Santander Customer Satisfaction

Arpit Nigam and Weicheng Dong
Department of Computer Science
Viterbi School of Engineering
University of Southern California
Los Angeles, CA 90089
`arpitnig@usc.edu and weichend@usc.edu`

Abstract

Develop a identification system for Santander Bank to help them identify dissatisfied customers early in their relationship. Identifying dissatisfied customers would help Santander Bank to take proactive steps to improve a customer's happiness its too late. From frontline support teams to C-suites, customer satisfaction is a key measure of success. Unhappy customers don't stick around. What's more, unhappy customers rarely voice their dissatisfaction before leaving.

1 Details of Solution

After 66 successful attempts we are able to achieve highest accuracy of 0.841059 %.
For data preprocessing and cleaning we have used MATLAB and mainly in preprocessing we replaced the outliers with the medians of the interquantiles in every feature column.
Our final solution for training and prediction is developed using Python programming language and some external libraries, most important of them being eXtreme gradient boosting (XGBoost)

2 Data and Preprocessing/Cleaning

Data for this problem is taken from Kaggle website and also this problem was posted as competition on Kaggle ”Santander Customer Satisfaction”. There were 3 files in the dataset provided by the Kaggle, first one was the train data file, second was the test data file and third one was the sample submission. Making use of the train file we first performed preprocessing and the trained our model using python code and later used the same trained model on the test file to generate the output file which is similar to the sample submission file with prediction corresponding to every unique customer number.

In our final solution for preprocessing we replaced all the outliers with the mean value from interquantile range in the train as well as the test file. But we tried several methods like taking equal amount of class 0 records and class 1 records to train our model; and converting all the outliers to 0 and even median value but mean values in the interquantile range gave us the best result.

3 Learning Algorithms and models

3.1 Support Vector Machine

We started developing the solution to this problem using Support Vector Machine(SVM) algorithm in MATLAB. In initial attempt we were close to accuracy of 70 % but even after trying a lot of possible pre-processing techniques and parameter values we were not able to improve accuracy beyond 71 %. With 71 % we are standing no where on the leader board and then we decided to move to some other techniques and we arrived at XGBoost algorithm.

3.2 XGBoost

We then switched from MATLAB to python for implementation of our solution using XGBoost. We started with accuracy of 0.837945 % using XGBoost and after several attempt with different parameters we are able to achieve highest accuracy of 0.841059 %

`https://xgboost.readthedocs.io/en/latest/`

XGBoost is an optimized distributed gradient boosting library designed to be highly efficient, flexible and portable. It implements machine learning algorithms under the Gradient Boosting framework. XGBoost provides a parallel tree boosting(also known as GBDT, GBM) that solve many data science problems in a fast and accurate way. The same code runs on major distributed environment(Hadoop, SGE, MPI) and can solve problems beyond billions of examples.

3.3 Principal Component Analysis

In our initial attempts we also tried making use of Principal component analysis in MATLAB for preprocessing and later used the result from the preprocessed file as our train file but we realized that we are loosing information using PCA. We were only able to achieve highest accuracy of around 0.82 % using PCA as one of the preprocessing step and XGBoost algorithm for training the model. So in our final solution we dropped the idea of PCA as it is making the result even worst.

3.4 Linear Regression

We also tried linear regression in our initial attempts using MATLAB but we didn't managed to get any better result than SVM. Using linear regression we were able to obtain the accuracy of around 0.70 % which was almost close to the accuracy of the SVM from initial attempts.

4 Results

We are able to achieve maximum accuracy of 0.841059 % on 65^{th} attempt out of 66 total successful attempts. Highest accuracy is achieved using the preprocessing using MATLAB in which we replaced all the outliers in every feature column with the mean value of the interquantile range and then by using python code to train our model using XGBoost algorithm and the preprocessed train file mentioned above; and later using the same trained model on the preprocessed test file to obtain the final output file with predictions.

5 Steps for executing the code

In our code, we have two main files one for preprocessing which is a MATLAB file and the other one is for training the model and generating the output file based on the trained model. For executing the python file there are some prerequisites which need to be fulfilled beforehand:

- numpy - package for scientific computations using python
- pandas - package for data analysis using python
- xgboost - package for extreme gradient boosting

Once all the prerequisites are met we can follow the steps below to generate the final output file with predictions:

1. Place the train and test file in a location and modify the MATLAB file to read these files from the location

2. Run the MATLAB file to generate the two preprocessed file one for th training data and the other one for the test data.
3. Place the preprocessed train and test data file in some location and modify the python file accordingly to read these files.
4. Run the Python file to generate the output file.

6 Insightful Thoughts

Algorithm in which we have hyper parameters there are always risk of overfitting and underfitting the model. Best values of these hyper parameters can only be found through cross validation.
As the number of hyper parameters increases it even get harder and harder to find the perfect value of these hyper parameters.

7 References

- XGBoost : `https://github.com/dmlc/xgboost`
- Pandas : `http://pandas.pydata.org/`
- Numpy : `http://www.numpy.org/`