

Santander Customer Transaction Prediction

Can you Identify who will make a transaction?

By: Noah Foilb

Analytical Plan

February 13th, 2019, Banco Santander released a big dataset (606.35 MB) on [Kaggle.com/c](https://www.kaggle.com/c/santander-customer-transaction-prediction/overview) (competitions) of over 200 independent columns and 400,000 rows with the purpose of predicting if a customer will make a transaction. They provide the binary [Target] column and a string [ID\_code] alongside many other attributes with randomized names while keeping the same structures as real data for confidentiality purposes. We can see how the data generally looks throughout the whole dataset as we can see in ***Image 1:***

***Imagine 1:*** This is a display of some of the data within the training dataset Santander released. It has 200,000 rows and 202 independent columns and as we can see, the column names and values are insignificant.

Table

Description automatically generated with low confidence

As we can see for confidentiality purposes, we will not be able to use a lot of data analytics throughout the model creation process. We will have to mostly stick to data mining and data science techniques to decipher the deeper meanings behind the data. Another note from this dataset, it comes clean with no NA’s. This shortens the process as we do not have to clean the dataset anymore. As we are no longer able to analyze the data via observing the meaning behind the names of the columns given, we are usually resorted to unsupervised algorithms. Yet while trying to predict a specific target variable, one has to analyze the data using other methods. Such as using a Random Forest to find what the model finds important and use such data.

Looking into the data originally, some concerns are raised from the imbalanced dataset as we can see from the disruption of the target variable in ***Image 1***. This may lead to the creation of invalid models as the null hypothesis will be too accurate for other models to compare. In this case the null model would be 90% accurate. We can fix this imbalance by using a mixture of bootstrapping and duplicating some rows where the target value equals 1 and deleting some rows where the target value equals 0. Once the dataset has been balanced and cleaned, then can the creation of the predictive models begin.

As by default, we will be trying to create a prediction model as we are trying to predict customer transactions. The objective is to create and compare three supervised predictive classifiers.

1. Stochastic Gradient Descent
2. Kernel Approximation
3. Random Forest

Stochastic Gradient Descent was chosen as not only does it support loss functions and penalties for classifications but also it is a very credible classifier. Kernel Approximation was also chosen as they are well known as a last resort if the SGD does not turn out to be a good classifier. Lastly, random forest was chosen as a backup. Random Forests tend to be a valid solution to many predictive models as there is a lot of automation behind the scenes that don’t require human intervention. Once the creation and evaluation of these optimized classifiers are complete, these will be evaluated based on their performance via k- fold cross-validation and the ROC/AUC curve.

The expected outcome will be a set of probabilities and predictions. After evaluating the convergence of such models, if the situation arises where an ensemble method may be used to optimize the models, it will be done. The final predicted values of these classifiers will consist of the probabilities and predictions per entrée from the test dataset and be exported into its own excel sheet. This is to be exported into tableau for presentation purposes to clearly display the data to the group.

Preliminary Results

Review of preliminary outcomes: To predict if a customer makes a transaction, we will create three models: Stochastic Gradient Descent, Kernel, and Random Forest, with the potential of using these models in an ensemble method later. After creating statistically significant models, we will compare them via ROC/AUC curves and k-cross fold validations.

Initial Results: In terms of data cleaning, the data did not have to be cleaned as it had received a clean version beforehand. Yet when looking into the data, the target variable was significantly imbalanced as there were more people who did not make a transaction when compared to those who did, in which one had to address. To address this, oversampling those who make a transaction using the SMOTE function from the imblearn package in Python. This SMOTE function isn’t a simple copy paste of other rows but rather they take other potential values based on what information is given. We can see how this works in ***Image 2*** and ***Image 3.***

***CODE 1:*** This is a display of the SMOTE command that was used to fix the Imbalanced data.



Chart

Description automatically generated***Imagine 2:*** This is what the imbalanced data would look like before SMOTEChart, scatter chart

Description automatically generated

***Imagine 3:*** This is what the data would look like after correction by SMOTE.

Once the data had been fixed, analyze the data was the next step. As stated before, this data has no meaningful labels in term of its predictors, so one has to find ways around this dilemma. Using a sample of the data to find the importance of each variable in relation to target variable as we can see in ***Image 4*** by using this sample in a random forest model. Random forest models are created by prioritizing one variable over another to create the best model. By doing this we can see which variables explain the variance of the model when looking at new data, hence variable importance.

***Imagine 4:*** Variable Importance

Chart, bar chart, histogram

Description automatically generated

By only selecting an optimal threshold of importance, only 20 attributes were left for the models. After finding the best predictors, we can finally create our first model. As we have a big dataset, one of the best models to run is a Stochastic Gradient Descent Classifier from the Python library Sklearn. As this model runs fast on Big Data, we were able to analyze the results and optimize the threshold based off the resulting ROC/AUC curve. This model’s confusion matrix can be seen in ***Confusion 1***. After this creation of a poly kernel and random forest, which created the biggest issue when creating these models. Big Data takes long to process. To process this data each respectively took over 30 minutes, which made one question if these models should be included. After running each model once, by kepping the models and ran their predictions which we can see in ***Confusion 2*** and ***Confusion 3***, respectively.

Graphical user interface, text, application

Description automatically generated***Confusion 1***

***Text, letter

Description automatically generated***

***Confusion 2***

***Text

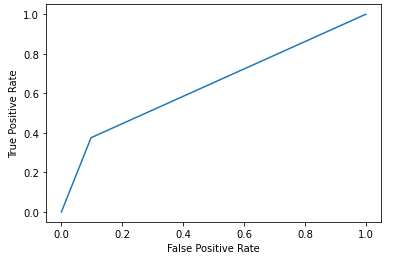
Description automatically generated***

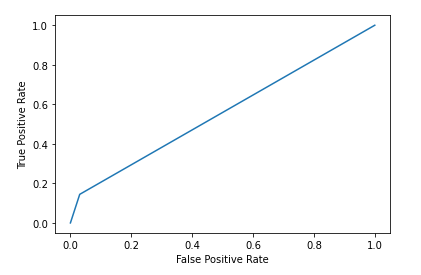
***Confusion 3***

Since the Kernel and Random Forest took so long to run, running a k-fold cross validation would take kx as long as a normal runtime which makes validating them suboptimal. In this preliminary round, by only taking a 3-fold cross validation from the SGD Classifier. This validation was run among the accuracy score, in which we got 88%, 90% and 90% which is great! This model is not only now statistically significant but also usable. After evaluation, the next step was display and analyzing the ROC Curves. As we can see the SGD classifier is the better model when compared to the others. We can tell this as the SGD Classifier ROC curve is the closest curve to reach the top left corner.

***Chart, line chart

Description automatically generatedSGD Classifier***

***Key***

***Random Forest***

***Poly Kernel***

Final Analysis: As we can see each model has their benefits and drawbacks but contrary to what the models would have suggested one would believe that the Stochastic Gradient Descent Classifier is the most realistic, simply because it can accurately predict the most True Positives. This leads to a higher balanced accuracy and while the recall is low, it is believed that to be potential for Santander. This model predicts a lot more people should be making transactions, so if Santander gives these customers a push in terms of advertisement or deals, they may make a transaction. The objective of testing these models was to predict accurately which Santander customers would make a transaction. After these preliminary results, the SGD Classifier was the best model as not only was the mis-classification error for 0 was low but also gave potential growth for the company as there are many who should be making a transaction but are not. Those with high probability should be encouraged to make a transaction as there is so much more potential in these clients.

Final Results

*Executive Summary*

The key findings from this these models were of different magnitudes. The first key finding was that more data and context behind the data would help create and optimize models in the future. Maybe talking with Santander bank may be a future step if pursuit of creating better models is of interest. The second key finding is from the models themselves. Their balanced accuracy was not ideal, butt they gave valid information. The best model, the Stochastic Gradient Descent Classifier (SGD Classifier) was the best model with an extremely high false positive classification rate and an okay true positive classification rate. This information shows that Santander is missing out on potential clients who could be making transactions. Based on this information, more pursuit towards these clients is recommended and procedure is shown below of how this information was derived from the data and models.

*Data & Approach*

An overview of the data would show one key takeaway, its all randomized and masked. When Santander created this dataset, they did not want to release any private information, so they randomized and made the data untraceable. The fact that the data was untraceable made analyzing the data extremely difficult and was the biggest obstacle in this project. The goal of this project was to create a predictive classification model that would provide insight on if a customer would make a transaction with Santander bank. The approach was to analyze the correlations, find meaningful information, fixed unbalanced data using the SMOTE function **Image 2,3 (Other section)** and to find variable importance from a random forest model. After this, putting these predictors into a Stochastic Gradient Descent Classifier will create the final model that will be used for our analysis of the data.

*Detailed Findings*

The first step was to analyze the data. In which was done from the Random Forest model. We can see each features importance from **Image 4.**

***Imagine 4:*** Variable Importance

Chart, bar chart, histogram

Description automatically generated

Each feature was then filtered for those of variable importance less than .0055 which led to only keeping 50 random and meaningless features. After that, the next step was to separate the data, create an A/B testing split (75/25), and then train the SGD Classifier. As the SGD Classifier is optimized for big data, the model was trained in under one minute. This model created a confusion matrix as can be seen in ***Confusion 1.***

***Confusion 1***

Graphical user interface, text, application

Description automatically generated

As seen in the confusion matrix, this model has an extremely accurate True Negative classification rate and an okay True Positive classification rate.

*Validity & Reliability Assessment*

The metrics used to validate and show the reliability of this was the combination of ROC/AUC curves and k-cross validation. The cross validation when ran for three folds had a relatively consistent accuracy value of 90% which was excellent. Being able to stay consistent under bootstrapped values is an excellent way to show consistency and reusability. For reliability, ROC/AUC curve was formed when comparing the SGD Classifier to the other potential models, as can be seen in ROC/AUC curve.

**SGD Classifier**

***Chart, line chart

Description automatically generated***

As seen, this model does well above randomly guessing, showing that this model has some actual potential in a practical sense!