SEP23\_ALLIANZ\_DATA\_SCIENTIST

**Road accidents in France**

Report 2 - Modelling

horizontal line

# 

Jennifer Pousada Alvarez  
Alessandro Perani  
Ahed Abdelky  
Falk Kegler  
20240623

# Classification of the problem

The aim of our study is to design a classification model for predicting the severity of road accidents in France. Predicting the severity of road accidents in France is a task related to predictive analytics and risk assessment. This type of task involves analyzing historical data and other relevant factors to predict future outcomes or the severity of incidents.

We evaluated the performance of each model by calculating performance metrics such as accuracy (Accuracy) and F1-score (F1-score) on the training set and test set.

Accuracy was used to measure the overall precision of the model. We used Accuracy as it is straightforward to understand and interpret. It measures the proportion of correctly predicted instances (both true positives and true negatives) out of the total instances. If the classes (severity levels) are balanced as it is the case in our data, accuracy can give a good overall measure of the model's performance.

Furthermore, the F1-score was taken into account. This metric is as well useful for binary classification problems, as it takes into account both precision and recall to calculate an overall score.

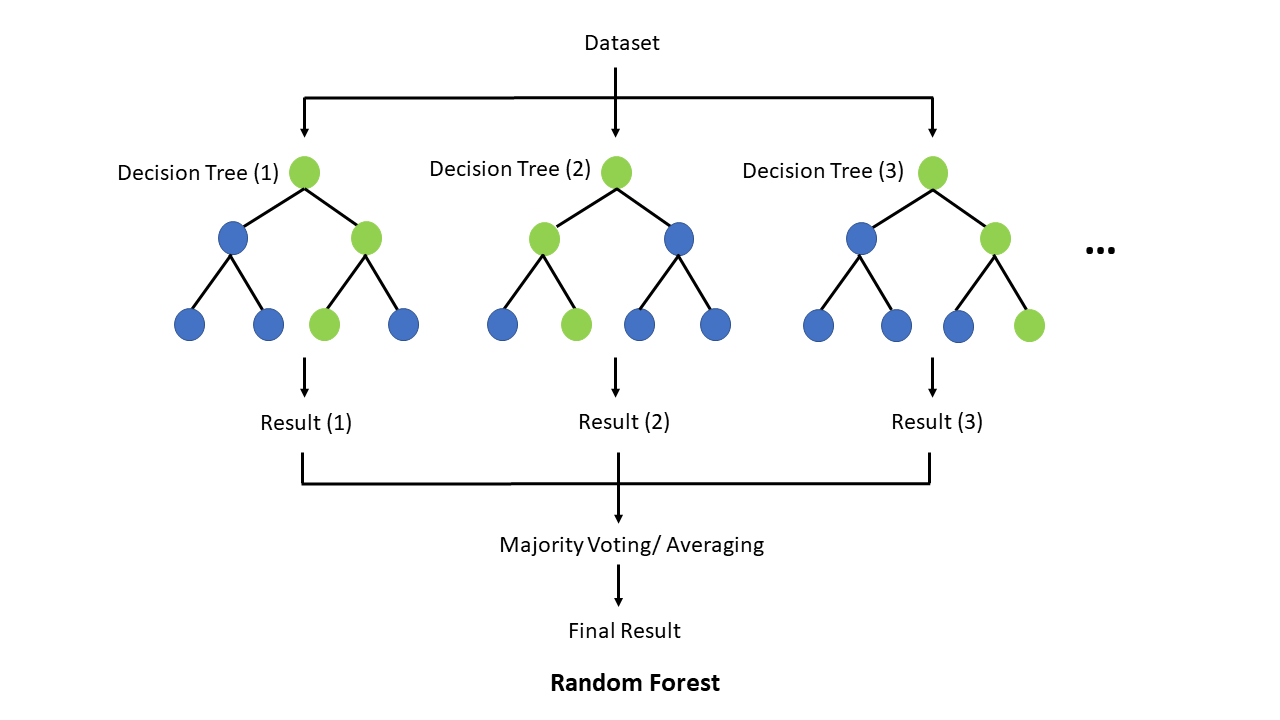
For the model’s performance, we also considered the Area under the ROC curve (Roc-AuC).

We tried 5 models to predict the severity of the accidents.

* Random Forest Classifier for a ternary target
* Random Forest Classifier for a binary target
* KMeans Cluster with n\_cluster = 2
* XGBoost with GridSearchCV for a binary target
* Dense Neural Network for a binary target

The details of the outcomes are described on the following pages of the report.

## RANDOM FOREST



Our first models were RandomForest Classifiers. We modelled with a ternary and a binary target for the severity of the accident. We also examined the feature importance and visualized it with the help of the SHAP-package for the binary-target model on an instance for target 0 and target 1 respectively.

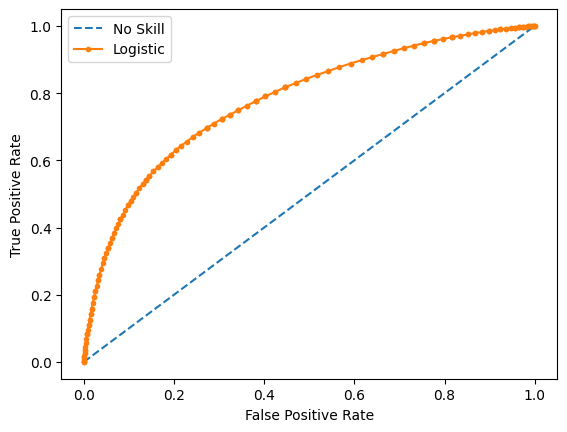
### Random Forest Classifier with ternary target

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| RandomForest | precision | recall | f1-score | support |
| 0 | 0.69 | 0.82 | 0.75 | 83.718 |
| 1 | 0.62 | 0.56 | 0.59 | 66.542 |
| 2 | 0.54 | 0.05 | 0.09 | 9.694 |
| accuracy |  |  | **0.66** | 159.954 |
| macro avg | 0.62 | 0.48 | 0.48 | 159.954 |
| weighted avg | 0.65 | 0.66 | 0.64 | 159.954 |

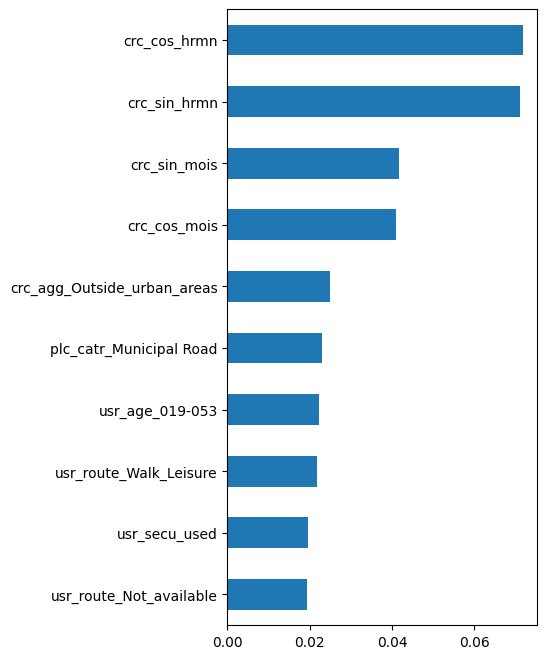
### Random Forest Classifier with binary target

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| RandomForest | precision | recall | f1-score | support |
| 0 | 0.70 | 0.80 | 0.75 | 83.718 |
| 1 | 0.74 | 0.63 | 0.68 | 76.236 |
| accuracy |  |  | **0.72** | 159.954 |
| macro avg | 0.72 | 0.71 | 0.71 | 159.954 |
| weighted avg | 0.72 | 0.72 | 0.71 | 159.954 |

### ROC AUC=0.780



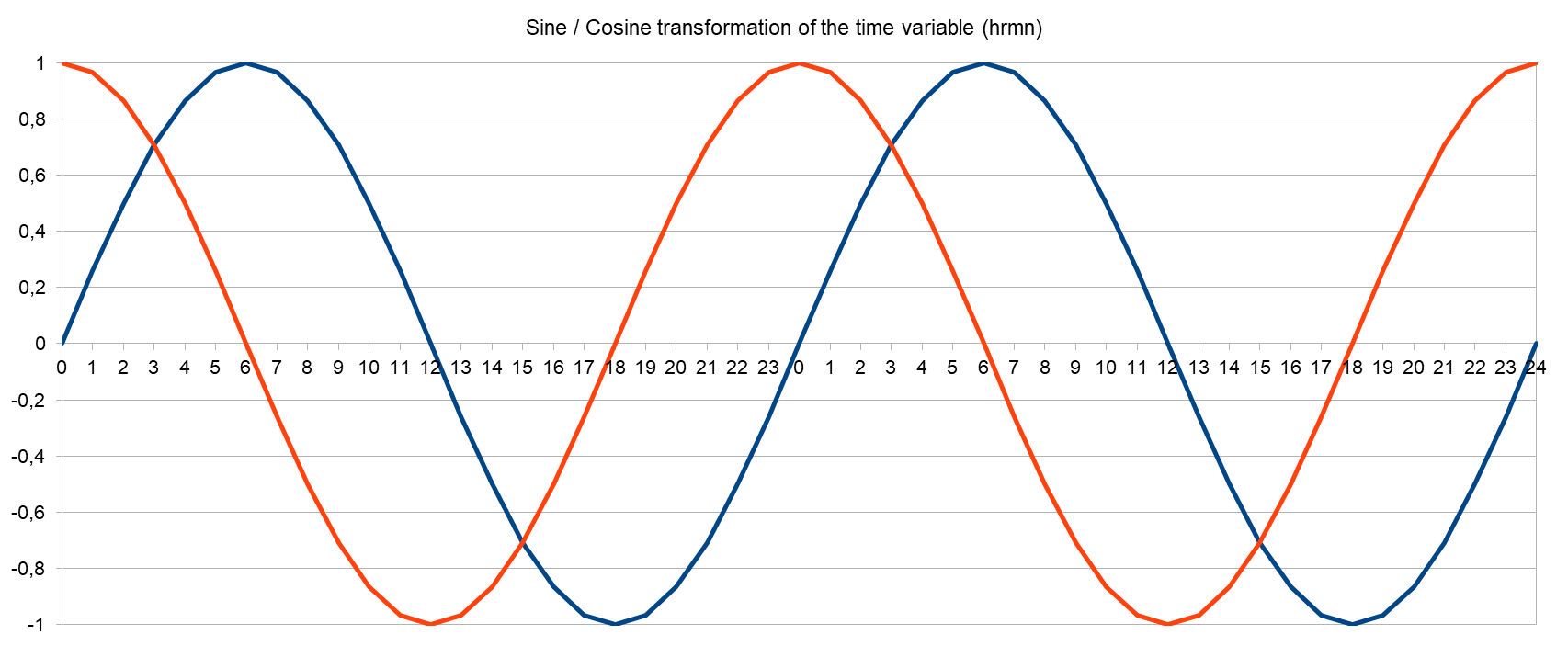
### Feature importance from Random Forest Classifier (Top 10):



The prefixes of the feature names are:

* crc for characteristic-features
* plc for place related features
* usr for user related features
* veh for vehicle related features

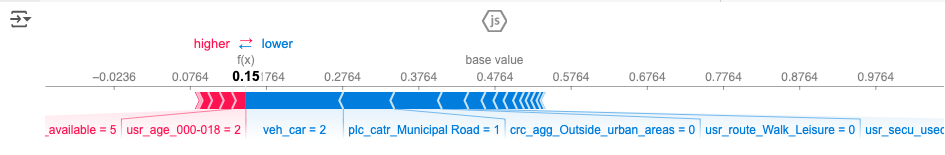
The time related features “minute of the day” and “month” in the characteristics-Tables have been replaced using a sin/cos decoding to have nearby times and dates better represented for the models to learn from them:



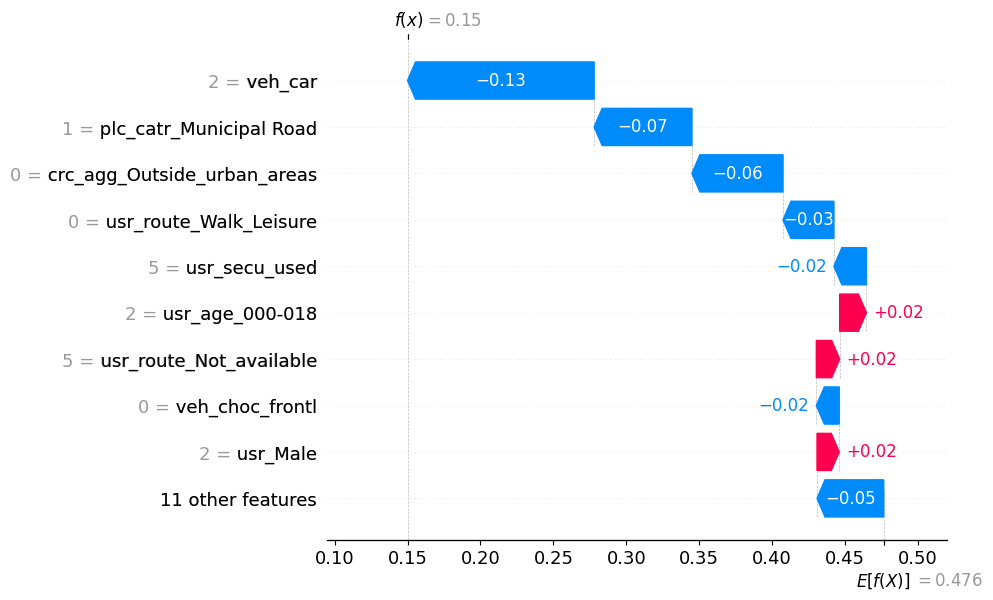
### SHAP

For the Random Forest Classifier we used the SHAPly values to visualize the feature importance and to show how and to what extent each feature contributed to the final prediction result. 2 instances were chosen, one with prediction result 0 and another with prediction result 1.

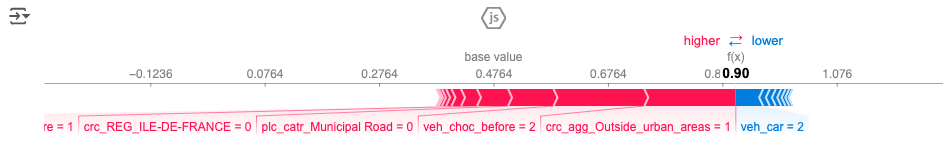
Shap-explainer (force) for instance with prediction result 0



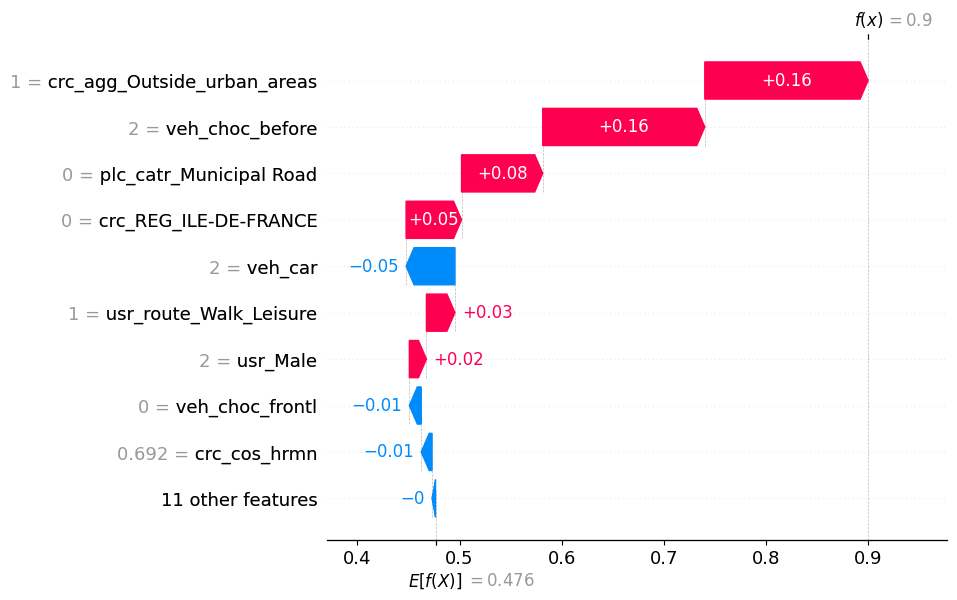
Shap-explainer (waterfall) for instance with prediction result 0:



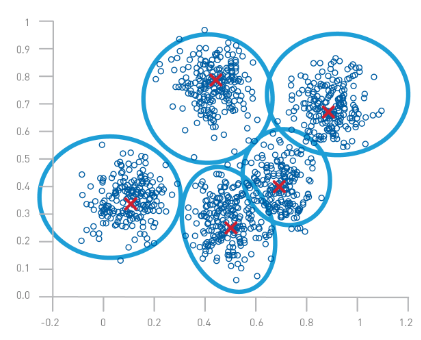
Shap-explainer (force) for instance with prediction result 1:



Shap-explainer (waterfall) for instance with prediction result 1:



## KMeans Cluster



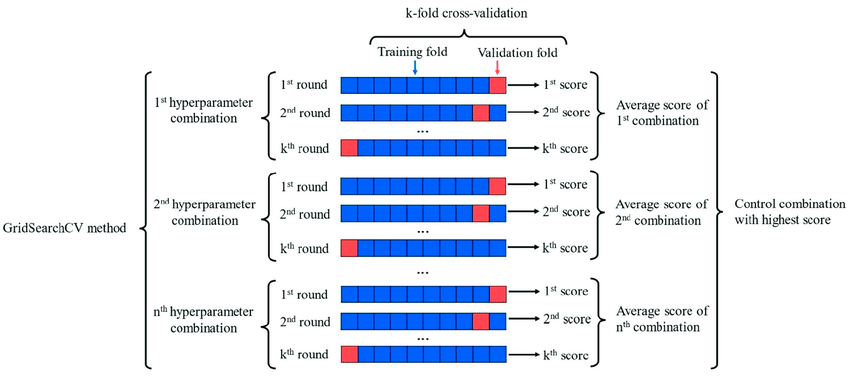
We also tried a KMeans Clustering model to see, if there is a natural cluster-mean for the binary target representing the severity of the accident. It turned out, that the correlation of the clusters found to the target was 0.25, so there is no indication of a natural severity-based clustering in the dataset.

### KMeans Cluster (n\_clusters = 2)

np.abs(dfc.corr()['cluster'][:]).sort\_values(ascending=False)[:12]

|  |  |
| --- | --- |
| **correlations** | **cluster** |
| cluster | 1.00 |
| crc\_agg\_Outside\_urban\_areas | 0.91 |
| plc\_catr\_Municipal Road | 0.61 |
| plc\_catr\_Highway | 0.40 |
| crc\_lum\_Night\_without\_public\_lighting | 0.34 |
| crc\_int\_Outside\_intersection | 0.29 |
| plc\_plan\_Curvature | 0.29 |
| veh\_obsm\_pedestrian | 0.27 |
| veh\_obsm\_unknown | 0.25 |
| plc\_situ\_Roadside\_verge | 0.25 |
| **target** | **0.25** |
| crc\_lum\_Night\_with\_public\_lighting\_on | 0.22 |

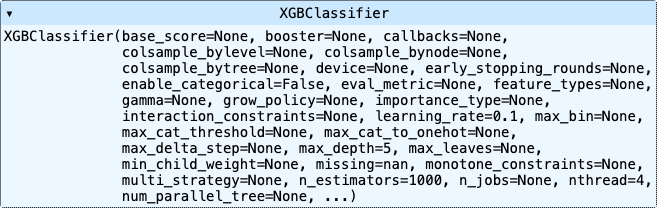
## XGBoost with GridSearch CrossValidation



The XGBoost-Algorithm is famous for its high accuracy, especially on Kaggle. We tried to find the best model with the help of a grid search cross-validation. Indeed, the model with the best estimators achieved the highest accuracy of all our models (73%) and the highest value for the ROC-AuC (0.793).

### XGBoost Classifier with Hyperparameter Tuning (grid search)

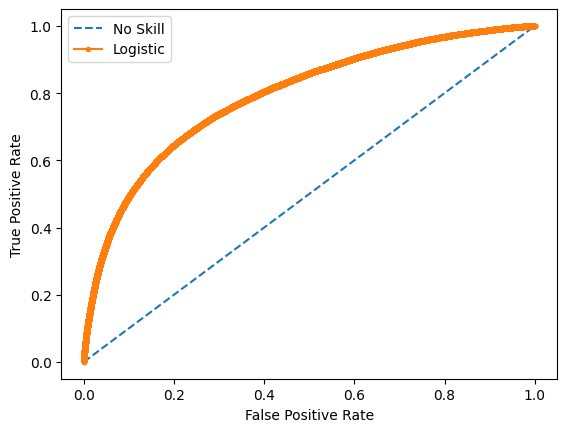
grid\_search.best\_estimator\_:



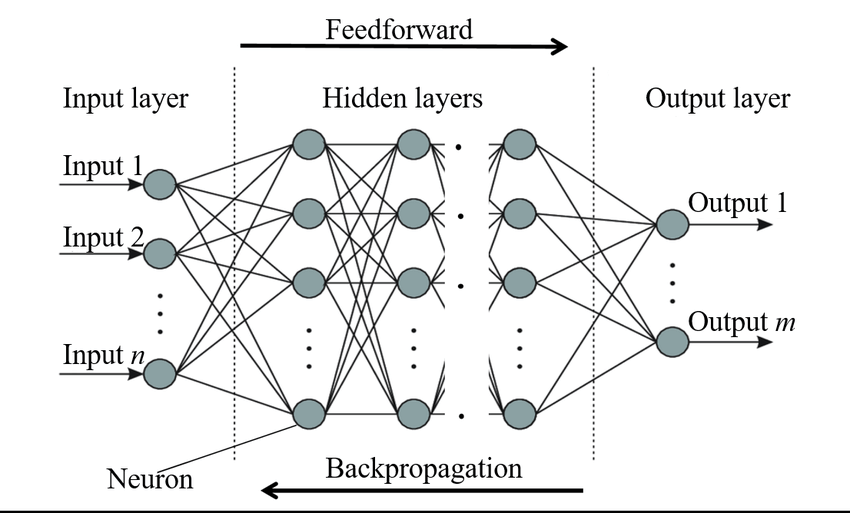
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| XGBoost | precision | recall | f1-score | support |
| 0 | 0.71 | 0.80 | 0.75 | 83.718 |
| 1 | 0.74 | 0.65 | 0.69 | 76.236 |
| accuracy |  |  | **0.73** | 159.954 |
| macro avg | 0.73 | 0.72 | 0.72 | 159.954 |
| weighted avg | 0.73 | 0.73 | 0.72 | 159.954 |

The best parameters across ALL searched params: grid\_search.best\_params\_

### {'learning\_rate': 0.1, 'max\_depth': 5, 'n\_estimators': 1000} ROC AUC=0.793



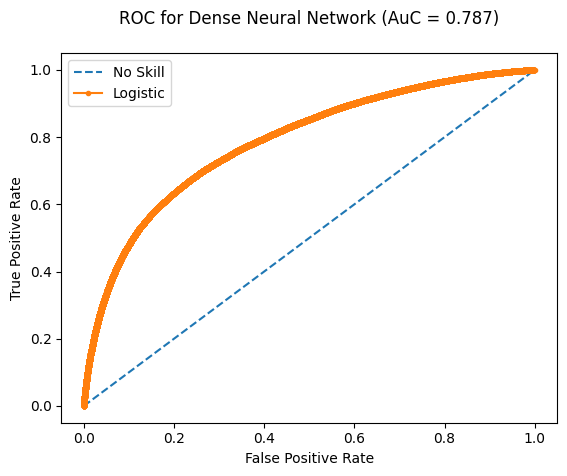
## Deep Learning



To practice what we have learned, we decided to try a deep learning model as well. With two hidden layers in a dense neural network we were able to achieve an accuracy of 71%.

### Dense Neural Network with 2 hidden layers

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| DNN | precision | recall | f1-score | support |
| 0 | 0.70 | 0.80 | 0.75 | 83.718 |
| 1 | 0.74 | 0.62 | 0.68 | 76.236 |
| accuracy |  |  | **0.71** | 159.954 |
| macro avg | 0.72 | 0.71 | 0.71 | 159.954 |
| weighted avg | 0.72 | 0.71 | 0.71 | 159.954 |



As we received the best accuracy with the XGBoost estimator, we will use the following **model** for all future tasks:

## OUR MODEL

**from xgboost import XGBClassifier**

**model = XGBClassifier(**

**objective = 'binary:logistic',**

**learning\_rate = 0.1,**

**n\_estimators = 1000,**

**max\_depth = 5,**

**seed = 42)**

# Interpretation of results

The different models all resulted in an accuracy of about 72%. The best model we found was an XGBoost Classifier with an accuracy of 73% and an Area under the curve of 0.787. The addition of external data, such as holiday dates in France, did not improve the forecast accuracy.