UNIVERSITY OF LONDON
LONDON SCHOOL OF ECONOMICS AND POLITICAL SCIENCE

ST2195 Programming for Data Science
Coursework Project

Barbash Lev
lb237@student.london.ac.uk

Haifa
2023

Table of contents

Describing the data

For the analytics, 2006 and 2007 data on US flights from Harvard Dataverse is used. The coma-separated files were loaded into the local database. It makes the extraction of needed data simple using SQL queries. Particularly, the queries contain filtering the cancelled and diverted flights as they are not valuable in answering the questions stated in the coursework. Also, the flights from 2 years are merged into one "ontime" table, so analysing them together is possible. There are 4 tables:

1. Airports – contains the data on the 3376 US airports (identification and location info);
2. Carriers – contains the data on the 1491 airlines (identification info);
3. Ontime – contains the data on more than 14 million flights made in 2 full years. The table with the most important for the analysis information:
    a. Identification of the plane and flight;
    b. Data and time of flight;
    c. Origin and destination;
    d. Duration of flight, taxiing and delay (in minutes);
    e. Delay and cancellation codes and indicators.
4. Planes – contains the data on 5029 planes (identification, type, manufacturing, issue info).

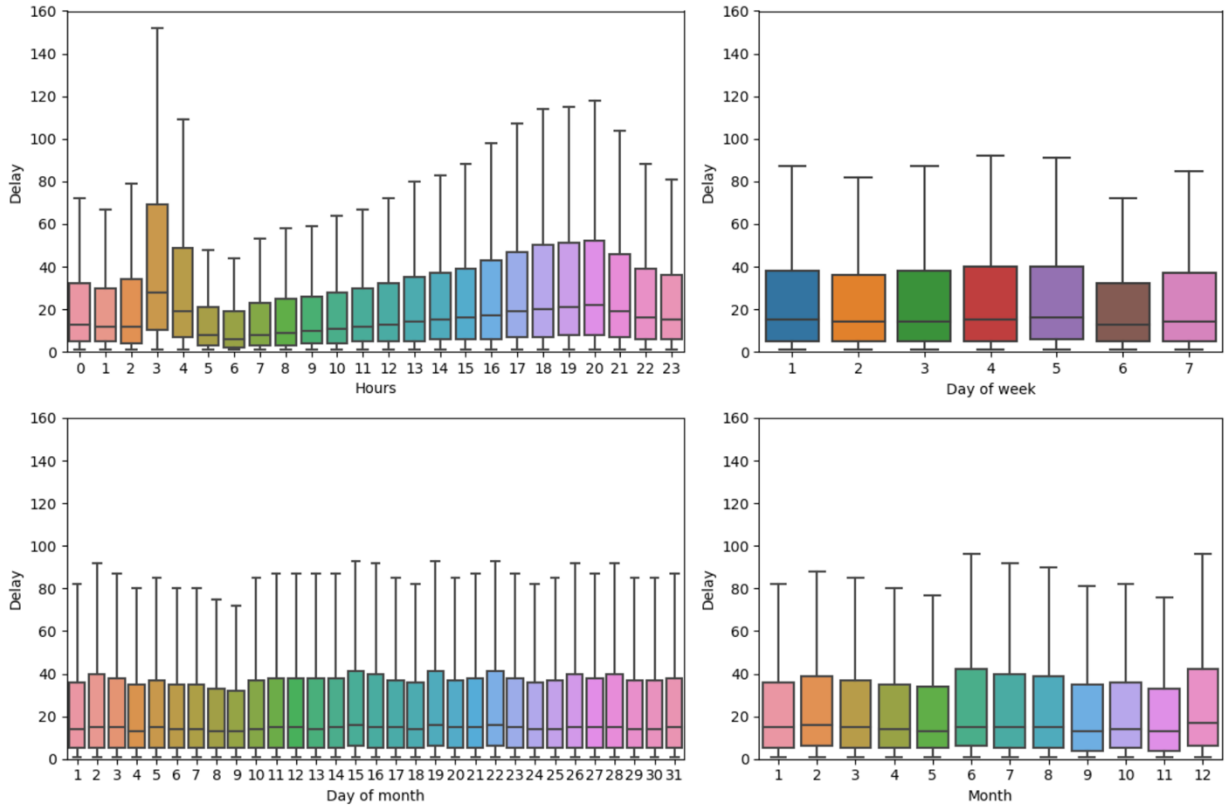
The processing, analysis, and visualization of the data allow us to answer the following questions.

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

There were made 4 SQL queries to answer that question for the scheduled time of departure, day of the week, day of the month and the month of the flight. Cancelled, diverted and not delayed flights are filtered out. Thus, 4 tables containing flight delay distribution for each dimension of time were extracted. The time of the

day table is generalized to a 24-hour format without minutes, so it could be represented and interpreted more conveniently.

The box plot for each time dimension has been chosen as the graphical representation for that question. Boxplots are simple yet informative, and they work well when plotted next to each other to visualize many distributions at once (Wilke, 2017).



The line in the middle of the boxplot represents the median, and the box encloses the middle 50% of the data. The top and bottom whiskers extend either to the maximum and minimum of the data or to the maximum or minimum that falls within 1.5 times the height of the box, whichever yields the shorter whisker (Wilke, 2017). The outliers were removed from the plot as the data has a large amount of them that could interfere with the visualization. To define the time with minimized delays the least boxes should be found.

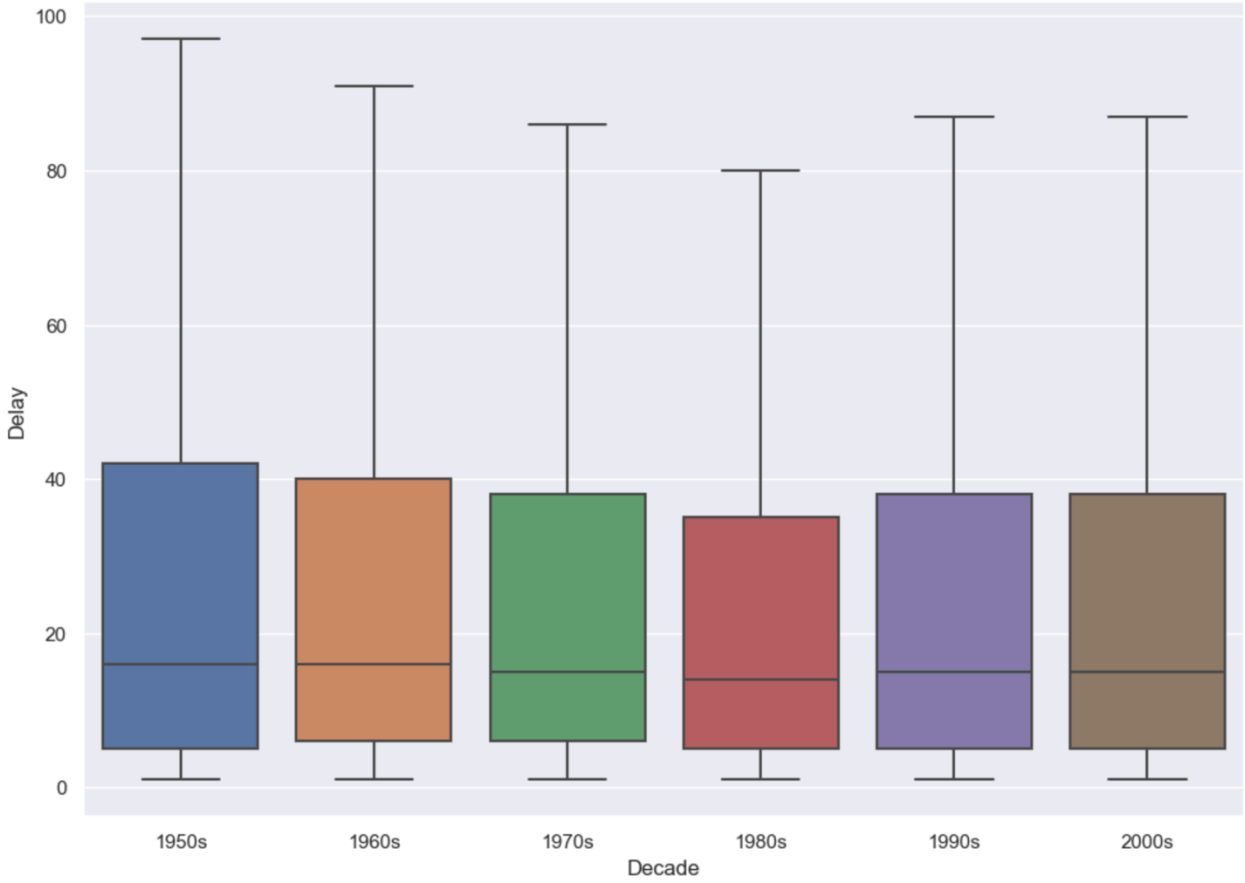
Thus, the best time is 6 o'clock, Saturday, 9th of May or November.

(The code for the question can be seen in best_time.R and best_time.py files).

Do older planes suffer more delays?

The year of manufacturing data from the “planes” has been joined to the flight delays using the talinum (plane id). Cancelled, diverted, not delayed flights and not appropriate year values are filtered out. The remained year values are generalized to decades from the 1950s to the 2000s. The generalization should allow capturing the possible delay decrease tendency.

The box plot without outliers is used to answer the question as the previous one.



The decrease in the boxes from the 1950s to the 1980s decades is displayed. There is a delay decrease tendency despite the slight increase in 1990s and 2000s planes.

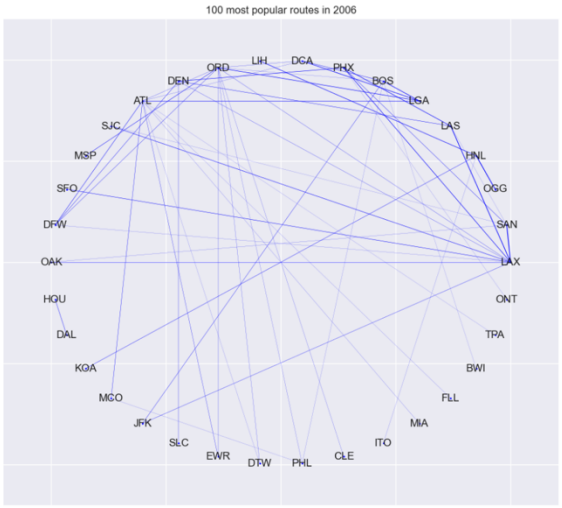
Thus, older planes suffer more delays.

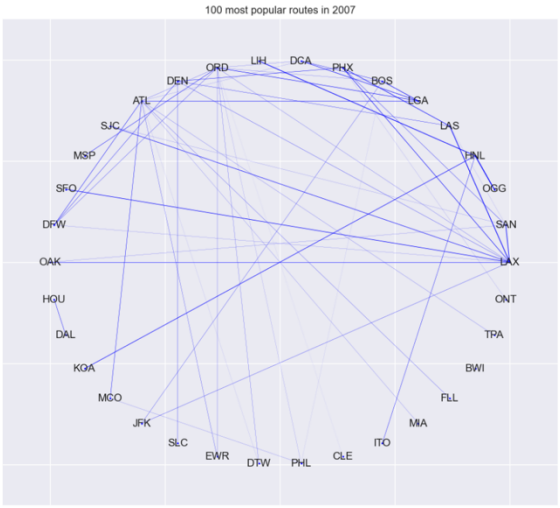
(The code for the question can be seen in older_planes.R and older_planes.py files).

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

There were extracted 2 tables for each year. The tables contain the origin and destination airport data. Thus, the data has been grouped by origin and destination to count the number of flights in each route. The number of flights can be compared between 2 years to answer the question. There have been selected number of flights in the 100 most popular routes of 2006 and compared with 2007 for more convenient representation.

Comparing network plots allows us to follow the changes in the number of flights. The number of flights is displayed as the width of edges between nodes. So, the values have been normalized.





The plot displays some changes in the number of flights, for example, the width of edges to and from Los Angeles Airport (LAX) increased. Also, a significant increase is captured between HNL and KOA - Hawaiian airports. However, less popular routes decrease, for example, the edges between ORD and PHL, ORD and CLE, BOS and BWI start disappearing.

Thus, the number of flights on the popular routes increases and the number of flights on the not popular routes decreases.

(The code for the question can be seen in number_of_flights_changes.R and number_of_flights_change.py files).

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

The date, scheduled and delayed time, origin, destination and flight identification data have been extracted. There should be a delayed flight that could create delays in the next flights from the destination airports. The third quantile of the departure delay distribution has been selected as the appropriate value that could create cascade delays. So, there is the origin flight:

	Year	Month	DayOfMonth	CRSDepTime	DepTime	DepDelay	CRSArrTime	ArrTime	ArrDelay	Origin	Dest	FlightNum
0	2007	12	27	1130	1207.0	37.0	1330	1404.0	34.0	CHS	DFW	3474

Now the flights from the DFW that have had to depart after the scheduled arrival time of the origin flight but departed later because of the arrival delay should be inspected:

	Year	Month	DayOfMonth	CRSDepTime	DepTime	DepDelay	CRSArrTime	ArrTime	ArrDelay	Origin	Dest	FlightNum
0	2007	12	27	1355	1458.0	63.0	1745	1829.0	44.0	DFW	ORF	682
1	2007	12	27	1335	1437.0	62.0	1745	1820.0	35.0	DFW	PHL	348
2	2007	12	27	1400	1458.0	58.0	1505	1606.0	61.0	DFW	DEN	1351
3	2007	12	27	1335	1427.0	52.0	1445	1542.0	57.0	DFW	MAF	3667
4	2007	12	27	1400	1447.0	47.0	1515	1548.0	33.0	DFW	SGF	3609
5	2007	12	27	1355	1436.0	41.0	1730	1804.0	34.0	DFW	PIT	3621
6	2007	12	27	1400	1440.0	40.0	1545	1641.0	56.0	DFW	BNA	1694
7	2007	12	27	1355	1430.0	35.0	1715	1738.0	23.0	DFW	CAE	3831
8	2007	12	27	1340	1413.0	33.0	1500	1537.0	37.0	DFW	BUR	2055
9	2007	12	27	1400	1427.0	27.0	1520	1536.0	16.0	DFW	JAN	3857
10	2007	12	27	1400	1427.0	27.0	1619	1621.0	2.0	DFW	ORD	538
11	2007	12	27	1345	1408.0	23.0	1445	1502.0	17.0	DFW	AUS	1587
12	2007	12	27	1355	1412.0	17.0	1455	1533.0	38.0	DFW	PSP	1767
13	2007	12	27	1350	1405.0	15.0	1730	1742.0	12.0	DFW	MIA	1246
14	2007	12	27	1355	1407.0	12.0	1530	1534.0	4.0	DFW	PHX	1457
15	2007	12	27	1400	1412.0	12.0	1510	1524.0	14.0	DFW	SHV	3267
16	2007	12	27	1400	1407.0	7.0	1745	1739.0	-6.0	DFW	FLL	1386

17 flights from DFW have been delayed because of the 1 flight from CHS.

Also, there have been detected another flight with the departure delay value equal to the third quantile:

	Year	Month	DayofMonth	CRSDepTime	DepTime	DepDelay	CRSArrTime	ArrTime	ArrDelay	Origin	Dest	FlightNum
258	2006	1	11	1545	1622	37	1825	1858	33	MSY	CLT	1244

The flights from MSY at the appropriate time:

	Year	Month	DayofMonth	CRSDepTime	DepTime	DepDelay	CRSArrTime	ArrTime	ArrDelay	Origin	Dest	FlightNum
40582	2006	1	11	1845	1927	42	2009	2051	42	CLT	IAD	7189
123202	2006	1	11	1852	1906	14	2003	2021	18	CLT	ORD	4325

Thus, there can be detected cascading failures as delays in one airport create delays in others.

(The code for the question can be seen in cascade_delays.R and cascade_delays.py files).

Use the available variables to construct a model that predicts delays

Only variables that are not related to the departure delay can be selected for the model. Also, instead of the airport IDs, should be used their coordinates as this information is more valuable for the prediction. The task of the delay prediction is the task of classification, so the departure delay data should be represented as binary variables:

Value “0” – departure delay is 0 or negative;

Value “1” – departure delay more than 0 minutes.

The logistic regression has been selected as the model. Logistic regression is essentially a classification algorithm. The word “regression” in its name comes from its close sister in the regression domain known as linear regression. Given that the classes are discrete in supervised classification problems, the goal of the algorithms is to find the decision boundaries among the classes. Decision boundaries separate examples of one class from another. The logistic regression model parameters are roughly the weights for the features. Each weighted feature vector is mapped to a value between 0 and 1 via the S-shaped logistic function. This value is interpreted as the probability of an example belonging to a particular class. (V.N. Gudivada et al, 2016).

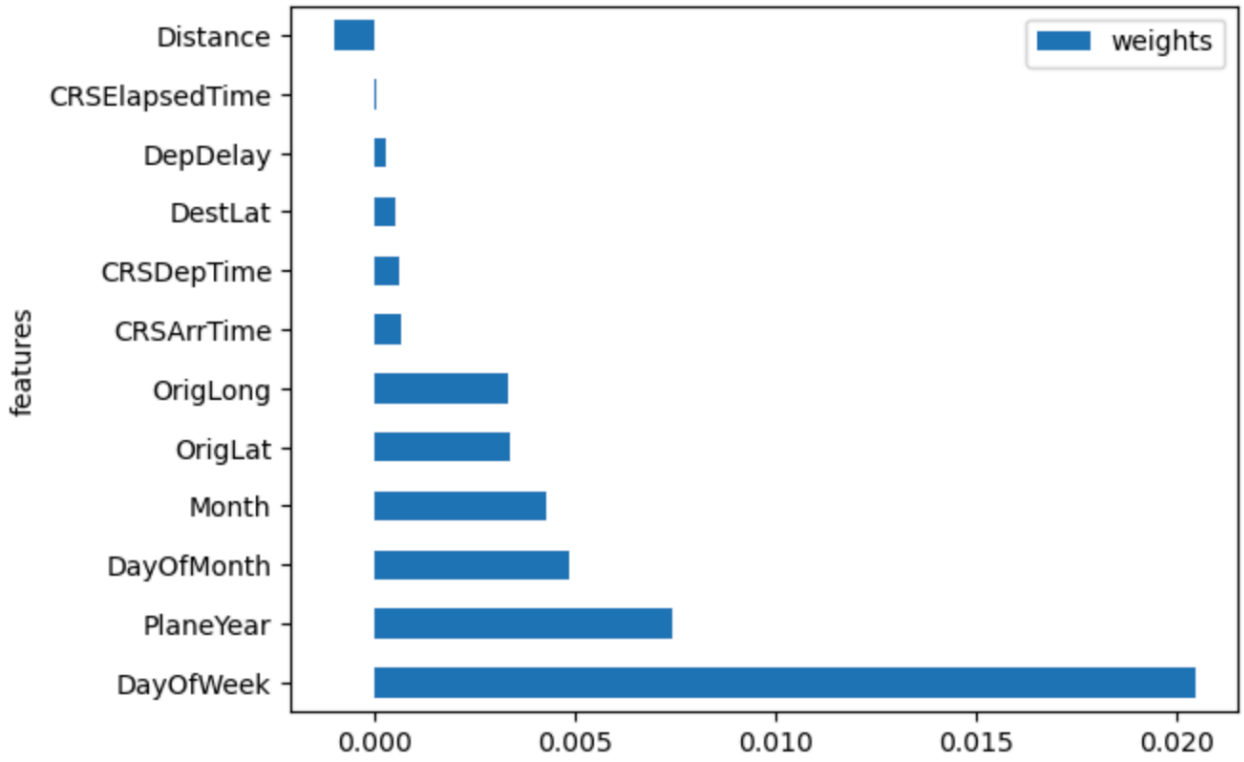
The model is appropriate for the task, as all the variables used are numeric and their effect on the potential delay can be interpreted easily as weights of the features.

The model should be trained, so the data has been divided into the train and test data in 50% proportion. The model scores:

Python – sklearn LogisticRegression – accuracy = 0,59;

R – used ml3 learner classif.log_reg – classification error = 0,31.

Also, the values of the coefficients of the Python model have been visualized:



The plot displays the significant effect of the “day of week” feature on the delay.

(The code of the model can be seen in ml_model.R and ml_model.py files).

References

1. Wilke, C. O. (2017). Fundamentals of data visualization. O'Reilly Media.
2. Venkat N. Gudivada, Vijay V. Raghavan, Venu Govindaraju, C.R. Rao. (2016). Chapter 5 - Cognitive Analytics: Going Beyond Big Data Analytics and Machine Learning. Handbook of Statistics, 35, 169-205.