Santander Customer Satisfaction, Kaggle

Neel Shah[1] and Prabhanjan Belagodu[2]

[1]University of Southern California, neelshah@usc.edu
[2]University of Southern California, belagodu@usc.edu

May 3, 2016

Abstract

The use of Data Science for building the most appropriate model in solving a real life problem is gaining popularity through competitions held on websites like Kaggle. In this paper, we will try to justify the different algorithms and the accuracies they give with respect to the Santander Customer Satisfaction problem on kaggle.

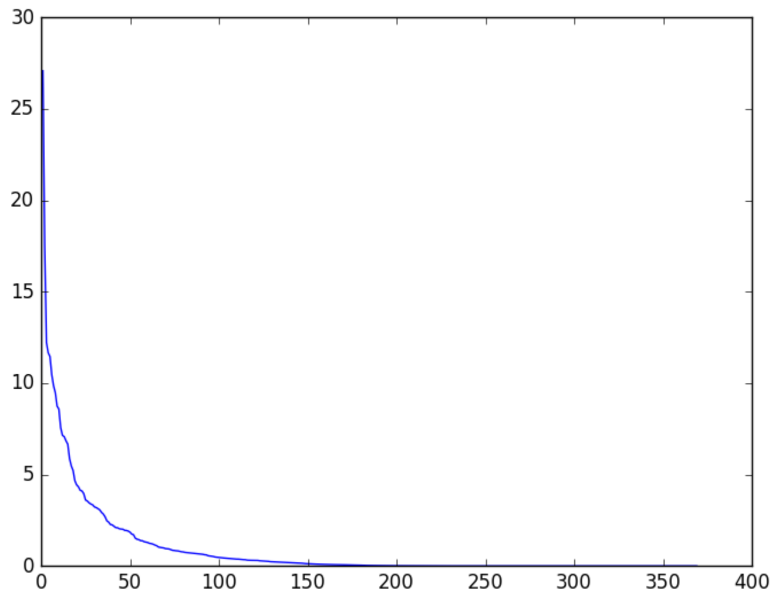
0.1 Keywords for paper submission

`Principle Component Analysis (PCA)`
`Scikit Learn`
`Neural Networks`
`LSTM`
`TensorFlow`
`Random Forest`
`Xgboost`

1 Principle Component Analysis (PCA)

Principle Component Analysis (PCA) [1] is a statistical procedure used to project data onto a lower dimensional linear space in such a way that the variance of the projected data is maximized. It is different from feature selection. Feature selection selects a subset of features from the given features, whereas PCA projects data onto a different dimension altogether. Scikit learn PCA method was used to fit and then transform the train and test data. Before applying PCA the data was centered using scaler method of pCA.

The data consists of 371 features. After analysing the data using python libraries numpy, scipy matplotlib etc., it was observed that many features had the same values for all rows. Also, in order to reduce the number of features for further building the model, PCA was performed and the variance was plotted with respect to each feature as shown below:



The x-axis represents the 371 features and y-axis represents the value of variance.

Thus, after observing the above plot, we took 142 columns into consideration as the variance after that seems to be not changing much.

2 Random Forest

A Random Forest [2] fits a number of decision tree classifiers on various sub-samples of dataset. Scikit learn has a random forest classifier method which was used to fit training data using v=cross validation. The trained model was then used to perform prediction on test data. The accuracy obtained with random forest was 72.3655 %

3 Xgboost

Xgboost [3] stands for eXtreme Gradient Boosting. It is used for supervised learning problems. It used the concept of tree ensemble method. Tree ensemble is a set of Classification and Regreesion Trees (CART). The score of each individual tree based on particular attribute is summed up to get the final score. The accuracy achieved with xgboost is 82.5373 İt gave the maximum accuracy amongst all the above method, random forest and neural networks.

3.1 Effect of Cross Validation

Without applying cross validation [4], the accuracy was 82.2727 %. After applying cross validation method in scikit learn with number of folds as 5, the accuracy was boosted and the final accuracy achieved was 82.5373%.

3.1.1 Model Designing

Cross validation and tuning parameters of XGBoost: We started with a random `n_estimator` value of 567 and later performed cross validation using the below steps to calculate the optimum values for `n_estimator` and `max_depth` and `min_child_weight` parameters.
First we performed 5-fold cv using the cv() method available in the direct xgboost library.(not the XGBClassifier scikit wrapper) This gave us a result of 441 for `n_estimators`. (which we used in one of our 2 submissions that gave us an accuracy of 82.27We further tuned `max_depth` and `min_child_weight` parameters using the GridSearchCV() method of sklearn lib that performs cross-validated grid-search over a parameter grid(for different ranges of `min_child_weight` and `max_depth`). We performed this for `max_depth` range of 3-11 and `min_child_weight` range of 1-7 by constructing a parameter grid and passing it to GridSearchCV(). This grid search takes a large amount of time to run (from hrs to days to run depending in the system). We have included both our submissions(with and without CV) in the folder.

4 Neural Networks

LSTM is a type of recurrent neural network (RNN) architecture which unlike the traditional RNN's is well-suited to learn from experience to classify. We wanted to use a deep learning method using Tensorflow for classification. LSTM was one among the wrongly chosen models for the experiment. The accuracy that we obained using this model with 1 hidden layer was 54.1274%. The accuracy on the training set however was around 96%.
The reason for this dip in the accuracy could be attributed the huge percentage of data belonging to a single class(73000 of 76000 belongs to one class and the remaining belong to another). Overfitting on the training data usually causes a neural network to not perform well as it generally remembers the classification pattern in the training set and then performs the same on the test set. The more a single class outperforms one, the more is the probability that during testing the classification is biased towards the class.

5 Tables

The accuracies obtained with different models have been mentioned in the above table. See Table 1.

Table 1: Model Accuracy

MODEL	ACCURACY
Random Forest	72.3655%
Neural Network	54.1274%
Xgboost (without cross-validation)	82.2727%
Xgboost (with cross-validation)	82.5373%

References

[1] Scikit-Learn PCA documentation [http://scikit-learn.org/stable/modules/generated/sklearn.decomposition.PCA.

[2] Scikit- Learn Random Forest Classifier [http://scikit-learn.org/stable/modules/generated/sklearn.ensemble.
RandomForestClassifier.html]

[3] Xgboost Documentation [https://xgboost.readthedocs.io/en/latest/]

[4] Scikit-Learn Cross Validation Documentation [http://scikit-learn.org/stable/modules/`cross_validation`.html]