Santander Customer Transaction Prediction

Nicola Poggialini – ML project work

Task description

- The task is about predicting the future behaviour of customers
- We’re interested in whether a transaction will be made, regardless of the amount of money involved
- Binary classification problem
  - 0: the customer won’t make a transaction
  - 1: the customer will make a transaction

Data exploration

- Dataset provided:
  - 200000 instances
  - 202 features
    - ID code: identifies the transaction (not used in the solution)
    - Target: binary variable that we want to predict
    - 200 anonymous features: predictors

	ID_code	target	var_0	var_1	var_2	var_3	var_4	var_5	var_6	var_7	...
0	train_0	0	8.9255	-6.7863	11.9081	5.0930	11.4607	-9.2834	5.1187	18.6266	...
1	train_1	0	11.5006	-4.1473	13.8588	5.3890	12.3622	7.0433	5.6208	16.5338	...
2	train_2	0	8.6093	-2.7457	12.0805	7.8928	10.5825	-9.0837	6.9427	14.6155	...
3	train_3	0	11.0604	-2.1518	8.9522	7.1957	12.5846	-1.8361	5.8428	14.9250	...
4	train_4	0	9.8369	-1.4834	12.8746	6.6375	12.2772	2.4486	5.9405	19.2514	...

First rows and columns of the dataset

Data exploration

- A brief analysis showed that there are no relevant correlations between the predictiors

- Then, it was noticed that the data are also imbalanced with respect to the target variable

Distribution of the target values

Methods – Approach overview

- The approach followed to tackle the problem was the following:
  - Find a good ‘configuration’ of the dataset
    - The data are imbalanced and present a high dimensionality, both factors that can impact negatively the performances of a model
    - To do that, I’ve started with a simple model (*Logistic regression*) to test the different approaches
  - Find the best model
    - Once the right configuration is found, I’ll tune and test different models on it

Methods – Techniques

- Baseline
  - Logistic regression with adjusted weights for unbalanced data
- High dimensionality
  - Feature selection
    - ANOVA
    - Mutual information
    - Random Forests
  - Dimensionality reduction
    - PCA

Methods – Techniques

- Unbalanced data
  - Undersampling: delete instances from the over-represented class
    - *Near Miss algorithm*: selects examples from the majority class that have the smallest average distance to the three closest examples from the minority class
  - Oversampling: add copies of instances from the under-represented class
    - *SMOTE algorithm*: it works randomly picking a point from the minority class and computing the k-nearest neighbors for this point. The synthetic points are added between the chosen point and its neighbours
  - Since these approaches could suffer from the curse of dimensionality, they are also tested with less/reduced features

Methods – Techniques

- Classifiers:
  - Decision Tree and Random Forests
  - Nayve Bayes
  - SVM

- Alternative approach
  - One class classification
    - Approach used in anomaly detection
    - Tested using the one class version of SVM

Operating environment

- Language used:
  - Python (Jupyter notebook)
- Hardware:
  - Local CPU
  - Accelaration (GPU/TPU): None
- Principal libraries:
  - sklearn
  - imblearn (for implementing the algorithms for unbalanced data)

Results – Comparison on validation set

- Metric
  - F1 score for the less-represented class

- The best score was obtained with the *Logistic regression* when combined with *ANOVA* and *Near Miss*

	Models	F1 score
0	Logistic regression baseline	0.420251
1	Logistic regression with ANOVA - 150 features	0.420180
2	Logistic regression with Mutual Info - 170 fea...	0.411099
3	Logistic regression with Random Forests Feat. ...	0.377583
4	Logistic Regression with PCA - 0.95	0.420680
5	Logistic Regression with PCA - 0.90	0.420120
6	Logistic regression with Near Miss	0.429212
7	Logistic regression with SMOTE	0.422319
8	Logistic regression with SMOTETomek	0.422319
9	Logistic regression with ANOVA and Near Miss	0.430241
10	Logistic regression with ANOVA and SMOTE	0.420504
11	Logistic regression with ANOVA and SMOTETomek	0.420504
12	Decision Tree with ANOVA and Near Miss	0.272710
13	Random Forests with ANOVA and Near Miss	0.343106
14	Naive Bayes with ANOVA and Near Miss	0.309361
15	Support Vector Machines with ANOVA and Near Miss	0.252796
16	One class approach - SVM	0.302181

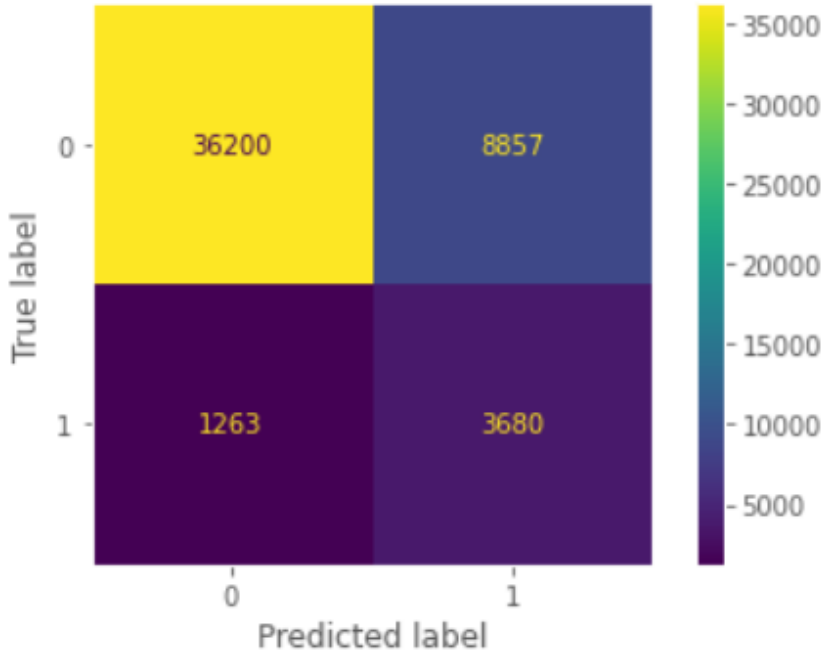
Scores on the validation set

Results – Test set

Test set: classification report

	precision	recall	f1-score	support
0	0.97	0.80	0.88	45057
1	0.29	0.74	0.42	4943
accuracy			0.80	50000
macro avg	0.63	0.77	0.65	50000
weighted avg	0.90	0.80	0.83	50000

Classification report on the test set



Confusion matrix on the test set

Conclusions

- Overall good score (0.83), but still under 0.5 for the less representative class

- Future improvements
  - Better technique to tackle the imbalance problem
    - With Near Miss, in the training set, the number of instances of the more-represented class goes from ~120.000 to ~13.000
  - More complex classifier (e.g. a well-tuned neural network)