ST2195 Programming
for Data Science Coursework

Name: Victor Min Thura Shwe

Student Number: 190398965

Contents

1. Coursework Question 1

1.1 Background Information

For question 1, I will be trying to find the best time of day, day of the week, and time of the year in which the average delay is at its lowest across datasets ranging from 2006~2008.

1.2 Implementation

1. Data Wrangling

Before starting on the question itself, I will be going through the process of how I utilised the data that was given initially using the files, “airports.csv”, “carriers.csv”, “plane-data.csv”, “2006~2008.csv” respectively. As I aim to use querying functions via SQL, I would first have to establish a connection to a .db file and in order to do that I would have to create one. Afterwards, I would have to establish several tables in the database that correspond to the .csv files that I have listed above with the “2006~2008.csv” files merged together into a table called “ontime”.

2. Best month of the year to travel

Before finding the best time of day and day of the week to travel, I first find the best time of the year to travel by comparing the months in terms of average delays, since there are 3 years the months are collated amongst them then averaged. By using a querying function to siphon specific variables that I require which in this case is the month, average departure delay and average arrival delay. I also created a new variable for this data frame that is essentially a sum of the average departure delay and average arrival delay called Average Delay so that it’s easier to interpret which month actually experiences the least amount of overall delays.

I then created a bar plot to depict the average delays between the different months of the year.

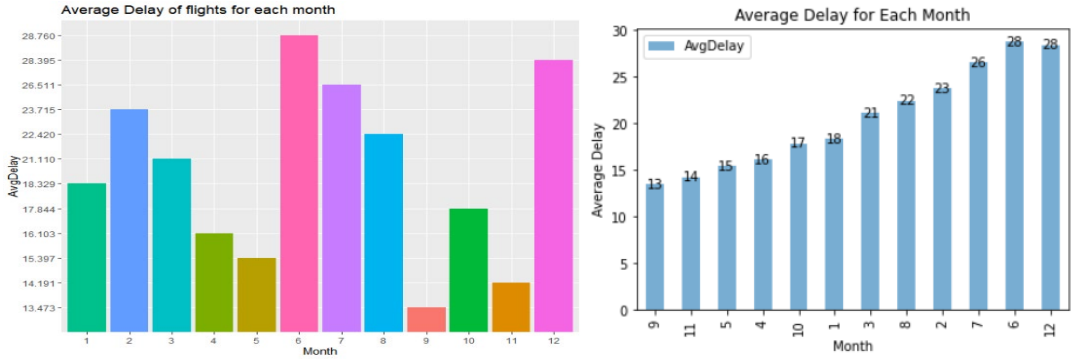

Figure 1: Bar Plot,R and Python for Average Delay in flights for each month

The bar plot depicts which months have the lowest average delays for flights and which months have the highest and in this case September, month 9 has the lowest average delays for flights therefore being the best month of the year to travel.

3. **Best time of day, Day of the week to travel**

   Using the query function once again, I can find that for the month of September across the years the average departure and arrival delays for each day of the week then by once again creating a new variable for the data frame that is essentially the sum of the delays. I then am able to answer this question by displaying a table that showcases the results with the average delay rounded to 3 significant figures.

Month	Day	Avg_DepDelay	Avg_ArrDelay	DepTime	AvgDelay
9	6	5.54	2.34	1112	7.89
9	3	4.72	3.21	1941	7.94
9	2	6.58	5.41	2012	12.0
9	4	7.03	6.97	1932	14.0
9	7	8.21	6.11	830	14.3
9	1	8.37	6.58	1927	15.0
9	5	11.1	11.0	1939	22.1

Figure 2:Table, Average delays for each day of the week for September

Therefore to answer the question, September would the best time of the year to travel in which the best Day of the week would be Saturday at a departure time of 1112 garnering the lowest average delay at 7.881 minutes or roughly 7 minutes and 53 seconds. It is to be noted however that Wednesdays at a 1941 departure time has a very close average delay time to the best being only 3 seconds slower.

2. Coursework Question 2

2.1 Background Information

For question 2, I will be trying to find whether or not older planes suffer more delays as compared to newer planes.

2.2 Implementation

1. **Data Wrangling**

   The standard in which I will be using to define a plane's age on would be the variable issue_date where from the data given, a plane's issue date can range from 1976 to 2008. I will also be using the database that was created in question 1.

2. **Whether older planes really suffer more delays in terms of top outliers**

   Initially to check whether older planes actually experience more delays, I start by checking the total of both departure delay and arrival delay grouping them based off their Issue dates to give a sense of essentially how the top outliers would look like. I then display the top outliers by essentially filtering the data that is a combination of the ontime and planes dataset for rows that have more than 0 in either arrival delay or departure delay then grouping them based on their issue dates while arranging them based on highest to lowest delay. The results are placed into issuedate. The results seem to lean more into the idea that newer planes experience more huge delays as compared to its older counterparts.

   However since the delay variables used are more generalised and some of the high delays can potentially be attributed to one-off incidents, I will be querying the dataset based on the five types of delays instead, CarrierDelay, WeatherDelay, NASDelay, SecurityDelay and LateAircraftDelay. It is to be noted too that the type of delay variable that is most applicable to this question would be CarrierDelay since it's the only variable that is within the control of the air carrier and how the age of a plane can only affect this variable since a plane's age can potentially make the plane more prone to damage. I'll be querying the delay variables based on their averages and then grouping them based off their issue date while ordering them from highest to lowest carrier delay. The results shown from the top 10 highest average carrier delays seem to once again lean more into the idea that newer planes experience more huge delays as compared to its older counter parts.

3. **General overview of whether older planes suffer more delays**

   An overall effect of how age affects delays can be visualised using a scatter plot. To do this, I would have to convert the data type of the variable Issue_Date into a date format of mm/dd/yy. I then am able to plot with reference to the data frame that I have created in part 2 with the 5 delay variables a scatterplot with Issue_Date as the x-axis and CarrierDelay as the y-axis.

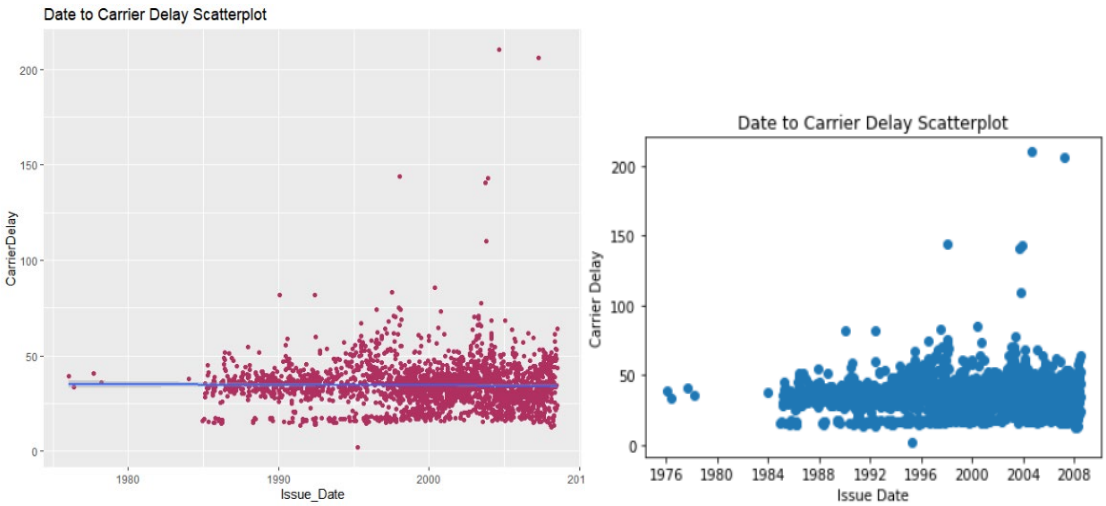

Figure 3: Scatterplots,R and Python for how the age of a plane affects delays

Then after plotting the scatterplots, I create a new linear regression model that is based off the line shown in the scatterplot to more accurately see whether carrier delays are increasing with newer planes or decreasing.

Therefore to answer the question, the linear regression models from both R and Python show issue date having a negative coefficient implying older planes do indeed suffer more delays as compared to newer planes although to a very small extent considering both coefficients are small. It is to be noted too that newer planes also have more outliers in terms of carrier delay explaining the findings in the part 2 and could possibly be due to one-off events that can cause massive delays.

3. Coursework Question 3

3.1 Background Information

For question 3, I will be trying to find how the number of people flying between different locations change over time or in this case from 2006 to 2008. As the dataset given does not have a variable that showcases the number of people in each flight, I will be supplementing this variable with the amount of flights between locations instead.

3.2 Implementation

1. Data Wrangling

To get a more accurate sense of the flights between different locations, I will be combining Origin and Destination to create a new variable, FlightPath. For example, ATLPHX can be read as ATL for the origin and PHX as the destination for the flight. With reference to question 1, the origin and destination variables will be taken from dataframe “ontime” to create a temporary dataframe which will then be column

binded to another dataframe “ontime2” and then be pushed into the database as another table. I will also be querying with use of the database created in question 1.

2. How flight patterns change over time

The idea is to create a table that is able to showcase how the number of people flying between different locations change over time for popular flights, in order to ease readability. Using the query function, I essentially made dataframes for 2006, 2007 and 2008 that each encapsulates the year, flightpath and the amount of flights that took place for each respective flightpath.

Year	FlightPath	TotalFlights	Year	FlightPath	TotalFlights	Year	FlightPath	TotalFlights
2006	LAXSAN	14283	2007	OGGHNL	15924	2008	SFOLAX	4469
2006	SANLAX	14257	2007	HNLOGG	15713	2008	LAXSFO	4344
2006	OGGHNL	13743	2007	LAXLAS	14137	2008	OGGHNL	4265
2006	HNLOGG	13690	2007	LASLAX	13586	2008	HNLOGG	4146
2006	LAXLAS	13385	2007	HNLLIH	13093	2008	LAXLAS	4057
2006	LASLAX	12937	2007	LIHHNL	12959	2008	LASLAX	3974
2006	LGABOS	11832	2007	LAXSAN	12606	2008	LASPHX	3794
2006	BOSLGA	11815	2007	SANLAX	12585	2008	LGABOS	3778
2006	PHXLAX	11650	2007	LASPHX	11925	2008	BOSLGA	3774
2006	LASPHX	11539	2007	PHXLAX	11624	2008	HNLLIH	3727

Figure 4: Table(Exactly the same on R and Python) representing popular flightpaths

By combining the dataframes for each year, I am able to present it in a table format where I have highlighted different changes in terms of flights between destination. Rapidly dropping in popularity are flights between LAX, Los Angeles and SAN,San Diego where they experienced a dip in popularity for both 2007 and 2008 possibly inferring that there is a high correlation for to-and-fro flights between these cities. In fact most flightpaths follow this trend where a “to” flight will likely be as popular as a “fro” flight likely due to people opting for round trips. A consistenly popular flight route would be between OGG, Kahului and HNL, Honolulu where they remained popular all 3 years.

However, it is to be noted that not all roundtrip flights follow this trend as for the case of LAS, McCarran and PHX, Pheonix sky harbour where the “to” flights are way more popular as compared to the return flights as evident from 2008 where return flights are not even as popular as the “to” flights.

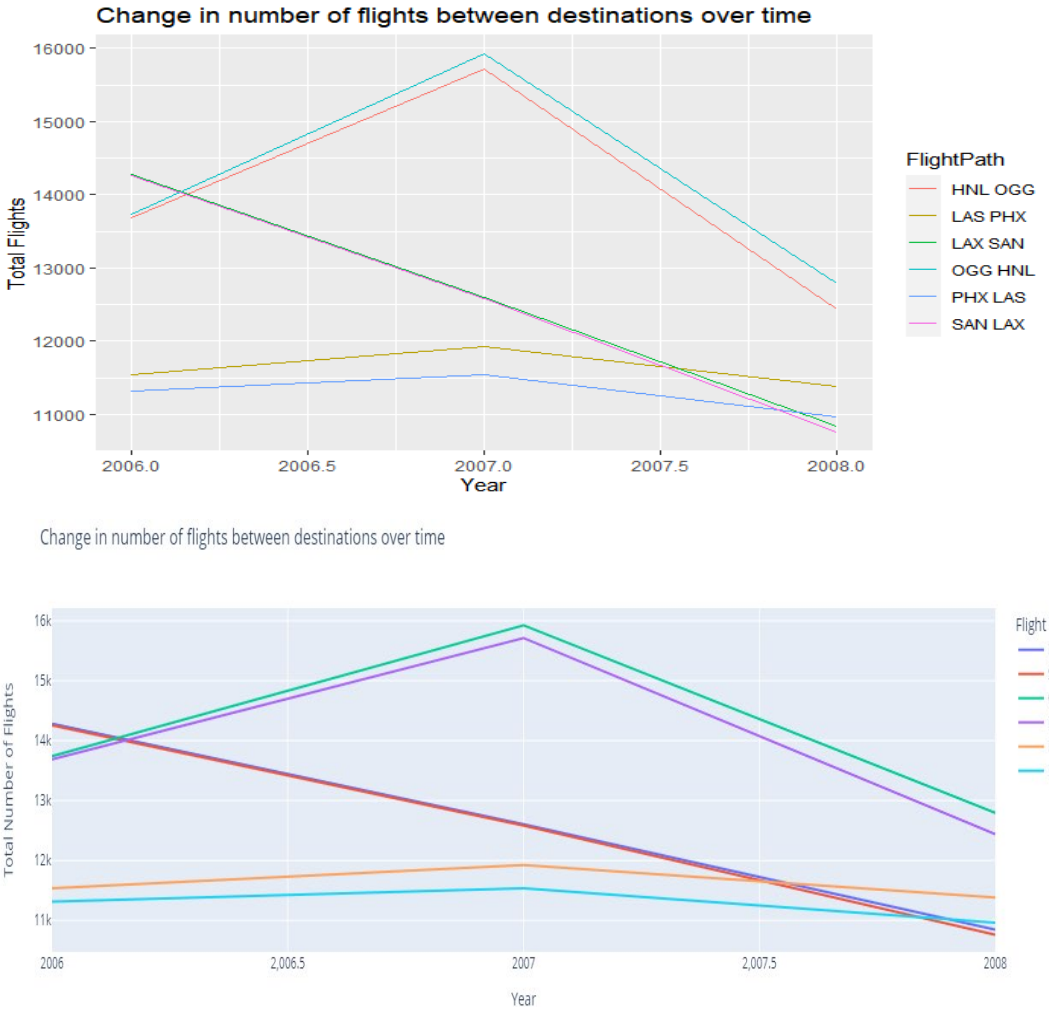

Figure 5: Change in number of flights between destinations over time,R and Python

Since 2008 only has data up to april, I multiplied Total Flights for 2008 by 3 as a rough estimation for the year's total number of flights. The lineplot depicts the changes that were mentioned where flights between OGG and HNL are consistently popular, flights between LAX and SAN are dropping in popularity and finally flights between LAS and PHX varies in the way that there is a lot more flights for LAS to PHX then PHX to LAS.

4. Coursework Question 4

4.1 Background Information

For question 4, I will be trying to detect cascading failure as delays in one airport creates delays in others. The way in which I intend to tackle this question would be to find if there is a correlation in terms of delays for the first "to" flight for departure and arrival and whether it is correlated to the following flight that takes place after this "to" flight and then determine if cascading failures is necessarily the issue. Therefore to ease this process, I'll only be using two airports in particular, MQT and ORD, Marquette and Chicago.

4.2 Implementation

1. Data Wrangling

In order to siphon the data I need, I will be querying with the use of the database created in question 1. The data that I am siphoning is essentially two dataframes with the first one having origin as MQT and the second one having destination as MQT, both dataframes are then merged on where the first one's destination aligns with the second one's origin at ORD in order to create the dataframe. Note that _x variables represent ones from the first airport and _y in the second airport.

I then drop all the origin and destination variables from the dataframe keeping in mind that the dataframe represents flights between MQT and ORD. All rows with NA cases are taken out, likely representing flights that have either diverted or were cancelled as they may interfere with clustering techniques later on.

2. Hierarchical Clustering

Before I am able to perform hierarchical clustering, I scaled the dataframe in order to put equal weight on each variable and then transpose it in to a different dataframe so that it is ready to be inputted into a dendrogram.

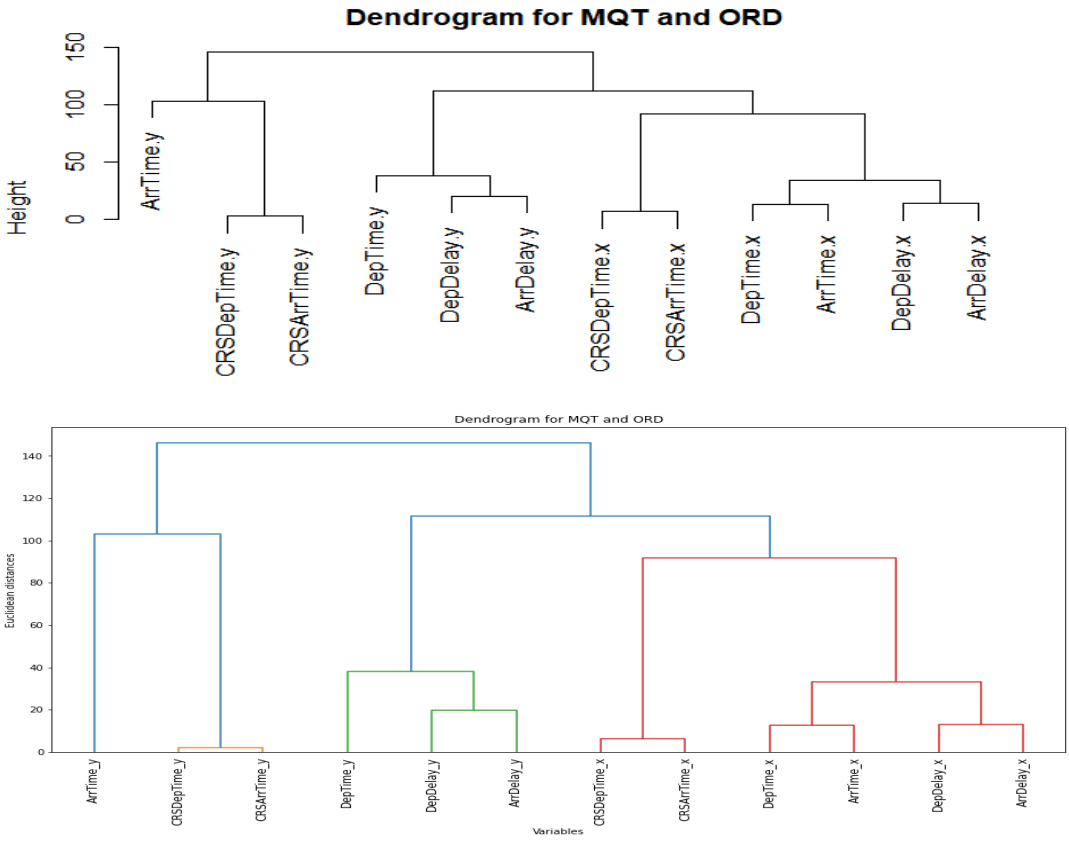

Figure 6: Dendrogram,R and Python for Hierarchical Clustering

From the dendrogram, we can see that 2 main clusters have been created with arrival/departure delay in the second airport being put in the same cluster as

arrival/departure delay in the first airport rather than the other cluster with the other second airport variables. To further summarise the data, I will be performing K-means clustering to find out how significant Arrival/Departure delays in the first airport are as differentiators for the Arrival/Departure delays in the second airport thereby checking for cascading delays.

3. K-Means Clustering

I performed K-means on the dataset while dropping the scheduled Arrival/Departure times in order to create two clusters, Cluster1 and Cluster2. The variables for Arrival/Departure delay in the second airport were also changed depending on how long the delay was into 4 factors, Not Delayed for values =< 10, Short Delay for >10 & =< 60, Medium Delay for >60 and =<150, Long Delay for the rest. The same variables but for the first airport are changed into only two factors which are Delayed for values >= 60 and Not Delayed otherwise. This is with the assumption that delays of an hour or more in the initial airport is more representative of a cascading delay effect occuring for flights that follow after it reaches the second airport.

Analysing the clusters, Cluster1 results show that for Arrival/Departure delay in the second airport has a majority of both Medium Delay and Long Delay as is also the case for Cluster2.

- For Cluster1, the Arrival/Departure delay in the first airport are both mostly filled with Not Delayed being 88.9% and 85.2% in terms of proportions. Therefore although Cluster1 is mostly full of there being no delays in the first airport, in the second airport however, there was still a majority of flights that still had huge delays implying that there is not much of a correlation in terms of delays in the first airport affecting the second airport.
- For Cluster2, the Arrival/Departure delay in the first airport are both mostly filled with Delayed being 92.5% and 87.5% in terms of proportions although this would align itself well into the idea that delays in the first airport can result in cascading delays, I am inclined to believe otherwise judging from Cluster1's case.

Therefore at least in the case for trips between MQT and ORD, it is not possible to detect cascading failure as delays in one airport, MQT creates delays in ORD in the flights back to MQT as the delays are more likely a result of other third-party interferences and also the correlation in delays between MQT to ORD and ORD to MQT is weak.

5. Coursework Question 5

5.1 Background Information

For question 5, I am tasked with constructing a model that predicts delays. In order to construct this model however, I would have to figure out suitable variables that would be helpful in predicting the delays which technically means that from the dataset I cannot use variables that are written only after a flight has completed its trip.

With this restriction in mind, the variables that I will be using are CRSDepTime, CRSArrTime, CRSElapsed time and Distance. Note that for timings with CRS in front, it represents the scheduled time.

5.2 Implementation

1. **Data Wrangling**

   Since the data in use is large, I will only be using one year's worth of data, 2007 for this question in order to streamline the process and make it more appropriate to run. The variables ArrDelay and DepDelay will be summed into TotalDelay as this is the variable that will be important to both models. I have also removed all cases that have NA's in them in fear that it would interfere with the regression calculations.

2. **Linear Regression Model**

   Firstly, I created a train-test split of a 7:3 ratio where the training data will be used to fit a model and the testing data for testing. I then created a linear regression model with all the variables that I listed previously along with TotalDelay, this resulted in a root mean square error[1] of **5412.41**. Ideally it would be best for a model to have a low RMSE hence in order to improve the model, I will be performing Ridge and Lasso regression. The results listed are with reference to the results attained from R.

   By performing Ridge regression on the dataset, I obtained a best lambda of **1.07958**. Noting that the lambda acts as the tuning parameter in which the higher it is the more the model variance reduces. I then derive the RMSE of the ridge regression which is **72.21144**. For comparison, performing Lasso regression resulted in a best lambda of **0.004792871** and a RMSE of **72.26283**.

   Once I have derived both regression model's RMSE, I can see which model serves as a better estimate for TotalDelay which in this case for delays is the ridge

[1] Root mean square error will be listed as RMSE from this point on

regression model. However, the difference in root mean square error is not a very significant amount and lasso regression also shows that disregarding distance as a variable would also work for the prediction model hence for cases where speed is key, the lasso model might be the preferred choice.

To answer this question however the model that I will be using is the ridge regression model for the prediction of delays. The ridge regression equation is as follows with variables coefficient being subjected to 3 significant figures for easier interpretability.

$$\boldsymbol{TotalDelay} = -17.7 + 0.0149\boldsymbol{CRSDepTime} + 0.00992\boldsymbol{CRSArrTime} + 0.0992\boldsymbol{CRSElapsedTime} - 0.0111\boldsymbol{Distance}$$

Judging by the first 5 predicted values of 17.0, 34.6, 12.1, 24.4 and 18.0, the values do not align very closely to the true values with the closest value being the 3rd value of 12.1 being off by 5.1 seconds to its true value of 7. Although they do not align very closely, it would serve as a helpful reference point for future scheduling of flights.

3. Logistic Regression Model

To create a logistic regression model, it would require a binary dependant variable. The binary dependant variable, Delayed can be derived from the variable TotalDelay where if there is an integer that's above 10 minutes, it can be displayed as a 1 and 0 otherwise. The reason for the 10 minutes would be that it is unfair to consider a flight that's for example only a few minutes late to be considered delayed hence 10 minutes would act as a buffer.

After the creation of the logistic regression model with Delayed as the dependent variable. I then create a test set of generated predicted numbers derived from the model which will be used to compare against the original test set.

The derived outcomes is then summarised and placed in a confusion matrix table to help compare model predicted values against the actual values for the test set.

	Predicted Delay: No, 0	Predicted Delay: Yes, 1
Observed Delay: No, 0	**1391065**	707585
Observed Delay: Yes, 1	44627	**39251**

Figure 7: Confusion Matrix

The logistic regression model sees an accuracy rate of 65.5%.

Both models can be used to predict delays, however the choice in which model to use would depend on whether the user require a specific value for delays.

6. Pages used

Excluding title page, table of contents and references. The amount of pages used is 10 pages.

7. References

Hierarchical Clustering

https://www.datacamp.com/community/tutorials/hierarchical-clustering-R

https://www.geeksforgeeks.org/cutting-hierarchical-dendrogram-into-clusters-using-scipy-in-python/

K-Means Clustering

https://www.datanovia.com/en/lessons/k-means-clustering-in-r-algorith-and-practical-examples/

https://realpython.com/k-means-clustering-python/

Line Chart

https://plotly.com/python/line-charts/

Pandas Guide

https://pandas.pydata.org/