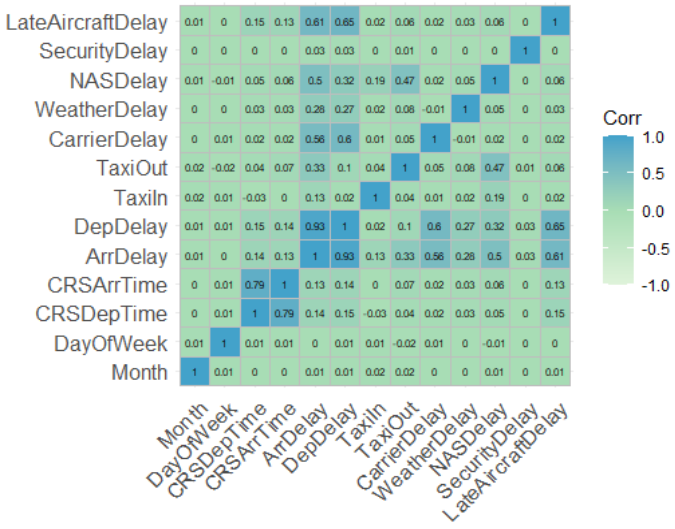

Figure 18 : correlation heatmap – R

The delay that is going to be predicted is the Arrival delay in the model. To build a model that can predict delays, we first created a target variable based on whether the arrival delay was greater than 0 or not. Then, we selected relevant features for the model based on their correlation with the target variable and their importance in predicting delays. These features included Month, DayOfWeek, CRSDepTime, CRSArrTime, DepDelay, CarrierDelay, WeatherDelay, NASDelay, LateAircraftDelay, and TaxiOut. To ensure that all features were on the same scale, we standardized the data using the StandardScaler.

Next, we trained a logistic regression model on the standardized dataset to predict delays. We split the dataset into training and testing sets using the train_test_split function, with a test size of 0.2 and a random state of 0. After training the logistic regression model, we evaluated its performance using various metrics, such as precision, recall, F1 score, confusion matrix, and ROC curve. These metrics provided us with insights into the model's ability to predict delayed and non-delayed flights, and helped us identify areas where the model can be improved.

```
Model: LogisticRegression

Confusion Matrix:
 [[1412060  114413]
 [ 361490  965823]]

Classification Report:
              precision    recall  f1-score   support

           0       0.80      0.93      0.86   1526473
           1       0.89      0.73      0.80   1327313

    accuracy                           0.83   2853786
   macro avg       0.85      0.83      0.83   2853786
weighted avg       0.84      0.83      0.83   2853786
```

Figure 19 : Classification Report

The classification report provided insights into the precision and recall of the model. The precision of 0.80 for 0 indicates that out of all the predictions made by the model for non-delayed flights, 80% were correct.

Similarly, the precision of 0.89 for 1 indicates that out of all the predictions made by the model for delayed flights, 89% were correct. The recall of 0.93 for 0 indicates that out of all the actual non-delayed flights, the model correctly predicted 93%. However, the recall of 0.73 for 1 indicates that the model was only able to correctly predict 73% of actual delayed flights.

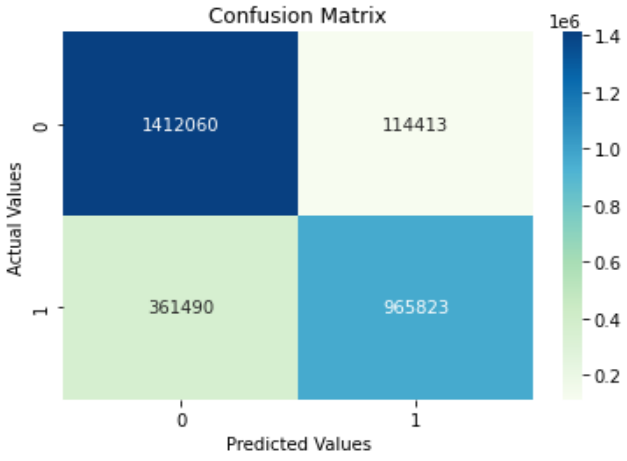

Figure 20 : Confusion Matrix – python

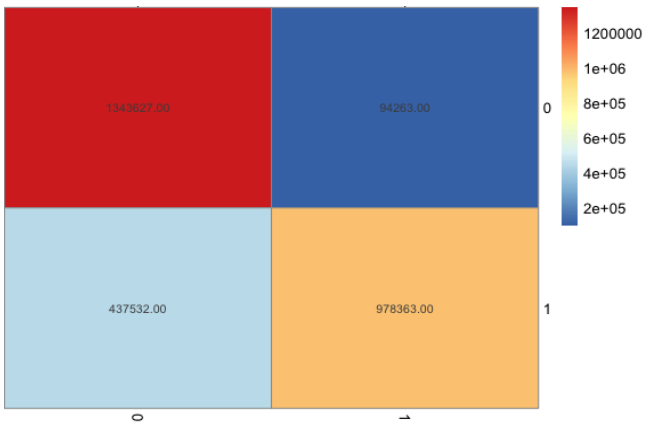

Figure 21 : Confusion Matrix - R

The confusion matrix provided us with a visual representation of the model's predictions. It showed us that our model correctly predicted non-delayed flights a majority of the time (true negatives), but there were some cases where it wrongly predicted a non-delayed flight as delayed (a false positive). Overall, the model's accuracy in predicting delayed flights was good, but there is still room for improvement in predicting non-delayed flights.

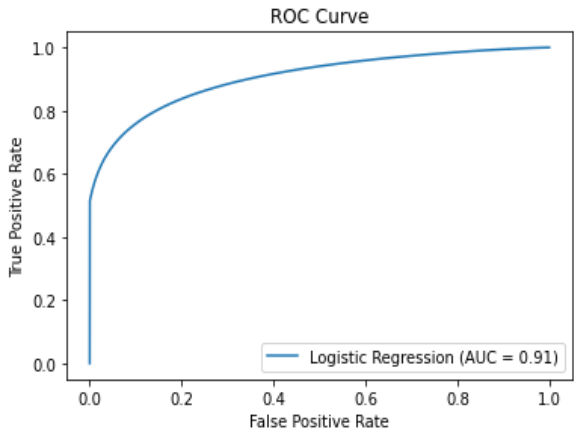

Figure 22 : ROC curve – python

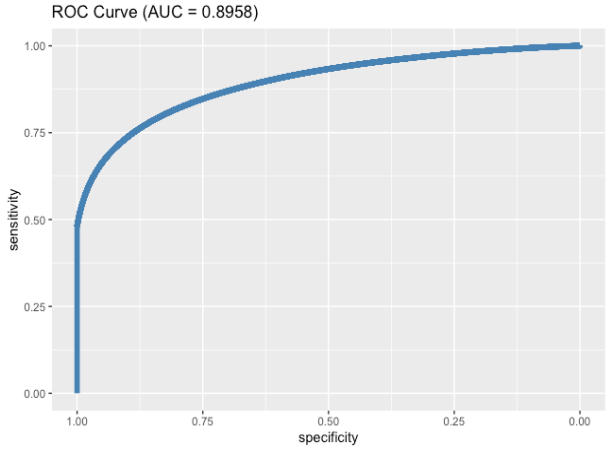

Figure 23 : ROC curve – R

The ROC curve showed an AUC of 0.91, indicating that the model has good predictive power. The accuracy score of 0.833238 means that the model was able to predict the correct delay status for 83% of the flights in the test dataset.

Overall, our model performed well in predicting delays. However, there is still room for improvement, especially in predicting delayed flights. We believe that additional feature engineering and fine-tuning of the model parameters could help improve the model's performance.