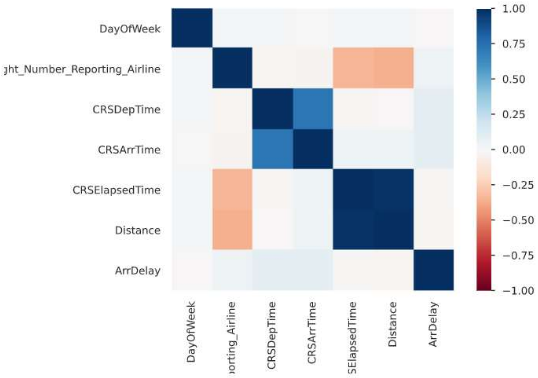
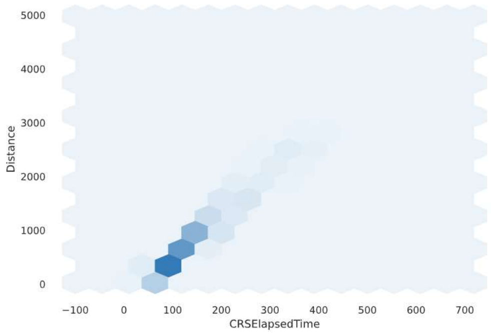
IML Hackathon - Your Flight Has Been Delayed

Team members: Chana Goldstein, Miriam Goldstein, Liron Girshoni, Yuval Dellus

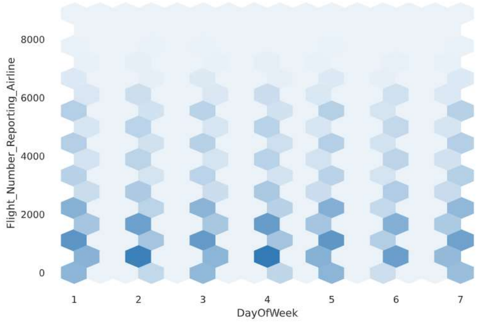
Work process:

1. Data portioning
   As a first step, we divided the data into a test set (25%) and train set (75%) which we then divided again into a train (75%) and validation set (25%). Since the data is divided evenly between delayed and not delayed flights, we can assume that the random split will also include the same distribution of those two classes.
2. Exploratory data analysis (EDA) and data visualization
   After manually scrolling through the data and getting a sense of what the dataset contains, we ran the pandas profiling report to understand some of the basic features. First, we noticed that the dataset did not have any missing values, other than the "Delay Factor" variable, which was missing for flight that arrived on time or early. We saw that there was a high correlation between the flight number reporting airline and the elapsed time and distance, as seen in the charts below.





   The correlation between distance and time of flight makes a lot of sense, since the further the distance we must travel, the longer the flight time will be. Interestingly, it seems like the days of the week do not play an important role, since all the features seem to be evenly distributed between the different days of the week. The chart to the right demonstrates just one feature, but all the others also look very similar.



3. Preprocessing and feature creation \ selection
   After getting to love our data, and splitting it into a train, validation and test set, we decided on the following steps for preprocessing the data.
   - Dividing the dates of the flights into months and years.
   - Converting the time of the arrival and departure to four categories (morning, afternoon, evening, night).
   - Adding additional information about holidays. We did this since we assume that around major holidays, for example Christmas or Easter, more people are traveling.
   - Creating dummy variables for the Reporting_Airline , Origin, Dest , DaysOfWeek variables.

When determining what variable to use and which ones do not use, we kept in mind the bias-variance tradeoff. Adding too many variables increases the complexity of the model, and therefore increases the variance. However, a model that is too simple can cause high bias. An example of this consideration is that instead of creating 24 dummy variable to indicate the time of arrival and departure of the flight, we created only 4 variables - morning, afternoon, evening, night.

Although our final model includes approx. 300 features, compared to about half a million samples, we are still in the case where $d\ (features) \ll m\ (samples)$.

Weather preprocessing – In this model we preprocess all the weather information, in which we decided to drop irrelevant columns as general and not specific information and the subjective data as “feel like”. In addition, we filled and replaced all the miss data by taking the average of 10% of the samples (in case of very big data set) and replaced the missing data with that average. Next we iterated on each unique airport to find all the flights leaving that airport on a specific date, which we can use to help predict future flight. We grouped the information by airport since all flights leaving the airport have the same weather information. Doing this in one bulk speeds up the whole process.

4. Model selection

   In the next stage we decided to use linear regression for the base algorithm for predicting how late or early a flight will be. For the classification stage we experimented with different multiclass classifiers including binary relevance, label power set, and classifier chains. We also experimented with using the wrapper classifier OneVsRestClassifier in which we ran classifiers that we learned about in the course (SVM, Decision Tree, Random Forest, etc.). The OneVsRestClassifier consists of fitting one classifier per class. For each classifier, the class is fitted against all the other classes.

5. Evaluating the model
   Since this hackathon is the first one that our team has actually participated in, we did not anticipate how long each of the stages would take us. Therefore, by 10:45 on Friday morning our models still haven’t finished the training stage, so we are unable to evaluate the performance. Although our model might not pass your automatic testing, we would be very grateful if when grading you could consider the difficulties we were experiencing and appreciate that although we stayed up until the early hours of the morning, and then woke up again in the early morning to continue working, we did our best for newly ML candidates.
   We appreciate your consideration!