

**Flight Delay Predictor**

**Spark Application**

Big Data

M.Sc. Data Analytics

Authors:

Elena Gómez

Nacho

María Lara Trullenque

December 2022

CONTENT

1. Introduction 2

2. Exploratory Data Analysis 2

3. Application Development 4

4. Results 5

5. Running the Application 6

1. Introduction

This report presents the development of a Spark application consisting in a flight delay predictor based on machine learning techniques. The goal of the application is to develop a model that, given a set of data available at takeoff, can estimate the arrival delay time of a commercial flight. Publicly accessible data from commercial domestic flights in the USA is used to accomplish that. This data can be found in the Harvard Dataverse [1]. Concretely, the subset of data from year 2000 has been selected in order to work with a dataset with a reasonable size.

In the following sections, the whole application development process is explained, starting with a first general exploratory data analysis, which will enlighten the variable selection criteria. Hereafter, the application main code where the model is created is explained step by step. Finally, a brief summary of the performance results is presented, followed by the instructions for the user in order to run the application.

1. Exploratory Data Analysis

This part of the project was performed in an extra notebook in order to get a global perspective of the dataset and establish the preprocessing and feature selection steps that will be performed later in the application. This notebook can be found in X.

In the first place, to gain a general overview of the dataset, a first general analysis is performed, obtaining the number of rows and columns contained in the dataset and a statistical summary of the numerical variables, as well as the different datatypes. An explanation of the dataset is exposed in Table 1.

Table 1: Variable Description

|  |  |
| --- | --- |
| Variable | Description |
| Year | Year of the flight |
| Month | Month Of the flight 1-12 |
| DayOfMonth | Day of the month 1-31 |
| DayOfWeek | Day of the Week 1-7 |
| DepTime | Actual Departure Time (local, hhm m) |
| CRSDepTime | Scheduled departure time (local, hhmm) |
| ArrTime | Actual arrival time (local, hhmm) |
| CRSArrTime | Scheduled arrival time (local, hhmm) |
| UniqueCarrier | Unique carrier code |
| FlightNum | Flight number |
| TailNum | Plane tail number |
| ActualElapsedTime | Actual elapsed time (min) |
| CRSElapsedTime | Scheduled elapsed time (min) |
| AirTime | Air time (min) |
| ArrDelay | Arrival Delay (min), target variable |
| DepDelay | Departure delay (min) |
| Origin | Origin IATA airport code |
| Dest | Destination IATA airport code |
| Distance | Distance (miles) |
| TaxiIn | Taxi in time (min) |
| TaxiOut | Taxi out time (min) |
| Cancelled | Cancelled flight (1 = yes, 0 = no) |
| CancellationCode | Reason for cancellation (A = carrier, B = weather, C = NAS, D = security) |
| Diverted | Diverted (1 = yes, 0 = no) |
| CarrierDelay | Carrier delay (min) |
| WheatherDelay | Weather delay (min) |
| NASDelay | NAS delay (min) |
| SecurityDelay | Security delay (min) |
| LateAircraftDelay | Late aircraft delay (min) |

After analyzing the structure of the dataset and the meaning of the variables, a brief preprocessing process is carried out to check for missing values and rows corresponding to cancelled flights, which do not provide any useful information. As the data set is large, these rows containing missing values will be removed. Also, rows corresponding to cancelled flights will be eliminated. Moreover, variables that contain information that is unknown at the time the plane takes off are removed as they cannot be used in the prediction model, because it could not be implemented in a real situation. These “forbidden” variables are: ArrTime, ActualElapsedTime, AirTime, TaxiIn, Diverted, CarrierDelay, WeatherDelay, NASDelay, SecurityDelay and LateAircraftDelay. Also, we consider other non-informative variables, that will not be used in the model, which are UniqueCarrier, FlightNum, TailNum, Origin, and Dest. On the one hand, information about the aircraft itself is not informative for the delay prediction as it is from an organizational nature. On the other hand, the variables Origin and Dest are categorical and therefore, in order to include them in the linear regression model, they would be treated as “dummy” variables. As there are a very large amount of different locations corresponding to origin and destiny, including them as dummy variables would generate an unnecessary complexity in the model, taking into account that they are not likely to provide much useful information. Finally, Cancelled, and CancellationCode will also be removed, given the fact that cancelled flights are already being discarded.

Once the data was clean, a bivariate analysis based on the correlation matrix for the numerical variables was performed. The goal is to observe which variables are more correlated with the target variable ArrDelay. The results are shown in Figure 1. It can be seen that seasonal data such as the year, month, day of month/week, etc are not relevant variables, as they have very low correlation values regarding ArrDelay. The same happens with the distance variable. On the other hand, the most correlated variables to ArrDelay are DepDelay (0.9), TaxiOut (0.31), DepTime (0.17), CRSDepTime (0.13) and CRSArrTime (0.12). Note that DepDelay is the result of the difference between CRSDepTime and DepTime, so these two variables can also be removed to avoid linear relationships between the predictor variables.

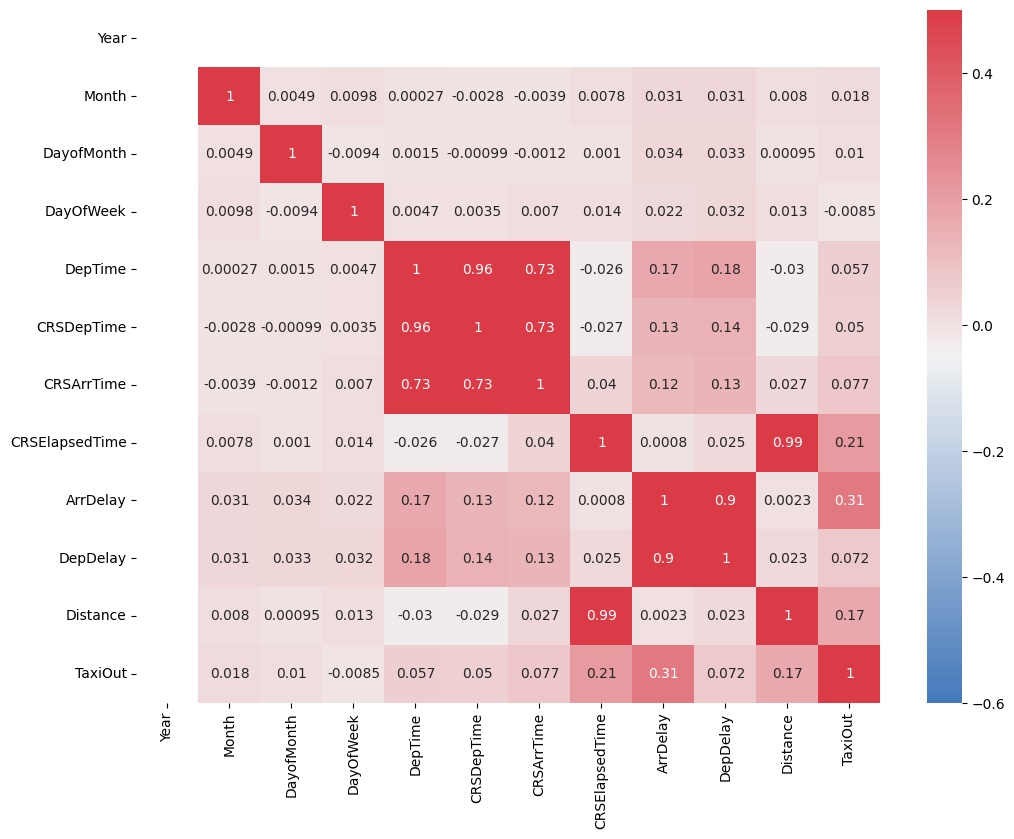


Figure 1: Correlation matrix for the numerical variables

From these results we can conclude that, for the linear regression model, the best variable selection might be DepDelay, TaxiOut and CRSArrTime. Analyzing the context, this seems like a good solution. The scatter plots of these variables against the target variable are going to be analyzed in order to check these relationships (Figure 2). Firstly, DepDelay is the departure delay which is of course very highly related to the arrival delay. Secondly, the TaxiOut time is defined as the time spent by a flight between its actual off-block time (AOBT) and actual take-off time (ATOT). We can easily conclude that if the aircraft lasts much time before actually taking off once it is considered as departed, the arrival delay will be increased. Finally, the CRSArrTime (scheduled arrival time) also seems to influence the arrival delay, although the correlation is lower and the scatter shows a more random behavior. This variable will be included in the model only after checking if it was improving the evaluation metrics.

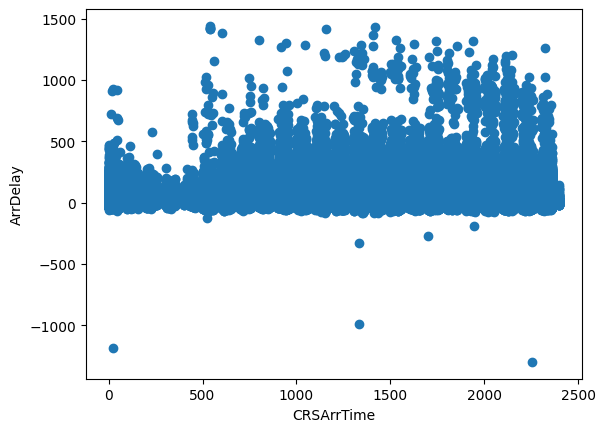
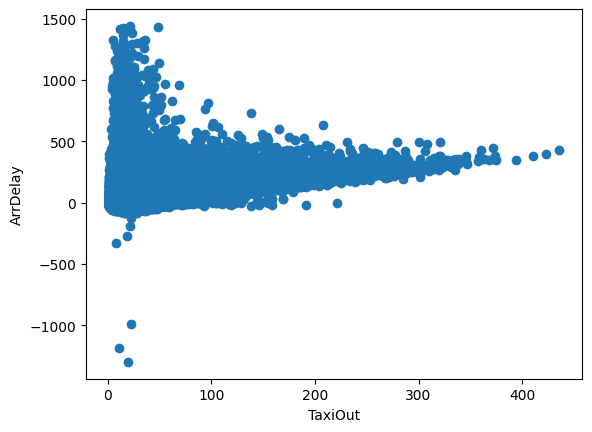
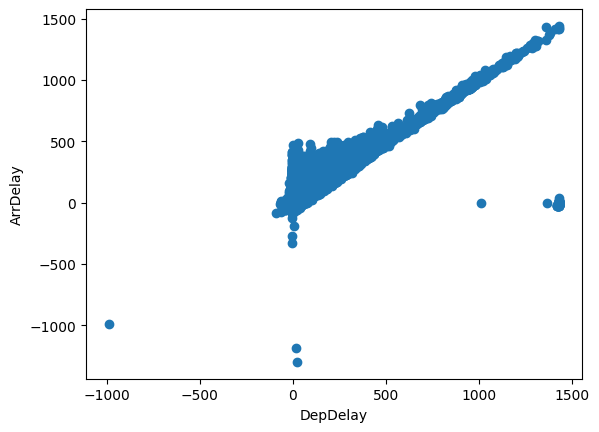


Figure 2: Scatter plots of the selected variables

Note that the correlation matrix also shows how these variables are not highly correlated to each other, which could produce multicollinearity problems in the linear regression model.

All in all, the results of these exploration are the preprocessing operations that will be performed in the application, from handling missing values to eliminating forbidden and non-useful columns and feature selection.

1. Application Development
   1. Loading and Processing the Data

When the application is run, first of all the data is loaded. This is done… Escribir cuando esté la interfaz.

Once the data is loaded, the preprocessing phase is carried out. Here, first of all, the rows corresponding to cancelled flights are deleted. Also, the forbidden variables and other non-useful columns are removed from the dataset. Moreover, to avoid linear dependencies in the useful columns, the columns DepTime and CRSDeptime are eliminated, given the fact that the feature DepDelay the difference of these two. Once the useful columns are selected, the numerical columns are transformed to double datatype, in order to avoid problems when developing the model. Finally, the missing values are handled. On the one hand, the rows with missing values in the target variable are eliminated, given the fact that they are not going to be useful to train or test the model. Secondly, the missing values in the feature columns are filled out with 0. This strategy was chosen based on the fact in general, there are no missing values in the columns, but sometimes in other datasets from other years, there are columns where all the values are missing. Therefore, these columns will be filled out with a constant value of 0 to avoid problems in the model without altering the information provided. When the preprocessing stage is done, the schema of the clean dataset is printed.

* 1. Model Generation

In this step, the machine learning model is generated. To this aim, we chose a linear regression model, which is appropriate to predict the continuous target variable (ArrDelay). In order to create the model, the application prepares the predictor variables and the dependent variable using *VectorAssembler*. The chosen predictor variables given the results of the exploratory analysis are DepDelay, TaxiOut and CRSArrTime, because of their high correlation with the target variable. Hay unas filas ahi (55-58) que no tengo muy claras. Also, the dataset is splitted into training (70%) and test set (30%) with a random seed using the *randomSplit* method. After splitting the dataset, the model can be initialized. This is done through the imported *LinearRegression* class. Here, the predictor and target variables are specified together with other parameters for the model generation. On the one hand, maxIter refers to the maximum number of iterations done when building the model. This is defined as 100. The parameter regParam has been set at 0.2 after performing a little exploration around parameters between 0.1 and 0.5, where 0.2 was providing the best results. Lastly, the parameter elasticNetParam is set at 0.8 after another exploration around this point, where the results were not changing. Ver si encontramos algo de explicacion sobre esos parámetros porque en la documentación no encuentro nada. <https://spark.apache.org/docs/latest/api/python/reference/api/pyspark.ml.regression.LinearRegression.html>

Once the building parameters are specified and the model is initialized, it is fitted to the data with the *fit* function, which takes the train data set as input.

* 1. Model Validation

For the model validation, we test it with the *transform* function taking the test dataset as an input (esto creo que está bien explicado así pero no estoy muy segura). Here, some of the obtained predictions for the unseen data are printed, so that the user can see and compare some results. Lastly, for the model evaluation we use the *summary* function, from which we later obtain the selected model evaluation metrics: the root mean squared error (RMSE) and the R2 measure. The former is an absolute measure of the goodness for the fit, which is calculated as follows:

On the other hand, the R2 measure indicates the amount of variability in the dependent variable explained by the model. More concretely, it is calculated as:

These two evaluation metrics are printed for the user to evaluate them. Ver si esto lo metemos en la interfaz

1. Results

The obtained model with three predictor variables (DepDelay, TaxiOut and CRSArrTime) is able to explain an 86,9% of the variability in the target variable (ArrDelay) based on the R2 measure. Moreover, the RMSE shows a value of 13.04 which is relatively low. Therefore, it can be stated that this model is an appropriate and simple model for predicting the arrival delay of aircrafts.

1. Running the Application

* Instructions

1. References

[1] 2008, "Data Expo 2009: Airline on time data", https://doi.org/10.7910/DVN/HG7NV7, Harvard Dataverse, V1