Data Camp

EMINES - UM6P

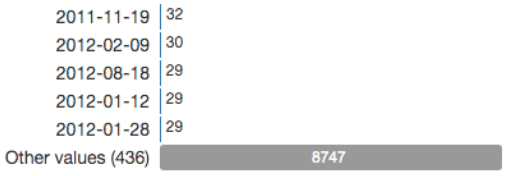
Flight Competition 2022

Zakarya JOUHAFA
Mohamed Amine CHAFIK
17 January 2022

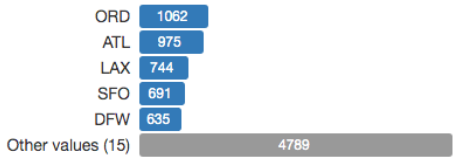
I - Data Exploration

The addressed data we will study revolves around flight data, its columns and distributions are represented in the following :

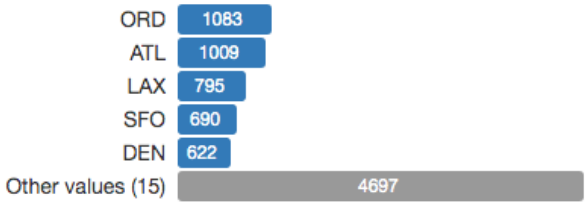
- **flight_date** : the date of the flight code



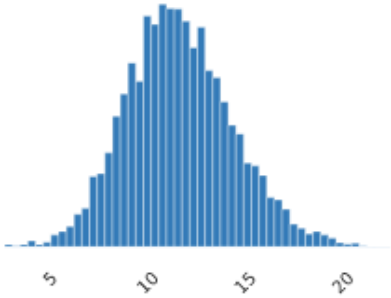
- **from**: the departure's airport



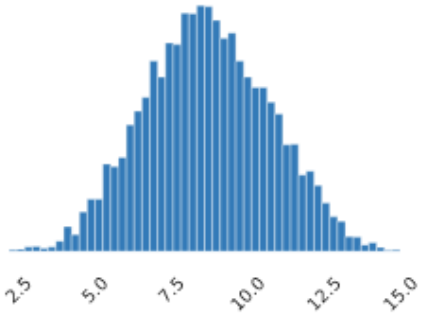
- **to**: the arrival's airport code



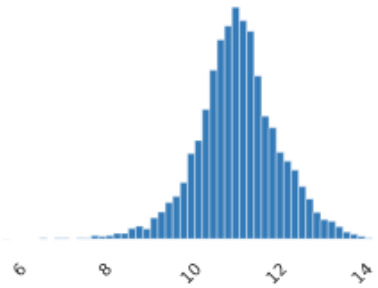
- **avg_weeks**: the average number of weeks between ticket purchase and flight date



- **std_weeks**: the standard deviation of the number of weeks between ticket purchase and flight date



- **target**: the variable to predict. It relates to the number of passengers on the flight.



No missing data has been found. We should highlight the high correlation between **std_weeks** and **avg_weeks.**

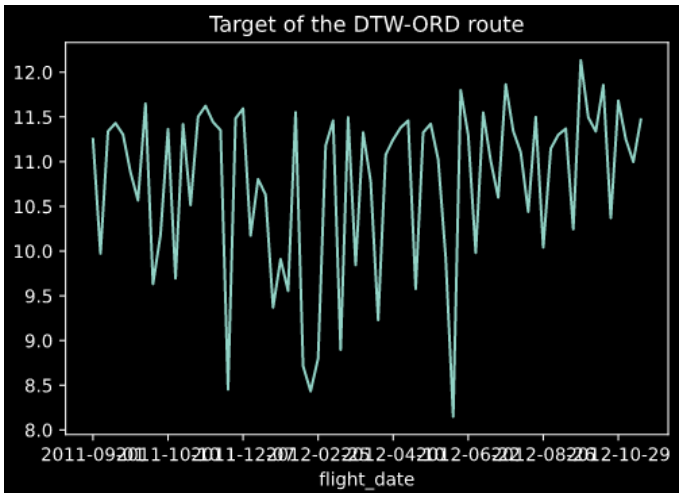
II - Data Processing and Modeling

Time Series Approach

- **GridSearchCV :**

It is clear that the approach using time series is a genuine one as data is instanced by Date. Yet, we should subset the data by route which is a column we created by adding the departure airport to the arrival. Thus, we now have 126 time series specified by route.

Here is one example :



- **Forecast and parameters tuning :**

We should now perform forecast using SARIMA over all time series, we chose to do a 80-20 split and mean over all the rmse in order to find the best parameters for the sarimax model.

We mean over all time series to get the best (p,d,q) parameters.

(p,d,q)	Rmse
(0, 0, 1)	0.66
(0, 1, 0)	0.95
(1, 0, 0)	0.64
(0, 1, 2)	0.72

(p,d,q)	Rmse
(1, 2, 1)	1.12
(2, 1, 1)	0.73
(2, 2, 1)	1.15

The best rmse on the training set is 0.64.

With Numerical Data

The first method to approach this dataset is trying to train our ML models only on the numerical variables because they're easy to manipulate.

In our case, there's two numerical columns in the dataset beside the target, "avg_weeks" and "std_weeks".

- **Train test size :**

X : numeric data; y : target

Train_data : 75% Vs test_data : 25%

```
The training dataset contains 6672 samples and 2 features
The testing dataset contains 2224 samples and 2 features
```

- **Models :**

We perform two of the very popular ML models :
- XGBoost : n_estimators = 500, max_depth = 4, min_samples_split = 5,

learning_rate = 0.01

- RandomForest : n_estimators = 500, max_depth = 2

- **GridSearchCV :**

The performance of the 2 models was poor. To boost this performance, we perform a HalvingGridSearchCV for the RandomForest Model.

By taking the best parameters given by this approach we get the following results :

Models	MAE	RMSE	Accuracy
XgBoost	0.7222	0.9373	2.5

Models	MAE	RMSE	Accuracy
RandomForest	0.7216	0.9350	2.9
RandomForest with GridSearchCV	0.7216	0.9352	2.9

However, the model still not performing well which demonstrate the inadequacy of this approach (only numerical data).

Including Categorical Data

Surely after assessing the results of the first approach it was clear that it's necessary to exploit also the information given from the categorical columns on our dataset.

- **Date preprocessing:**

One the important given in the variables if the date of flight. To extract the maximum insights from this variable, we opted for the following processing :

- Add a column of the Year
- Subtract the min year (make it reference)
- Add a column of the Month (from 1 to 12)
- Add a column of the Week-day (from 1 to 7)
- Add a column of Weekend (binary variable)
- Add a column of Holidays (binary variable)

- **Time series CV :**

Since the dataset is a time-ordered event, we will use a time-sensitive cross-validation splitter to evaluate our demand forecasting model as realistically as possible. We do so using TimeSeriesSplit from sklearn.

- **Encoding Categorical data :**

To benefit from the information encapsulated in the categorical variables, we should encode their inputs to a numerical representations. We spot the categorical columns and with their unique categories respectively. We apply mostly the One Hot Encoder with a MinMaxScaler.

- **Sin & Cos Transformation :**

In this part we try to encode each of those periodic features using a sine and cosine transformation with the matching period.

Each ordinal time feature is transformed into 2 features that together encode equivalent information in a non-monotonic way, and more importantly without any jump between the first and the last value of the periodic range.

- **Adding kernel :**

We use the Nyström method to compute an approximate polynomial kernel expansion to explain the non-linear interaction between some variables.

Results

One-hot-Encoding only :

Models	MAE	RMSE	Accuracy
Ridge	0.5569	0.7171	45.9
HistGradientBoosting	0.3394	0.4702	76.8
AdaBoost	0.6418	0.8018	32.4

With Sin&Cos Transformation :

Models	MAE	RMSE	Accuracy
Ridge	0.5527	0.7151	46.2
HistGradientBoosting	0.3356	0.4632	77.4
AdaBoost	0.6235	0.7854	35.1

With Spline Transformation :

Models	MAE	RMSE	Accuracy
Ridge	0.5472	0.7033	48.0
HistGradientBoosting	**0.3347**	**0.4612**	**77.6**
AdaBoost	0.6262	0.7818	35.7

With Kernel :

Models	MAE	RMSE	Accuracy
Ridge	0.4138	0.5532	67.8
HistGradientBoosting	0.4130	0.5517	68.0
AdaBoost	0.6400	0.8010	32.6

Conclusions

After applying several methods we come to the conclusion that the best model to predict our target from this dataset was the histogram-based gradient boosting regressor with the sin&cos transformation which can allow us to say that wise to approach the problem from its variability in time, however our algorithm of pure time series approach didn't perform as well as expected.

What's next :

Our approach at the end had decent results but there's always a room of improvement. One of the approach that we couldn't finish implementing was to apply a GNN on our dataset. Secondly, maybe adding other variables can help express better our target.

References

- https://scikit-learn.org/stable/auto_examples/applications/plot_cyclical_feature_engineering.html#sphx-glr-auto-examples-applications-plot-cyclical-feature-engineering-py