Executive Summary

The report will cover the best time to travel in order to minimise flight delays, as well as factors that affect flight delays such as the months, day, time and age of plane. We will also be comparing how the number of people flying between different locations change over time. We will also be researching if airports are a cause of cascading delay. Various Python and R packages be will used to make the data analysis regarding the Data expo 2009 flights. We will be taking a look at Years 2000 and 2001 and make analysis for those two consecutive years. Machine Learning and prediction models will also be created to tested out to check for flight delays and the amount of delay. Lastly, an ROC curve will be made to judge which model is the best and which one we should utilise

Note

Due to the missing values of various columns such as CarrierDelay, WeatherDelay, NASDelay, SecurityDelay and LateAircraftDelay from the 2000/2001 ontime tables, they will not be used in the analysis below. As most of R and python has similar graphs, only 1 of each will be included most of the time to reduce repetition and to save space due to 10 page constraint.

Setting Up Libraries and Importing to Database

Setting up in Python

Firstly, we will import the required libraries for the dataset/codes that we will be working with, such as sqlite3, pandas, seaborn, scikit etc. We would start creating a database by setting up a connection. Additionally, we have to create a cursor object order to fetch results. Next, we will utilise pandas to import the required csv files into python. Afterwards, they can be transferred into the database in table formats. Since we're adding multiple csv files to a table, append code is needed to add new values to the existing table, and set index=false to not create an index number.

To get queries, we will use c.execute command to create the query and put the required information in place, and end the query with .fetchall to retrieve the related data from the database.

Setting up in R

We will start by loading the required libraries for the codes that we will be working with such as RSqlite3, DBI and ggplot2 etc. A database can be created by running a connection with DBI and SQL. The csv files are first loaded into R followed by dbWriteTable to transfer them into the database. header=True shows the column names while the append=True is added to place more csv files into a table that already have content in it.

To get queries, we will use dbGetQuery to create and retrieve the data from the database. conn is included inside the sqlite query in order to establish a connection to the database.

Question 1 (Python and R)

Question 1 has been separated into 3 separate sections to show the average delay for different time frames of months, day of week and time of day.

We will get the Average of delays by averaging the sum of ArrDelay and DepDelay. We will use CASE expression for short-circuit evaluation, to separate the DepTime into time frames, from the ontime table. Next, we will use WHERE syntax to place conditions in the query such as Departure Delay and Arrival Delay more than 0, Cancelled = 0 and Diverted = 0 so that we only account for the flights that are delayed and not cancelled or not diverted. This ensures that our data is not skewed with inaccurate information. Furthermore, empty fields in the deptime column is also excluded. We group them according to time so that we get the output in the 5 different time groups, and then they are ordered from lowest delay to highest delay. The same is repeated for days and month.

Transforming from list to database (Additional Step for Python)

Secondly, we will transform the query from a list to dataframe so that the data can be inserted into the seaborn graph syntax. This will have to be done for every query in python as lists cannot be inserted into the seaborn graphs.

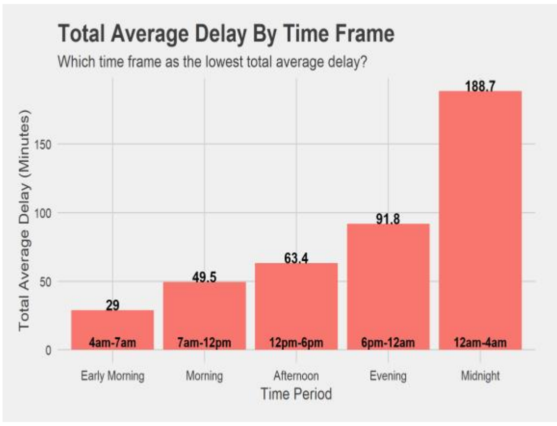
Graphing With Python

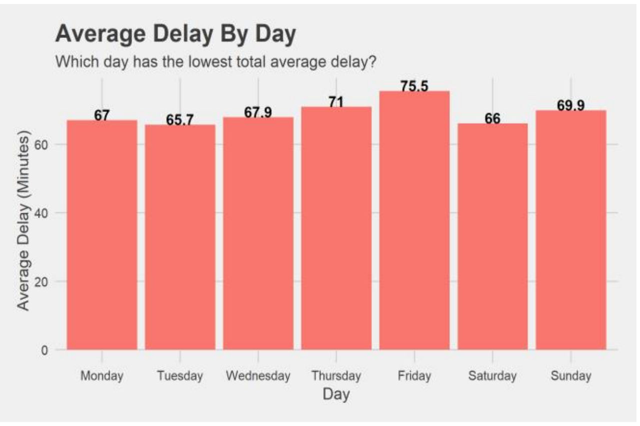
We will plot seaborn graph using a barplot by inserting the dataframe into the seaborn syntax and also set the axis accordingly, order is added to ensure that the variables are in the order that we want it to be. Loops are used to get the values of the bars for readability. Labels are added so that the graph will be easier to understand. Additional texts are also added to annotate the actual time frame for the bars graphed.

Question 1 (Graphing with R)

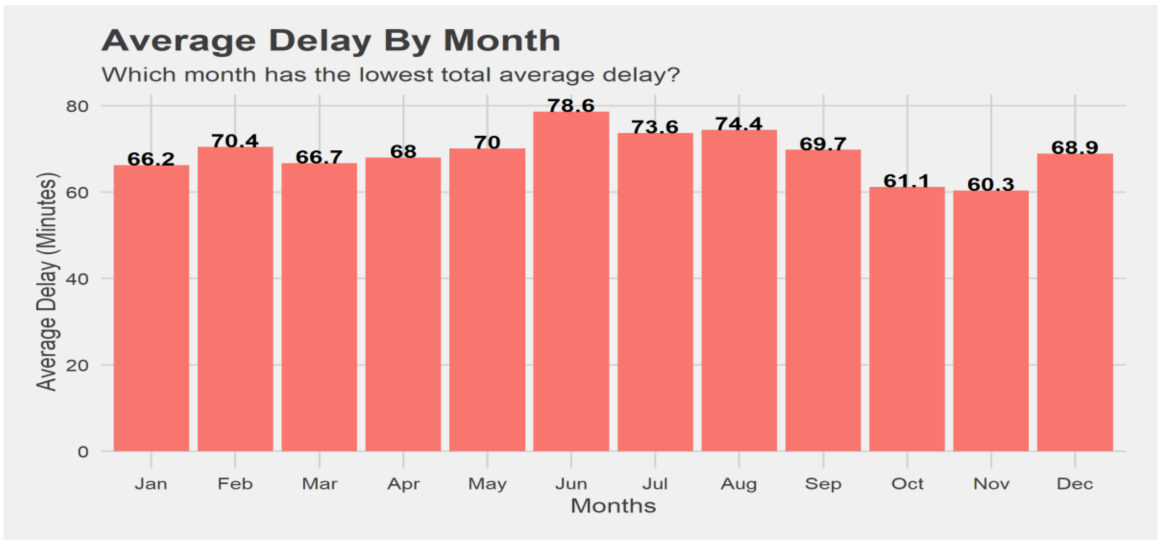
A vector is created to order the time groups manually. We will then pipeline the query q1a and drop rows with empty fields in the time query. The axis can be placed inside the aesthetics. We will place the timevector we created into the scale to order and arrange the bars accordingly. A bar graph is then created, and the theme is set using theme_fivethirtyeight. However, theme(axis.title = element_text()) has to be added in as the fivethirthyeight theme labels automatically removes axis labels. Labs enables us to add in titles, subtitles and also edit the axis. Annotate syntax allows us to include additional texts into the graphs, here which allows us to add in actual time frames for readability.[1]

(Q1 R Graphs Figures 1 ,2,3 Respectively)





[1] Basic labelling will be referred to as "usual labelling" as to not repeat the points over again. Any new labelling that is not mentioned inside here will be mentioned.



Analysis of Q1

The time with the lowest average delay would be a Tuesday early morning in the month of November with an average delay of (29+60.3+65.7)/3 = 51.6 Minutes. However, October also seems to be a close second after November with monthly average delay of 61.1. Flight delays only get worse as the day go on, with midnight averaging 188.7 minutes of delay. The reason why early morning has the lowest delays, is because the flights are less prone to turbulence and most thunderstorm tend to occur in the afternoon, and airports starting getting packed with passengers from 11am to 9pm onwards. Hence, morning periods have the lowest delay due to the better atmospheric conditions. Hence, the best time to travel with the least average delays would be the period of October and November, on an early morning.

Question 2 (Python and R)

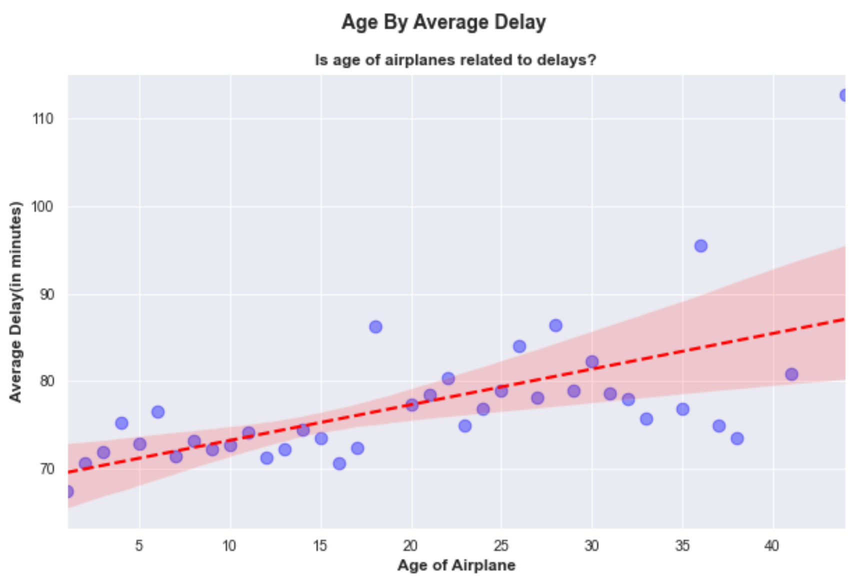
Age is deduced by subtracting plane_data.year from ontime.year. We use natural join to link columns from 2 different tables together by using their common column, Tailnum. The manufacture date of the plane has been conditioned to record planes which are only made before 2002, in addition to the conditions stated above. The condition of ontime.year > plane_data.year has also been set so that manufacture year will always be earlier than the year of the flight. We group the planes by Age so that all planes of same age are grouped together in order to see if age is a factor of delay. Q2 is also converted from a list to a data frame in order to work with seaborn.

Question 2 (Graphing with Python)

sns.set_style('darkgrid') is done to set a dark theme for the graph. Regplot plots a regression graph and the axis and the dataframe has been placed inside the syntax. Additional styling has been added to change the opacity, size, colour and line styles. Ci = 95 is added to set the confidence interval to 95%. Usual labelling has been done to make the graph neater and easier to read.

Question 2 (Graphing with R)

Q2 is pipelined into ggplot and the axis is inserted into the aesthetics. We create a scatterplot with the size and opacity adjusted to see different datapoints. Usual labelling is done according to the data. A theme is also added. A regression line is inserted into the graph, and the line type and colours are adjusted. The regression line displays the connection between the scatter data points which helps us to see the correlation between the two variables clearer.



(Q2 Python Graph, Figure 4)

Analysis of Q2

From the figure above, we can infer that yes, there is a correlation between the age of airplane and the average delays. There is a gradual increase in the delays as the airplanes advance in years. However, there are datapoints which are quite far from the regression line, this mean that there may be other factors of the plane that be affecting the delays such as the Model of airplane, the airline and the new technologies used in creating the airplanes as a later model of an airplane would be more optimised and perform better with age. E.g (An airplane made in 1960 vs an airplane made in 1995). All in all, there is a correlation between age of airplanes and delays but there are also other factors that may play a part in the delays.

Question 3 (R and Python)

A query with year, origin and dest and count of dest is done from the ontime table. However, aside from the usual conditions, we also have the origin set to “ABE”. This allows us to target only ABE as the origin and count for the total flights from ABE to other airports in 2000 and 2001. Grouped by Origin, Dest and followed by year. They are also in the order of Origin. Limits are set so that we would only see the first 200 results. The origin can be changed to any other origin to track for outgoing flights from the airports that we wish to.

Question 3 (Graphing with Python)

A Facetgrid is created with the query, and the columns and rows relating to Dest and Origin respectively are added. All the graphs also have their own y axis instead of a single y axis. Y limit is also set to 3000, with the colour of bars being set to yellow.

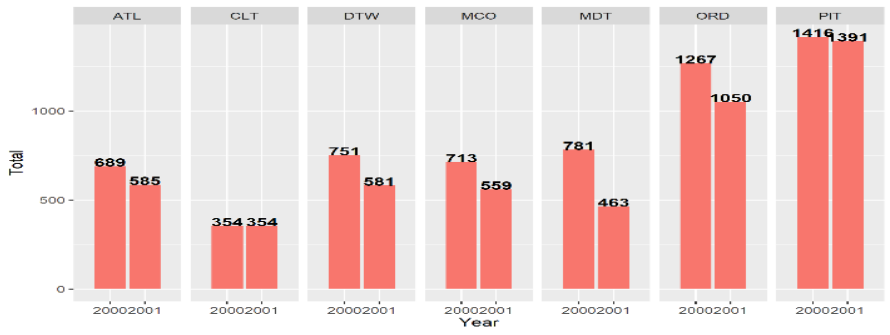
A function is created to show values on top of the bars to see the numerical differences clearly. Keyword arguments are used to handle named arguments in the function. Next, For loops are utilised to loop for the number of items in x, and annotations are added inside the loop function to annotate accordingly for every bar that shows up according to the origin. Lastly, the function is added into the map syntax with scaling and markers.

Question 3 (Graphing with R)

Q3a is pipelined into the ggplot, with the aesthetics of x and y axis set to Year and Total respectively. Geom_col is used to make height represent the values of the data, like bar charts.

The scaling of X ticks is coded to set the range of the data into 2000 and 2001 (only 2000 and 2001 ticks instead of 2000.2 ,2000.2…2001). The graph is also coded to set the graphs to show the Destinations based on the input of origin. Lastly, theme(legend.position=”none”) is added to remove legends created by the fill=””.

(Question 3 Graph of Total no. of Flights from ABE to other countries, Figure 5)



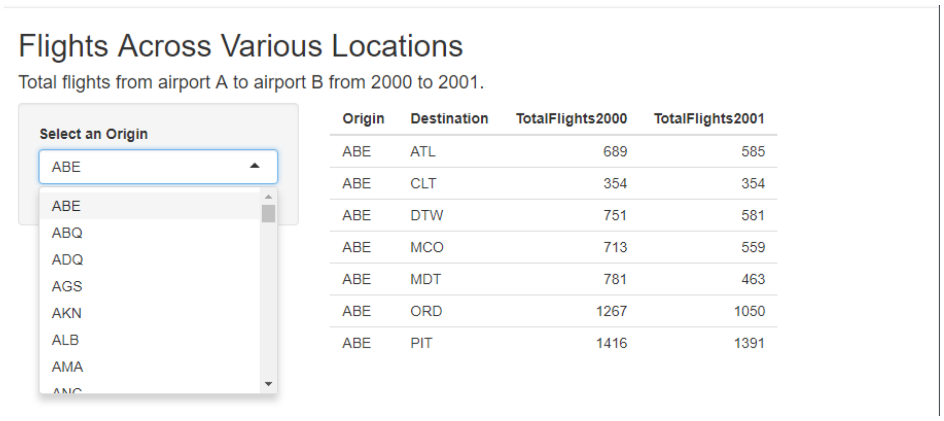
Question 3 (Visualisation with R Shiny)

Separate queries of 2000 and 2001 with their Origin, Destination and count of total flights are done. Once finished, we will combine with 2 queries of 2000 and 2001 with append, and turn them into a dataframe with as.data.frame. Once we’re done, we will take out of unnecessary columns by setting NULL to their column name.

Within the fluidpage function, we have a titlepanel and h4 for putting our title and text. Within the sidebarlayout and panel, there is selectinput which is a dropdown box for choosing our selections. Choices = q3e$Origin allows us to choose any Origin from the query. The main panel for viewing the output of our selections in a table format of “q3edata”.

For the shiny server, shinyServer(function(input,output) allows shiny to take the input from the user and give the output back to the user. The output$q3edata links back to the ui main panel function, renderTable is a reactive function which changes table values according the the user’s selection. With the help of subset function, we are subsetting the values based on Origin and giving it back to the Origin filter. Shiny takes the input, filter the q3e data and assign it back to the Originfilter object.

(R Shiny Visualization for comparison of flights from x country to y country, Figure 6)



Flights Across Various Locations

Total flights from airport A to airport B from 2000 to 2001.

Select an Origin: ABE

Origin	Destination	TotalFlights2000	TotalFlights2001
ABE	ATL	689	585
ABE	CLT	354	354
ABE	DTW	751	581
ABE	MCO	713	559
ABE	MDT	781	463
ABE	ORD	1267	1050
ABE	PIT	1416	1391

Analysis Of Q3

From 2000 to 2001, there is a decrease in the number of total flights from ABE to other locations. For some destinations, there is a considerable decrement in the number of flights while other locations maintained or lessened. There could be other reasons for the decrease in flights, but one of the main factors could be the 9/11 terrorist attack which have left many fearful of travelling in an airplane for their holiday season from September to December 2001.

Question 4 (R and Python)

For Python and R, We have selected departure delay and arrival delay to see the correlation between them. Conditions of departure delay and arrival delay more than 0, Diverted=0 and Cancelled=0 is set. They are grouped according to DepDelay in order of lowest to highest.

For R specifically, we have also discovered the % of Delays in various airports. We created 2 separate queries from the database, one with Delayed flights and another with Total Flights. Functions such as changing from list to data frame, ordering, and removing of a column was done. Another calculated column was created to count the percentage of delayed flights. Columns are also renamed for easier readability, and lastly top 10 busiest airports and least 10 busiest airports were calculated.

As additional analysis material for Python specifically, NetworkX package is used in order to create a network between the carriers and the Origin(iata). We have given weights to carrier and 1 line in the network resembles 1 flight by a particular carrier. This would mean that the busier airports would have more carriers, which would show how busy it is. The network graph is placed at the end of the report as it is an additional graph.

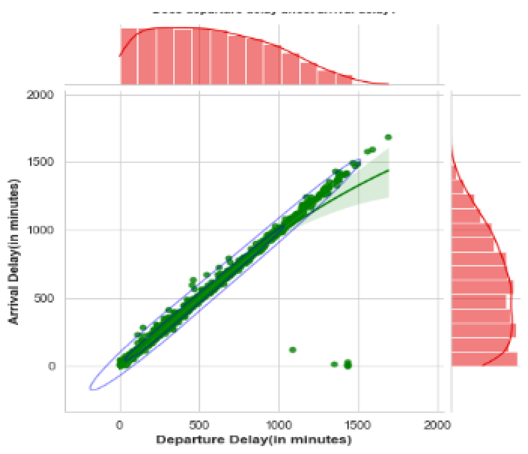
Question 4 Graphing with Python

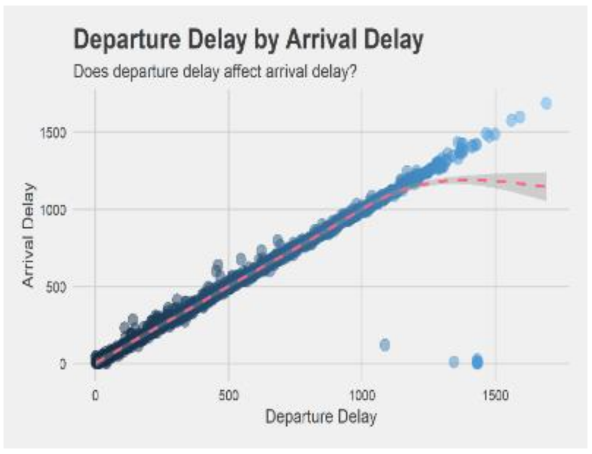
Q4 is transformed from a list to a dataset using pandas. The dataframe is placed into the jointplot with the x and y axis being set to DepDelay and ArrDelay respectively. The kind is set to regression with colors and space being set as well. Confidence interval is set to 95% and the order is set to 2. A kernel density estimator (kde) plot is also added into the jointplot with the colour being blue and the opacity set to 0.5, and levels set to 3. Labelling is also done according to the axis and titles.

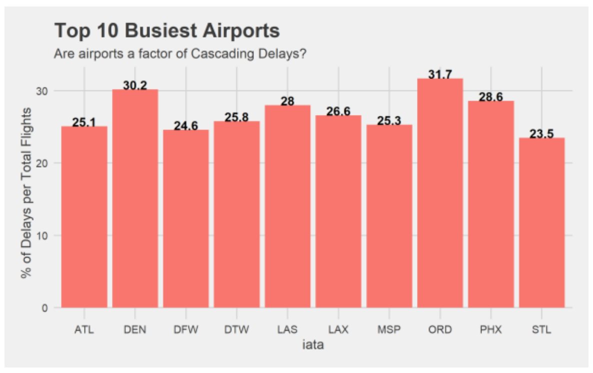
Question 4 Graphing with R

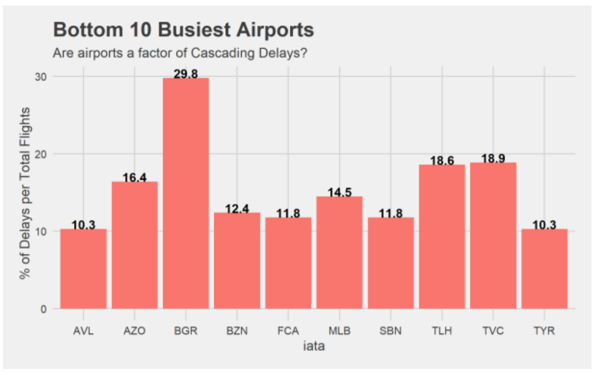
Q4 is pipelined into ggplot, with the x and y axis being set to DepDelay and ArrDelay respectively, the colour is also set according to DepDelay. A scatterplot is done with size 3.5 and opacity of 0.5 and Geom Smooth is used to add a regression line.

(Python and R graph for Departure Delay Against Arrival Delay, Figure 7,8 respectively)

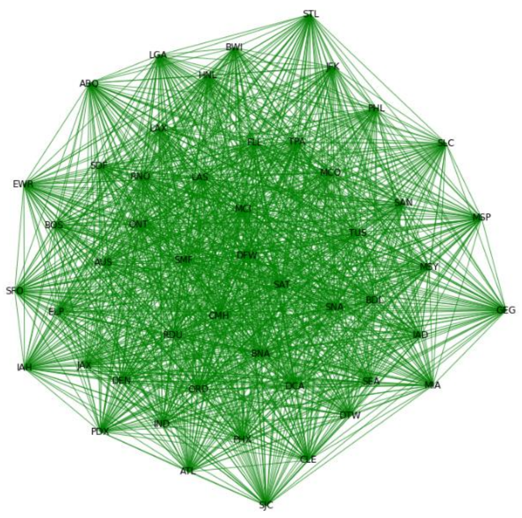








(R Graph for Top 10 Most Busy Airports and 10 Least Busy Airports Figure 9,10 respectively)



(NetworkX Graph, Figure 11)

<u>Q4 Analysis</u>

From the figures above, we can conclude that there is a direct relationship between departure delay and arrival delay. This can definitely cause a chain reaction for customers who are taking connecting flights. When a flight from airport A to airport B is delayed, it is apparent that the connecting flight from B to C will also be delayed. Hence, cascading failures is bound to happen when flights are delayed for more than a certain period of time. From the jointgraphs, the high number of delays is mostly from 0 to 500 minutes and slowly decline after 500 minutes.

We can also see that the busyness of an airport is also a factor of delays. As top 10 busiest airports suffer 25-30% of delays per total flights while only 15-20% of the least 10 busy airports suffer delays. Hence, we can infer that airports play a part in causing cascading delays. Some of the reasons may be that the popular airports may not be big enough to make up for the high incoming traffic/landings. Another reason may be that as many flights are happening, there is higher chance for errors and mishaps to happen which causes further delays. Last reason could be that the infrastructure and technology of the airport may not be as advanced, which causes delays. High number of airlines flying to and from an airport shows the busyness of an airport. All in all, from number of airlines to airports are all factors of cascading delays.

Question 5 (Python Regression Model)

Query is created by selecting the variables that are related to the Departure delay from the ontime table, with conditions of Departure Delay and Arrival Delay more than 0, and Diverted=0 and Cancelled=0 being set to filter out the rows that may affect our overall results. They are grouped by DepTime to see the instances in the time that they flight departs.

Q5 is then transformed into a data frame q5df from a list using pandas.

We set a variable x to the q5df data frame with DepDelay excluded, and set variable y to DepDelay which we dropped earlier, as we want DepDelay to be the target that we are predicting, which is transformed it into an array.

Using sklearn.model_selection, we will split the data using train test split. We split the data into 70% to be training data and 30% to be the testing data with the 2 variables x and y created earlier inserted into train_test_split. Random state is set to 100 in order to achieve reproducible results.

We will set our model to be model to be Linear Regression and then fit the 2 training models of x_train (Factor Variables) and y_train (DepDelay) into the Linear Regression.

Next, we set variable y_pred to be the prediction result of the x_test (which are factor variables) from the LinearRegression prediction model.

Once this is done, we can start predicting by inserting our own values into the prediction model, in order of the variables. IF condition statement is done to check the value of our prediction and print according statements to the minutes of delay.

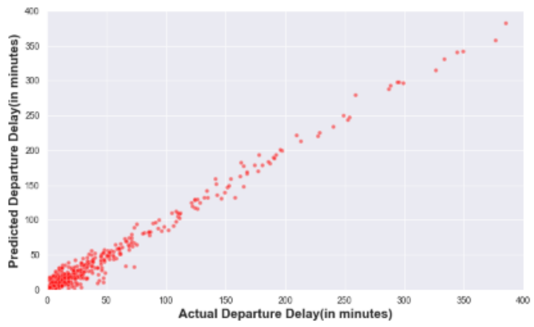
We can then check the accuracy of our model using r2 _score by measuring the y test result against the y_pred. Mean squared error can be calculated by setting variable mse for mean squared error with y_test against y_pred.

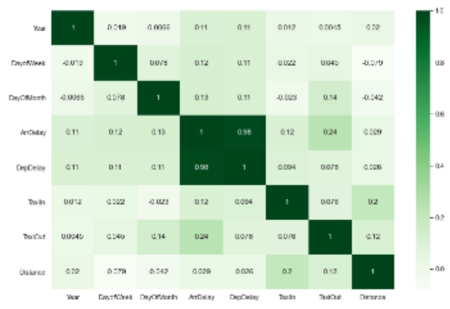
Lastly, Random Forest regression is done with the same steps as above and finally, the 2 models are compared using their accuracy and error.

Question 5 (Regression Prediction Model with Python)

Figsize is set to 10x6 to see the figure clearly. A scatterplot is done to place the datapoints on the graph, with Actual Departure Delay plotted against Predicted Delay with opacity of the datapoints set to 0.5 and the color changed to red. Labels are also done to understand the graph better, and limits being set to 400 for both axis as there is only 3 or 4 outliers after 400.

Additionally, a heatmap is also done in order to see the correlation between the variables. Figure size being set to 12x8 to see it clearer. Colour of the heatmap is also set to Greens, as the greener/darker the hue, the higher the correlation.





(Q5 Linear Regression Prediction Model and Heatmap for correlation, Figure 12 and 13)

<u>Q5 Python Classification Model (Python)</u>

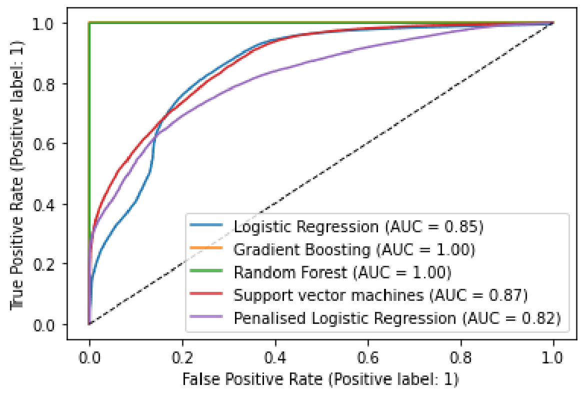
Various Classification Models have been done for python to check whether there will be a delay. A new column called "DelayClass" is created to check whether there is any delay using ArrDelay and DepDelay. All the features/variables that we will be using is separated from the original dataframe and named X.

Afterwards, we separate the numerical features and categorical features. We will apply simple imputation to numerical features and additionally for categorical, we will apply onehotencoder to it for the creation of dummy variables, and lastly combine everything together.

Next, we will then create the training the test set and assign it to 70% for training and 30% for testing, with random state set to 1 to reproduce the same results. For numerical features, we will be using the variable's mean and median values to make up for the variable's missing values while categorical features are imputed using constant and most frequent values. Y_train is also transformed from a multidimensional array into a 1-Dimensional Array.

Finally, we can place our training sets into the various machine learning models such as Logistic Regression, Service Vector Machine, Gradient Boosting, Random Forest and Penalised Logistic Regression. We will then rely on a ROC curve to compare the results of the 5 models.

On the whole, we can see that Gradient Boosting and Random Forest have an AUC of 1, this means the that classifier is able to perfectly distinguish between all the positive and negative class points correctly. Hence, we should choose either Gradient Boosting or Random Forest as each classifier has its advantages and disadvantages.



(ROC Curve for the 5 Classification Models, Figure 14)

Question 5 (R Regression Model)

A query is created with the variables that are could be a factor of DepDelay. A new task called "delay" is created with the backend q5 and the targeted variable to be DepDelay, as we are predicting the minutes of DepDelay.

We will be using mean squared error as our measure. We choose the ridge regression learner and ,inside param_set values, alpha is set to 0 to define the usage of ridge regression.

We will assign train set and test set to 75% and 25% respectively. We will then impute mean to account for the missing values using the variable's other values.

Next, we will set up tuning environment and specify hyperparameters and the range to be explored. The search method is also defined to be grid search, and the evaluations is to be repeated 20 times.

Lastly, we will put everything into the new learner and resampling done using cross validation, and place the assigned variables (imputation and tuning) into the new learner.

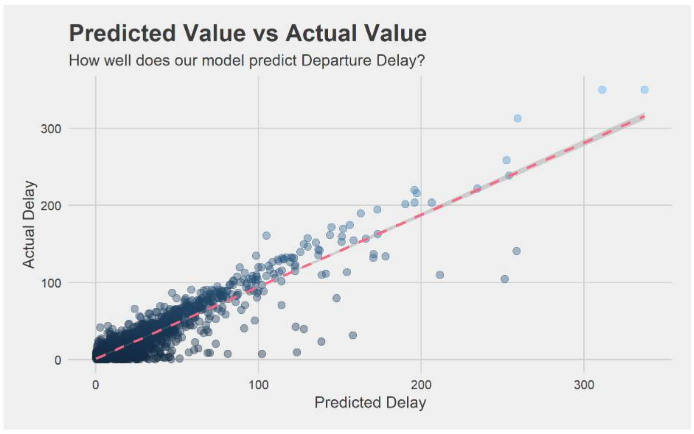
Afterwards, we can train the new learner and then predict the results.

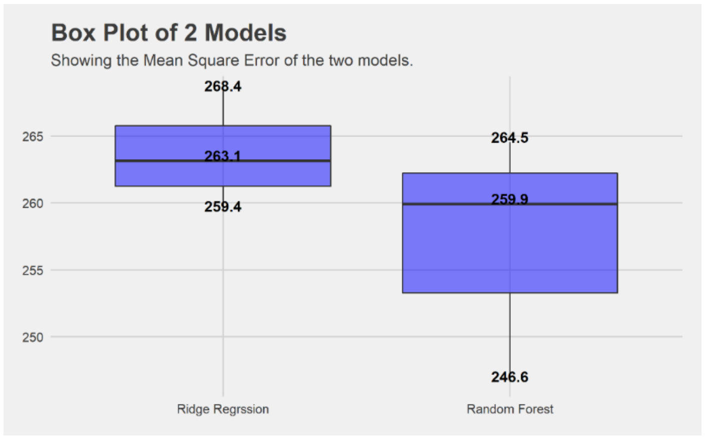
We repeat the same process with Random Forest with aid of the Ranger package, and afterwards we will benchmark the two models to compare which model is better overall.

We can lastly conclude which model is better by the results that the benchmarking gives us.

Q5 Prediction and Modelling (R)

We will place our prediction into the ggplot and set the axis to be response and truth for x and y respectively. Using labs to label it accordingly and theme styling to make it look neat and nice. Geom smooth is used to add in a linear regression line with a confidence interval of 90%/Below is a boxplot to compare the ridge regression model and random forest model and their results using MSE. Hence, we can see that Random Forest is a better model to predict the delays.





(Q5 R Ridge Regression Prediction Model and Benchmarking Model for Ridge and Random Forest, Figure 15 and 16.)

All in all, we can see that there are many factors that affect delays of a flight. We have discovered that even day of week are a factor of delays. As an airplane age, it is just natural for a vehicle to become slower and last productive than when they are just manufactured. People flying over different locations also tend to change over time with things that go on around the world, such as disasters of 9/11 as mentioned above. Various airports also have a higher chance for delays to compared to other airports which are likely causes of technology, airport size, and the high traffic amount. On the whole, there are many factors which play a part in delays of a flight and we can only predict them with the historical data given, however, outliers are still bound to happen once in a while.

References

The best time of day to avoid flight delays. (2022). Retrieved 30 March 2022, from https://www.businessinsider.com/best-time-of-day-to-fly-to-avoid-delays-2015-9#:~:text=They%20found%20that%20early%20flights,which%20they%20taper%20off%20slightly.

Linear Regression using Python. (2022). Retrieved 30 March 2022, from https://medium.com/analytics-vidhya/linear-regression-using-python-ce21aa90ade6

Beginner’s Guide to Creating an R Shiny App. (2022). Retrieved 30 March 2022, from https://towardsdatascience.com/beginners-guide-to-creating-an-r-shiny-app-1664387d95b3