Flight Delay Prediction

Machine Learning Engineer Nanodegree - Capstone Proposal

Diogo Dutra November 11th, 2018

Domain Background - Flight delays

Aerial commute is increasingly important as the globalization advances and the world population grows. However, the air traffic is also becoming a issue, especially for the most used regional hubs. While transportation infrastructure is mainly a role for the governments, predicting the flight delays may be accessible for private initiative and it will benefit those passengers running tight on schedule by allowing them to reorganize their tasks on advance.

The most common causes of flight delays are varied [reference]. While some are not related to accessble data, some other might be within reach. The inaccessible data will remain as noise caused from security, maintenance and distater issues. The accessible data are weather and congestion that hopefully will shed some light to predict some of the flight delays.

The inspiration for such topic is clear for the author because of a combination of being a frequent airplane traveller and an experienced aeronautical engineer.

Problem Statement - How much will be the flight delay?

Basically, the predictive model shall be able to answer ther following question:

> ***Given the departure information, how many minutes will be the flight delay?***

The input data are all the available data before the take-off, including scheduled time, date, airliner, flight number, air temperature, air pressure, visibility and so on.

Datasets and Inputs - ANAC and BDM

The considered data of flight departures and arrivals are the world most busy regional aerial commute in order to maximize the ammount of available data. The Brazilian one-way commute from São Paulo (Guarulhos airport) to Rio de Janeiro (Sandos Dumont airport) is the most numerous by quantity of departures per day.

The 2018 historical dataset from January to September will be downloaded from the ANAC website. The weather data will be the METAR type and specific to the departure airport extracted directly from the INPE's BDM website.

Both departure and weather data will be merged into one single table considering the nearest date and time. Such table will be the sample input to the pipeline.

Solution Statement - Supervised learning regression

The predictive algorithm shall be a supervised learning regression in order to estimate the flight delay in minutes.

Different options of regression algorithms will be considered by applicability and compared by accuracy. The chosen one will be further polished for even better accuracy.

Benchmark Model - Better than naive

The obtained predictive model should ideally be more accurate than the "naive guess", where "naive guess" means estimating null delay for every flight. For comparison, the same evaluation metrics will be considered for both the predictive model and the "naive guess".

Evaluation Metrics - Sum of errors

The considered metrics are: - Sum of MSE (mean squared error) - Sum of absolute errors

Additionally, there shall be a linear plot of percent of successfull flight delay estimatives (y axis) by threshold of error tolerance (x axis) comparing both the predictive model and the "naive guess". This plot will be a support for extra insights.

Project Design

The pipeline will be divided in 4 different files.

1st File - Download

The departure data will be downloaded directly from the source as CSV files for replicability. The source are CSV files separated by months and are directly accessible by url. For this reason, the algorithm will dowload all of them separatedly and then append.

The weather files are also separated by months. However, they are not directly accessible by url as the website requires a login. For this reason, each of the files will be manually downloaded as XLS and then saved as CSV. All the files will be available at the repository for public offline access. Then, the weather tables will be appended similarly as the departure tables.

2nd File - Preparing

The downloaded data will be loaded into Panda's dataframe tables.

The departure table will be filtered to discard all routes except the one's regarding the airliners one-way flights from Guarulhos (SBGR) to Santos Dumont (SBRJ) airports.

The departure and weather tables will be merged by the nearest common date and time. The merge will be from left to right, meaning that useless weather data will be discarded.

Then, the table will have its columns modified for adaptation to the scikit-learn type. For instance, there will be conversion from string to datetime type and the **"Flight Delay"** label in minutes will be calculated from the difference between the scheduled and the real arrivals.

The final input table will be saved as local file for persistent access. This file will also be available at the repository for public offline access.

3rd File - Exploration

The prepared input table will then be used for a series of plots to better understand how is the data variation over its dimensions. We will focus on how each feature is correlated with the label "Flight Delay".

The series of plots will be of two kinds: scatter with transparency and violin distribution. These plots may give us enough insights to choose some candidates for types of regression algorithms and which features should be considered for the last phase of this pipeline.

4th File - Predicting

The last file will be used to create the predictive model.

Firstly we will filter down the feature columns. Secondly, we will hot-encode the categorical features. Thridly, we will split the samples between train and test samples.

Then, the selected candidates from the previous phase will each be instantiated with standard parameters, trained with the train sample and compared to each other by the evaluation metrics (sum of MSE and sum of absolute errors) over the test sample. The lowest evaluation metric will define the best algorithm for further tune. In the case of even results, then the linear plot comparing the percent of predict success (y axis) per threshold of error tolerance (x axis) will also be considered to choose the final algorithm.

Once the final algorithm is chosen, it will be fine tuned for better prediction accuracy. In order to achieve this, a grid of different values of parameters will be rerun to search and find the combination that gives the best accuracy considering again the evaluation metrics.

Finally, the final evaluation metrics over the test samples will be compared with the "naive guess" as benchmark for the predictive model.