Telecom Churn Classification

Havan Patel
Supervised Machine Learning
DATA 55000-002
Lewis University
Romeoville, US
havangpatel@lewisu.edu

***Abstract*— Customer churn occurs when a customer or a subscriber stop doing business or unsubscribes to their service. In this report we will try to predict if a customer will remain or part ways with your subscription. By knowing or getting an idea of it, business can strategize a new idea to retain the customers. This will only benefit the company to improve their service and the customers with better experience when they use their services.**

I. Introduction

In this report we will take look at the 'Telecom Churn' dataset from Kaggle[1] which contains data from the 'Orange Telecom's Churn Dataset' which is a French multinational telecommunication corporation.[2] This dataset contains all the activity of the customers. There contain two dataset files that are available: 'Churn-80' for training and 'Churn-20' for testing purposes. However – we will combine these two files and make it into one file because we want to clean up the data and want to apply our own training and testing data from the dataset.

I performed 4 different classification techniques on the dataset: Logistic regression, Decision Tree, Random Forest, and Support Vector machine (SVM). But we will only look at two models, Decision Tree, and Logistic Regression as they are different techniques, and they contrast each other. There is a python notebook included with the report which contains the code for all four techniques. The main reason for me to choose these two techniques were the contrasting result and accuracy on same dataset.

II. DATA CLEANING

The first step in the process of classification was to understand the data and apply some preprocessing techniques on the dataset to give us more accurate correctness in our results. After combining the two files into one dataset we had 3333 entries and 20 features, including the target feature. There were series of steps taken to better understand the data. Below are the steps taken:

A. Removing NULL values

The dataset itself came with cleaned up data and I didn't have to perform any extra step to remove null values. Discarding null values can be good and bad, because you will have loss of information and this step really depends on what you are trying to achieve.

B. Mapping the target variable to 0 (False) or 1 (True)

The target variable contains Boolean data type, and it is best to convert the true and false values to 1 or 0. One of the reasons we do this is so that the program doesn't have to do conversion so that way converting them beforehand can help with performance. For our case 0 represents customer who will not cancel the plan or not churn and 1 represents customer who will cancel the plan or will churn.

C. Finding Corelation between the variables.

This step is very important because there are a lot of data and we do not always need to use them all because they might overlap with other columns or would be irrelevant to what we want to do. Therefore, I used the correlation technique shown in figure 1 to find the relation between the features to remove them as they will contain or cover most of the data. Several of the numerical data are very correlated. (Total day minutes and Total day charge), (Total eve minutes and Total eve charge), (Total night minutes and Total night charge) and lastly (Total intl minutes and Total intl charge) are also correlated. We would only have to select one of them. So, then I pick only one column from the set and drop the rest of the columns so that way our dimensions are reduced further.

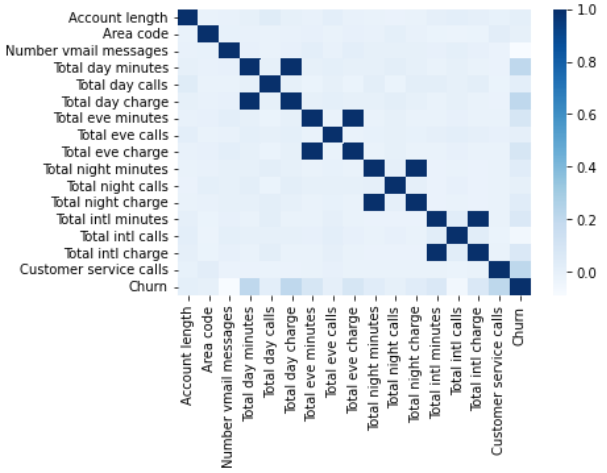

Fig 1. Correlation between features

D. One Hot Encoding

One hot encoding is a technique which converts categorical data variables so they can be provided to machine learning algorithms to improve predictions. One hot encoding is a crucial part of feature engineering for machine learning. One hot encoding makes our training data more useful and expressive, and it can be rescaled easily. By doing this we will have more features which is not what we would like to have.

E. Shuffle dataset and create training and testing dataset

After merging the files, it is better to randomize the data so that we can have good variety in the training and testing data. We split the dataset into 80% training and 20% testing.

F. Feature Importance

I do repeat one more step of finding the feature importance by using the Random Forest Classifier technique

and below is the result of the columns I chose from the correlation from step C above and its score. These are the columns that will be used for our report and findings shown in figure 2.

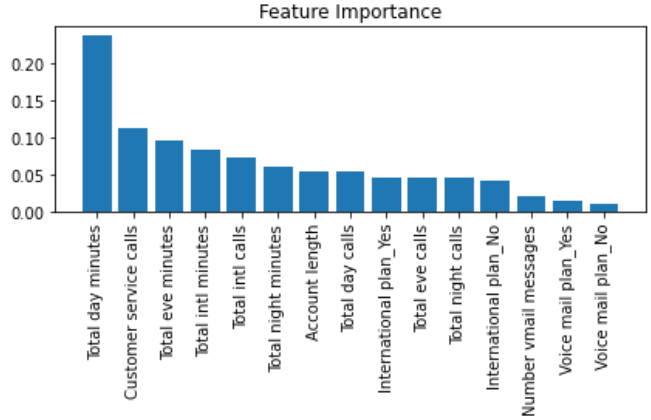

Figure 2. Feature Importance

G. *Balancing our training models*

In figure 2 we can see that our data is not very balanced for out target variable. This imbalance can mess up our analysis. So, to avoid this I used the SMOTETomek technique [3]. SMOTETomek generates examples based on the distance of each data (usually using Euclidean distance) and the minority class nearest neighbors, so the generated examples are different from the original minority class.

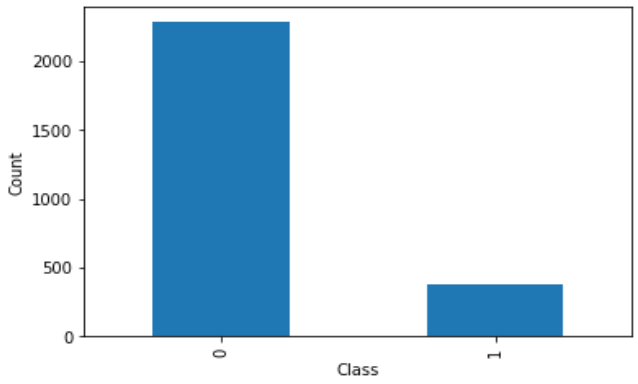

Figure 3. Imbalanced dataset on target variable

After applying this technique, we can see out data is balance and much usable now as seen in figure 4.

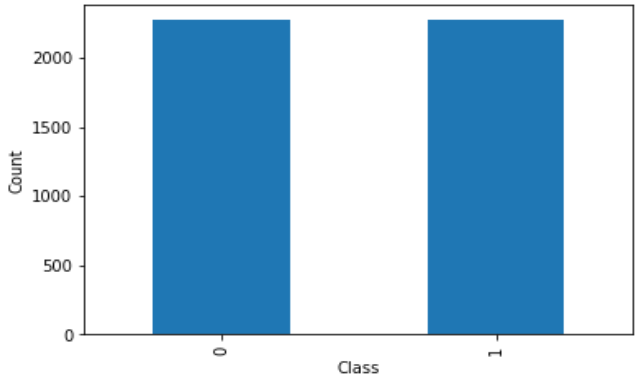

Figure 4. Applying SMOTETomek

III. Decision Tree vs Logistic Regression Classification

Below we will look at how these two different techniques performs by looking at their accuracy scores.

A. *Decision Tree*

Decision Tree (DTs) are a supervised machine learning technique that predicts value of responses by learning decision rules derived from the features. They are also often used for both regression and classification context.

For python we have sklearn libraries that provides the DT imports called DecisionTreeClassifier which has different parameters. One such parameter is called max_depth that controls the size of the tree to prevent from overfitting. Larger the tree depth the more accurate but the slower it is. The best accuracy result I got was when I set the max_depth to 7 after trying out different values. We can then predict our model after doing our training on DT.

	precision	recall	f1-score	support
0	0.97	0.97	0.97	566
1	0.85	0.83	0.84	101
accuracy			0.95	667
macro avg	0.91	0.90	0.91	667
weighted avg	0.95	0.95	0.95	667

Figure 5. DT Report

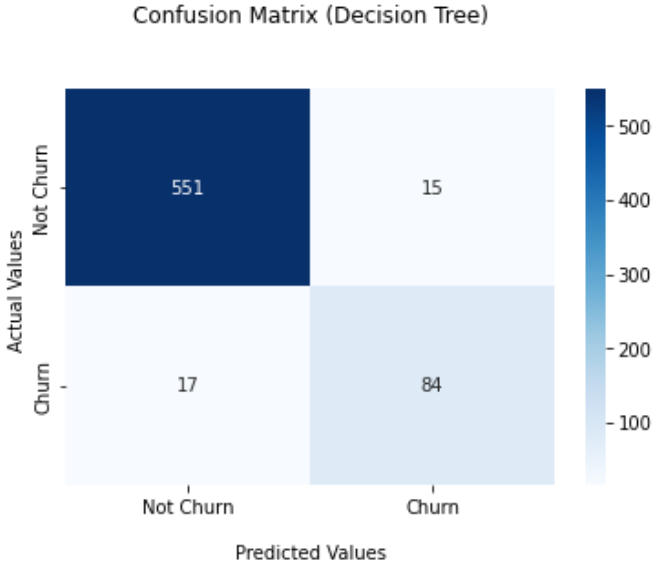

Figure 6. DT Confusion Matrix

Let's analyze figure 5, where we see that the precision score of 85%, which tells us that how precise the classifier id when predicting churned or positive cases. The recall values are 83% showing that how frequently the expected values are correct if the real values are positive.

Now let's look at the confusion matrix of DT. There are 551 instances where classifier accurately predicted that the customers did not churn or known as true negative. 15 instances where the classifier incorrectly predicted that customers churned but, they did not (False Positive). 17 instances where the classifier incorrectly predicted that customer did not churn but they did or called false negative. Which also known as false negative, Lastly, 84 instances

where the classifier accurately predicted that the customers churned or called true positive.

For this dataset and for the business model, it is more fitting to accept false negative then false positive because we do not want to lie and create a model for our business that will hurt both the business and customers. With false positive you are very prone to losing customer due to lying to them with false information. The goal for the business to retain old customer, bring in new customers and generate more revenue.

B. Logistic Regression

Logistic regression estimates the probability of an event occurring, such as voted or didn't vote, based on a given dataset of independent variables. Since the outcome is a probability, the dependent variable is bounded between 0 and 1.

I used the library from the sklearn to predict the logistic regression model. There are a lot of parameter options for logistic regression, but I didn't use any special type of inputs. After training the dataset I used the predict method to generate the report and confusion matrix shown in Figure 7 and 8.

	precision	recall	f1-score	support
0	0.93	0.85	0.89	566
1	0.44	0.65	0.53	101
accuracy			0.82	667
macro avg	0.69	0.75	0.71	667
weighted avg	0.86	0.82	0.83	667

Figure 7. LR Report

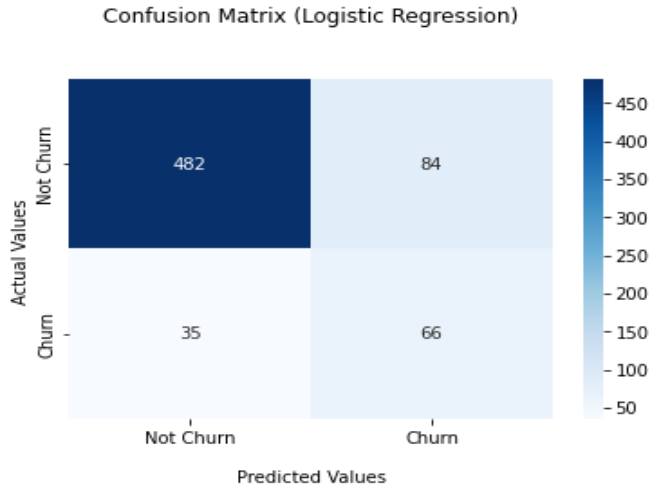

Figure 8, LR Confusion Matrix

From the previous model we now know how to interpret the information from the above two figures. Let's create a table for the report and see which model did better and which model we should use for our business.

Table I. Comparison of Confusion Matrix

	Decision Tree	Logistic Regression
True Negative	551	482
False Positive	15	84
False Negative	17	35
True Positive	84	66

From table 1. We can clearly see that decision tree has performed way better then logistic regression. With DT we had FP of 15 instance and FN of 17 instanced compared to LR with FP of 84 and FN of 35 instances. As we discussed that we wanted to decrease our score on false negative or Type II error and decision tree very efficiently does it. With that we also get boost in the TP and TN score as well in decision tree compared to logistic regression. If we look at the accuracy percentage for both we can clearly see decision tree with 95% accuracy clearly beats logistic regression's accuracy at 82%.

Final way of comparing the model is looking at the receiver operating characteristics curve or ROC. Below figures contains the ROC curve for both models.

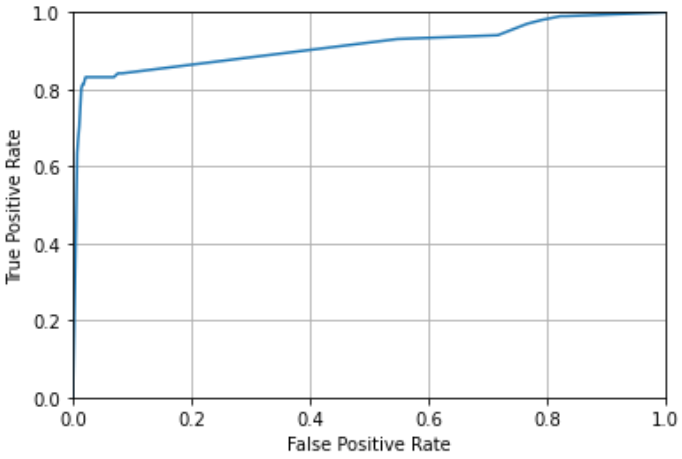

Figure 9, Decision Tree ROC Curve

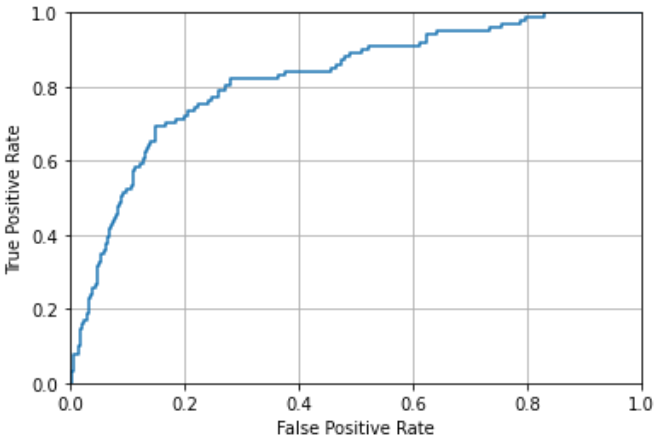

Figure 10, Logistic Regression ROC Curve

Let's analyze the above ROC curves for both models. It is very straightforward to understand. It is constructed by plotting the true positive rate (TPR) against the false positive rate (FPR). It is essentially mapping various possible outcome thresholds and showing the area under the curve. For DT the area under the curve was 91% and for LR it was 82%. These percentages tell us about the percentage for the model to be able to distinguish between TPR and FPR. If you have greater area under the curve the better the model classifier. Therefore, again we prove that decision tree is better fit for this dataset.

Conclusion

In this report we looked at step by step process of training and testing our dataset by using different techniques. We started with applying preprocessing to our dataset. Luckily this dataset was already cleaned, and we didn't have to perform extra steps to clean it but that is not always the case. As data grows, we have some unwanted values that we would have to filter out for our models to predict the results better. Then we also must balance out our data and to reduce the dimension of our data for the machine and for us humans to understand and analyze the data. We applied two techniques such as correlation to remove redundant columns and random forest classifier to score the feature importance on out selected features. After all that we then train out model on different algorithms and produced different metrics on the target feature to see which model gave us the best results. And we saw that decision tree with max depth of 7 gave an accuracy or 95%.

References

[1] BALIGH MASSRI, "Telecom Churn Dataset" Kaggle, 5 July 2019, https://www.kaggle.com/datasets/mnassrib/telecom-churn-datasets?resource=download

[2] "Orange S.A." https://en.wikipedia.org/wiki/Orange_S.A.

[3] Raden Aurelius Andhika Viadinugroho "Imbalanced Classification in Python: SMOTE-Tomek Links Method" 18 April 2018 https://towardsdatascience.com/imbalanced-classification-in-python-smote-tomek-links-method-6e48dfe69bbc