CLADAG 2019 DS Competition

Daje Scientists Report

Alessandro Flaborea, Paolo Gentile, Davide Toma

ABSTRACT

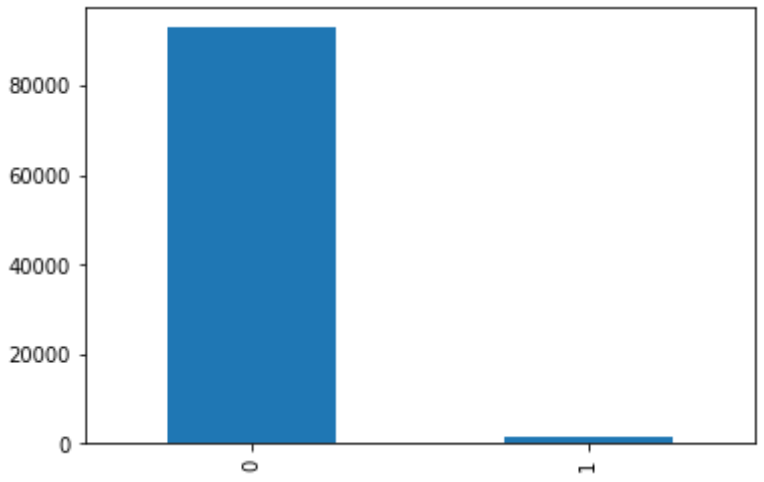
In this project, we were asked to deal with a real world dataset made available for us from TIM Telecommunications Company, and to explore how machine learning algorithms can be used to find patterns in data.
The dataset for the whole challenge has been internal TIM's historical data, related to different domains (customer records, commercials, contacts, traffic, network quality, customer needs) in a date span of 9 months.
The main goal was to build a model able to predict the risk to churn of fixed telephony customers and to identify the discriminant variables that best describe the customers with higher churn risk. We tackled this problem treating it as a highly imbalanced binary classification problem performing a final clustering analysis on the predicted results.

INDICE

1. DATA OBSERVATION

The best way to approach any classification problem is to start by analyzing and exploring the dataset. The sole purpose of this section was meant to generate as many insights and information about the data as possible to find any problems that might exist in the dataset.
First of all we wanted to deal with null observations and fill the 'NaN' values. This part of the analysis was completed by using Pandas library mostly.

Analyzing the train set, it was immediately clear that we were facing an unbalanced dataset that showed an unequal distribution of 0 and 1 values of the 'TARGET' label with a 98:1 ratio between the two classes.



If we had trained a binary classification model without fixing this problem, the model would have been completely biased impacting also the correlations between features. To tackle the issue of class imbalance, Synthetic Minority Over-sampling Technique (SMOTe) was used. This technique is based on nearest neighbors judged by Euclidean Distance between data points in feature space. Basically what was done wass changing the dataset in order to have more balanced data on which training the predictive model by adding copies of instances from the under-represented class ('1').

2. FEATURES ENGINEERING

For this part of the pipeline, first of all, we created new classes for the feature “NEEDS_RECLAMI_00_90_DAYS”, because we wanted to codify if the customer has had any problem with the company.

Then we passed to the management of the Nan values; first we tried to fill the Nan values with numerical features: we filled them with the imputer built in the python library “sklearn.impute”, and we decided to use the Simple imputer with the parameter of the strategy set with the median of the column we were considering.
For the categorical features, we used different strategies to impute the missing values in the columns.

Then we dropped some columns for the following different reasons: the columns named “S_VAR” and “S_TARGET” because of multicollinearity with the feature “T_RUN”, that we kept; in fact, there was not any reason to keep features that are linear combination of “T_RUN” column, so we decided to drop them without loss risk of information.
Then we dropped some columns because we wanted to use the *get_dummies* function later on, and these columns could create a really high number new columns; so for memory safety reasons and because we saw that these features were not so correlated with the target variable we dropped them.

Then we applied the *get_dummies* function to create new binary features starting from the initial features we had, in order to find new features correlated with the target variable.

The application of the *get_dummies* function created a lot of new features that could add some noise when we make the prediction, so we need to apply dimensionality reduction.
We tried two different approaches: in a first moment we tried the Principal Component Analysis, but the results with this dimensionality reduction were not so good with our models, so we tried to find another approach to reduce the dimension of the dataset.
Our final strategy was to find the columns with a number of zeros greater than 80%, then we saved the column names in a list called “overfitting list” and we dropped the columns in the overfitting list from the dataframe.

We applied Robust Scaler to normalize the columns, because in general normalize the variables is useful when we tried to understand the data and the results we got.

In the end, from the previous analysis on the dataset, we knew we were dealing with an highly unbalanced problem, so we used the strategy of oversampling the train set in order to avoid this problem for the analysis with our models.
Oversampling in data analysis is technique used to adjust the class distribution of a dataset, and in our specific case we used SMOTE that is a technique based on nearest neighbors judged by Euclidean Distance between data points in feature space. What this method does

is changing the dataset in order to have a balanced dataset where we can train our algorithm.

3. CLASSIFICATION

In order to predict the risk to churn of fixed telephony customers, we tried to use different classifiers seeking for the model that best fit our data and also trying to avoid overfitting and underfitting.
These are the classifiers tested:

1. Logistic Regressor
2. XgBoost
3. Extra Tree Classifier
4. Random Forest Classifier

It can be seen that we selected simpler models, as logistic regressor, and models that use the ensemble techniques of bagging (Extra Tree Classifier and Random Forest Classifier) and boosting (XgBoost).

For each model we tried to have minimize the risk looking at multiple metrics such as precision, recall, f1 and Roc Auc. Our approach to solve this problem and to have the best predictions was mainly focused on the searching and the tuning of the best hyperparameters.

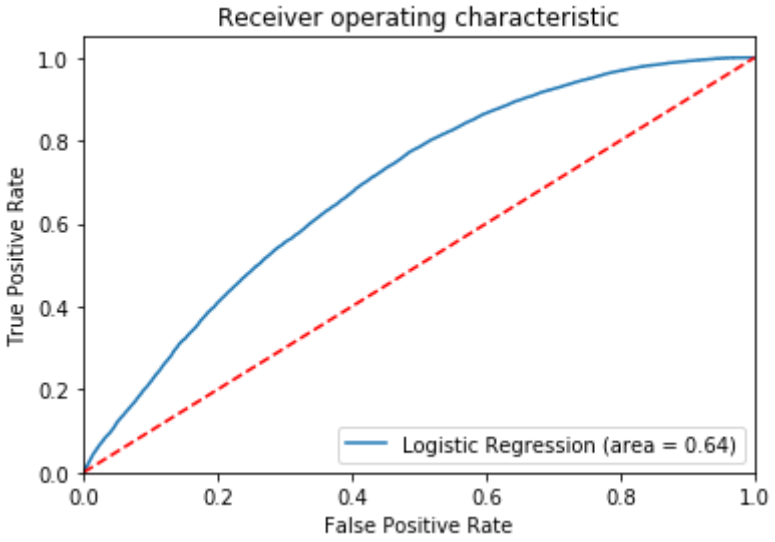
To configure optimal values for these parameters we rely on Grid Search Cross Validation. Grid search is an approach to parameter tuning that will methodically build and evaluate a model for each combination of parameters specified in a grid. It feeds all distinct parameter combinations through the algorithm and then reports back the ones that had the highest f1_score.

In order to check the goodness in our predictions of different set of parameters, we needed to have same test from which we already know the target variable, for this reason we divide the train dataset into another train and test (for which we know the result) thanks to train_test_split.

After all the tuning parameter phase, we discovered that the model that best fit our data was Random Forest Classifier.

Here we show the results obtained with two different models, Logistic Regressor and Random Forest Classifier, that had the worst and the best predictions according to our metrics.

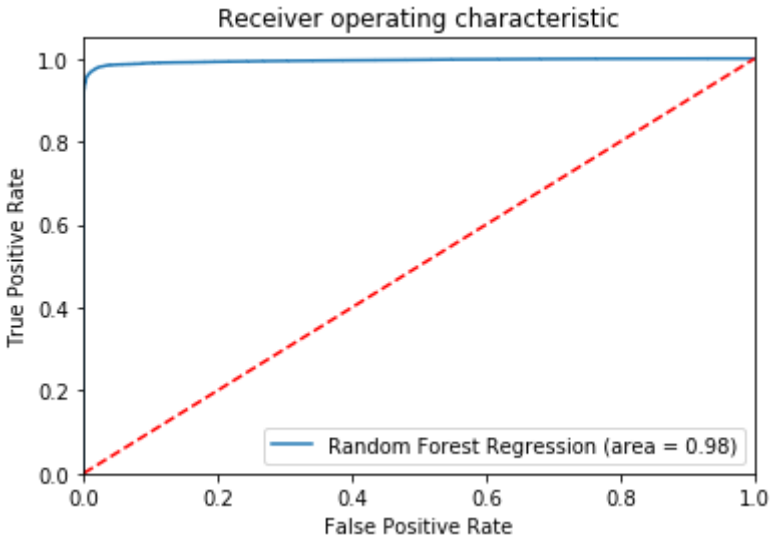
Logistic Regressor



	precision	**recall**	**f1-score**
0.0	0.64	0.62	0.63
1.0	0.63	0.66	0.64

We can see that all the metrics values are around the 0.65, that means that the model poorly performed on our data.

Random Forest Classifier



	precision	recall	f1-score
0.0	0.97	0.99	0.98
1.0	0.99	0.97	0.98

In contrast with the previous model, here we can see that all the metrics had values above the 0.9. That means that the model is fitting data really well. The hyperparameters chosen were n_estimators = 500 and max_depth=7.

4.RESULTS

First we want to show the most important feature we got from our classification algorithm.
In other words we found the features that for our model are the most important in the classification problem and we will show the first ten features that contains the 50% of the information needed to classify correctly the people in the test set:

- AGING_FASCE_LINEA_Oltre 10 anni
- DES_CAUSALE_ATT_LINEA_Attivazione

- FLAG_INTEGRATO_CONVERGENTE_INTEGRATO NON CONVERGENTE
- DES_CAUSALE_ATT_LINEA_*ND
- FASCIA_ETA_SINTESI_da 60 a 70
- DESC_CANALE_CC 187
- DURATA_PROMO_non in promo
- PROMO
- PROMO_ATTIVE_nessuna
- FLG_FATTURE_SCOPERTE_Y

We can say that some of the most important features we find with our algorithm are features that we created from the starting ones with the feature engineering part.

The first most important feature is “AGING_FASCE_LINEA_Oltre 10 anni” that has the 9% of the importance over all the dataset and of course this is correct, because what we wanted to do is to predict the loyalty grade of a customer, and a customer that is with the company for over ten years, is likely to be a really loyal customer. So for us it was a confirm of something we already could imagine that is that a long period of time with the same company is a really good predictor for customer’s loyalty.

For the numerical results and the correctness of the forecast we got some plot of the RocAuc curves, that are the method we knew would be the one chosen for the final evaluation.

UNSUPERVISED ANALYSIS

Once we made the predictions, we decided to clsterize the test dataset into two groups expecting to see some analogies with the predictions obtained with our classifier.
To clusterize the data we used to different models, kmeans and kmeans++, and we could discover some interesting evidence.
We saw that also the distribution of the test dataset grouped in an unbalanced way that was not so distant from the distribution of the 'TARGET' label in the trai dataset. Indeed, we had one cluster with 97% of customers that is really close to the 98% of target-customers with label '0' in the train.

Comparing the distribution of the predictions with label '1' made by our model and the distribution of the cluster with less customers, we discovered that even if the two smaller groups were made more or less by the equal number of customers, they totally differ on which customers were taken by the two smaller groups.