Machine Learning Engineer Nanodegree

Capstone Project

Seif El-Eslam Ibrahim Hegazy
seif.eleslam1990@gmail.com
May 30th, 2017

I. Definition

- ### Project Overview

  Customer is the focal point of any business, and his satisfaction is directly proportional to business growth. *Santander Bank*, started in Spain and it has been serving customers in the Northeast since 2013, cares about its customers satisfaction and work to take proactive steps to improve a customer's happiness before it's too late.
  Based on variables that describes the banking experience and actual satisfaction status of given customers, this project aims to classify dissatisfied/ satisfied customers using Supervised learning techniques.

  Input data is CSV file contains anonymized variables and target label.
  This file contains 76020 rows and 371 columns. For academic purposes, a part of these training data will be used as testing data. More precisely, 20% of training data will be set as a testing set.

  Problem link: https://www.kaggle.com/c/santander-customer-satisfaction
  Dataset link: https://www.kaggle.com/c/santander-customer-satisfaction/data

- ### Problem Statement

  Developing a Supervised learning classification model to classify dissatisfied/ satisfied customers.
  Work strategy is:
  - Explore the data-set to get better insight.
  - Prepare the data-set to feed in learners by some preprocessing steps.
  - Develop a base model to compare with.
  - Develop some models and compare their performance to pick the best.
  - Tune the best model and compare its performance with the base model.

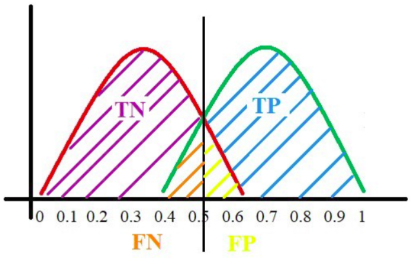
- <u>Metrics</u>

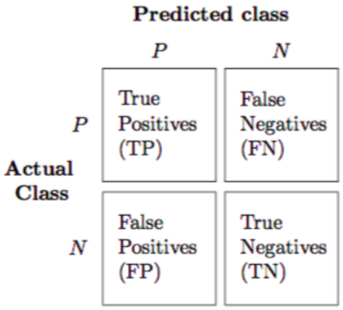
  However using “Accuracy” as metric is easy to understand and makes comparison of the performance of different classifiers trivial, but it ignores many of the factors which should be taken into account when honestly assessing the performance of a classifier. Instead, Area under the ROC Curve (AUC ROC) metric handles this issue. AUC ROC indicates how well the probabilities from the positive classes are separated from the negative classes.

  For any binary classification, the following terms are important:

  - Sensitivity: out of the points that are labeled positive, how many of them are correctly predicted as positive = TP/(TP + FN)

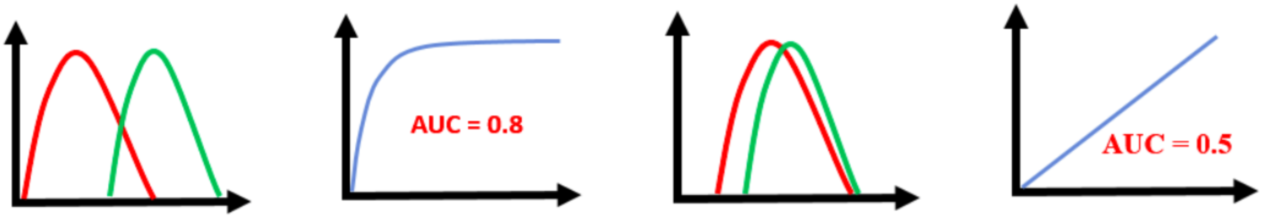
  - Specificity: out of all the points predicted to be positive, how many of them are actually positive = TN/(TN + FP)





		Predicted class	
		P	*N*
Actual Class	*P*	True Positives (TP)	False Negatives (FN)
	N	False Positives (FP)	True Negatives (TN)

ROC curve is obtained by plotting Sensitivity on y-axis aganist (1-Specificity) on x-axis at several thresholds. The area under curve is the model’s performance. The closer it gets to the top left corner, the better the model is doing at distinguishing the two classes apart. Here’s an example:



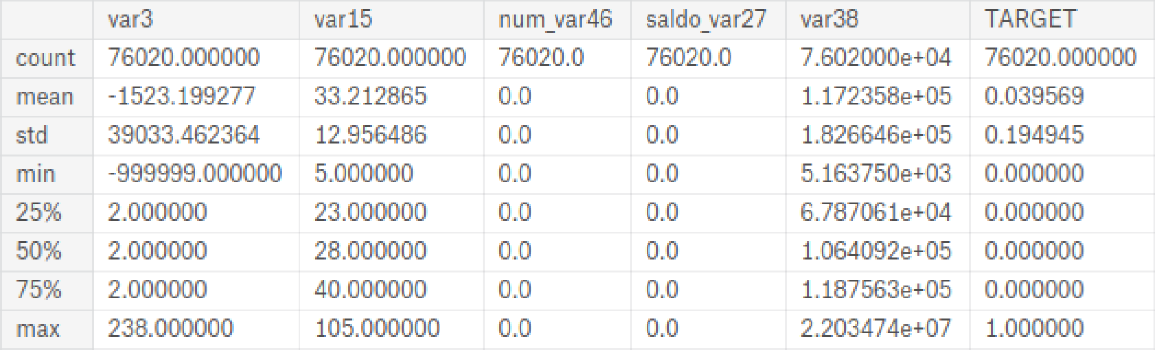
II. Analysis

- <u>Data Exploration</u>

  The data-set contains 369 numeric variables and 1 binary target variable. Traning-set has 76020 records. Testing-set has 75818 records. The target variable represents if a customer is satisfied or not. By exploring this target variable, *there is a huge imbalance between satisfied and unsatisfied customers*. **Out of 73012 customers, only 3008 customers are satisfied**, so

Santander Bank has a big trouble. The data-set has anonymized features with no description or even units such as *‘ind_var2_0’*, *‘num_reemb_var13_hace3’ and ‘var15’*.

Statistics sample for features:
[‘var3’ - ‘var15’ - ‘num_var46’ - ‘saldo_var27’ - ‘var38’ - ‘TARGET’]



	var3	var15	num_var46	saldo_var27	var38	TARGET
count	76020.000000	76020.000000	76020.0	76020.0	7.602000e+04	76020.000000
mean	-1523.199277	33.212865	0.0	0.0	1.172358e+05	0.039569
std	39033.462364	12.956486	0.0	0.0	1.826646e+05	0.194945
min	-999999.000000	5.000000	0.0	0.0	5.163750e+03	0.000000
25%	2.000000	23.000000	0.0	0.0	6.787061e+04	0.000000
50%	2.000000	28.000000	0.0	0.0	1.064092e+05	0.000000
75%	2.000000	40.000000	0.0	0.0	1.187563e+05	0.000000
max	238.000000	105.000000	0.0	0.0	2.203474e+07	1.000000

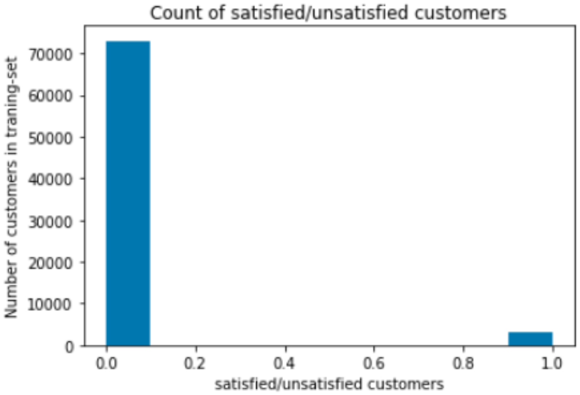
This shows that:

- Some features has values of zero only such as **‘num_var46’** and need to be removed.
- Some features has abnormal distribution such as **‘var38’** and **‘var3’**.
- Some features has skewed distribution such as **‘var15’**.
- Some features has missing values (i.e. -999999) such as **‘var3’.**

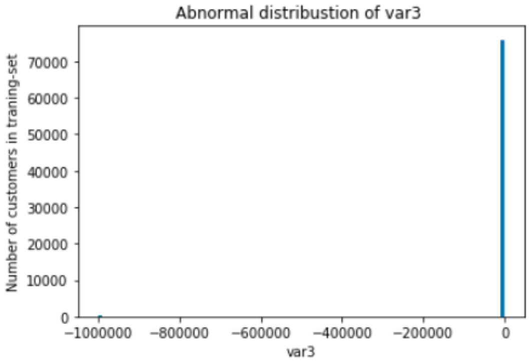
Almost all features has outliers. Outliers amount ranges from 1 to 15983. For example **‘var38’** has 15983 outliers, **‘var15’** has 14196 outliers and **‘saldo_var27’** has 11254 outliers. Removing all outliers will reduce the data-set dramatically. Instead, using algorithms that don’t affect by outliers can handle this issue.

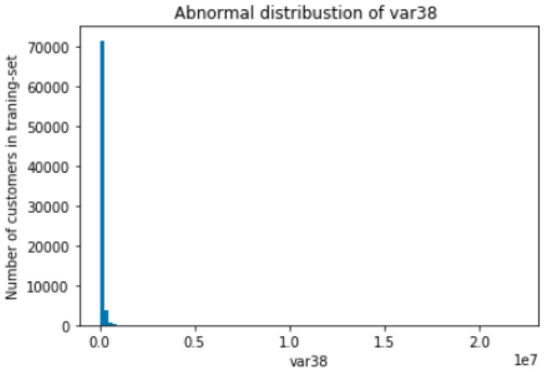
- Exploratory Visualization

In order to summarize the imbalance in ‘TARGET’ variable, a histogram plot can be used. It displays the frequency of each class.
Only 3008 customers are satisfied and 73012 customers are not.

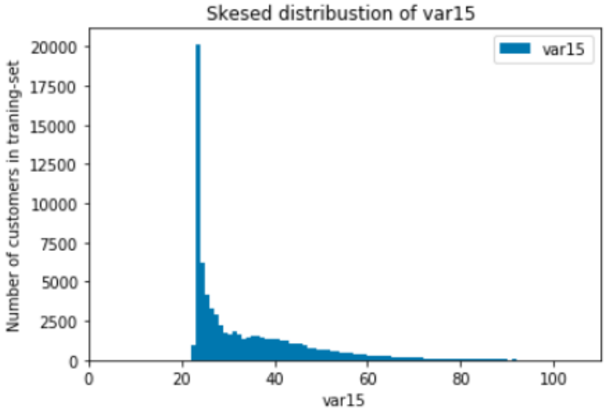


Abnormal distribution of features can be shown using a histogram plot, for example for features ‘var15’ and ‘var3’:

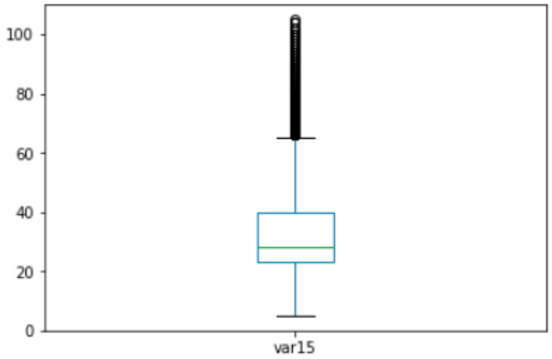

Abnormal distribustion of var3

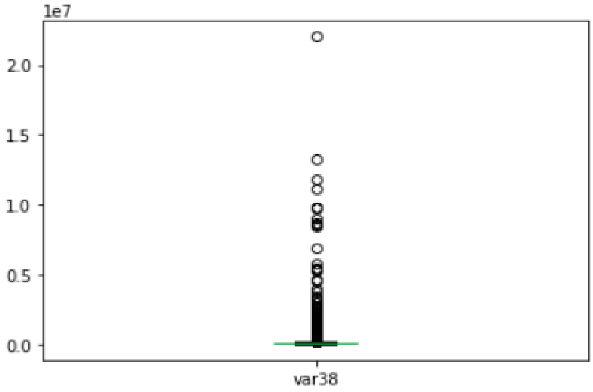

Abnormal distribustion of var38

Abnormal distribution of features can be shown using a histogram plot, for example for features ‘var15’:


Skesed distribustion of var15

Outliers in features can be shown by using box plot. For example, features ‘var15’ and ‘var38’:





Algorithms and Techniques

‘Tree based classification algorithms’ empower predictive models with high accuracy, stability and ease of interpretation. These algorithms are robust to outliers, as the given data-set has high number of outliers, so this type is the best candidate to handle this issue.

Given high number of features (i.e. 369 features), a normal ‘Decision tree’ will overfit the data. Hence, using ‘ensemble methods’ is the best way to handle this high number of features.

Ensemble methods include:

- **Bagging algorithms** where Bootstrap Aggregation or bagging involves taking multiple samples from your training dataset (with replacement) and training a model for each sample. The final output prediction is averaged across the predictions of all of the sub-models. **‘Random Forest’** is an example of this type.

- **Boosting algorithms** creates a sequence of models that attempt to correct the mistakes of the models before them in the sequence. Once created, the models make predictions which may be weighted by their demonstrated accuracy and the results are combined to create a final output prediction. **‘XGBoost’** is an implementation of gradient boosting which is an example of this type.

Benchmark

The benchmark model is **LogisticRegression.** It is trained on unscaled training data, it gets low AUC_ROC score. SO, it is decided to scale the training data to help convergence of the model used for optimization.

Observations:

- Using *unscaled* training data:
  - AUC_ROC score: 0.60253
  - Traning time: 2.0460 sec.
  - Prediction time: 0.01581 sec.

- Using *scaled* training data:
  - AUC_ROC score: 0.7815
  - Traning time: 50.09 sec.
  - Prediction time: 0.00728 sec.

- Using *testing* sample:
  - AUC_ROC score: 0.7894
  - Prediction time: 0.12 sec.

The benchmark score for the testing sample is 0.7894.

III. Methodology

- Data Preprocessing

  1st step is to remove the constant columns that won't affect the prediction. The training data has 35 constant columns.

  2nd step is to handle missing values. It is found that variable 'var3' has 116 missing value. These missing values are replaced with the mode of this variable.

  3rd step is to scale the data. Since the data has wide range, this will affect the models performance. Also, principle component analysis needs scaled data. *StandardScaler* from *sklearn* is used in this step.

- Implementation

  Due to the high dimensionality of the data. So, principle component analysis (PCA) is used to reduce the dimensionality and feature selection.
  It reduced the dimensionality from 369 to 90.
  PCA analysis results:
  - 90 components
  - 0.93% explained variance.

  Scaled/reduced training data is splitted to training/testing sets and the last 15204 rows are reserved as final testing sample.
  A pipeline is build to train, testing and report the auc_roc score. A *RandomForest* model (with n_estimators=100, max_depth=10 as initial parameters) and *XgBoost* model (with default parameters) are passed to pipeline. Then running the models on scaled data and reduced data by PCA to compare the results. Results are:

	Using scaled data			Using reduced data		
	score	**Training time**	**Prediction time**	**score**	**Training time**	**Prediction time**
RandomForest	0.813367	20.97 sec	0.4411 sec	0.803186	39.8147 sec	0.3928 sec
XgBoost	0.82567	30.73 sec	0.0634 sec	0.8181	32.9046 sec	0.0441 sec

Score is decreased due to PCA dimensionality reduction. It’s clear that XgBoost performance is better even on scaled data or reduced data. So, XgBoost is packed for tuning.

- <u>Refinement</u>

Parameter tuning is a dark art in machine learning. Some strategies available for this task is *GridSearchCV* and *RandomizedSearchCV*. The randomized search and the grid search explore exactly the same space of parameters. The result in parameters settings is quite similar, while the run time for randomized search is drastically lower. The performance is slightly worse for the randomized search. *GridSearchCV* will be exhausted search that takes alot of time. Instead, *RandomizedSearchCV* has a probability of 95% of finding a combination of parameters within the 5% optima with only 60 iterations. Also compared to other methods, it doesn't bog down in local optima.

StratifiedKFold cross validation technique is used, as it is generally a better scheme, both in terms of bias and variance, when compared to regular cross-validation. It rearranges the data as to ensure each fold is a good representative of the whole.

Using randomized search, the following parameters of XgBoost are tunned:

- *n_estimators*: Number of boosted trees to fit.
- *min_child_weight*: Minimum sum of instance weight needed in a child.
- *gamma*: Minimum loss reduction required to make a further partition on a leaf node of the tree.
- *subsample*: Subsample ratio of the training instance.
- *colsample_bytree*: Subsample ratio of columns when constructing each tree.
- *max_depth*: Maximum tree depth for base learners.

Using 3 folds, 60 iteration in *RandomizedSearchCV:*

- *Best parameters combination is:*

{'subsample': 0.8, 'n_estimators': 200, 'min_child_weight': 10, 'max_depth': 3, 'gamma': 1.5, 'colsample_bytree': 0.6}

- Best score: 0.8231
- Tuning process takes 30.1 min.

Results comparing with untuned model:
- using scaled data:

- roc_auc score is 0.8271, by 0.0014 increase.

- using reduced data:

- roc_auc score is 0.81297, by 0.00012 increase.

IV. Results

- <u>Model Evaluation and Validation</u>

The final model is the *tuned XgBoost model*. It is derived by tuning the hyper-parameters of the default XgBoost model using *RandomizedSearchCV* by trying 60 parameters combination. It is chosen because its better score that beats the default XgBoost score.

Comparing the hyper-parameters of the two models:

	Tuned model	Un-tuned model
n_estimators	200	100
min_child_weight	10	1
gamma	1.5	0
subsample	0.8	1
colsample_bytree	0.6	1
max_depth	3	3

Robustness check:
The tuned model shows robustness even using un-reduced data with 369 features with testing score of 0.8271. It gets score of 0.81297 using reduced data by PCA.
Even with reducing the dimensionality from 369 to 90 features, score is decreased by 0.01413 only.
So, this model is robust and its results can be trusted.

- <u>Justification</u>

Comparing to the benchmark results:

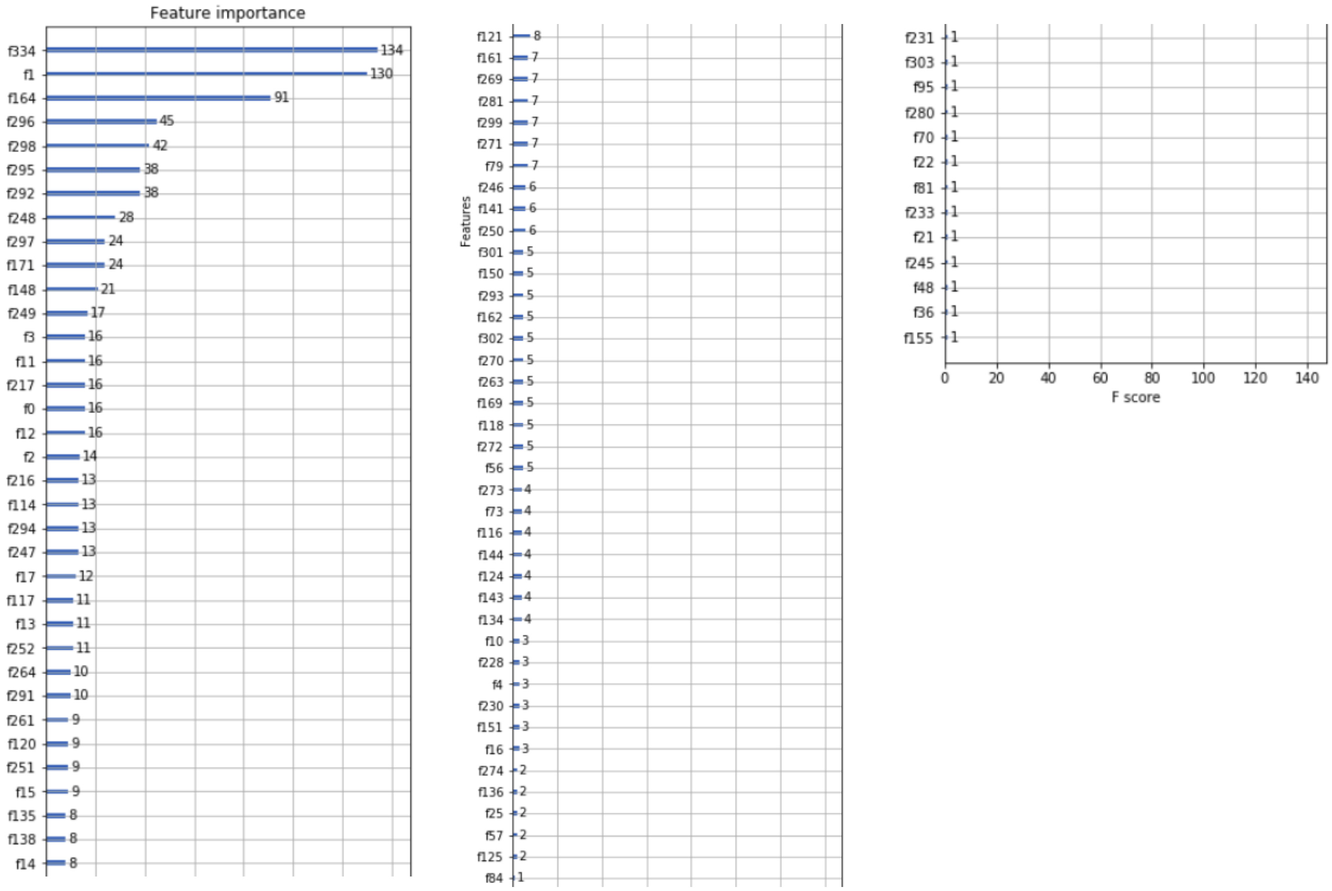
	Final model		Benchmark model	Difference	
	Scaled data	Reduced data		Scaled data	Reduced data
Training score	0.8271	0.81297	0.7815	0.0456	0.03147
Testing score	0.8359	0.81102	0.7894	0.0465	0.02162

Based on this comparison, the final model is better than the benchmark model and it can be deemed as a satisfactory solution.

V. Conclusion

- <u>Free-Form Visualization</u>

According the final model, the feature importance is as follows:



Top 10 important features are:

1. ‘var38’ - feature no. 334
2. ‘var15’ - feature no. 1
3. ‘saldo_var30’ - feature no. 164
4. ‘saldo_medio_var5_hace3’ - feature no. 296
5. ‘saldo_medio_var5_ult3’ - feature no. 298
6. ‘saldo_medio_var5_hace2’ - feature no. 295
7. ‘num_var45_hace3’ - feature no. 292
8. ‘num_var22_ult1’ - feature no. 248
9. ‘saldo_medio_var5_ult1’ - feature no. 297
10. ‘saldo_var42’ - feature no. 171

As the input features are anonymized with no description, the meaning of these features is missing. But at least, we get the top 10 features that affect the customer satisfaction.

- <u>Reflection</u>

This project can summarized as follows:

1. Searching for a real problem that needs a machine learning technique. This project problem is a Kaggle competition that has a business domain.
2. Exploring the data and provide some visualizations.
3. Deciding the algorithms to be used to solve this problem.
4. Preprocessing the data and reduce the dimensionality using PCA.
5. Developing a benchmark model.
6. Applying selected algorithms and visualizing the results.
7. Tuning hyper-parameters of the best algorithm and reporting the test score of best model.
8. Discussing importance of selected features and check the robustness of model.

The 1st point was very interesting. It gives me better insight how the real machine leaning problems are.

The 2nd point wasn’t easy due to the high dimensionality of the data, in addition to the missing meaning of the features. But, using sample of features, it could be get a better insight of the data.

The 3rd point was interesting. The given data-set has some issues such as high dimensionality and high outliers amount. This requires specific

algorithms to handle these issues. *RandomForest* and *XgBoost* algorithms are selected for solving this problem.
The 4th point was important to prepare the data for algorithms.
The 5th point was critical. Defining a good benchmark model was important to get high score that hard to beat. So, *LogisticRegression* is selected as a base model and it gets a good score.
Build a pipeline for training, testing and visualize the results in the 6th point was very helpful.
The 7th point was very important. As any machine learning task required tuning the best model to get the best performance.
The 8th point was very important. Using the final model, the main features that affect the customer satisfaction are determined.

Based on some analysis, it’s found that the final model can be trusted and it can be used to solve these types of problems.

- <u>Improvement</u>

A few points for improvement:

- Performing more aggressive feature engineering.
- Using Grid search can be used instead randomized search to search the parameter space exhaustively and determine the best parameters combination.
- Apply deep learning algorithm using neural networks as it is trending machine learning technique.