Team Salvi

Eugenio Lomurno

Giovanni Gianola

Giacomo Fusetti

Descriptive analysis and interesting findings

Pre-Processing

- Feature engineering
- Unsuccessful approaches

Model selection

- Scikit-learn models
- Neural network
- Gradient boosting

Final model

- Categorical Gradient Boosting Classifier

Summary

- Final considerations
- Further implementations

Pre-Processing

Feature Engineering

- Data cleaning
  - Mostly empty columns (DataAllowanceOneShot, EstimatedDevicePrice) are dropped
  - Columns Region, Province, CustomerAge and Product are transformed from string to numerical values
  - The remaining missing values are filled with -999

	Raw_CustomerAge	CustomerAge
0	(40, 50]	45
1	(20, 30]	25
2	(30, 40]	35
3	(50, 60]	55
4	(60, 70]	65

- Features generation
  - IsModified: a binary column in which each row is set to 1 if there are missing values in that row, to 0 otherwise

IsModified	CustomerAge	Region	Province	Product
0	55.0	12.0	7.0	0
0	45.0	2.0	26.0	0
0	45.0	2.0	26.0	0
1	55.0	-999.0	-999.0	2
0	35.0	8.0	25.0	0

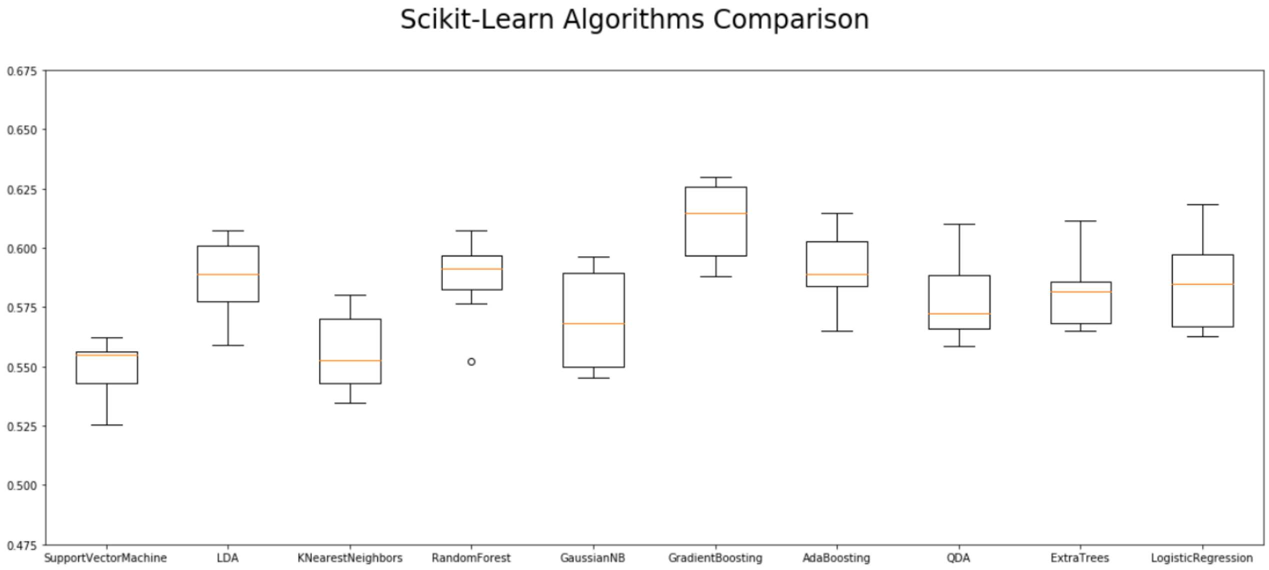
Unsuccessful approaches

- Drop all the rows with a missing value
- Fill all the NaN values of the dataset with the mean/mode/zero value of the corresponding column
- Grenerate the colums ConnectionsCount, ConnectionsDuration and RegionsCluster
- One hot encoding of categorical features
- Label ensembling (EasyEnsemble, SMOTE, SMOTEENN)

Model Selection

Scikit-learn models

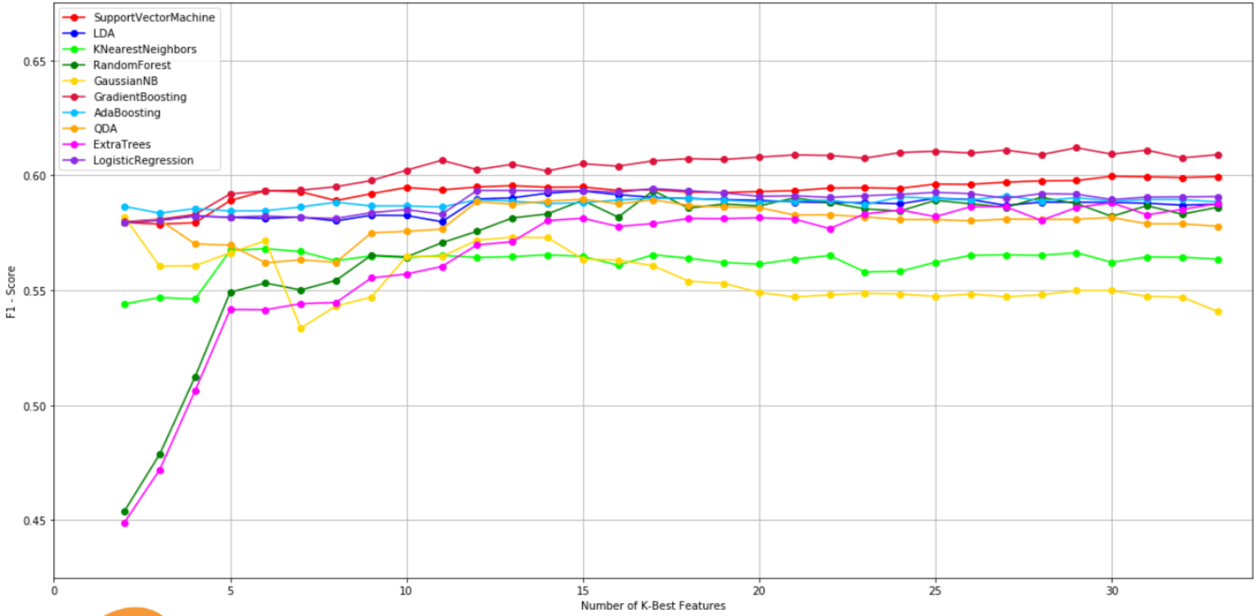
- Model analysis to understand which family of models can better fit the problem



Model	F1_score	Margin
SupportVectorMachine	0.549	+/-0.011
LDA	0.588	+/-0.015
KNearestNeighbors	0.556	+/-0.016
RandomForest	0.587	+/-0.014
GaussianNB	0.569	+/-0.020
GradientBoosting	**0.611**	**+/-0.015**
AdaBoosting	0.590	+/-0.015
QDA	0.578	+/-0.016
ExtraTrees	0.580	+/-0.014
LogisticRegression	0.585	+/-0.019

Scikit-learn models

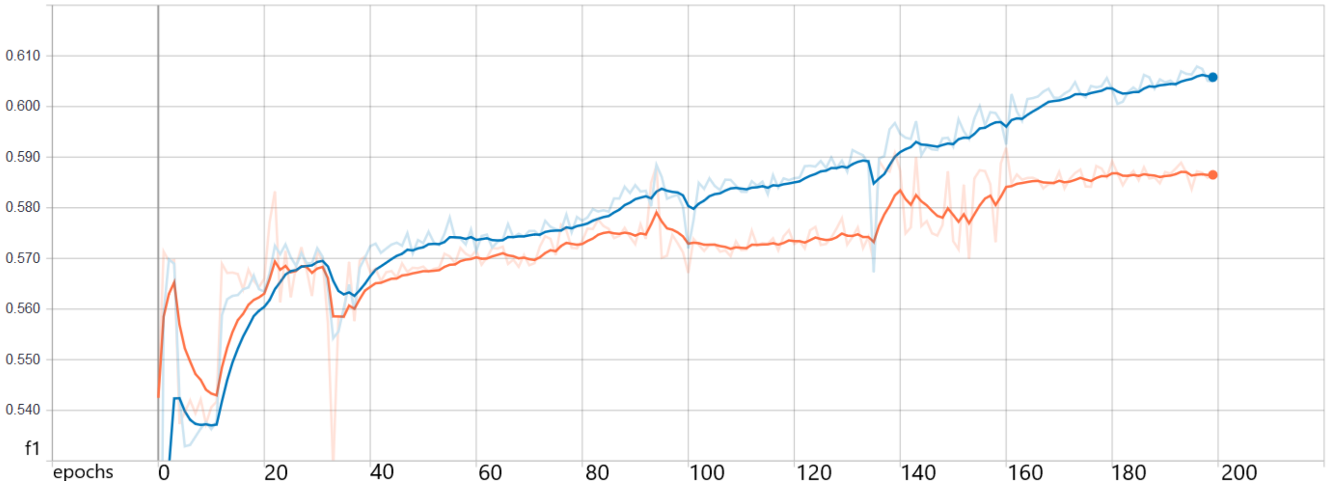
- SelectKBest features with Chi-squared test to evaluate the best features for each model
- There are many features with a weak correlation with the label



Model	F1_score	N Feat
SupportVectorMachine	0.600	30
LDA	0.593	15
KNearestNeighbors	0.568	6
RandomForest	0.593	17
GaussianNB	0.582	2
GradientBoosting	**0.612**	**29**
AdaBoosting	0.591	27
QDA	0.590	15
ExtraTrees	0.588	30
LogisticRegression	0.594	17

Neural Network

- Neural network built with Keras and plotted with Tensorboard
- The following architecture lets us able to reach **F1 score = 0.5866**

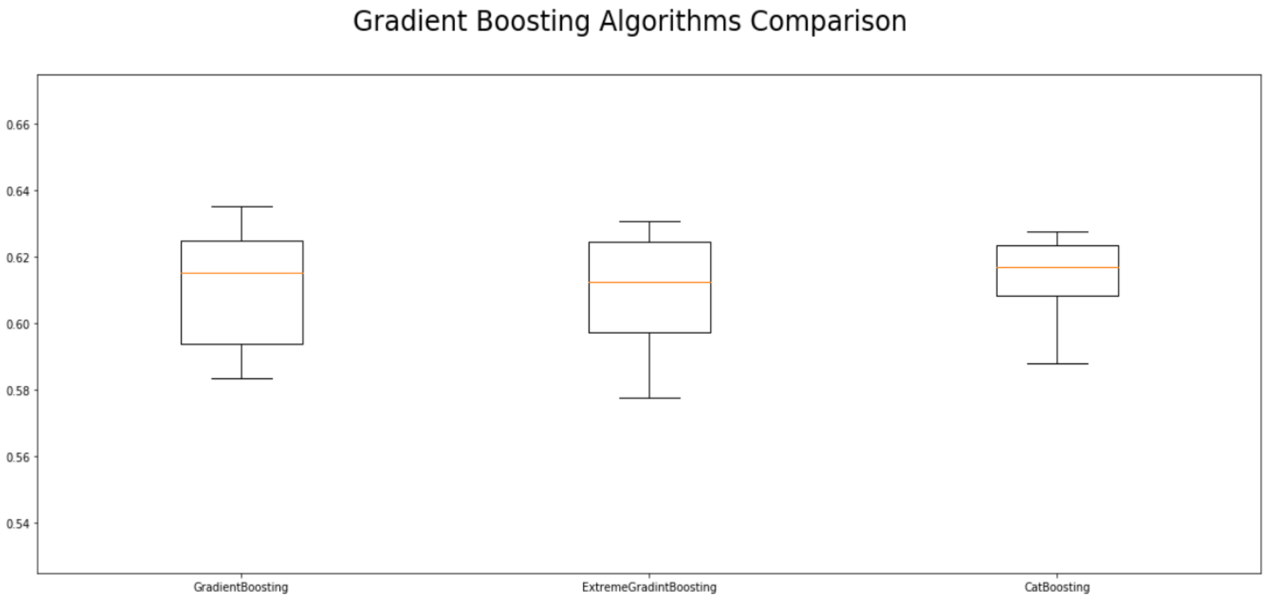



```
Layer (type)                 Output Shape              Param #
=================================================================
dense_1 (Dense)              (None, 64)                1920
_________________________________________________________________
activation_1 (Activation)    (None, 64)                0
_________________________________________________________________
dropout_1 (Dropout)          (None, 64)                0
_________________________________________________________________
dense_2 (Dense)              (None, 64)                4160
_________________________________________________________________
activation_2 (Activation)    (None, 64)                0
_________________________________________________________________
dense_3 (Dense)              (None, 4)                 260
=================================================================
Total params: 6,340
Trainable params: 6,340
Non-trainable params: 0
_________________________________________________________________
```

Name	Smoothed	Value	Step
training	0.6058	0.6050	199.0
validation	0.5865	0.5866	199.0

Gradient boosting

- Gradient Boosting models are really powerful algorithms for this task, here we compare some of the most robust



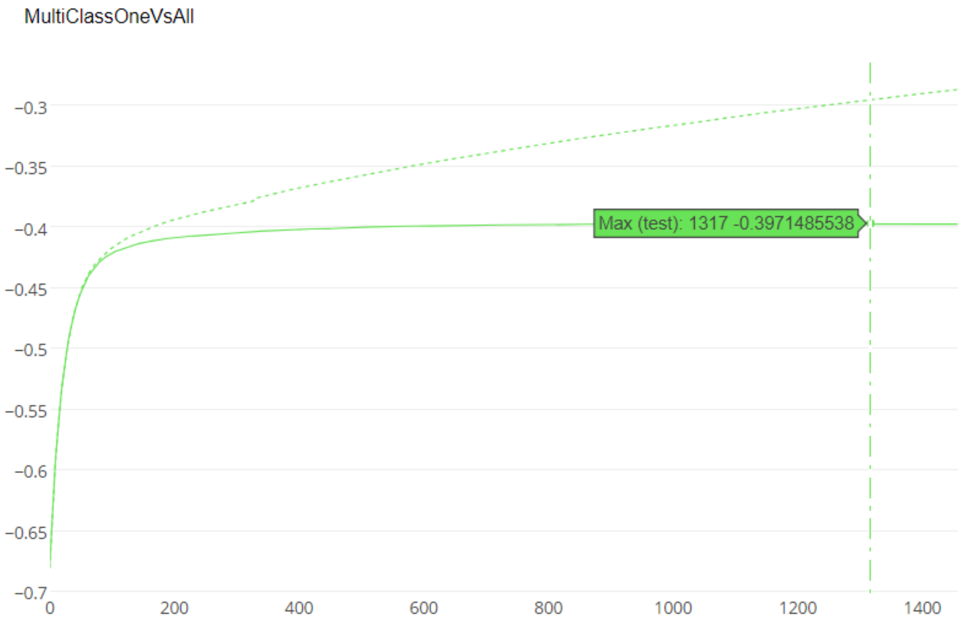
Model	F1_score	Margin
GradientBoosting	0.611	+/-0.017
ExtremeGradintBoosting	0.610	+/-0.017
CatBoosting	**0.614**	**+/-0.012**

Final model

Categorical Gradient Boosting Classifier (CatBoost)

- Final tuned algorithm, evaluated with 10Fold on all the features
- F1 score with crossvalidation: 0.617
- **F1 score on test set: 0.6316**

```
final_model = CatBoostClassifier(
     learning_rate=0.03,
     iterations=1350,
     bootstrap_type='Bayesian',
     depth=6,
     leaf_estimation_method='Gradient',
     random_seed=seed,
     logging_level='Silent',
     loss_function='MultiClassOneVsAll',
     eval_metric='MultiClassOneVsAll',
     custom_metric='F1',
     od_type = 'Iter',
     od_wait=100
)
```



Yandex

Summary

Summary

- Final considerations
  - Even if missing values have negatively influenced the prediction’s quality, gradient boosting algorithms still perform well
  - The reason why the crossvalidation has a poor performance compared with the test score is due to this imbalance inside the fold’s labels

- Further implementations
  - VotingClassifier between different gradient boosting algorithms may increase the results of prediction
  - Because of the high label imbalance, it may be possible to split the problem in two subproblems: a binary classification between customer and non-customer labels, and than a multiclass classification to select the correct device for customes