STATISTICAL LEARNING PROJECT

Arianna Morandi, Beatrice Spagnolo, Tomas Guarini

RESEARCH QUESTION

Our analysis aim is to study the impact of multiple regressors, as flights departure delays, on flights arrival delays and how the same regressors can help in classifying either a flight will be delayed or not.

OBTAINING DATA

We firstly imported the “aircrafts” dataset from Kaggle. It contains all the civil flights of 2023 in the USA, from all the airports to all the airports. It is made of 6,743,404 rows and 24 columns.

```
flights <- read.csv("US_flights_2023.csv")
```

CLEAN AND FILTER DATA

We first selected only the flights that, leaving from all the airports, arrived in the JFK airport of NY, because what we want to study is the delay in arrival, given the conditions in the departure.

```
flights_JFK <- flights[flights$Arr_Airport =="JFK", ]
```

Merge of flights and weather

Here we added the weather data to our dataset, merging “flights” and “weather” through “FlightDate” and “Dep_Airport” variables, to also have information about the weather conditions at the departure airport the day of the flight.

```
weather <- read.csv("weather_meteo_by_airport.csv")
dataset <- merge(flights_JFK, weather, by.x = c("FlightDate", "Dep_Airport"), by.y = c("tim
e", "airport_id"), all = TRUE)
cat(paste(colnames(dataset), collapse = ", "), "\n")
```

```
## FlightDate, Dep_Airport, Day_Of_Week, Airline, Tail_Number, Dep_CityName, DepTime_label, D
ep_Delay, Dep_Delay_Tag, Dep_Delay_Type, Arr_Airport, Arr_CityName, Arr_Delay, Arr_Delay_Typ
e, Flight_Duration, Distance_type, Delay_Carrier, Delay_Weather, Delay_NAS, Delay_Security, D
elay_LastAircraft, Manufacturer, Model, Aicraft_age, tavg, tmin, tmax, prcp, snow, wdir, wsp
d, pres
```

Omit na values

The reason why we have some rows with NA values is that the merge operation keep all rows from both datasets, even if there are no matching records in the other dataset, and since some departure airports don’t have flights to the JFK this rows will have NA values from the first dataset.

```
dataset <- na.omit(dataset)
```

Delete variables

We decided to delete some variables that are totally not useful for our analysis.

```
useless_variables <- c("Tail_Number", "Dep_Delay_Type", "Arr_Delay_Type", "Distance_type", "D
elay_Carrier", "Delay_Weather", "Delay_NAS", "Delay_Security", "Delay_LastAircraft", "Manufac
turer", "Arr_CityName", "Dep_Delay_Tag", "Day_Of_Week")
dataset_total <- dataset[, -which(names(dataset) %in% useless_variables)]
```

Select flights

In this part we took only the flights that are delayed

```
dataset <- dataset_total[dataset_total$Arr_Delay > 0, ]
```

Factor variable

Here we created a factor variable, to be able to include it in our analysis, from the categorical variable “DepTime_label”. This new variable takes values from 1 to 4, that respectively represents the different departure moments of the day: Morning, Afternoon, Evening and Night.

```
dataset$DepTime_categoric <- factor(dataset$DepTime_label, levels = c("Morning", "Afternoon",
"Evening", "Night"),labels = c(1, 2, 3, 4))
head(dataset$DepTime_categoric)
```

```
## [1] 4 3 3 1 3 2
## Levels: 1 2 3 4
```

This is our final dataset shape (49597, 20)

```
cat(nrow(dataset), "rows and", ncol(dataset), "columns")
```

```
## 49597 rows and 20 columns
```

Variables

We then created a list containing all the numeric variables of interest to make it easier for further analysis.

```
variables_of_interest <- c ("Arr_Delay", "Dep_Delay",  "Flight_Duration", "Aicraft_age", "tav
g", "tmin", "tmax", "prcp", "snow", "wdir", "wspd", "pres")
```

Our variables are: “Arr_Delay” that represents the delay in arrival and is our dependent variable; “Dep_Delay” that represents the delay in departure; “Flight_Duration” that represents how long is the flight; “Aicraft_age” that represents how old is the plane; “tavg” that represents the average temperature in the departure airport; “tmin” that represents the minimum temperature in the departure airport; “tmax” that represents the maximum temperature in the departure airport; “prcp” that represents the mm of rain; “snow” that represents the mm of snow; “wdir” that represents the direction of the wind; “wspd” that is the speed of the wind and “pres” that is the air pressure.

```
head(dataset[variables_of_interest])
```

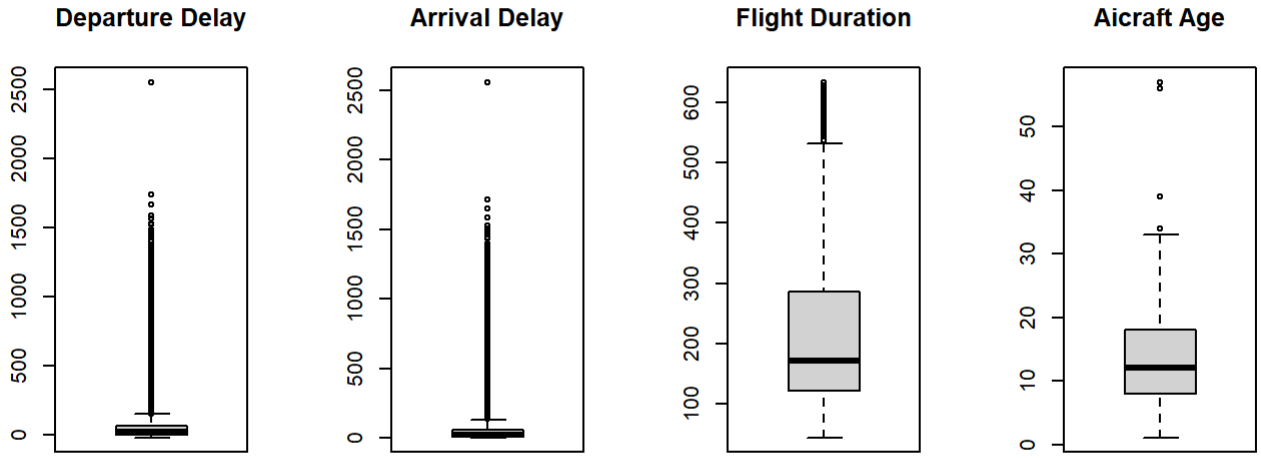
```
##    Arr_Delay Dep_Delay Flight_Duration Aicraft_age tavg tmin tmax prcp snow
## 22        50        38             145          24 14.1  8.9 21.1    0    0
## 24         9         9             134          11 14.1  8.9 21.1    0    0
## 25        28        27             135          13 14.1  8.9 21.1    0    0
## 27        17        19             133           6 14.1  8.9 21.1    0    0
## 28        43        46             136          23 14.1  8.9 21.1    0    0
## 29        31        36             130          25 14.1  8.9 21.1    0    0
##    wdir wspd pres
## 22  336  3.7 1019
## 24  336  3.7 1019
## 25  336  3.7 1019
## 27  336  3.7 1019
## 28  336  3.7 1019
## 29  336  3.7 1019
```

EXPLORE DATA

Boxplot

We draw 4 boxplots of all the numerical variables strictly regarding the flight and then 8 boxplots of all the numerical variables regarding the weather conditions.

```
par(mfrow = c(1, 4))
par(pin = c(1, 2))
boxplot(dataset$Dep_Delay, main = "Departure Delay", ylab = "Minutes")
boxplot(dataset$Arr_Delay, main = "Arrival Delay", ylab = "Minutes")
boxplot(dataset$Flight_Duration, main = "Flight Duration", ylab = "Minutes")
boxplot(dataset$Aicraft_age, main = "Aicraft Age", ylab = "Years")
```



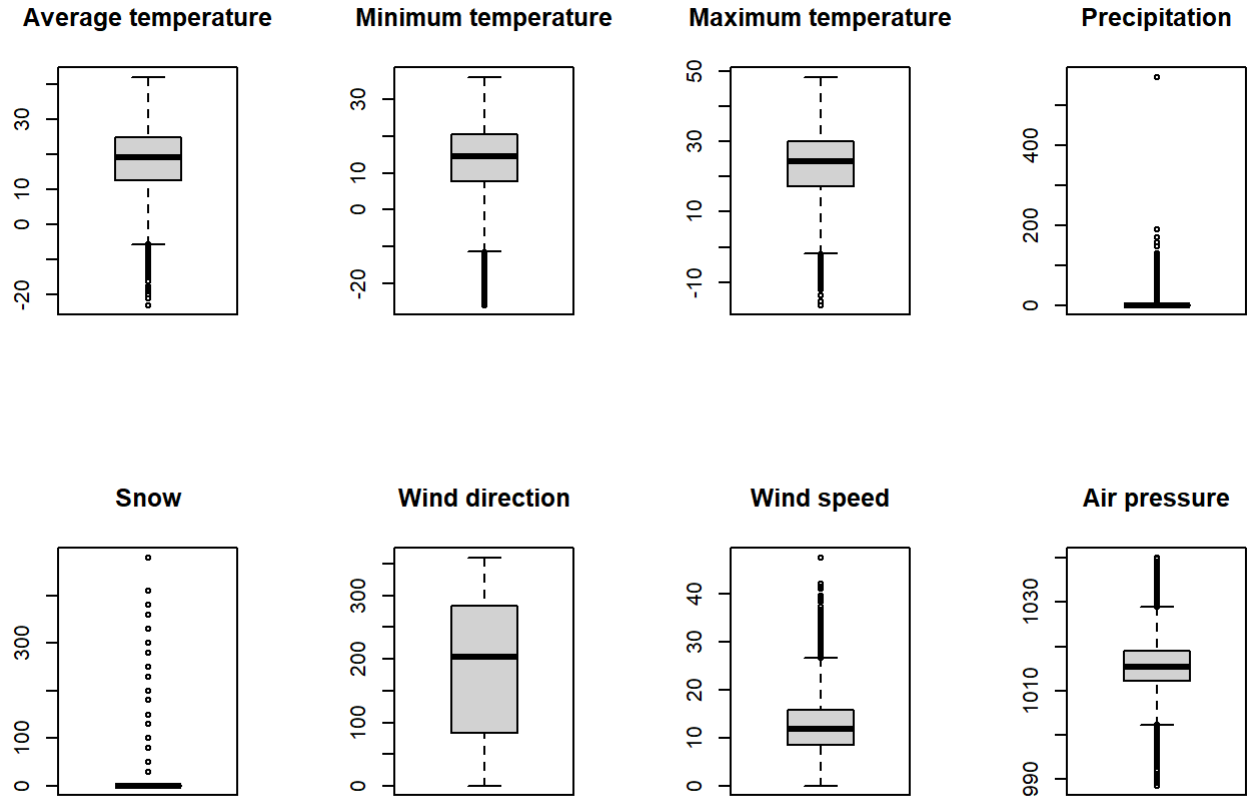

```
par(mfrow = c(1, 1))
```

Starting from the first two boxplots, and given that the line inside the box indicates the median of the data and the box contains all the values in between the first and the third quartile, we can observe that:

- the median is close to zero that indicates that the majority of our flights are on time or have a slight delay,
- we have a long tail of outliers that are the delays that we are studying and represent the natural distribution of our data,
- and there is a point that is extremely far from the others that we can consider as an outlier to subsequently delete.

For the other two boxplots we can notice that they have a more uniform distribution, but still with some outliers.

```
par(mfrow = c(2, 4))
boxplot(dataset$tavg, main = "Average temperature")
boxplot(dataset$tmin, main = "Minimum temperature")
boxplot(dataset$tmax, main = "Maximum temperature")
boxplot(dataset$prcp, main = "Precipitation")
boxplot(dataset$snow, main = "Snow")
boxplot(dataset$wdir, main = "Wind direction")
boxplot(dataset$wspd, main = "Wind speed")
boxplot(dataset$pres, main = "Air pressure")
```




```
par(mfrow = c(1, 1))
```

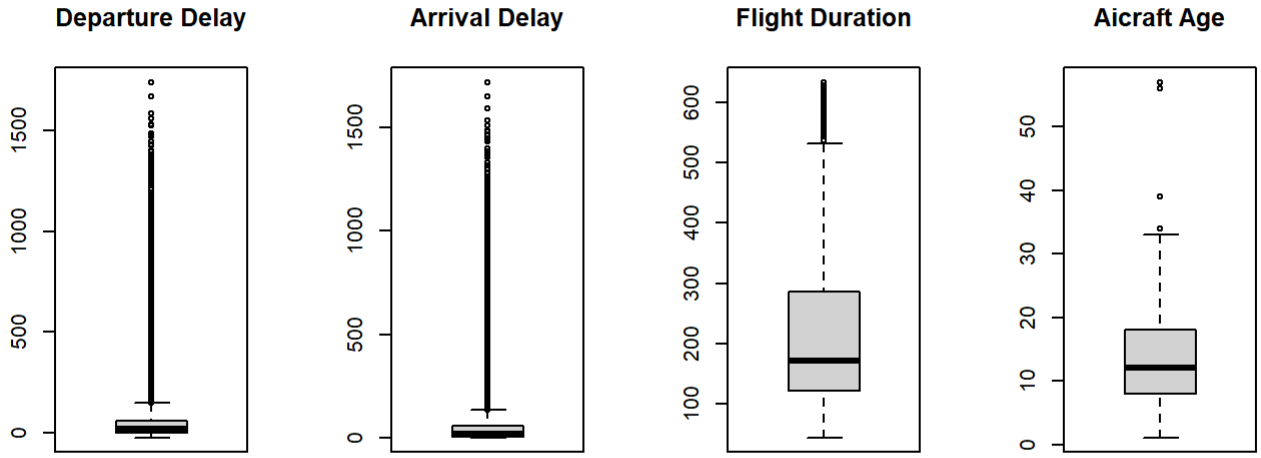
Starting from the boxplots of “Precipitation” and “Snow” we can notice that they have a median approximately equal to zero, that represents the fact that during the majority of the flights this two conditions didn’t verify.

For the rest of the boxplots instead we notice a uniform distribution, where the median value is close to the middle. “Average temperature”, “Minimum temperature” and “Maximum temperature” present outliers only before the first quartile, while “Wind speed” only after the third quartile. “Air Pressure” have outliers on both sided of the box and “Wind direction” doesn’t have any since it has higher variability and all the possible directions verify often.

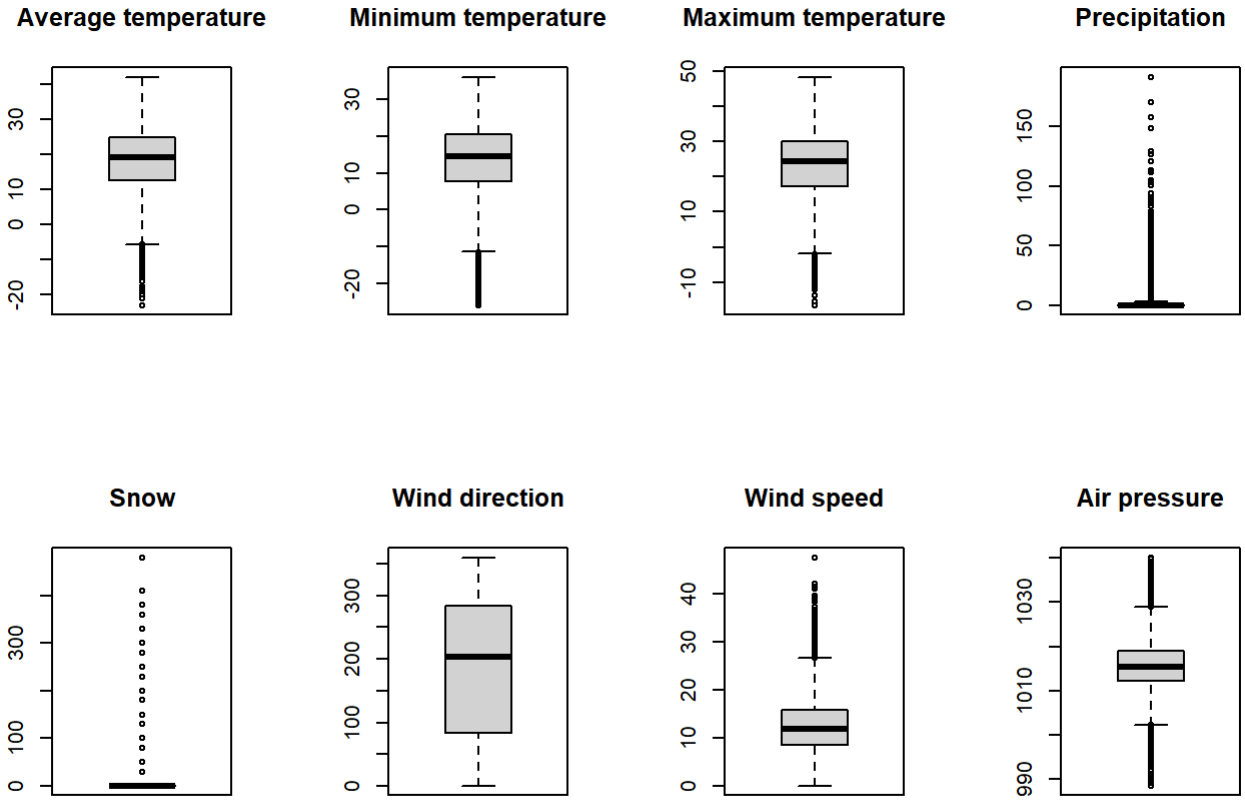
Outliers

We eliminate outliers related to variable “Dep_Delay”, “Arr_Delay” and “prcp”.

```
outlier_Dep <- which.max(dataset$Dep_Delay)
outlier_Arr <- which.max(dataset$Arr_Delay)
dataset = dataset[-c(outlier_Dep, outlier_Arr), ]
```




```
outlier_prcp <- which.max(dataset$prcp)
dataset = dataset[-c(outlier_prcp), ]
```



Covariance

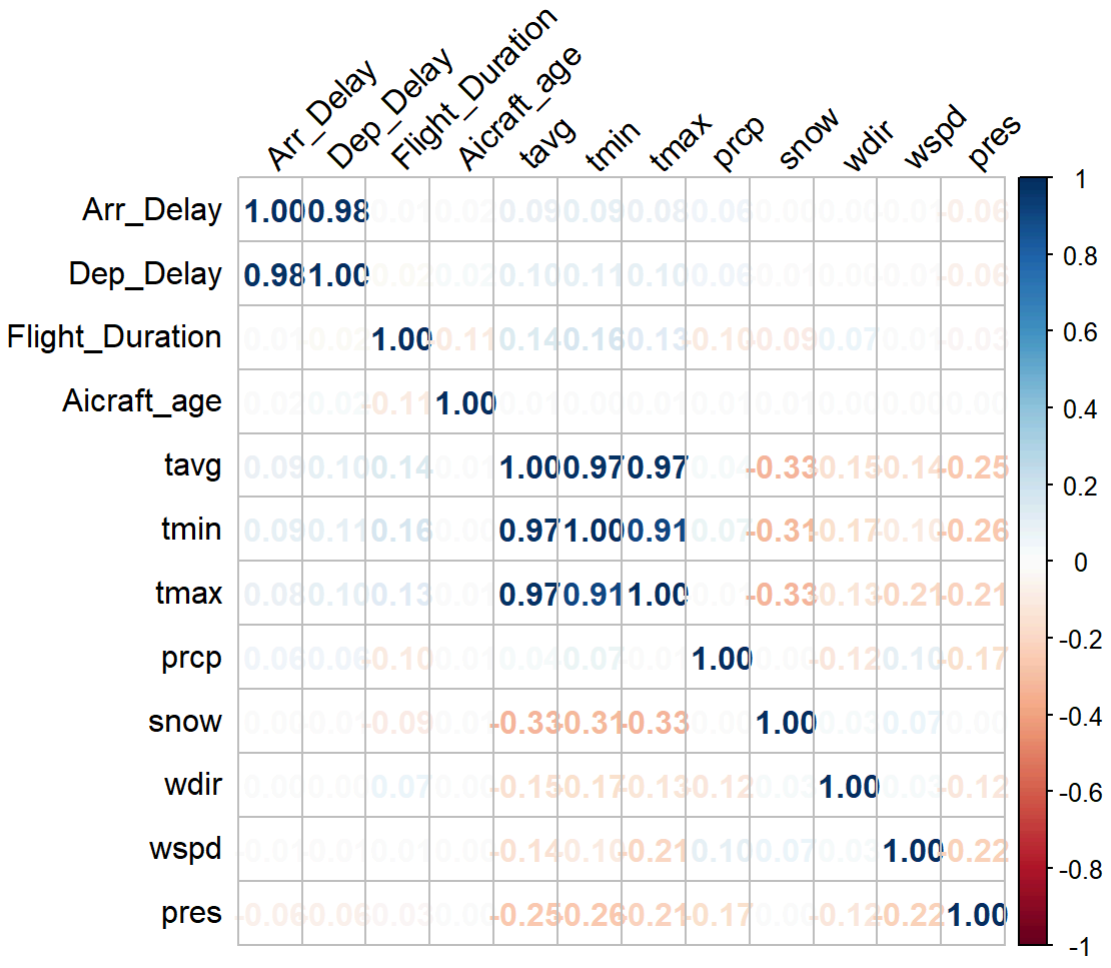
```
cov_matrix <- cov(dataset[variables_of_interest])
print(cov_matrix[ ,1])
```

```
##       Arr_Delay       Dep_Delay Flight_Duration     Aicraft_age            tavg
##    10988.279530    11076.286379      116.883910       13.257958       84.116435
##            tmin            tmax            prcp            snow            wdir
##       89.186819       82.101217       58.659624       -3.410022       19.125641
##            wspd            pres
##       -4.745323      -38.732823
```

Since the covariance describes how the variations in one variable are associated with the variations in the other variable, we can notice that we have 3 inverse relationship. For the variables “snow” and “wspd” the value is close to zero that means there is almost no covariance, while for the variable “pres” the value is low that means there is a stronger inverse relationship that is explained by the fact that when the air pressure decreases there is bad weather and so the flights delay could increase. In addition it is clear that the variable “Dep_Delay” is the one with the strongest covariance.

Correlation

```
cor_matrix <- cor(dataset[variables_of_interest])
corrplot(cor_matrix, method = "number", tl.col = "black", addCoef.col = "black", tl.srt = 45)
```



Since correlation measures the strength and direction of their linear relationship on a scale from -1 to 1, we can notice how “Dep_Delay” is the most correlated variable, while all the others are closer to zero so they have a weak correlation with our dependent variable.

MODEL DATA: REGRESSION

Train and test split

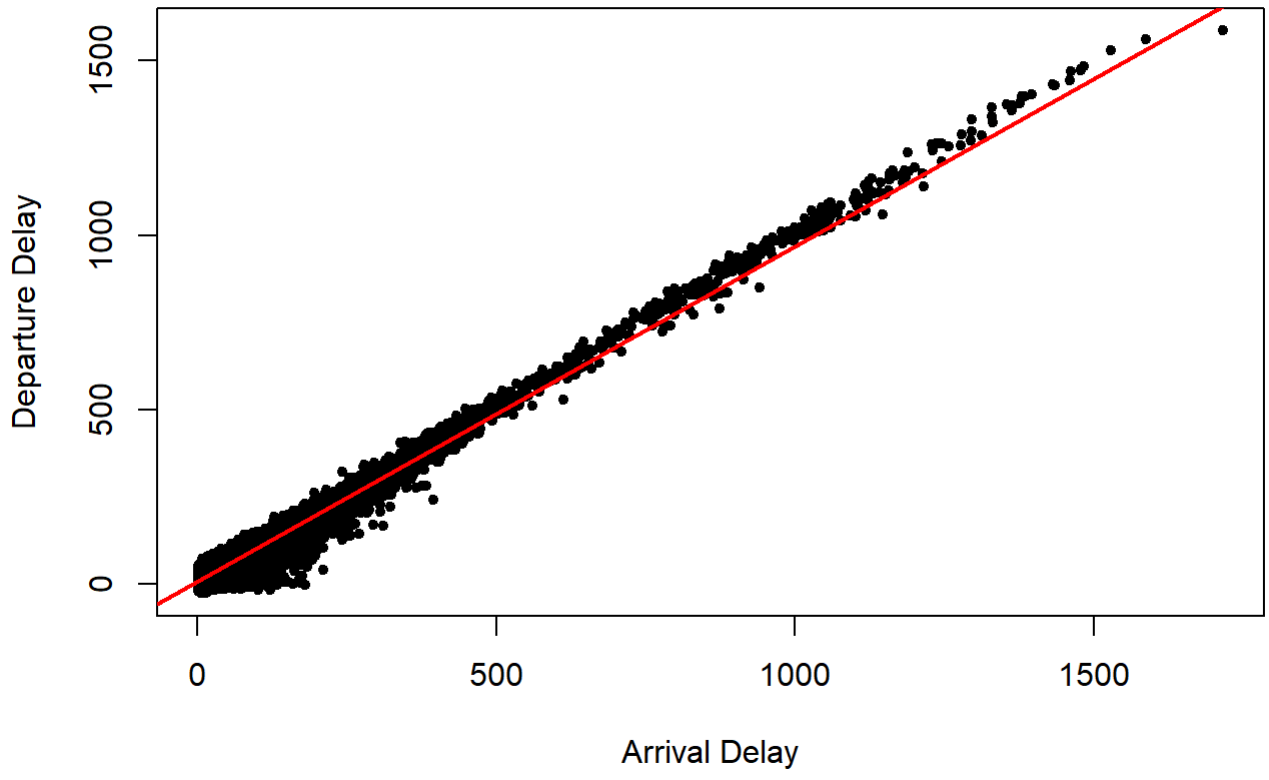
We initially split the data, using the flight date and we decided that all the flights from January to September were becoming the train set while the ones from October to December the test set.

```
train <- (dataset$FlightDate < "2023-09-30")
data_train <- dataset[train,]
data_test <- dataset[!train,]
arr_delay_test <- dataset$Arr_Delay[!train]
```

Simple linear regression

We decided to use as a regressor “Dep_Delay” for our simple linear regression model because of it’s strong correlation with our dependent variable.

```
linearmodel = lm(Arr_Delay ~ Dep_Delay, data = data_train)
plot(data_train$Arr_Delay, data_train$Dep_Delay, pch=20, xlab = "Arrival Delay", ylab = "Depa
rture Delay")
abline(linearmodel, lwd=2, col="red")
```



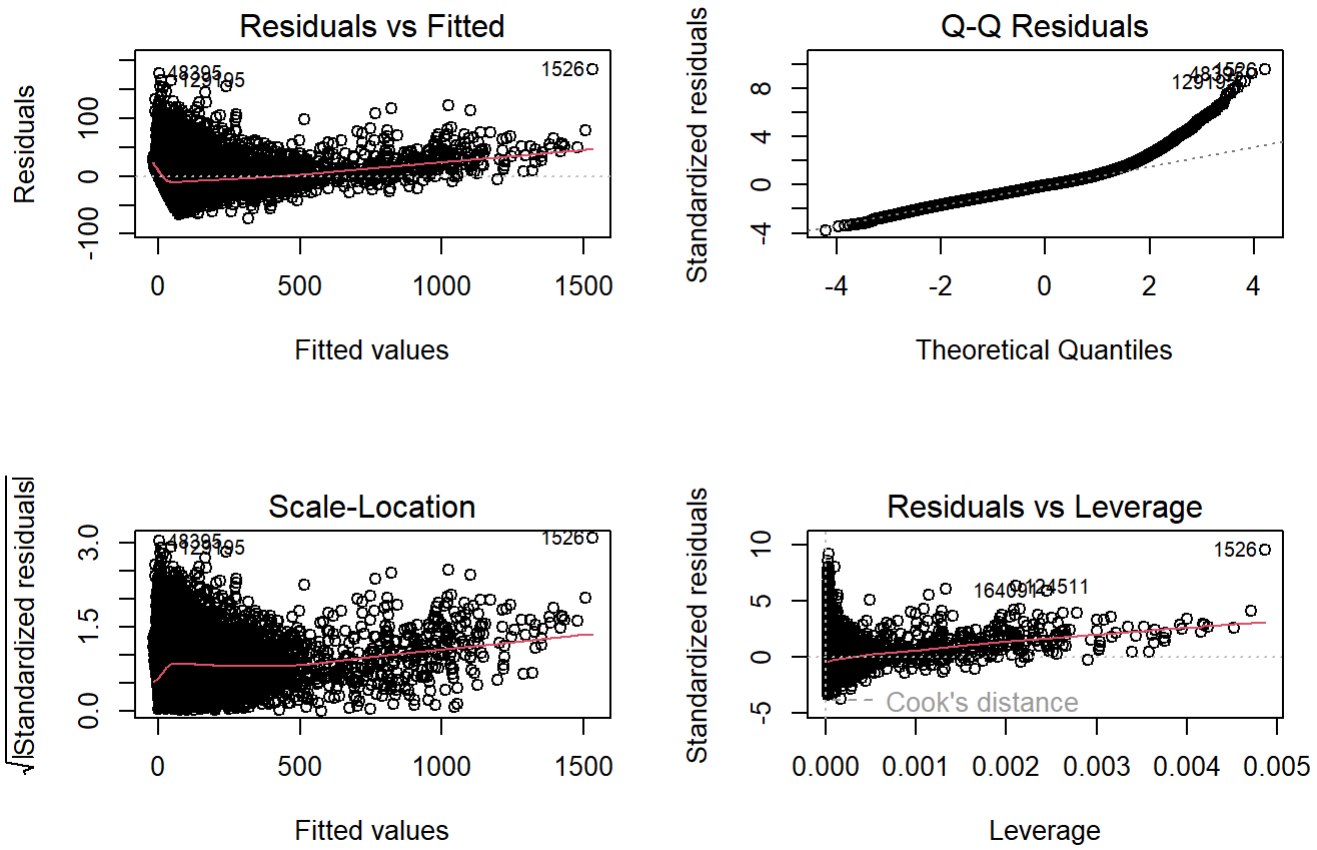
```
summary(linearmodel)
```

```
##
## Call:
## lm(formula = Arr_Delay ~ Dep_Delay, data = data_train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -73.392 -12.212  -1.367   8.976 184.473
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 6.7919020  0.1072867   63.31   <2e-16 ***
## Dep_Delay   0.9613715  0.0008784 1094.50   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 19.37 on 39920 degrees of freedom
## Multiple R-squared:  0.9678, Adjusted R-squared:  0.9677
## F-statistic: 1.198e+06 on 1 and 39920 DF,  p-value: < 2.2e-16
```

Our linear regression model, which examines the relationship between departure delay and arrival delay, has an R-squared value of 96.77%, that means we can explain nearly all the variance in arrival delay using departure delay. The coefficient for departure delay is 0.96, indicating that each minute of departure delay results in almost a minute of arrival delay. Both the intercept and the slope are statistically significant with p-values well below 0.001, confirming the robustness of our model. In addition a residual standard error indicates that, on average, the actual arrival delay differs from the predicted arrival delay by approximately 19.37 minutes.

Plots

```
par(mfrow = c(2, 2))
plot(linearmodel)
```




```
par(mfrow = c(1, 1))
```

These diagnostic graphs are essential tools to validate the assumptions of our regression model. The Residuals vs Fitted chart verifies homogeneity and linearity, ensuring that residues are evenly distributed across the entire mounted value range. The Q-Q chart evaluates the normality of residues, a critical assumption for the validity of many statistical tests. The 'Scale-Location' chart further examines homoscedasticity, while the 'Residuals vs Leverage' chart identifies influential data points that could disproportionately affect our model's predictions. In this case the Residuals vs Fitted and the Scale-Location plots show a tendency of the linear model to overestimate the biggest arriving delays, it could suggest that the residuals have a non-normal behaviour. This is confirmed by the Q-Q plot: the distribution of the residuals have a bigger right tail than the normal distribution. We may suppose that the relation between our variables is more complex than linearity. The graph 'Residuals vs Leverage' highlights the presence of an outlier: the point of observation 1526 has an high leverage and also an high residual, we therefore decided to eliminate this point from the graph. Together, these charts provide a complete check-up of our linear model.

```
outlier <- which.max(linearmodel$residuals)
data_train <- data_train[-outlier, ]
linearmodel = lm(Arr_Delay ~ Dep_Delay, data = data_train)
summary(linearmodel)$r.squared
```

```
## [1] 0.9676324
```

Even if the R2 value doesn't show a significant change, we decided to eliminate this point because it is an extreme point that, differently from other observations, doesn't represent the distribution of the data, and so it is considered an outlier.

Prediction

```
new_x <- data.frame(Dep_Delay = 10)
pred_y <- predict(linearmodel, newdata = new_x)
conf_int <- predict(linearmodel, newdata=new_x, interval ="confidence")
conf_int
```

```
##        fit      lwr      upr
## 1 16.42575 16.22244 16.62906
```

```
pred_int <- predict(linearmodel, newdata=new_x, interval ="prediction")
pred_int
```

```
##        fit       lwr      upr
## 1 16.42575 -21.48971 54.34121
```

When you set interval = “confidence” it gives you a range that is likely to contain the true mean of the response variable for a given set of predictor values; while when you set interval = “prediction” it gives you a range that is likely to contain an actual single future observation for the response variable given a set of predictor values. So it is understandable why the second range is way wider than the first one.

Polynomial regression

Considering the previous results we decided to try adding polynomial terms of different powers.

```
polynomial_model1 = lm(Arr_Delay ~ Dep_Delay + I(Dep_Delay^2), data = data_train)
summary(polynomial_model1)$r.squared
```

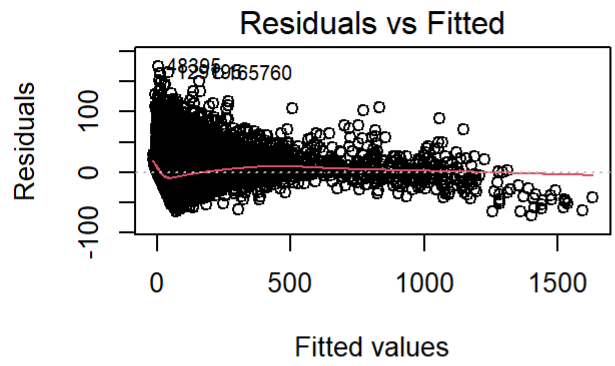
```
## [1] 0.9693564
```

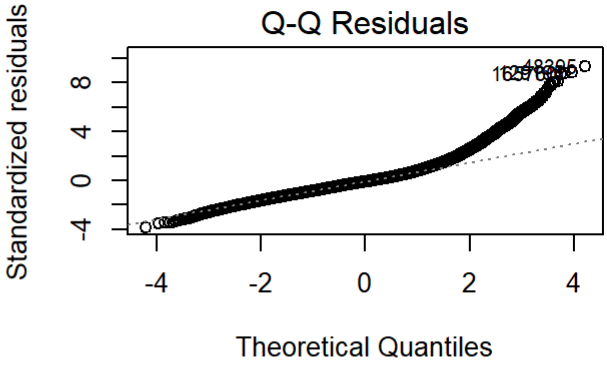
```
polynomial_model2 = lm(Arr_Delay ~ poly(Dep_Delay,3, raw=TRUE), data = data_train)
summary(polynomial_model2)$r.squared
```

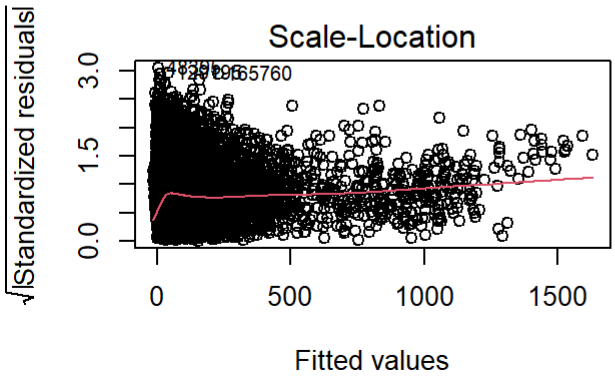
```
## [1] 0.9701764
```

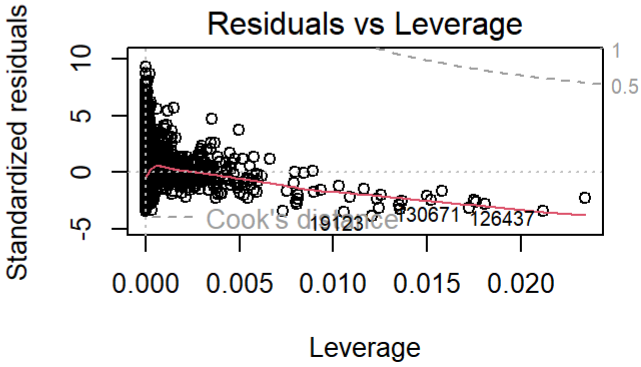
Our analysis includes two polynomial regression models to predict arrival delay based on departure delay. The first model includes linear and quadratic terms, while the second model adds a cubic term. All models fit the data exceptionally well, with R-squared values of 0.9694 and 0.9702, respectively, indicating they explain over 96% of the variance in arrival delay. The last model provides a slightly better fit with a lower residual standard error and marginally higher R-squared value. Observing that, as expected, increasing the polynomial’s degrees the model was performing better. We decided to stop after the third degree to avoid adding exaggerated complexity and reducing the interpretability.

Plots

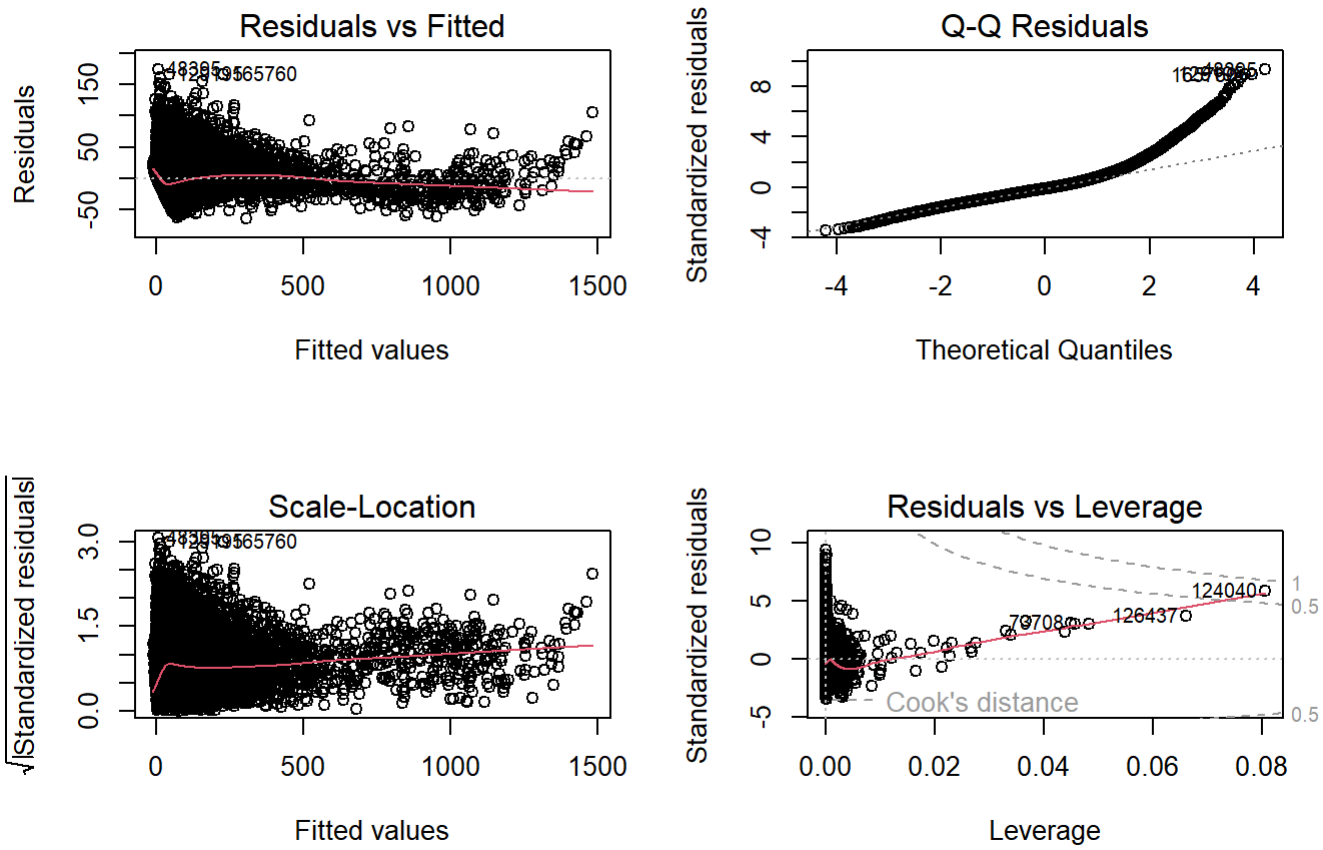








Using the polynomial model of degree 2 we can notice that the previous error in overstimating the biggest delays turned in understimating it. For the rest of it, we exhibit similar behaviour.



In the model with polynomial of degree 3 we can notice again that the biggest delays are overestimated by the model. It's meaningful see that some points have an high leverage.

Non-linear transformation

We did a log transformation of the variable "Arr_Delay"

```
logmodel <- lm(log(Arr_Delay) ~ Dep_Delay, data = data_train)
summary(logmodel)$r.squared
```

```
## [1] 0.40324
```

Here we explored a different transformation with a log-transformed linear regression model that reveals a significant positive relationship between departure delay and the log-transformed arrival delay, with an R-squared value of 40.32%, indicating a moderate level of explanatory power. We can conclude that this model is less explanatory than the model with the polynomial transformation, meaning that the type of non-linear relationship between the two variables is of the polynomial type.

Multiple linear regression

```
multiplemodel = lm(Arr_Delay ~ Dep_Delay + Flight_Duration + Aicraft_age + tavg + tmin + tmax
+ prcp + snow + wdir + wspd + pres + DepTime_categoric, data = data_train)
summary(multiplemodel)
```

```
##
## Call:
## lm(formula = Arr_Delay ~ Dep_Delay + Flight_Duration + Aicraft_age +
##     tavg + tmin + tmax + prcp + snow + wdir + wspd + pres + DepTime_categoric,
##     data = data_train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -84.372 -11.294  -1.207   8.787 168.720
##
## Coefficients:
##                       Estimate Std. Error  t value Pr(>|t|)
## (Intercept)         3.947e+01  1.906e+01    2.071  0.03833 *
## Dep_Delay           9.635e-01  8.723e-04 1104.533  < 2e-16 ***
## Flight_Duration     3.602e-02  1.044e-03   34.499  < 2e-16 ***
## Aicraft_age        -4.430e-02  1.347e-02   -3.288  0.00101 **
## tavg                2.680e-01  1.022e-01    2.622  0.00873 **
## tmin               -3.496e-01  5.682e-02   -6.153 7.70e-10 ***
## tmax               -1.139e-01  5.657e-02   -2.014  0.04400 *
## prcp                5.250e-02  1.011e-02    5.190 2.11e-07 ***
## snow                2.577e-02  3.962e-03    6.503 7.95e-11 ***
## wdir                9.295e-05  9.309e-04    0.100  0.92047
## wspd               -1.954e-02  1.819e-02   -1.074  0.28263
## pres               -3.596e-02  1.861e-02   -1.932  0.05332 .
## DepTime_categoric2  8.878e-01  2.308e-01    3.847  0.00012 ***
## DepTime_categoric3 -2.385e+00  2.526e-01   -9.439  < 2e-16 ***
## DepTime_categoric4  1.059e+00  6.713e-01    1.578  0.11468
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 18.94 on 39906 degrees of freedom
## Multiple R-squared:  0.969,  Adjusted R-squared:  0.969
## F-statistic: 8.909e+04 on 14 and 39906 DF,  p-value: < 2.2e-16
```

Our multiple linear regression model explores the relationship between various factors and arrival delay in flights. As expected, departure delay emerges as highly significant predictor. Flight duration, Minimum temperature, precipitation and snow are significant influently for the model. While Aircrafr age, Average temperature and Maximum temperature are slightly significant, and Wind direction, Wind speed and Air pressure are not significant at all. The model exhibits a high explanatory power with an adjusted R-squared of 0.969, indicating that around 96.90% of the variance in arrival delay is accounted for by the predictors. In addition a residual standard error indicates that, on average, the actual arrival delay differs from the predicted arrival delay by approximately 18.94 minutes.

Moving on to the categorical variable “DepTime_categoric” we interpret the three levels as follow:

- Flights departing during the afternoon have an average delay in arrival that is approximately 0.887 minutes higher compared to the morning, holding all other variables constant.
- Flights departing during the evening have an average delay in arrival that is approximately 2.38 minutes lower compared to the morning, holding all other variables constant.
- Flights departing during the night have an average delay in arrival that is approximately 1.059 minutes higher compared to the reference category, holding all other variables constant.

The observation that we made on it is that, as we expected, the moment in which the flights are more delayed compared to the morning is the night.

Backward stepwise procedure 1

We now use the backward procedure to eliminate the less significant variables from our model and to try to see what changes at the end.

```
mod1 <- update(multiplemodel, ~.-wdir)
mod2 <- update(mod1, ~.-wspd)
mod3 <- update(mod2, ~.-pres)
mod4 <- update(mod3, ~.-tmax)
summary(mod4)$r.squared
```

```
## [1] 0.9689894
```

As expected we can see that nothing changed.

Aic & Bic

Now we use in order R2, AIC and BIC to evaluate the other variables and check if they are useful to the model. First try: delete predictor “average temperature”

```
cat("mod4: R2", summary(mod4)$r.squared, "AIC", AIC(mod4), "BIC", BIC(mod4), "\n")
```

```
## mod4: R2 0.9689894 AIC 348126.1 BIC 348229.3
```

```
mod5 <- update(mod4, ~.-tavg)
cat("mod5: R2", summary(mod5)$r.squared, "AIC", AIC(mod5), "BIC", BIC(mod5), "\n")
```

```
## mod5: R2 0.968986 AIC 348128.5 BIC 348223
```

Since in model4 the R2 is higher, the AIC is lower and the BIC is higher with respect to model5 we can conclude that they are pretty, but focusing on the BIC we should exclude it.

Second try: delete predictor “Aircraft_age”

```
mod6 <- update(mod4, ~.-Aicraft_age)
cat("mod6: R2", summary(mod6)$r.squared, "AIC", AIC(mod6), "BIC", BIC(mod6), "\n")
```

```
## mod6: R2 0.9689808 AIC 348135.2 BIC 348229.8
```

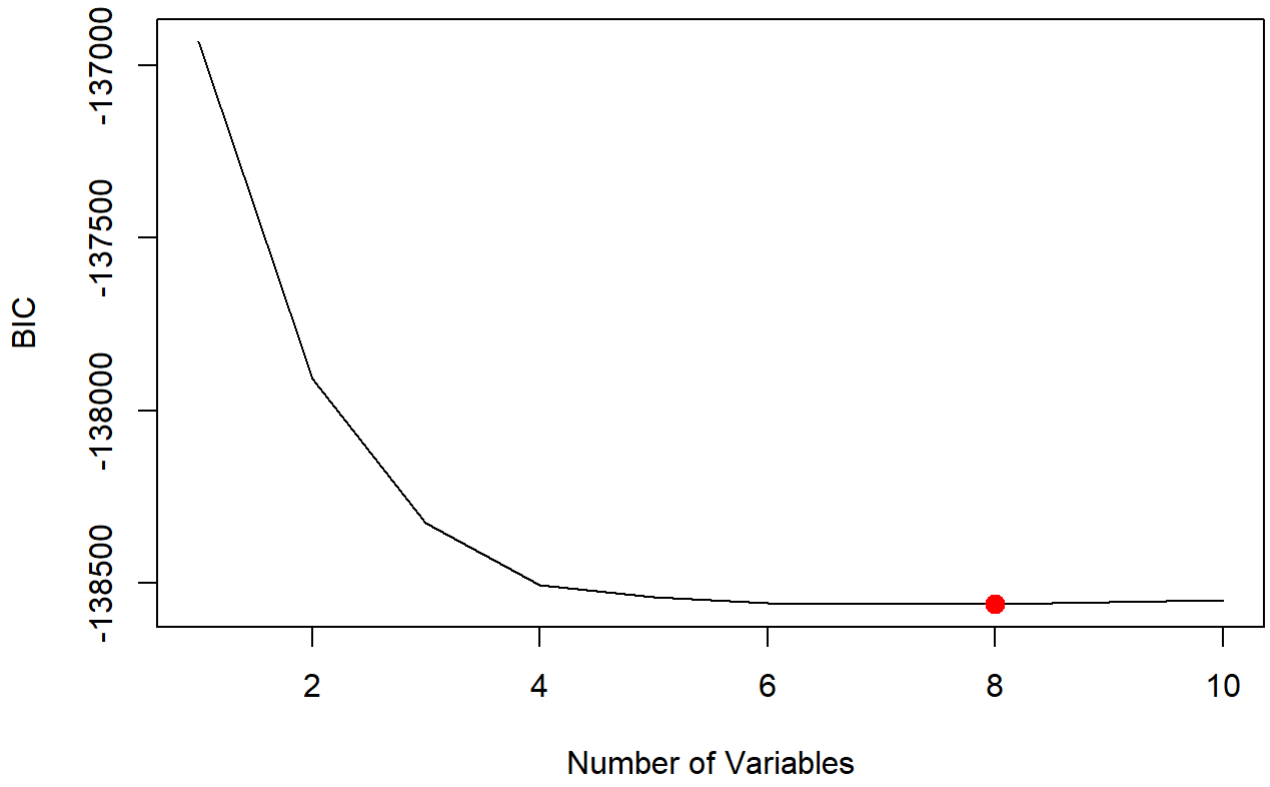
Since in model4 the R2 is higher and the AIC and BIC are lower we can conclude that this is the best model, so we will keep the regressor Aircraft age in our model even if the relationship between the two is negative.

Backward stepwise procedure 2

We then used a different approach that does by itself the backward procedure to check if what we manually did was right.

```
backward <- regsubsets(Arr_Delay ~ Dep_Delay + Flight_Duration + Aicraft_age + tavg + tmin +
tmax + prcp + snow + wdir + wspd + pres + DepTime_categoric, data = data_train, nvmax=10, met
hod="backward")

backward_summary <- summary(backward)
plot(backward_summary$bic,xlab="Number of Variables",ylab="BIC",type='l')
i <- which.min(backward_summary$bic)
points(i,backward_summary$bic[i],col="red",cex=2,pch=20)
```




```
best_model_vars <- backward_summary$which[i, ]
print(best_model_vars)
```

```
##        (Intercept)          Dep_Delay    Flight_Duration       Aicraft_age
##               TRUE               TRUE               TRUE              TRUE
##               tavg               tmin               tmax              prcp
##              FALSE               TRUE              FALSE              TRUE
##               snow               wdir               wspd              pres
##               TRUE              FALSE              FALSE             FALSE
## DepTime_categoric2 DepTime_categoric3 DepTime_categoric4
##               TRUE               TRUE              FALSE
```

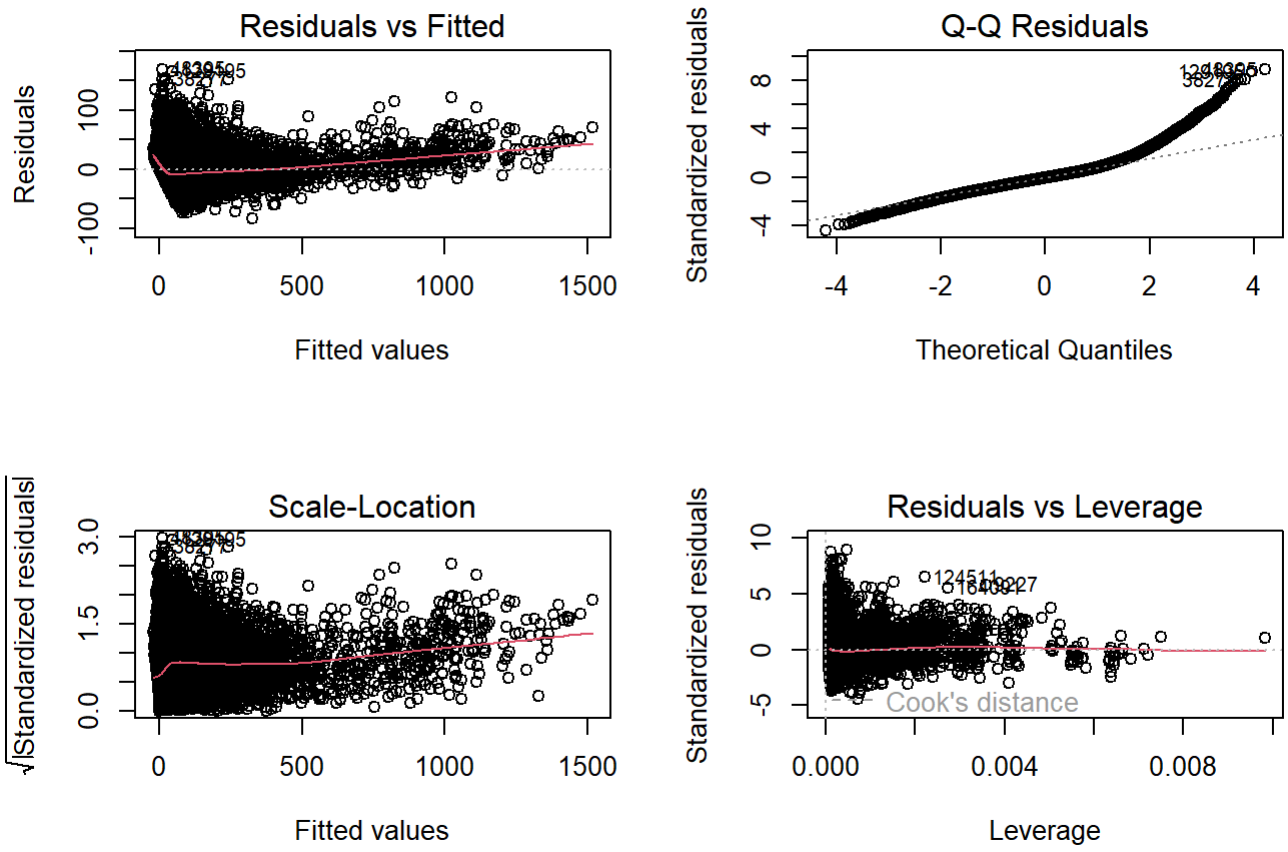
Looking at the plot, that set the best number of regressors to 8, and reading the “best_models_vars”, can notice how with this method the result and the selected variables, for the final model, are mostly the same. The table indicates that the variable Aircraft age will be kept in the model (as we studied), while the variable Average temperature will be excuded from the model. Regarding the factor variable, even if one of the three levels results to be not significant, it will be all kept in the model since it’s not possible to remove just one of them.

Final multiple model

```
finalmodel <- update(mod4, ~.-tavg)
summary(finalmodel)
```

```
##
## Call:
## lm(formula = Arr_Delay ~ Dep_Delay + Flight_Duration + Aicraft_age +
##     tmin + prcp + snow + DepTime_categoric, data = data_train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -84.062 -11.281  -1.224   8.800 168.627
##
## Coefficients:
##                       Estimate Std. Error  t value Pr(>|t|)
## (Intercept)          2.8950713  0.3593112    8.057 8.02e-16 ***
## Dep_Delay            0.9635788  0.0008718 1105.246  < 2e-16 ***
## Flight_Duration      0.0356062  0.0010226   34.818  < 2e-16 ***
## Aicraft_age         -0.0444781  0.0134670   -3.303 0.000958 ***
## tmin                -0.1936294  0.0111917  -17.301  < 2e-16 ***
## prcp                 0.0537006  0.0098080    5.475 4.40e-08 ***
## snow                 0.0258455  0.0039197    6.594 4.34e-11 ***
## DepTime_categoric2   0.9059724  0.2305591    3.929 8.53e-05 ***
## DepTime_categoric3  -2.3655203  0.2520207   -9.386  < 2e-16 ***
## DepTime_categoric4   1.0590617  0.6708058    1.579 0.114392
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 18.94 on 39911 degrees of freedom
## Multiple R-squared:  0.969,  Adjusted R-squared:  0.969
## F-statistic: 1.386e+05 on 9 and 39911 DF,  p-value: < 2.2e-16
```

Plots



In the case of multiple linear regression the Residuals vs Fitted and the Scale-Location plots suggest that the residuals have a non-normal behaviour that is confirmed by the Q-Q plot, as in the linear model, because the distribution of the residuals have a bigger right tail than the normal distribution. In conclusion, the graph 'Residuals vs Leverage' shows that all the points are within Cook's lines that indicate that there are no particularly influential observations that need further analysis.

Collinearity

```
vif(finalmodel)
```

```
##                      GVIF Df GVIF^(1/(2*Df))
## Dep_Delay        1.025292  1        1.012567
## Flight_Duration  1.049882  1        1.024638
## Aicraft_age      1.009058  1        1.004519
## tmin             1.181597  1        1.087013
## prcp             1.019826  1        1.009864
## snow             1.146478  1        1.070737
## DepTime_categoric 1.022690  3        1.003746
```

Since the collinearity represents the correlation between predictors, if it is high it avoids them from giving useful information to the model. A collinearity equal to 1 means that there is no collinearity, between 1 and 5 means that there is a moderate collinearity and higher that 5 means that there is high collinearity. In our case we have all values that are pretty close to 1 that means that there is low, almost zero, collinearity and we can keep all the predictors in the model.

REGRESSION CONCLUSIONS

Mse

```
y_pred <- predict(linearmodel, newdata = data_test)
mse_linear <- mean((y_pred - arr_delay_test)^2)
print(mse_linear)
```

```
## [1] 284.1507
```

```
y_pred <- predict(polynomial_model1, newdata = data_test)
mse_poly1 <- mean((y_pred - arr_delay_test)^2)
print(mse_poly1)
```

```
## [1] 262.4516
```

```
y_pred <- predict(polynomial_model2, newdata = data_test)
mse_poly2 <- mean((y_pred - arr_delay_test)^2)
print(mse_poly2)
```

```
## [1] 249.0253
```

```
y_pred <- predict(finalmodel, newdata = data_test)
mse_multiple <- mean((y_pred - arr_delay_test)^2)
print(mse_multiple)
```

```
## [1] 262.8106
```

To conclude we compared the MSEs of the regression models that we run, to have a final idea on what is the best one. Looking at the results, the considerations that we can make are multiples:

- the linear model, although its high R2 value, results to be the worst model in terms of MSE (284.1507), for its simplicity
- the quadratic model and the multiple model have similar MSE value (262.4516 and 262.8106) and, even if the residuals shown a polynomial pattern, suggesting that the quadratic model was going to be the most appropriate, we can notice how, adding more regressors, and so more complexity, we can obtain the same result.
- the cubic model, show a lower MSE value (respectively 249.0253), demonstrating that, keeping just one regressor, but exponentially increasing the complexity, we could obtain a better model. Its important to notice that some points have a high leverage and thus they have a significant impact on regression's coefficients. This means a small change in this points have a high impact on the coefficients.

In conclusion the decision about the model to use depends on the type of analysis and results that you are looking for: on one hand more complexity ensure better results, on the other hand less complexity ensure more interpretability.

MODEL DATA: CLASSIFICATION

Logistic Regression

In this section we restarted from the beginning, we merged the two datasets, we created the factor variable, we transformed the dependent variable “Ar_Delay” into a binary variable (1 = delayed, 0 = not delayed) and we deleted the variables that were useless for our analysis.

```
#datasetlogistic <- merge(flights_JFK, weather, by.x = c("FlightDate", "Dep_Airport"), by.y =
c("time", "airport_id"), all = TRUE)
datasetlogistic <- dataset_total
datasetlogistic$DepTime_categoric <- factor(datasetlogistic$DepTime_label, levels = c("Mornin
g", "Afternoon", "Evening", "Night"),labels = c(1, 2, 3, 4))
datasetlogistic$Arr_Delay_Binary <- ifelse(datasetlogistic$Arr_Delay > 0, 1, 0)
datasetlogistic <- na.omit(datasetlogistic)
useless_variables <- c("Airline", "Dep_CityName", "Arr_Airport", "DepTime_label", "Model", "D
ep_Airport")
datasetlogistic <- datasetlogistic[, -which(names(datasetlogistic) %in% useless_variables)]
```

```
cat("Number of delayed flights:", sum(datasetlogistic$Arr_Delay > 0), "Number of not delayed
flights:", sum(datasetlogistic$Arr_Delay <= 0))
```

```
## Number of delayed flights: 49597 Number of not delayed flights: 79945
```

Train and test split

For the train and test split phase we used, as before, the date, keeping the flights from January to September as train, and using the flights from October to December as test.

```
train <- (datasetlogistic$FlightDate < "2023-09-30")
dl_train <- datasetlogistic[train,]
dl_test <- datasetlogistic[!train,]
arr_delay_test <- datasetlogistic$Arr_Delay[!train]
arr_delay_test <- ifelse(arr_delay_test > 0, "delay", "no_delay")
cat("Dimension dl_train:", dim(dl_train), "Dimension dl_test:", dim(dl_test))
```

```
## Dimension dl_train: 97468 15 Dimension dl_test: 32074 15
```

Logistic model

We created our logistic model with all the numerical variables and the factor one, using the glm.

```
model_binary <- glm(Arr_Delay_Binary ~ Dep_Delay + Flight_Duration + Aicraft_age + tavg + tmi
n + tmax + prcp + snow + wdir + wspd + pres + DepTime_categoric, data = dl_train, family = bi
nomial)
summary(model_binary)
```

```
##
## Call:
## glm(formula = Arr_Delay_Binary ~ Dep_Delay + Flight_Duration +
##     Aicraft_age + tavg + tmin + tmax + prcp + snow + wdir + wspd +
##     pres + DepTime_categoric, family = binomial, data = dl_train)
##
## Coefficients:
##                      Estimate Std. Error z value Pr(>|z|)
## (Intercept)        -2.4500296  1.7280987  -1.418   0.1563
## Dep_Delay           0.1120685  0.0009600 116.733  < 2e-16 ***
## Flight_Duration     0.0031344  0.0000963  32.550  < 2e-16 ***
## Aicraft_age        -0.0050823  0.0011988  -4.239 2.24e-05 ***
## tavg                0.0078370  0.0092878   0.844   0.3988
## tmin               -0.0313453  0.0052384  -5.984 2.18e-09 ***
## tmax                0.0022702  0.0050859   0.446   0.6553
## prcp                0.0048327  0.0009696   4.984 6.22e-07 ***
## snow                0.0023759  0.0003546   6.699 2.09e-11 ***
## wdir               -0.0003480  0.0000840  -4.143 3.43e-05 ***
## wspd                0.0039614  0.0016693   2.373   0.0176 *
## pres                0.0010045  0.0016864   0.596   0.5514
## DepTime_categoric2  0.4431531  0.0200453  22.108  < 2e-16 ***
## DepTime_categoric3  0.2409394  0.0236278  10.197  < 2e-16 ***
## DepTime_categoric4  0.1069042  0.0512978   2.084   0.0372 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 131918  on 97467  degrees of freedom
## Residual deviance:  82105  on 97453  degrees of freedom
## AIC: 82135
##
## Number of Fisher Scoring iterations: 8
```

In the model each coefficient represents the expected change in the log odds, that is the natural logarithm of the ratio of probability of success to probability of failure, for a unit of change in the independent variable, keeping all other independent variables in the model constant. For example, incrementing of one unit the “Dep_Delay” independent variable, contributes positively to the likelihood of the observation being classified into class 1, that means that the more the flight is delayed more probable it is to classify it as delayed, while a one-unit increase in tmin contributes negatively. In addition the null deviance represents the deviation of the null model (i.e., a model without predictive variables) to the data while the residual deviance represents the residual deviation of the model with all the regressors. Since the residual deviation is lower than the null deviation it means that the model has better adaptability than the null model.

Reduced model

We then removed the not significant variable and created a reduced logistic model.

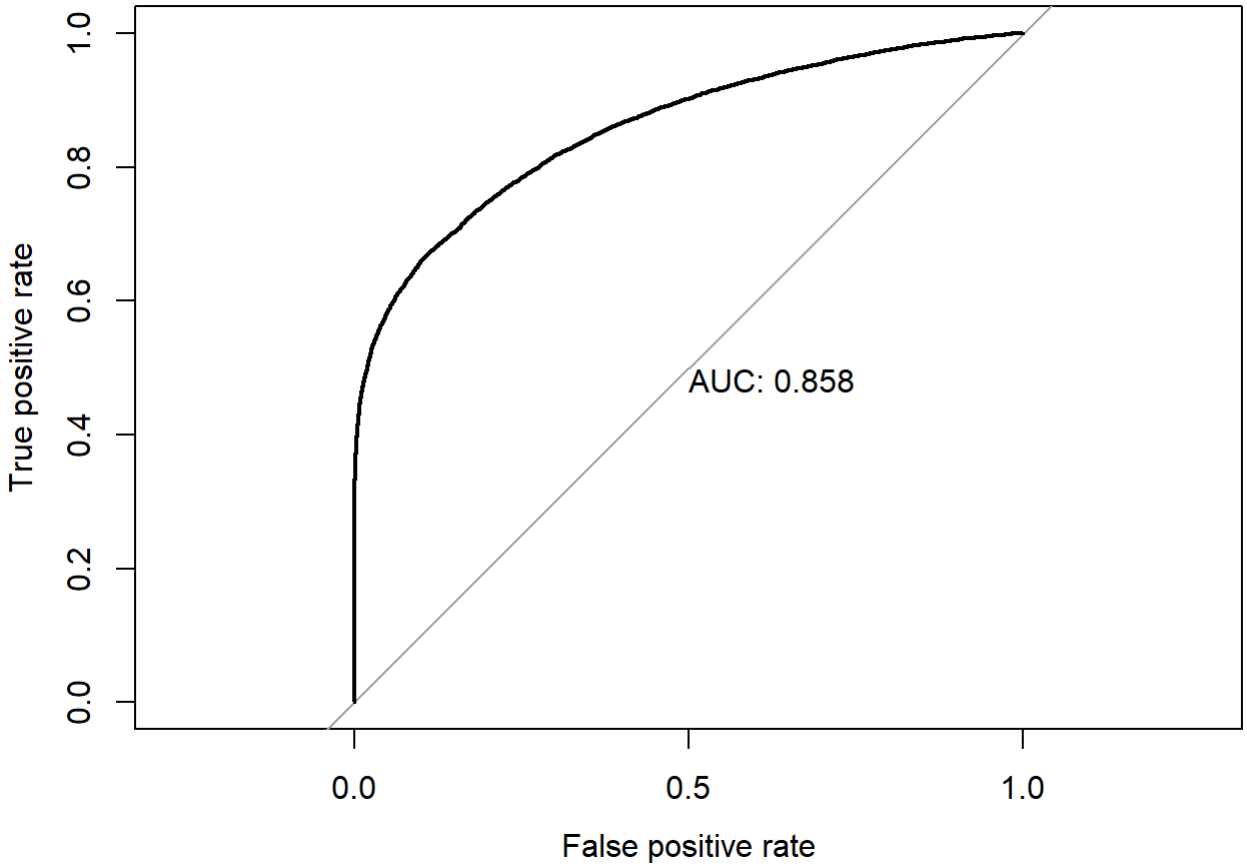
```
model_binary_reduced <- glm(Arr_Delay_Binary ~ Dep_Delay + Flight_Duration + Aicraft_age + tm
in + prcp + snow + wdir + DepTime_categoric, data = dl_train, family = binomial)
summary(model_binary_reduced)
```

```
##
## Call:
## glm(formula = Arr_Delay_Binary ~ Dep_Delay + Flight_Duration +
##     Aicraft_age + tmin + prcp + snow + wdir + DepTime_categoric,
##     family = binomial, data = dl_train)
##
## Coefficients:
##                      Estimate Std. Error z value Pr(>|z|)
## (Intercept)        -1.301e+00  3.406e-02 -38.189  < 2e-16 ***
## Dep_Delay           1.120e-01  9.594e-04 116.763  < 2e-16 ***
## Flight_Duration     3.103e-03  9.358e-05  33.157  < 2e-16 ***
## Aicraft_age        -5.083e-03  1.198e-03  -4.241 2.22e-05 ***
## tmin               -2.222e-02  1.068e-03 -20.815  < 2e-16 ***
## prcp                4.578e-03  9.330e-04   4.907 9.26e-07 ***
## snow                2.255e-03  3.510e-04   6.425 1.32e-10 ***
## wdir               -3.469e-04  8.267e-05  -4.196 2.72e-05 ***
## DepTime_categoric2  4.440e-01  2.004e-02  22.154  < 2e-16 ***
## DepTime_categoric3  2.414e-01  2.361e-02  10.225  < 2e-16 ***
## DepTime_categoric4  1.015e-01  5.124e-02   1.981   0.0476 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 131918  on 97467  degrees of freedom
## Residual deviance:  82115  on 97457  degrees of freedom
## AIC: 82137
##
## Number of Fisher Scoring iterations: 8
```

Looking at the results we can notice how the residual deviance has increased of 10 units, but the complexity of the model has drastically decreased. While talking about the factor variable “DepTime_Categoric”, the coefficients represent the changing in log-odds by moving from the morning to the other three levels, in order: afternoon, evening and nights. So for example the coefficient 4.440e-01 for “DepTime_categoric2” represent that the log-odds of obtaining the flight classified as class 1 increases of 0.444 units when the variable moves from morning to afternoon. For an easier comprehension we can covert this coefficients in odds ratio (OR) using the exponential. For example e^0.444 ≈ 1.559 that means that with respect to the reference class morning, the probability to obtain the flight classified as class 1 is 1.559 times higher if the variable is afternoon. And the same interpretation holds for the other two levels of the variable.

Plot ROC

```
probs <- predict(model_binary_reduced, dl_test, type = "response")
roc_out <- roc(arr_delay_test, probs, levels = c("no_delay", "delay"))
plot(roc_out, print.auc = TRUE, legacy.axes = TRUE, xlab = "False positive rate", ylab = "Tru
e positive rate")
```



Given that the diagonal line represents a random classifier that means that any point on this line suggests that the model is guessing randomly and that a ROC curve above the diagonal indicates that the model is performing better than random guessing, the further the curve is from the diagonal and towards the top left corner, the better the model. The AUC quantifies the overall ability of the model to discriminate between positive and negative classes and we get a score of 0.858 that means that our model performs well.

Linear discriminant analysis (LDA)

The LDA model calculates the posterior probabilities that a flight belongs to either of the two classes on these predictor variables and then classifies each flight into the class with the highest one. This model is based on the assumption of normal distribution of the predictor variables and common covariance matrix between classes.

```
lda_model <- lda(Arr_Delay_Binary ~ Dep_Delay + Flight_Duration + Aicraft_age + tmin + prcp +
snow + wdir + DepTime_categoric, data = dl_train)
lda_model
```

```
## Call:
## lda(Arr_Delay_Binary ~ Dep_Delay + Flight_Duration + Aicraft_age +
##     tmin + prcp + snow + wdir + DepTime_categoric, data = dl_train)
##
## Prior probabilities of groups:
##         0         1
## 0.5903681 0.4096319
##
## Group means:
##   Dep_Delay Flight_Duration Aicraft_age     tmin     prcp     snow     wdir
## 0 -3.162056        172.2426    13.47711 13.22037 2.487985 3.160144 187.1319
## 1 52.431674        198.7965    13.64494 13.97174 3.389591 4.030757 188.8948
##   DepTime_categoric2 DepTime_categoric3 DepTime_categoric4
## 0          0.3042821          0.1902610         0.04271662
## 1          0.4068527          0.2809447         0.02166508
##
## Coefficients of linear discriminants:
##                              LD1
## Dep_Delay           1.180273e-02
## Flight_Duration     3.187251e-03
## Aicraft_age         2.784885e-03
## tmin               -1.659446e-04
## prcp                9.774186e-03
## snow                3.618466e-03
## wdir               -4.856806e-05
## DepTime_categoric2  7.918853e-01
## DepTime_categoric3  7.629473e-01
## DepTime_categoric4 -7.925549e-02
```

Starting from the beginning we can see that 59.04% of the observations belong to class 1 (delayed), while 40.96% belong to class 0 (not delayed). Then moving to the group means section, and given that it represent the mean values of each predictor variable for each class, it is possible to notice how delayed flights tend to have higher values for Dep_Delay, Flight_Duration, tmin, prcp, snow compared to non-delayed flights. The coefficients, as in the logistic regression model, represent the weights assigned to each predictor variable in the linear combination used to form the discriminant function. That means that a positive coefficient indicates that an increase in the predictor variable value contributes to the likelihood of the observation being classified into class 1, while a negative coefficient suggests the opposite.

Quadratic Discriminant Analysis (QDA)

The QDA model classifies each class in the same way that LDA does; the difference is in the assumptions. This model is base on the idea that the predictor variables are normally distributed, but it allows each class to have its own covariance matrix.

```
qda_model <- qda(Arr_Delay_Binary ~ Dep_Delay + Flight_Duration + Aicraft_age + tmin + prcp +
snow + wdir + DepTime_categoric, data = dl_train)
```

Here the proportions of the prior probabilities and the group means are the same as LDA because we are doing it on the same data. What changes is how the model classify the point.

Accuracy

```
y_pred <- predict(model_binary_reduced, newdata = dl_test)
y_pred_class <- ifelse(y_pred > 0, "delay", "no_delay")
accuracy_logistic <- mean(y_pred_class == arr_delay_test)
accuracy_logistic
```

```
## [1] 0.8371578
```

```
lda_pred <- predict(lda_model,dl_test)
lda_class <- ifelse(lda_pred$class == 1, "delay", "no_delay")
conf_matrix <- table(lda_class, arr_delay_test)
accuracy_lda <- sum(diag(conf_matrix)) / sum(conf_matrix)
accuracy_lda
```

```
## [1] 0.7755191
```

```
qda_pred <- predict(qda_model,dl_test)
qda_class <- ifelse(qda_pred$class == 1, "delay", "no_delay")
conf_matrix <- table(qda_class, arr_delay_test)
accuracy_qda <- sum(diag(conf_matrix)) / sum(conf_matrix)
accuracy_qda
```

```
## [1] 0.8305169
```

To conclude we compared the accuracies of the classification models that we run, to have a final idea on what is the best one. Looking at the results, the considerations that we can make are multiples:

- Logistic Regression performs the best overall, suggesting that the data is well-suited to a model with linear decision boundaries without assuming normality.
- QDA's performance is also strong, indicating that the flexibility of quadratic boundaries is beneficial.
- LDA, while still effective, has the lowest accuracy, potentially due to its more restrictive assumptions about the data distribution.

In conclusion, Logistic Regression seems to be the most accurate model for this dataset, even if QDA is also good and allows for a more flexible decision boundary by considering quadratic terms, which can capture complex relationships between predictor variables and the response variable.