Flight arrival delay prediction

Advanced Data Mining

Klara Langerholc

AGH

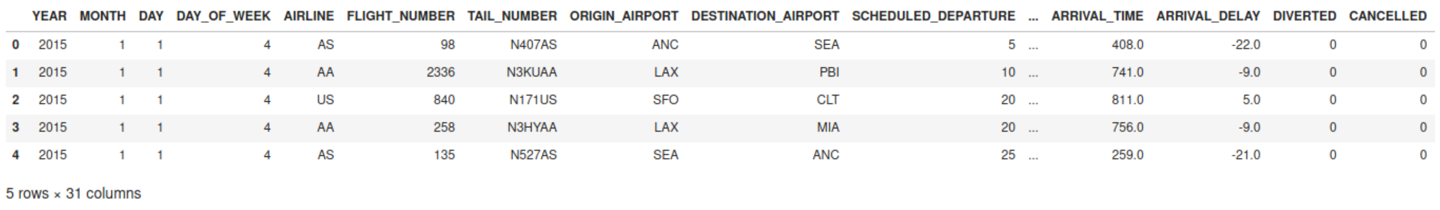
Krakow, Poland

21.1.2022

Description of the problem

In this project I try to create a model for flight arrival delay prediction. The dataset used is available at https://www.kaggle.com/usdot/flight-delays?select=flights.csv . I use only the flights.csv dataset, available on this link. The dataset contains around 6 million flight entries with 31 features.

For the purpose of this task, data preprocessing, feature selection and extraction, model fitting, optimization and validation are performed. In the end, two models are built and evaluated, one with data, which contains information about the departure, including the departure delay, and one with data with no flight departure information whatsoever.



	YEAR	MONTH	DAY	DAY_OF_WEEK	AIRLINE	FLIGHT_NUMBER	TAIL_NUMBER	ORIGIN_AIRPORT	DESTINATION_AIRPORT	SCHEDULED_DEPARTURE	...	ARRIVAL_TIME	ARRIVAL_DELAY	DIVERTED	CANCELLED
0	2015	1	1	4	AS	98	N407AS	ANC	SEA	5	...	408.0	-22.0	0	0
1	2015	1	1	4	AA	2336	N3KUAA	LAX	PBI	10	...	741.0	-9.0	0	0
2	2015	1	1	4	US	840	N171US	SFO	CLT	20	...	811.0	5.0	0	0
3	2015	1	1	4	AA	258	N3HYAA	LAX	MIA	20	...	756.0	-9.0	0	0
4	2015	1	1	4	AS	135	N527AS	SEA	ANC	25	...	259.0	-21.0	0	0

5 rows × 31 columns

Method used

The described problem is a regression problem, therefore multiple regression algorithms were tested and the best-performing ones were chosen. In this case the best results came from linear regression and gradient boosting regressor, both using the squared error as the loss function. Their performances changed when I added a feature selection method and were also different based on what data I had in the dataset. Both algorithms come from the sklearn library.

Linear regression attempts to fit a linear equation with multiple independent variables and one dependent variable to the data and therefore model the relationships between variables. Meanwhile, gradient boosting regressor uses decision trees as weaker learners. They output real values for splits, allowing subsequent model outputs to be added together and correct the predictions. Gradient descent is used to minimize the loss when adding trees. Trees are constructed in a greedy manner, but have limitations (for example, maximum number of layers), so that they remain weak.

The process of building the prediction model

The whole flight delay prediction model building process incorporates multiple steps: data preprocessing, feature selection and extraction, model fitting, optimization and validation.

Data preprocessing

First, the given dataset had to be analyzed and processed.

I first familiarized myself with the contents of the dataset and checked for the muber of missing and unique values. There were some missing values, but after removing the entries for flights that were diverted or canceled (those are irrelevant for our task), all of them were removed, therefore I did not have to remove them or replace them with any other values.

Next, I removed the columns of the dataset, that contained information about the flight arrival or also for flight departure. This reduced the number of features from 31 to 17 in the first case (where the flight departure information is available) and to 13 for the second case (flight departure information is not available), including the target arrival delay feature.

Most of the data was already in numerical form, even the dates and the times, so it was not necessary to further manipulate that. I did, however, encode the text data (for example the origin airport was written with a text code).

I also tried to repair the skewness of the data and scaled it.

In the end I ended up with a dataset containing only relevant numerical, scaled data, without any missing values. I split this dataset into the training and testing sets.

Regression model fitting

For the purpuse of this task, I have tried several regression models: decision tree regressor, support vector regression, K-neighbors regressor, linear regression, random forest regressor, Ada boost regressor and gradient boosting regressor.

Unfortunately, support vector regressor and random forest regressor were too time-consuming for my computer, so I stopped them before getting any results. For the other algorithms, I calculated the R2 scores, MAE, MSE and RMSE and concluded, that the linear regression and gradient boosting regressor performed the best, so I continued to work with these two.

Feature selection

I also tried several feature selection methods. I have tried a simple SelectKBest() function, recursive feature elimination (FRE) and the SelectFromModel() function, both of which are based on importance weights. I have also tried some other approaches, but they turned out to be too computationally complex for the computer I was working on.

I looked at the resulting selected features of each of these methods and transformed the training data set to contain only these features. I then tested the obtained datasets on the selected regression models and determined which one performs the best in combination with a certain regression model (I have tried them all on linear regression and gradient boosting regressor).

In the end I decided to use the SelectFromModel() method with a default median threshold, which extracts the most relevant features of a (possibly pretrained) model.

Optimization

Once I selected the regression and feature selection methods, I tried to optimize the hyperparameters of the gradient boosting regressor. For this purpose I used the grid search method (GlidSearchCV), where I specified which parameters I wanted to optimize and which values the algorithm should try. Optimizing many parameters at once is too computationally demanding, so I only focused on the learning rate and the number of estimators parameters. I noticed the tendency that with the number of estimators, the accuracy increases, but also the computational time increases, so I had to find a good trade-off also with the learning rate.

Validation

After every model fitting, I calculated and displayed R2 scores (for the test set and for the train set), mean absolute error, mean squared error and root mean squared error. Based on these results I was deciding which models were the better-performing ones. On linear regression and gradient boosting resgressor models I also used 5-fold cross validation, to recheck for any possible overfitting on the training data.

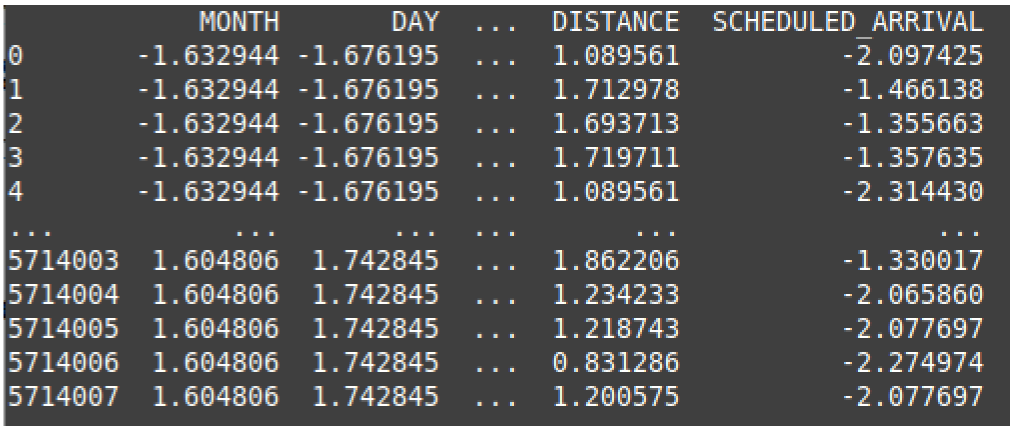
Results

Since I tested many combinations of algorithms, I have a lot of different results, some better, some worse and some very similar to each other. Here I will only discuss the most relevant ones.

First dataset

First I will present the results I obtained by processing the dataset, which includes information about the flight departure.

This is a sample of the final preprocessed training dataset:



```
            MONTH       DAY  ...  DISTANCE  SCHEDULED_ARRIVAL
0       -1.632944 -1.676195  ...  1.089561          -2.097425
1       -1.632944 -1.676195  ...  1.712978          -1.466138
2       -1.632944 -1.676195  ...  1.693713          -1.355663
3       -1.632944 -1.676195  ...  1.719711          -1.357635
4       -1.632944 -1.676195  ...  1.089561          -2.314430
...           ...       ...  ...       ...                ...
5714003  1.604806  1.742845  ...  1.862206          -1.330017
5714004  1.604806  1.742845  ...  1.234233          -2.065860
5714005  1.604806  1.742845  ...  1.218743          -2.077697
5714006  1.604806  1.742845  ...  0.831286          -2.274974
5714007  1.604806  1.742845  ...  1.200575          -2.077697
```

Following are the results of the linear regression and gradient boosting regressor before any feature extraction, so with all the 16 training features:

Linear regression	Gradient boosting regressor
R2_test: 0.9287425382775923	R2_test: 0.9386560851750879
R2_train: 0.9291711132512475	R2_train: 0.939094989323751
MAE: 7.546685397436756	MAE: 6.84072799072297
MSE: 109.3833103096617	MSE: 94.16558362747271
RMSE: 10.458647632923757	RMSE: 9.703895281147293
CV5 mean: 0.9277605533976596	CV5 mean: 0.9359666545199527

As we can see, they both perform well, gradient boosting regressor slightly better.

After performing feature selection, 3 features were selected for linear regression (departure_delay, taxi_out and scheduled_time) and only 1 feature for gradient boosting regressor (departure_delay). These are the errors after the models were trained only with these features:

Linear regression	Gradient boosting regressor
R2_test: 0.9275293908046929	R2_test: 0.8919183215766968
R2_train: 0.9279468486904299	R2_train: 0.8928202772977623
MAE: 7.6468045457569485	MAE: 9.061169150085453
MSE: 111.24554456937314	MSE: 165.91008834724715

RMSE: 12.880609005293467	RMSE: 10.547300345082297
CV5 mean: 0.929094989323751	CV5 mean: 0.89072799072297

It can be observed that the results are now slightly worse, but not by much in the linear regression model. The gradient boosting model, now trained only on one most valuable variable, departure delay, still performs well, but in this case worse than linear regression, which was trained on 3 selected variables.

Not much can be optimized in the linear regression model, so the results of the linear regression above were chosen as the final results for this dataset.

Second dataset

Following results were obtained by processing a dataset with no information about the flight departure whatsoever, therefore only containing information like scheduled departure and arrival times, departure and arrival airport and plane model.

Following are the results of the linear regression and gradient boosting regressor before any feature extraction, so with all the 12 training features:

Linear regression	Gradient boosting regressor
R2_test: 0.014493705495590925	R2_test: 0.05162558656076455
R2_train: 0.01457743033195169	R2_train: 0.05245030795806027
MAE: 20.96434023375088	MAE: 20.377289315350357
MSE: 1512.795126548866	MSE: 1455.796070299068
RMSE: 38.89466707080633	RMSE: 38.15489575793738
CV5 mean: 0.004892268662532473	CV5 mean: 0.0482813426402298

As we can see, the model without and departure data is very bad. Gradient boosting regressor performs better than linear regression.

After performing feature selection, 5 features were selected for linear regression (month, scheduled_departure, scheduled_time, distance, scheduled_arrival) and 4 features for gradient boosting regressor (month, day, airline, scheduled_departure). These are the errors after the models were trained only with these features:

Linear regression	Gradient boosting regressor
R2_test: 0.01317600270861674	R2_test: 0.04766787990904775
R2_train: 0.013163358211478404	R2_train: 0.04828147713902298
MAE: 20.987960173767615	MAE: 20.45342645015271
MSE: 1514.817857773913	MSE: 1461.871322551047
RMSE: 38.92066106547926	RMSE: 38.23442588232556
CV5 mean: 0.004892268662532473	CV5 mean: 0.0369668195730464

It can be observed that the gradient boosting regressor still performs better than linear regression. Nevertheless, the results are so bad, that such model is basically useless. However, I still tried to optimize the parameters of gradient boosting regressor and also tried to execute it with a different loss function (Huber) and optimized the alpha parameter for it. This later turned out to result in an even worse performance, so the results below are for the squared error loss function with optimized learning rate and number of estimators:

Gradient boosting regressor - optimized

R2_test: 0.0689648953628631

R2_train: 0.07135964500009606

MAE: 20.209150672049415

MSE: 1429.1794753571437

RMSE: 37.80449014809145

Discussion

Flight arrival delay prediction can be done well, within 90% accuracy, if the information about the flight departure is available. The most influencing feature in this case by far is the departure_delay, which is not surprising. So we are able to predict the flight arrival delay immediately after the flight takes off. But not knowing anything about the departure makes the prediction very difficult. I was quite unpleasantly surprised by such poor results for this case. I tried searching online for solutions to this problem from other people to check if I did something wrong, but I noticed that even if they excluded the departure delay from their dataset, they still left information such as wheels_off and taxi_out, which implicitly contain information about the actual departure and therefore also about the delay, so they were still able to get good results.

I have tried many algorithms during this project and they mostly required a lot of computational power because of a huge dataset, which took my computer a lot of time. Perhaps next time I should focus on one algorithm and make it as best as I can. I think the preprocessing could also be improved, but all in all I’m content with the results, at least for the dataset with departure information.