GROUP : 17

AML Challenge 1 : Airport Delay Prediction

Vincent Labarre, Ankita Sahoo, Yassine Elkheir, Lamia Salhi

https://www.oxfordsaudia.com/en/blog/airplane-movies-5-must-watch-movies-for-aviation-lovers/

I. Introduction :

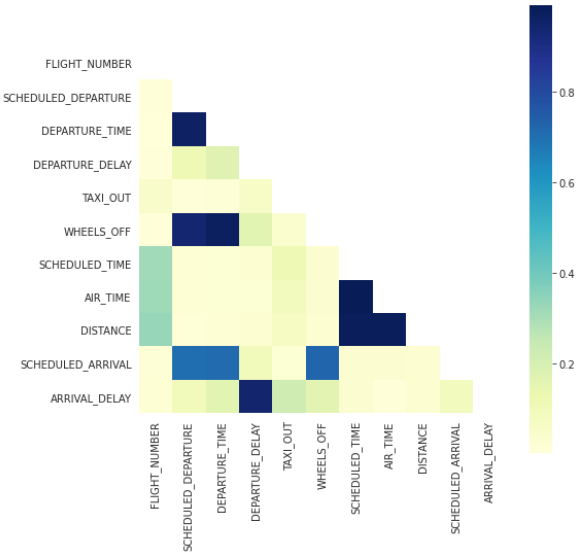
During this challenge, we are going to predict the arrival delay of each airplane. We are given 2 data sets: train and test. Each data set contains many features that we will explore, try to find eventual correlations between them and select the most useful ones to predict the house price. Finally, we will test and compare a few models trained on this data in order to select the model with the best performance.

II. Data Pre-processing :

1. Numerical Features :

A quick description of the numerical fields is quite useful to have an idea about our data (e.g. the data range, min/max, mean of each column, etc). The range of values of each column varies widely. So normalizing the data is an important step in order to avoid any scaling problem and also to speed up the training process of a model. We deleted empty features ('**CANCELLATION_REASON**'), and we eliminated features that aren't relevant to a predictive model ('**ID**', '**YEAR = 2021**').

As we can see we have so many features describing the data samples. Probably not all of these features are important to predict our target which is the delay arrival. This heatmap gave us a great overview of our data as well as the relationship between different features. We notice that there are many dark blue-colored squares: there are obvious correlations such as between **AIR_TIME, SCHEDULED_TIME, DISTANCE and WHEELS_OFF, SCHEDULED_DEPARTURE, DEPARTURE_TIME**. As we don't have enormous number of features, we decided to keep all numerical features except for **FLIGHT_NUMBER** is very random feature, which is normal, because it is Flight Identifier (Not Sequential) and we can see that it is not a continuous feature, it's a semi numerical but it takes too numerous values (~ 6100) so due to our material limitation, we cannot consider it as categorical.

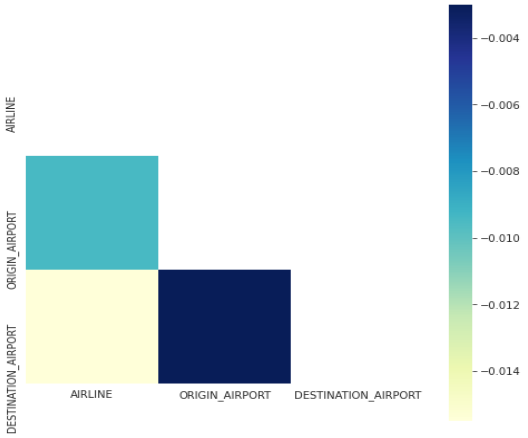


2. Categorical Features :

Categorical Features in that case are **AIRLINE**, **ORIGIN AIRPORT**, **DESTINATION_AIRPORT**, **CANCELLED** and **DIVERTED.**

- we don't need To include **CANCELLED** as categorical features (same value for all "0" No Cancelled Flight)
- we don't need To include **DIVERTED** as categorical features (same value for all "0" no Flight Diverted),

The common idea with the categorical features is to use the One Hot Encoding, but here we have a problem, **DESTINATION_AIRPORT**, and **ORIGIN_AIRPORT**, have total distinct values of 1257, which makes our data set very large.

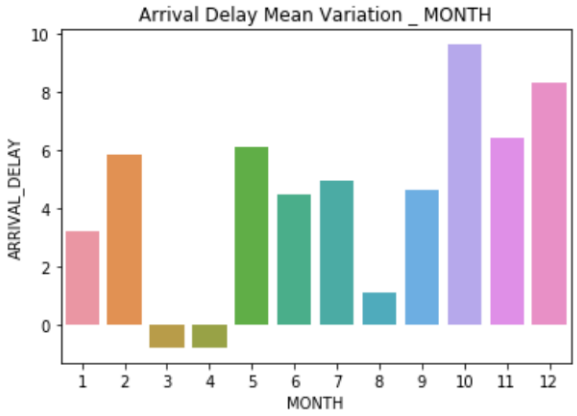


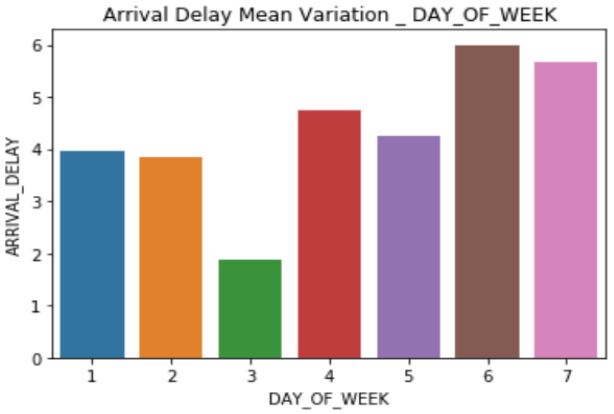
Solution : Instead of working with airports, we can work with there coordinates of airports given in **AIRPORTS file**, so:

DESTINATION_AIRPORT will be presented by numerical features **LATITUDE_dst, LONGITUDE_dst**

ORIGIN_AIRPORT will be presented by numerical features **LATITUDE_org, LONGITUDE_org.**

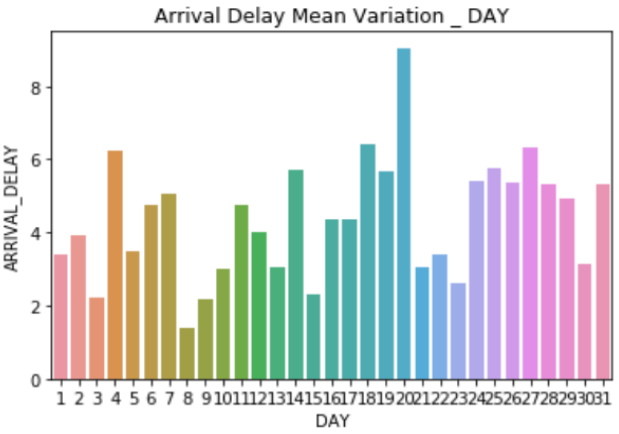
3. Semi-Categorical Features :





We observe that the arrival delay depends on the month, and on the day, according to the plots, a large average delay is completed during the end of the year period, and in summer (*which can be explained by the increase in numbers of trips in these periods*), and a significant variation in the arrival delay compared to the day, and the increase of the delay is seen in week-end days.

The limitation of the material, forces us to use these features when the prediction model allows. because the one hot encoding results in **50** additive features.



4. Missing Data and size manipulation

After examination, it appears that there is no missing data in the variables of the datasets, except for the empty feature '**CANCELLATION_REASON**'.

Concerning the size of the data, the feature selection step allowed us to suppress a certain among of data but we also experienced that we don't need the whole dataset to train our model. Thus depending on the models we introduce after we only used a subset of the training set.

Another preprocessing is used after, this is a polynomial feature and it is used on some models. This sklearn function allows to generate a new feature matrix with polynomial combinations of the features with degree less than or equal to the specified degree [1].

III. Model Selection :

1. Models considered

a. Kernel Ridge Regression

Kernel ridge regression (KRR) combines ridge regression with the kernel trick. It is one of the best models to adjust the bias to make the estimates reasonably reliable.The form of the model learned by the Kernel Ridge is identical to support vector regression (SVR) which is one of the good models for regression. But this model is seven times faster than SVR. Moreover, it avoids overfitting and performs well in cases of a large multivariate data.

b. Polynomial Linear Regression (degree=2)

We have chosen Polynomial Linear regression as it is a special case of linear regression where we fit a polynomial equation of degree 2 on the data with a curvilinear relationship between the target variable and the independent variables. It is one of the basic and non complex models. As it is non complex, a broad range of functions can be fit under it.Various observation states that it provides the best approximation of the relationship between the dependent and independent variable.

c. Elastic NET

Elastic net is a penalized linear regression model that includes both the L1 and L2 penalties during training. The benefit of an elastic net model is that it allows a balance of both penalties, which we thought could fetch us a better performance than a model with either one or the other penalty on some problems. Thus we considered this model.

d. Gaussian process Regression

We decided to try Gaussian Process Regression as it is a nonparametric, Bayesian approach to regression that is making waves in the area of machine learning for both regression and classification. Their greatest practical advantage is that they can give a reliable estimate of their own uncertainty.

e. Random Forest Regressor

Random Forest Regressor is a supervised learning algorithm that uses ensemble learning methods for regression. We thought that it could be a good model to predict the air travel delay as it is one of the models that works well with both categorical and continuous values and as an additional precaution it automates missing values present in the data.It is also expected to reduce overfitting in decision trees and helps to improve the accuracy.

f. DNN(Deep Neural Network)

Deep neural network (DNN) models are one of the advanced deep learning models which are less complex. Moreover it can address these limitations of matrix factorization**.** It can easily incorporate query features and item features (due to the flexibility of the input layer of the network), which can help capture the specific interests of a user and improve the relevance of recommendations. Thus we thought to try this model.

2. Model Evaluation

Here is the summary of the considered models.

MODEL NAME	TRAINING RMSE	VALIDATION RMSE	5-Cross Validation	Sample Number
Kernel Ridge (Polynomial Kernel degree 2)	5.2389	5.2571	5.3428	10000
Polynomial Linear Regressor (degree=2)	5.1341	5.5857	6.3629	10000
Elastic NET (Enet)	5.3525	5.2654	5.3529	10000
Gaussian Process (DotProduct + White Kernels)	6.0409	6.0167	6.3853	10000
Random Forest Regressor	3.9056	9.3886	11.3069	10000
DNN	7.5168	7.8266	////	100000

All tested models are from supervised learning because we know what the labels are.

The model evaluation is done with:

- the training/validation rmse error to assess the performance and generalization of the model.
- the cross validation (with 5 folds here) to give an idea on the variability of the test error. Indeed, it can assess the stability of the model by looking at the models parameter obtained for each fold.

Here, we see that the random forest regressor has the lower training rmse but also the largest validation rmse. That means it is very prone to overfitting, therefore we reject it.
There is the Polynomial Linear regression too that is a little prone to overfitting (but with much lower value for the validation rmse than for the random forest).
However, we can notice that the cross validation value is much bigger than the training and validation rmse. This implies that the model is susceptible to variability in its results when it encounters new data, which we want to avoid.

Then, we note that the DNN has the largest training rmse (but not so large compared to the others models) and the second largest validation rmse, which shows that it works less well than the other models. Moreover, it is important to note that we took much more data for this model but as a consequence it took much more time to train.

And also the cross validation took too much time to compute but we don't need this value here to evaluate the DNN. We have several ways to improve it. For example, we could use more layers to capture more non-linearities but this leads to an excessive demand for ram. Thus we don't keep it, and for the same reason (third largest training and validation rmse), we reject the Gaussian Process too.
Afterwards, it remains the Kernel Ridge Regressor and Enet models. Both have the best validation rmse and cross validation values.

3. Chosen models and further amélioration

To conclude the three best models regarding the cross validation results are KRR, ENet and the Polynomial Linear Regression. But Polynomial Linear Regression asks for a larger data on which to train on. Thus we decided to focus on the KRR and the Enet. In the next section we try to improve those models by combining them through an Average Model.
We can also highlight that Random Forest seems to predict well sometimes, and gives us good validation rmse scores but it almost overfits (high cross validation score).

IV. Average model of Enet and KRR VS Polynomial ENet.

As we had two models KRR and ENet which were giving us both great results for a small number of training dataset we decided to create an average model. Our Averaged model uses the out-of-folds predictions of these base models (KRR and ENet) to train a Lasso Model. The procedure, for the training part :

1. Train several base models on train dataset
2. Predict output using Cross Validation prediction
3. Use those predictions outputs as inputs of the lasso model, which tries to find a mapping function between those predictions outputs of our based models, and the train output.

We averaged the model on the KRR and the Enet with Polynomial features of degree 2. Indeed using this preprocessing method improves the Enet results. However the Size of the training set used has to be increased to have such results and we couldn't train on more than 200000 data. Indeed the RAM limitation didn't allow us more.
Here is a chart summarizing the results.RMSE is rounded upto 4 decimal digits.

Model	TRAIN RMSE	VAL RMSE	TEST RMSE	Cross Validation	Sample Number
Average Model, KRR, ENet with Polynomial Features degree 3	5.3458	5.2306	5.3017	5.3425	10000
ENet with Polynomial Features degree 2	5.2994	5.1150	5.2485	5.4449	200000
Kernel Ridge Regressor	5.2389	5.2571	5.4474	5.3428	10000

As we can see the 3 models have similar cross validation scores with KRR which has a slightly better score. Thus to choose our model we looked at the RMSE result.
The Enet with polynomial features gives the best result in terms of training, validation and test RMSE. However to be used this model needs a training dataset of 200000 samples whereas the kernel Ridge Regressor offers similar results trained on 10000 samples.
To conclude we choose the ENet with polynomial features of degree 2 for this challenge as we have a large amount of data and because the RMSE result is the best. However, Kernel Ridge Regression would be a better model in the case of a smaller dataset. We choose it as our second submission score on kaggle.

Conclusion:

Thus, with correct feature selection and less complex models chosen by us (according to the Occam's razor principle), we find that ENet with Polynomial Features degree 2 and Kernel Ridge Regressor predicts the air delay better considering the RMSE and Cross validation score. Further, if given some more time, we would have tried to optimize the hyperparameters of the models, optimise the RAM utilisation and would have tried a few more stacked average models to improve the performance of the model.