Predicting Customer Churn In Telecom To Keep Them From Opting-Out of Service

Havan Patel
havangpatel@lewisu.edu
DATA-51000-002, Fall-2022
Data Mining and Analytics
Lewis University

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.

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.

We 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 three models, Decision Tree, Logistic Regression and Random Forest 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.

The future sections of this report shall describe the preprocessing of data, three different supervised learning techniques, results of both methods and conclusion. In section II we will process of data cleaning and analysis to better understand. Section III will describe how the dataset was used after preprocessing. In section IV we will describe the techniques we utilize. In Section V, the result obtained from the techniques are discussed. Finally, section V Provides the conclusion.

II. Data Description

In the telecom industry, customers are able to choose from multiple service providers and actively switch from one operator to another. "In this highly competitive market, the telecommunications industry experiences an average of 15-25% annual churn rate."[3] After doing the data cleaning and removing some features that were overlapping, we are left with the following attributes mentioned in the Table 1 below.

TABLE I. Dataset attributes

Attribute	Type	Example Value	Descripton
Total day minute	Numeric (integer)	100	Day time call mins.
Total day calls	Numeric (integer)	4	Total Day time calls
Total eve minutes	Numeric (integer)	200	Evening time call mins
Total eve calls	Numeric (integer)	66	Total evening calls
Total night Minutes	Numeric (integer)	150	Total night call mins.
Total night calls	Numeric (integer)	20	Total night calls
Total intl minutes	Numeric (integer)	15	Total international call mins
Total intl calls	Numeric (integer)	5	Total international calls
Customer service calls	Numeric (integer)	3	Total calls made to customer service
International plan	Boolean	Yes	Does customer have international plan
Churn	Numeric (integer)	0	Did customer churn

We see in fig 1 shown below that how many people have churned if they have international plan and how many churned if they didn't even have any international plan. If we look at people with international plan, we see the churn rate and not churning rate is very close to each other and on the other hand people without international plan we see clear gap of people churning and not churning. This makes sense because if you don't have international plan, you are most likely to stay with one carrier and if you do have it then you will churn based on best price you get.

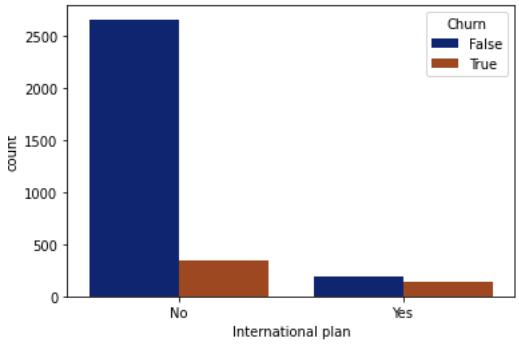

Fig 1. How many people churned with International Plan

III. DATA PREPROCESSING

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 we 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. 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.

D. 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.

E. Balancing our training models

In figure 2 we can see that our data is not very balanced for our target variable. This imbalance can mess up our analysis. So, to avoid this we used the SMOTE Tomek technique [3]. SMOTE Tomek 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.

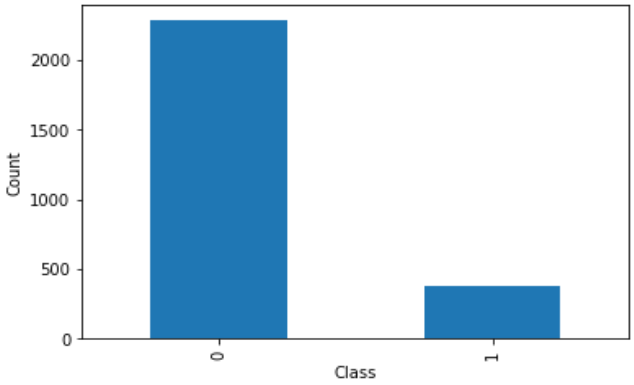

Fig 2. Imbalanced dataset on target variable

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

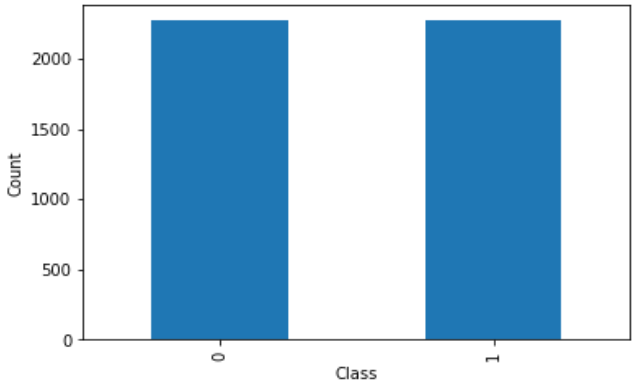

Fig.3 After applying SMOTETomek

IV. METHODOLOGY

The below figure shows the overall procedure adopted by us for this report.

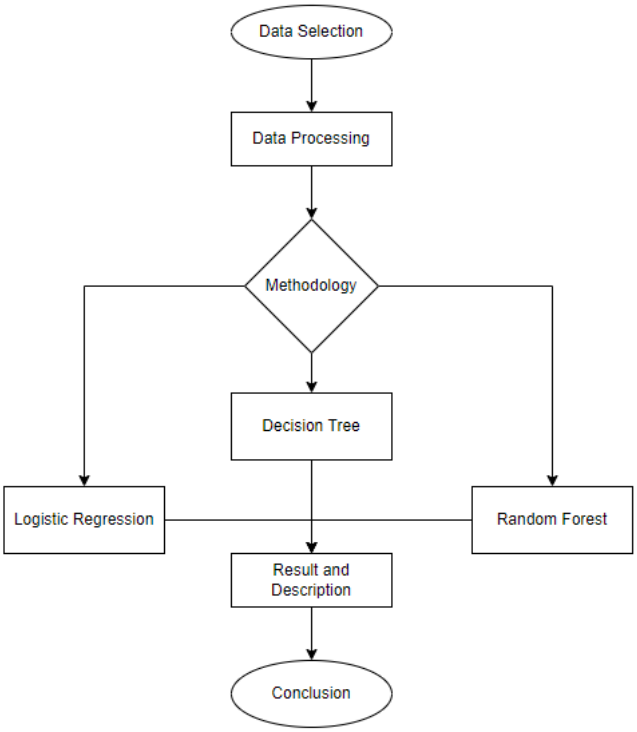

Fig.4 Flow chart of the prediction task

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 we got was when we set the max_depth to 7 after trying out different values. This made sense because the deep the tree rules are the better accuracy but very performance intensive because it has to go through all the conditions. We can then predict our model after doing our training on DT.

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.

We used the library from the sklearn to predict the logistic regression model. There are a lot of parameter options for logistic regression, but we didn't use any special type of inputs. Random Forest

C. Random Forest

The Random Forest creates a set of decision trees from a randomly selected subset of the training set. It is basically a set of decision trees (DT) from a randomly selected subset of the training set and then it collects the votes from different decision trees to decide the final prediction. We used the library from the sklearn to predict the random forest model.

V. Results and Discussion

After training the model on training data (80%), the models were applied on the testing data(20%) and the accuracy and confusion matrices were shown below

TABLE II. Decision tree Metrics

	Precision	**Recall**	**F1-score**	**support**
0	0.97	0.97	0.97	556
1	0.85	0.83	0.84	101
Accuracy	0.95			667

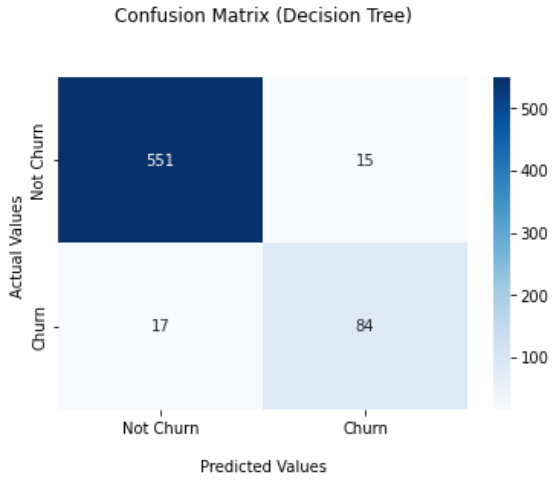

Fig 5. Decision Tree Confusion Matrix

Let's analyze table II, 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 from fig. 5. 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.

After training Logistic regression model on training data we used the predict method to generate the report and confusion matrix on testing data and shown in table 3 and figure 6.

TABLE III. Logistic regression metrics

	Precision	**Recall**	**F1-score**	**support**
0	0.93	0.85	0.89	556
1	0.44	0.65	0.53	101
Accuracy	0.82			667

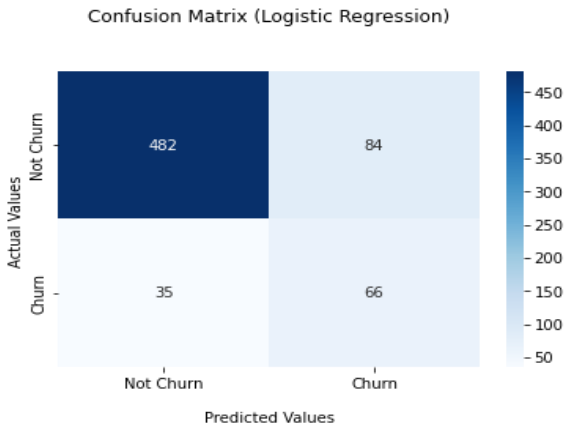

Fig 6. Logistic Regression Confusion Matrix

Let's analyze table 3, where we see that the precision score of 44%, which tells us that how precise the classifier id when predicting churned or positive cases. The recall values are 65% showing that how frequently the expected values are correct if the real values are positive.

Now let's look at the confusion matrix of LR from fig. 6. There are 482 instances where classifier accurately predicted that the customers did not churn or known as true negative. 84 instances where the classifier incorrectly predicted that customers churned but, they did not (False Positive). 35 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, 66 instances where the classifier accurately predicted that the customers churned or called true positive

After training Random Forest model on training data we used the predict method to generate the report and confusion matrix on testing data and shown in table 4 and figure7 .

TABLE IV. RANDOM FOREST METRICS

	Precision	**Recall**	**F1-score**	**support**
0	0.96	0.98	0.97	556
1	0.86	0.79	0.82	101
Accuracy	0.95			667

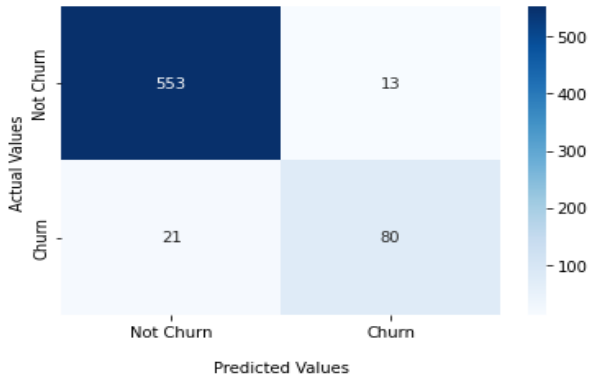

Fig 6. Random Forest Confusion Matrix

Let's analyze table 4, where we see that the precision score of 86%, which tells us that how precise the classifier id when predicting churned or positive cases. The recall values are 79% showing that how frequently the expected values are correct if the real values are positive.

Now let's look at the confusion matrix of Random Forest from fig. 7. There are 553 instances where classifier accurately predicted that the customers did not churn or known as true negative. 13 instances where the classifier incorrectly predicted that customers churned but, they did not (False Positive). 21 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, 80 instances where the classifier accurately predicted that the customers churned or called true positive

Let's create a table of all the information we obtained from modeling our 3 different techniques for the report and see which model did better and which model we should use for our business.

TABLE V. COMPARISON OF CONFUSION MATRIX

	Decision Tree (DT)	Logistic Regression (LR)	Random Forest (RF)
True Negative	551	482	553
False Positive	15	84	13
False Negative	17	35	21
True Positive	84	66	80

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

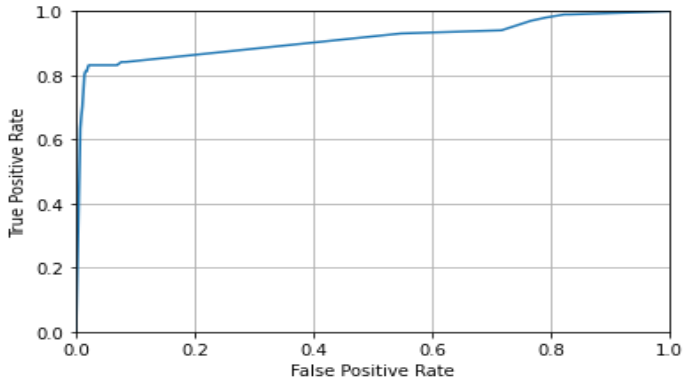

Fig.8 , Decision Tree ROC Curve

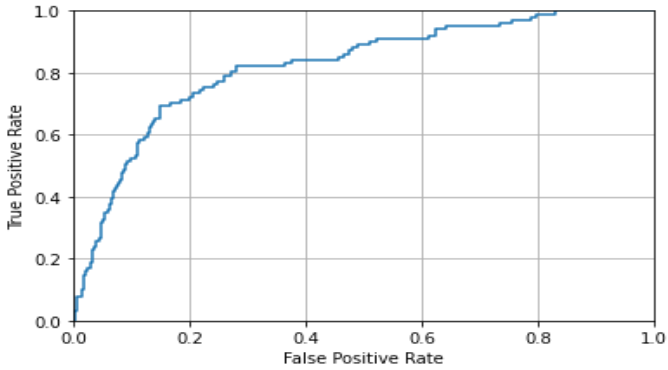

Fig.9 Logistic Regression ROC Curve

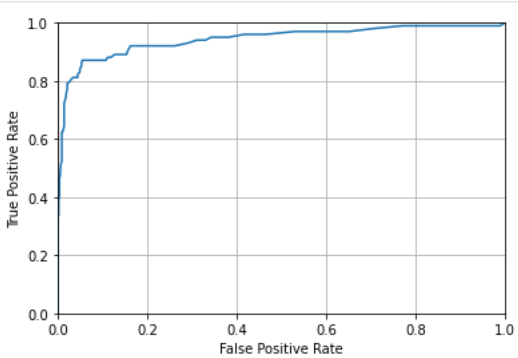

Fig.10 Random Forest ROC Curve

Let's analyze the above ROC curves for all 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% and for RF the area under the curve was 94%. 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.

Precision is a metric that is used to lower the false positive errors. It can be seen as a measure of quality. It refers to False Positive and tells us that in there are X number of people where it incorrectly predicted that a customer opted out from the service but in fact they never did.

When we analyze the metrics of different models, it is not all always necessary that if we see score that are high mean that it is best fit and accurate for our model, or it will produce the best result. We need to first define what is our goal and based on that we should use metrics and parameter that makes the most sense.

VI. Conclusions

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. we then trained model on different algorithms (Decision Tree, Logistic Regression, and Random Forest) 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 and random forest gave an accuracy or 95%. Therefore, we can conclude that random forest best suites this dataset and to do our analysis and generate report using that technique.

From table V. We can clearly see that random forest has performed way better then logistic regression and decision tree by slightest. With RF we had FP of 13 instance and FN of 21 instanced compared to LR with FP of 84 and FN of 35 instances and DT with FP of 13 and FN of 17. As we discussed that we wanted to decrease our score on false negative or Type II error and Random Forest model very efficiently does it compared to LR. With that we also get boost in the TP and TN score as well in decision tree compared and Random Forest to logistic regression. If we look at the accuracy percentage for all models, we can clearly see Decision Tree and Random Forest with 95% accuracy which clearly beats Logistic Regression's accuracy at 82%.

Using the models, we will be able to predict if the customer is likely to opt out or stay. If the result is true then the customer will churn, and we can look at different features to analyze why they left the service. Based on this information maybe we can work on our business model to improve them to retain them. Maybe they can work on providing service for better price or for same price maybe increase the call minutes. These are the possibilities for the business to make adjustment to their model. This way they can also attract to new customers and generate more revenue for the business.

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] Ankush Handa, "Case Study: Churn Prediction" 2 March 2020 , https://ankushhanda.github.io/machine%20learning/Churn_Prediction/

[4] 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