CUSTOMER CHURN ANALYSIS FOR A TELECOMMUNICATION COMPANY

Saeid Rostami

2018/10/15

1 Definition

1.1 Project Overview

Usually in business and marketing the term of customer churn refers to loss of customers, which can lead to loss of revenue. In the world of growing subscription base business it is extremely important to not loosing the customers. Customer churn is important since it can directly affect the amount of revenue of a company. In order to survive in this kind of business many companies are using data analyst techniques for understanding why some costumers are leaving the company and try to win them back. There is some research and papers on customer churn by different research group. For example, in [1] the authors show predictive modeling for customer churner. This paper demonstrates how to use decision tree for churning problem. In [2] the authors describes how to apply Naive Bayes algorithm with supervised learning for customer churn problem.

In order to keeping the customers, companies should be able to predict in advance customers that are more likely to leave the company. The next important step is to come up with actions and algorithms that can impact each particular customer and retain them in the company [3].

1.2 Problem Statement

Customer churn refers to when clients decide to cut their relationships with a company. Many companies like telephone and Internet providers companies, Banks, insurance companies etc. usually use data analysts to predict and prevent losing customer. It is very important for every company to predict the risk of churning of a particular customer, especially when there is still time to prevent him/her to leave the company. Besides the direct loss of revenue because of churning, it is always more expensive to gain a new customer than it is to keep a current paying customer. It is worth to mention that there is a difference between voluntary churn and involuntary churn. Voluntary churn happens because of customers own decision to leave the company and switch to another one, while involuntary churn occurs due to special circumstances like technical problem, customer relocating or death. Usually, involuntary churning is excluded from analysis and more focus on the factors that companies can control for avoiding churning.

The input data will be customers specifications and contract details such as, the customer is male or female, what kind of service he/she gets from the company, how he/she pays the bills, how often he/she pays the bill, is he/she senior citizen or not and so on. The output is a column of yes and no, which defines a customer keeps using the company services and pays or decides to leave the company. Customer churning is a classification problem since our output is a discrete type data. The output variable, Churn value, takes the binary form as "Yes" or "NO", it will be categorized under classification problem in the supervised machine learning.

Since this is a supervised classification problem we can apply popular classification algorithm like Decision Tree, Logistic regression, SVM, Random Forest and xgboos. For better results and increasing classification accuracy I used grid search for tuning hyper parameter for each algorithm. Although the mentioned algorithms are good, it is worth to apply ensembling techniques to have even better results. I decided to apply Adaboost technique on this problem. Since we have an unbalanced data it can be really helpful to use some techniques to convert unbalanced data to balance data. At this point I use upsampling algorithm to make the data balanced and then applied the mentioned algorithms on balanced data.

1.3 Metric

Confusion matrix is a good way to demonstrate the performance of classification algorithms. Since we have unbalanced data, the number of observation in each class is not equal; classification accuracy alone can be misleading. Confusion matrix is composed of four main parts; True Positive, True Negative, False Positive, False Negative [4].

	Actual Positive	**Actual Negative**
Predicted Positive	TP	FP
Predicted Negative	FN	TN

True Positive: It is predicted positive and its true.
True Negative: It is predicted negative and its true.
False Positive: It is predicted positive and its false.
False Negative: It is predicted negative and its false.

Precision:

Precision is defined as the number of true positives over the number of true positives plus the number of false positives. It is a number between zero and one and higher number means better result. Precision is a measure that tells us what proportion of customers that we predicted leaving the company , actually leave.

$$Precision = \frac{TP}{TP + FP} \tag{1}$$

Recall:

Recall is defined as the number of true positives over the number of true positives plus the number of false negatives. It is a number between zero and one and higher number means better result. Recall is a measure that tells us what proportion of customers that actually left the company was predicted by the algorithm before.

$$Precision = \frac{TP}{TP + FN} \tag{2}$$

AUC-ROC curve:

AUC-ROC curve is one of the most popular metrics when we have a binary classification problem. Receiver Operating Characteristics (ROC) is a probability curve and Area Under The Curve (AUC) represents degree of separability. AUC has a range of [0, 1]. The greater the value, the better is the performance of our model [5]. if we randomly choose customer that stay with the company and customers who leaving, the probability that the staying case outranks the leaving case according to the classifier is given by the AUC.

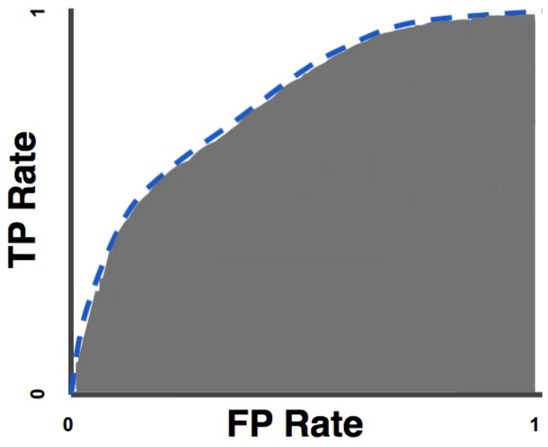


Figure 1: Area under the curve.

F1 Score:

The F1 score is the harmonic average of the precision and recall, it is a value between zero and one.

$$F1Score = 2 * \frac{precision * recall}{precision + recall} \tag{3}$$

2 Analysis

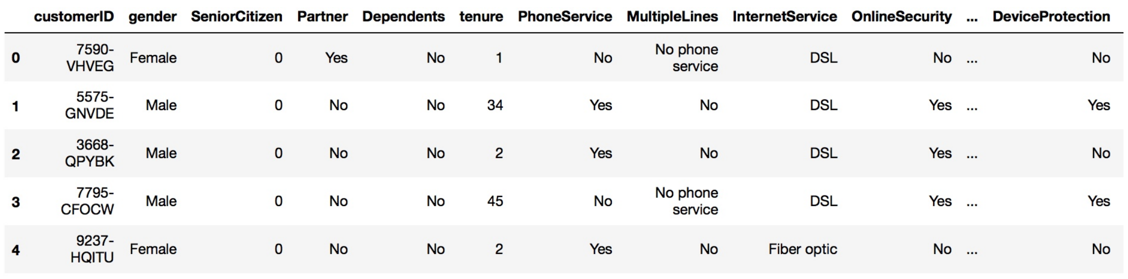
2.1 Data Exploration

I will use Telcom Customer Churn dataset, which is available at http://www.kaggle.com. The data was downloaded from IBM Sample Data Sets https://www.ibm.com/communities/analytics/watsonanalyt cs- blog/guide-to-sample-datasets/. The dataset provides 7043 customers information in 21 columns. We have both numerical and categorical type of information in this dataset. I am planning to use 30% of data for testing purposes and 70% of data for training purposes. The dataset is not perfectly balanced since the proportion of the people that stayed and left the company is about 73/27.

Input Data:
customerID: customerID
gender: Customer gender (female, male)
SeniorCitizen: Whether the customer is a senior citizen or not (1, 0)
Partner: Whether the customer has a partner or not (Yes, No)
Dependents: Whether the customer has dependents or not (Yes, No)
tenure: Number of months the customer has stayed with the company
PhoneService: Whether the customer has a phone service or not (Yes, No)
MultipleLines: Whether the customer has multiple lines or not (Yes, No, No phone service)
InternetService: Customers internet service provider (DSL, Fiber optic, No)
OnlineSecurity: Whether the customer has online security or not (Yes, No, No internet service)
OnlineBackup: Whether the customer has online backup or not (Yes, No, No internet service)
DeviceProtection: Whether the customer has device protection or not (Yes, No, No internet service)
TechSupport: Whether the customer has tech support or not (Yes, No, No internet service)
StreamingTV: Whether the customer has streaming TV or not (Yes, No, No internet service)
StreamingMovies: Whether the customer has streaming movies or not (Yes, No, No internet service)
Contract: The contract term of the customer (Month-to-month, One year, Two year)
PaperlessBilling: Whether the customer has paperless billing or not (Yes, No)
PaymentMethod: The customers payment method (Electronic check, Mailed check, Bank transfer (automatic), Credit card (automatic)
MonthlyCharges: The amount charged to the customer monthly
TotalCharges: The total amount charged to the customer

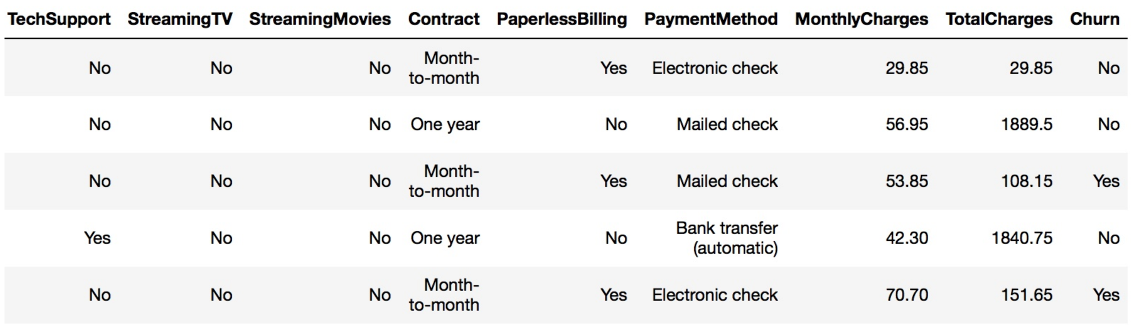
Output Data:

Churn: Yes for the customers that left the company and No for the customers that stayed with company



	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	InternetService	OnlineSecurity	...	DeviceProtection
0	7590-VHVEG	Female	0	Yes	No	1	No	No phone service	DSL	No	...	No
1	5575-GNVDE	Male	0	No	No	34	Yes	No	DSL	Yes	...	Yes
2	3668-QPYBK	Male	0	No	No	2	Yes	No	DSL	Yes	...	No
3	7795-CFOCW	Male	0	No	No	45	No	No phone service	DSL	Yes	...	Yes
4	9237-HQITU	Female	0	No	No	2	Yes	No	Fiber optic	No	...	No

Figure 2: Sample pf data



TechSupport	StreamingTV	StreamingMovies	Contract	PaperlessBilling	PaymentMethod	MonthlyCharges	TotalCharges	Churn
No	No	No	Month-to-month	Yes	Electronic check	29.85	29.85	No
No	No	No	One year	No	Mailed check	56.95	1889.5	No
No	No	No	Month-to-month	Yes	Mailed check	53.85	108.15	Yes
Yes	No	No	One year	No	Bank transfer (automatic)	42.30	1840.75	No
No	No	No	Month-to-month	Yes	Electronic check	70.70	151.65	Yes

Figure 3: Sample of data

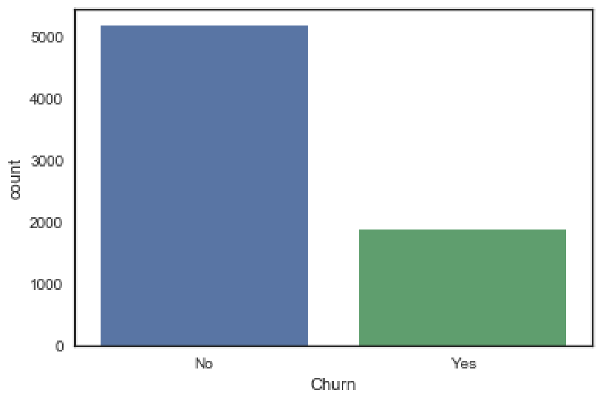
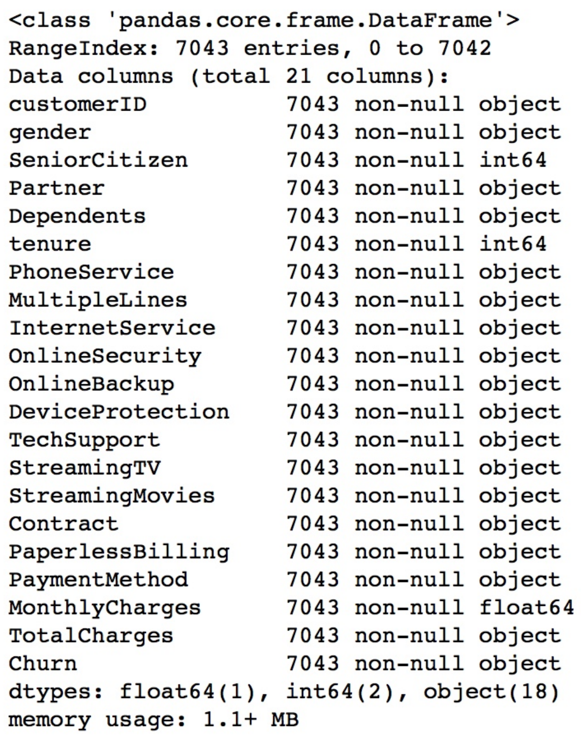


Figure 4: Churn Information

SeniorCitizen, tenure and MonthlyCharges columns are integer, integer and float respectively. Othe columns have categorical data. The range index of rows is [0,7043]. The time duration is not provided in this dataset. We do not have any NULL or duplicate data.

```
data = pd.read_csv("WA_Fn-UseC_-Telco-Customer-Churn.csv")
data.info()
```



```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 7043 entries, 0 to 7042
Data columns (total 21 columns):
customerID          7043 non-null object
gender              7043 non-null object
SeniorCitizen       7043 non-null int64
Partner             7043 non-null object
Dependents          7043 non-null object
tenure              7043 non-null int64
PhoneService        7043 non-null object
MultipleLines       7043 non-null object
InternetService     7043 non-null object
OnlineSecurity      7043 non-null object
OnlineBackup        7043 non-null object
DeviceProtection    7043 non-null object
TechSupport         7043 non-null object
StreamingTV         7043 non-null object
StreamingMovies     7043 non-null object
Contract            7043 non-null object
PaperlessBilling    7043 non-null object
PaymentMethod       7043 non-null object
MonthlyCharges      7043 non-null float64
TotalCharges        7043 non-null object
Churn               7043 non-null object
dtypes: float64(1), int64(2), object(18)
memory usage: 1.1+ MB
```

Figure 5: Dataset information

As we can see from the Figure 6 the number of people that stays with company is much bigger that people who leaves. Then we can conclude that our data is not balanced.

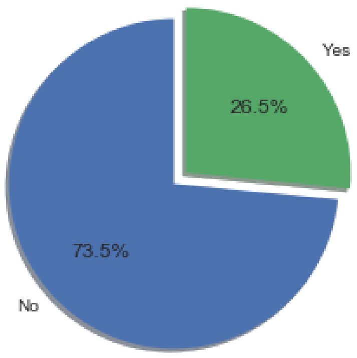


Figure 6: Churn

From the figure 7 we can see how different factors effects on people decision for staying or leaving the company.

From the figure 8 and Figure 9 we can see the distribution of numerical columns on churn column.

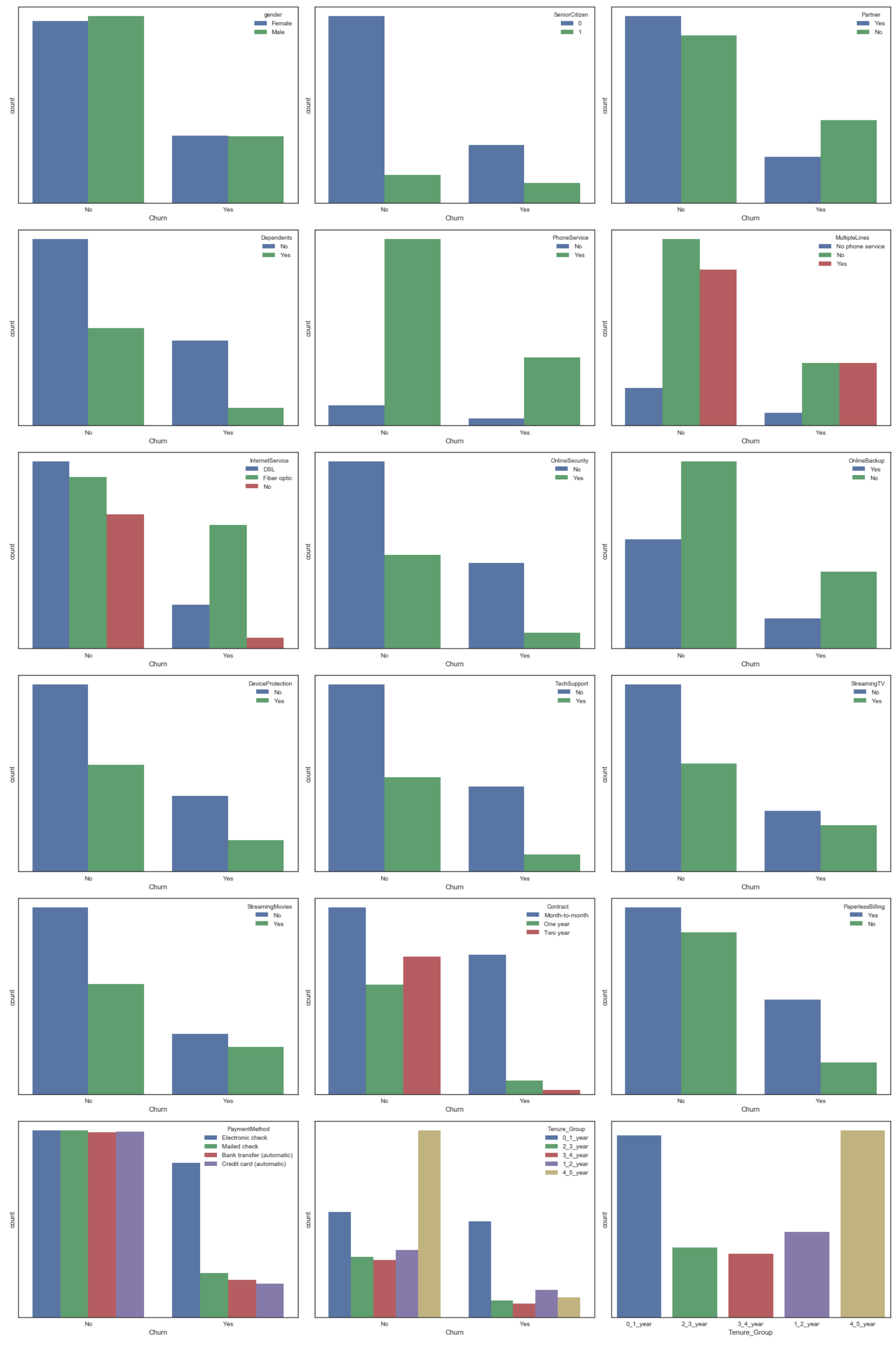


Figure 7: Data Visualization of dataset columns

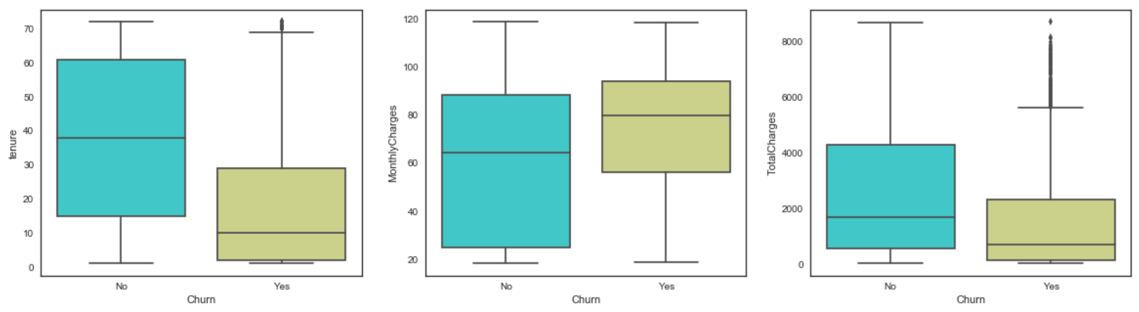


Figure 8: Data Visualization of dataset columns

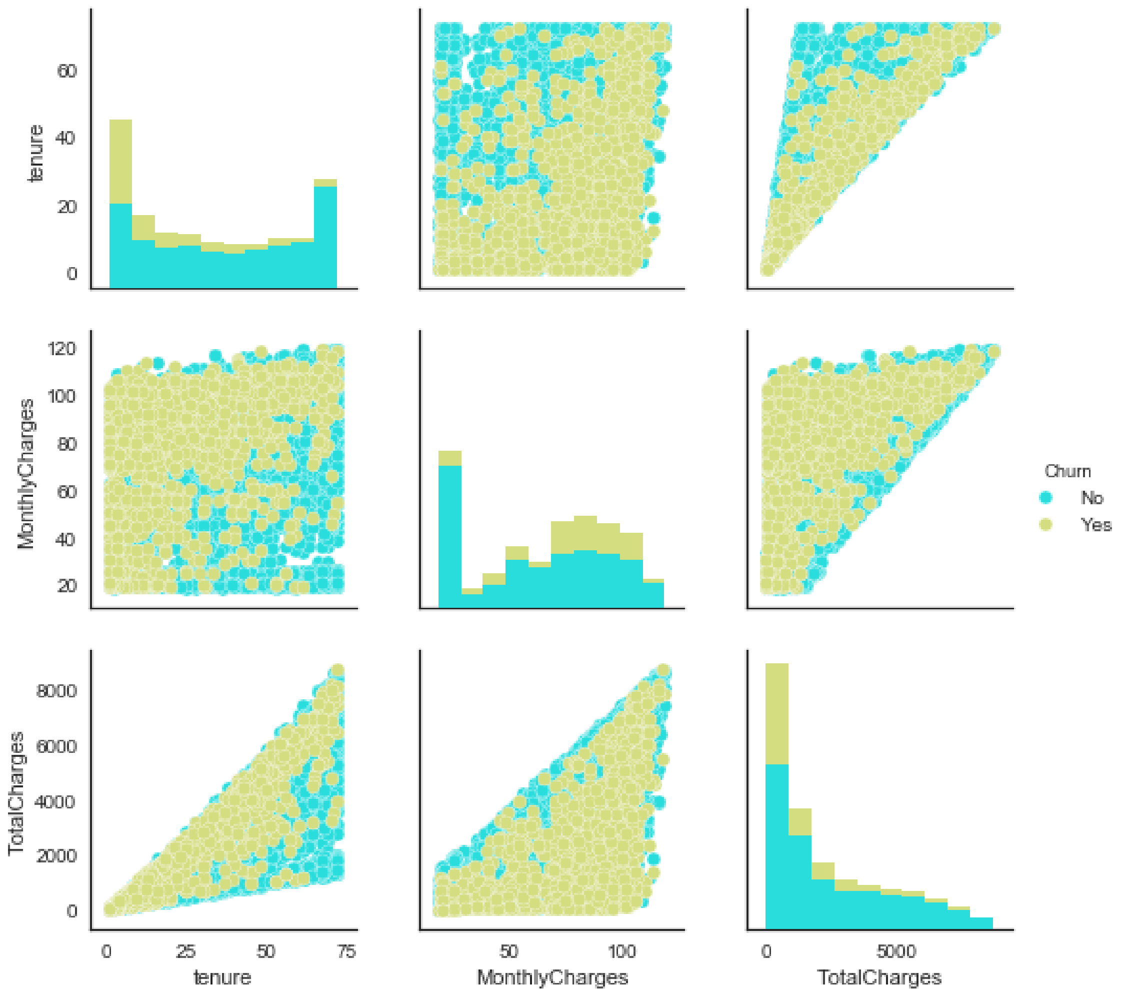


Figure 9: Data Visualization of dataset columns

```
def tenure_to_group(data):
    if data["tenure"] <=12:
        return "0_1_year"
    elif (data["tenure"] > 12) & (data["tenure"] <= 24 ):
        return "1_2_year"
    elif (data["tenure"] > 24) & (data["tenure"] <= 36) :
        return "2_3_year"
    elif (data["tenure"] > 36) & (data["tenure"] <= 48) :
        return "3_4_year"
    elif data["tenure"] > 48 & (data["tenure"] <= 60):
        return "4_5_year"
    elif data["tenure"] > 60 & (data["tenure"] <= 72):
        return "5_6_year"
data["Tenure_Group"] = data.apply(lambda data:tenure_to_group(data),axis = 1)
sns.countplot(data["Tenure_Group"]);
```

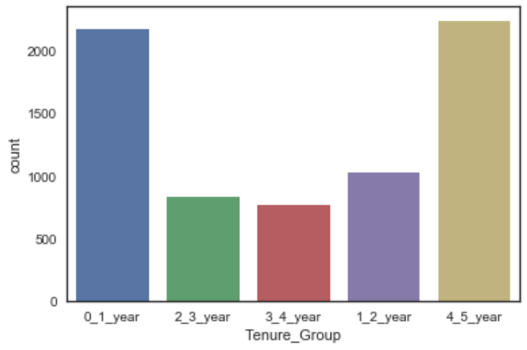


Figure 10: Tenure Group distribution

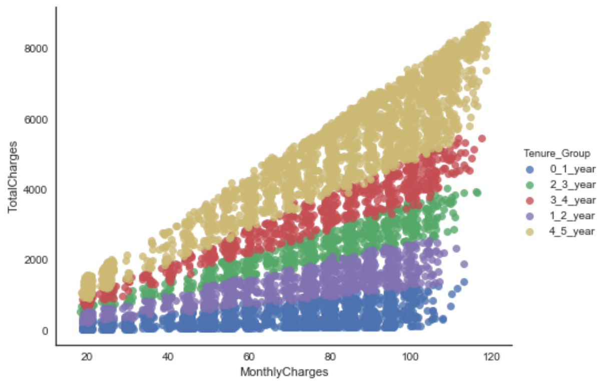


Figure 11: Total charges and monthly charges based on different tenure group

I decided to make different category of different tenures. I divided the whole tenure group into 5 different

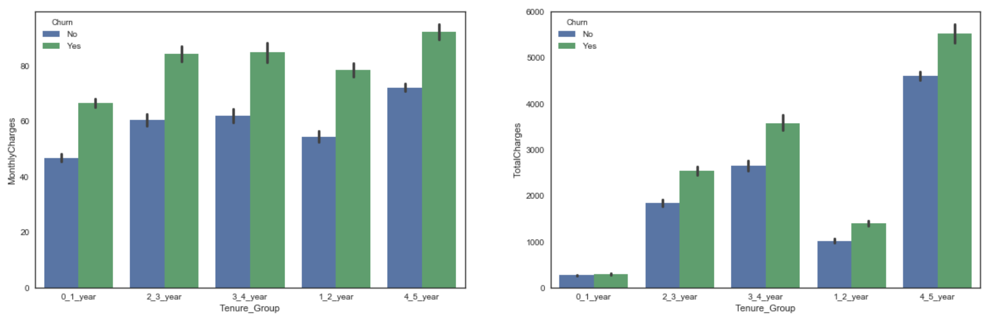


Figure 12: Different tenure group churning distribution

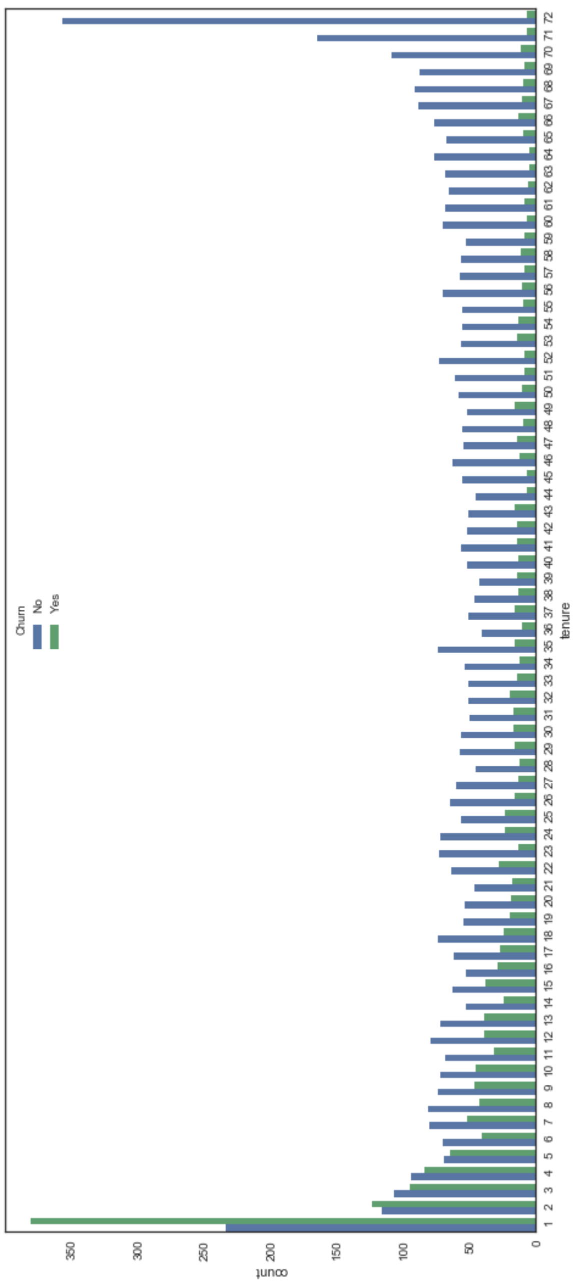


Figure 13: Different tenure group churning distribution

group;0-1, 1-2, 2-3, 3-4, 4-5. In Figure 9 we can see total charges and monthly charges based on different tenure group. In figure 10 we can see how different churning group reacted to churning.

2.2 Algorithms and Techniques

Since this is labeled data set, I am planning to apply supervised technique on that. I want to use different supervised algorithms like, Logistic Regression, Decision Tree, S.V.M., Random Forest and XGBoost and then compare the result together and see which one can better predict the customer churning.

Decision Tree: A decision tree is a tree that each node represents a feature and each branch represents a decision. It is a very popular algorithm in both classification and regression problems [6][7].

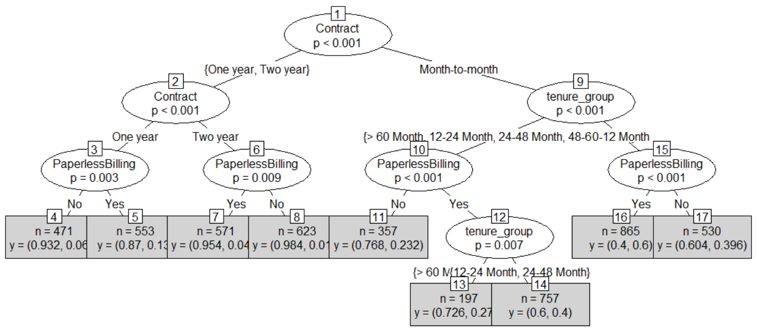


Figure 14: Decision Tree

Logistic Regression: Logistic regression is a statistical method for analyzing a dataset. The dataset should have one or more independent variables. In logistic regression, the dependent variable is binary 1 for True and 0 for False [7][12].

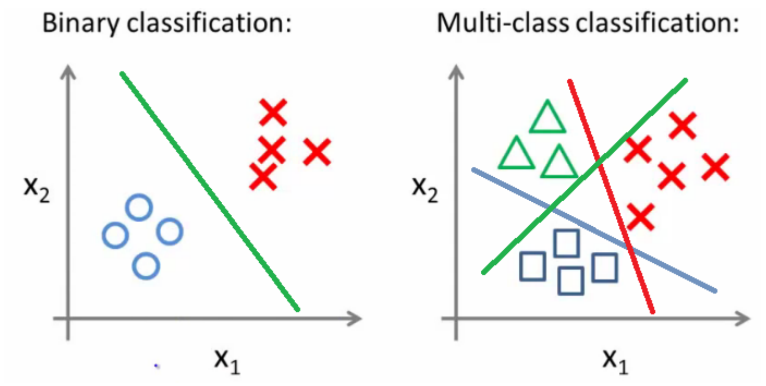


Figure 15: Logistic Regression

$$Sigmoid = \frac{1}{1 + \exp(x^2)} \tag{4}$$

The goal in Logistic Regression is to find the line that separates the data. first we need to apply apply the linear equation and then Sigmoid function for the result so we get the values between 0 and 1. Now we need to find the error by minimizing the cost function and next step is to apply Gradient descent[13].

$$J(\theta) = \frac{1}{m} \sum_{n=1}^{m} Cost(h_\theta(x^{(i)}), y^{(i)}) \tag{5}$$

Support Vector Machine: Support Vector Machine (SVM) is a supervised machine learning algorithm that can be used for both classification or regression challenges. In SVM, each data item is plotted as a point in n-dimensional space with the value of each feature being the value of a particular coordinate. Classification is applied by finding the hyper-plane that differentiate the two classes [8].

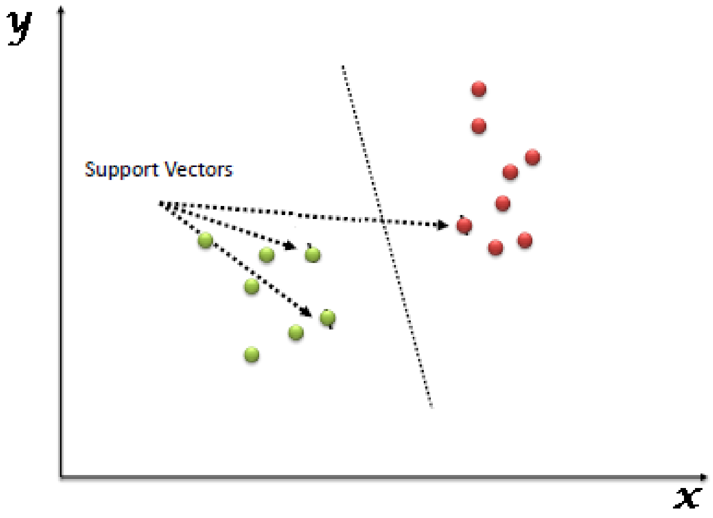


Figure 16: SVM

Random Forest: Random forest is a learning method for both classification and regression problems. In random forest multitude decision trees are constructed at training time and the result is the class that is the mode of the classes of the individual trees [9]. If each $h_k(x)$ is a decision tree, then the ensemble is a random forest[14]. The decision tree K leads to a classifier;

$$h_k(x) = h(x|\theta) \tag{6}$$

XGBoost: XGBoost is an implementation of gradient boosted decision trees designed for fast learning and better performance [10]. Boosting consists of three steps as follow [15];

1. An initial model m(0) is defined to predict the target variable y.
2. A new model m(1) is fit to the residuals from the previous step
3. Now, m(0) and m(1) are combined to give N(1), the boosted version of m(0).

2.3 Benchmark

The benchmark of this project is the output of decision tree model and I will try to beat the decision tree model by other algorithms. At the first step I applied algorithms with their default setting at Sklearn. Then I tune the parameter with grid search based on the better AUC. After that I applied Adaboost ensembling method. Since our data is unbalanced I applied upsampling method for making them balance and then applied above algorithm on them.

	precision	**recall**	**f1-score**	**AUC**
Decision Tree	0.73	0.73	0.73	0.6621
Logistic Regression	0.79	0.80	0.79	0.7090
SVM	0.78	0.79	0.77	0.6677
Random Forest	0.76	0.78	0.76	0.6649
XGBoost	0.79	0.80	0.79	0.7034

Table 1: Metric for default values of algorithms

	precision	**recall**	**f1-score**	**AUC**
Decision Tree	0.78	0.77	0.78	0.7296
Logistic Regression	0.78	0.80	0.79	0.7014
SVM	0.78	0.80	0.78	0.6903
Random Forest	0.80	0.74	0.75	0.7525
XGBoost	0.78	0.78	0.78	0.7015

Table 2: Metric after hyper parameters tune with grid search

	precision	**recall**	**f1-score**	**AUC**
AdaBoostClassifier:	0.80	0.74	0.75	0.7607

Table 3: Metric for AdaBoost Classifier

	precision	**recall**	**f1-score**	**AUC**
Decision Tree	0.87	0.87	0.86	0.8663
Logistic Regression	0.77	0.77	0.77	0.7694
SVM	0.77	0.76	0.76	0.7635
Random Forest	0.89	0.89	0.89	0.8896
XGBoost	0.78	0.78	0.78	0.7820

Table 4: Metric after upsampling

	precision	**recall**	**f1-score**	**AUC**
AdaBoostClassifier:	0.91	0.90	0.90	0.9022

Table 5: Metric for AdaBoost Classifier after upsampling

3 Methodology

3.1 Data Preprocessing

It is usefule to see how many different values we have for each column.

```
for col in temp_columns:
    print("{} : {}".format(col,data[col].unique()))
```

Output:

```
SeniorCitizen : [0 1]
Partner : ['Yes' 'No']
Dependents : ['No' 'Yes']
tenure : [ 1 34 2 45 8 22 10 28 62 13 16 58 49 25 69 52 71 21 12 30 47 72 17 27 5 46 11 70 63 43 15 60
18 66 9 3 31 50 64 56 7 42 35 48 29 65 38 68 32 55 37 36 41 6 4 33 67 23 57 61 14 20 53 40 59 24 44 19
54 51 26 39] PhoneService : ['No' 'Yes']
MultipleLines : ['No phone service' 'No' 'Yes']
InternetService : ['DSL' 'Fiber optic' 'No']
OnlineSecurity : ['No' 'Yes' 'No internet service']
OnlineBackup : ['Yes' 'No' 'No internet service']
DeviceProtection : ['No' 'Yes' 'No internet service']
TechSupport : ['No' 'Yes' 'No internet service']
StreamingTV : ['No' 'Yes' 'No internet service']
StreamingMovies : ['No' 'Yes' 'No internet service']
Contract : ['Month-to-month' 'One year' 'Two year']
PaperlessBilling : ['Yes' 'No']
PaymentMethod : ['Electronic check' 'Mailed check' 'Bank transfer (automatic)' 'Credit card (auto-
matic)']
```

We can interpret that for the columns withe values of ['No' 'Yes' 'No internet service'] No and No Internet Service have same meaning to us then I changed all the No Internet Servise to NO.

```
for col in temp_columns:
    if col in ("OnlineSecurity","OnlineBackup","DeviceProtection","TechSupport","Streami
        data[col] = data[col].replace({'No internet service':'No'})
```

For the next step, I divided the data set to categorical data, numerical data, target and customer ID categories. For the calculation and simplicity I changed the target data value to 0 and 1 instead of using Yes and No.

```
cat_cols = [x for x in data.columns if data[x].nunique()<6 and x!="Churn"]
num_cols = [x for x in data.columns if data[x].nunique()>6 and x!="customerID"]
id_customer = data["customerID"]
label = data["Churn"]
label = label.apply(lambda x: 1 if x == "Yes" else 0)
```

Due to different ranges of values in different columns, it is possible that some values will dominate the others. To avoid that I used Min-Max scaler to scale the data.

```
from sklearn.preprocessing import MinMaxScaler

features_log_transformed = pd.DataFrame(data = data[num_cols])
features_log_transformed[num_cols] =
data[num_cols].apply(lambda x: np.log(x + 1))
scaler = MinMaxScaler()
features_log_minmax_transform = pd.DataFrame(data = features_log_transformed)
features_log_minmax_transform[num_cols] =
scaler.fit_transform(features_log_transformed[num_cols])
```

From Figure 17 we can see the correlation between different numerical columns in the data set. As we can see from the heatmap there is a strong correlation between tenure and TotalCharge. For avoiding the redundant data I dropped the tenure columns.

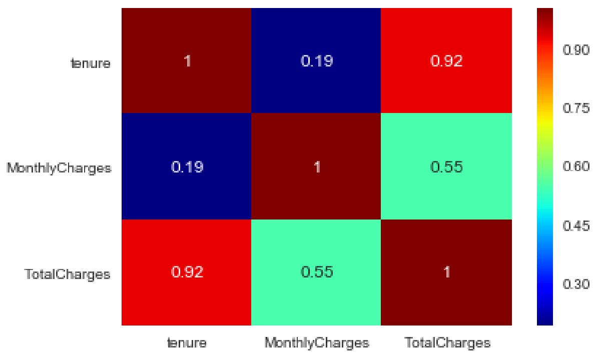


Figure 17: Correlation of different numerical data

For the final step I changed all the categorical data into numerical type of data using one hot encoding method.

```
data = pd.get_dummies(data = data,columns = cat_cols)
```

3.2 Implementation

After prepossessing and cleaning data it is the time for implement the algorithm. The model implementation is done in 3 main step.

- Splitting the data into training and testing part
- designing apply_classifier() function for applying different algorithms
- Creating grid_search() function for hyper parameter tuning

For the first part I used 30% of data for testing and 70% for training. In apply_classifier() function I applied different supervised algorithm on the data and then show the results of metrics and plot the AUC curve. Tnen I defined hyper parameter that I want to tune for each algorithm and send it to grid_search() function to find the best values.I used AUC For the scoring for the grid search. For the first step I applied Decision Tree since it is my base model and other models will be compared to this one. Then I applied Logistic Regression, SVM, Random Forest and XGboost. The hyper parameter for each algorithm is defined as follow;

```
Tree_parameters = {"max_depth": [3,4,5,6],
                   "min_samples_leaf":[1,2,3,4]}

LogReg_parameters = {
    "C":[0.25,0.5,0.75,1.0,1.5,2.0,2.5,3.0,4.0,10.0],
    "solver":["newton-cg", "lbfgs", "sag", "saga"],
    "tol":[0.01,0.001,0.0001,0.00001],
    "warm_start":["True","False"]}

SVM_parameters = {
    "C":[1.0,2.0,3.0],
    "cache_size":[100,200],
    "decision_function_shape":['ovo','ovr'],
    "kernel":['sigmoid',"linear"],
    "tol":[0.001,0.0001]}

RandomForest_parameters = {
    "n_estimators" :[10,15,20,25,30],
    "criterion": ["entropy","gini"],
    "max_depth" : [5,10,15],
    "min_samples_split":[2,4,8,16],
    "max_features":["sqrt","auto","log2"],
    "class_weight" : ["balanced_subsample","balanced"]}

Xgboost_parameters = {"max_depth" : [3,4,5,6],
    "learning_rate" : [0.001,0.0001],
    "booster" : ["gbtree","gblinear","dart"],
    "min_child_weight" : [1,2,3,4]

                      }
```

In Table 1 and Table 2 we can see the result of algorithms before and after parameter tuning respectively. The challenge in coding part was to write one helper function that can run all the algorithms. Since there was two plots for each algorithm, confusion matrix and RUC curve, it was time consuming to put everything in right places using subplot function.

3.3 Refinement

For the first step I applied Adaboost ensembling method to increase the classification accuracy. Since the working data is unbalanced data I applied upsamling method to make the data balanced and then applied the same method as above on new balanced data.

Upsampling:
Upsampling matches the number of samples in the majority class with resampling from the minority class.

```
from sklearn.utils import resample

upsample_data = data_original

majority = upsample_data[upsample_data["Churn"]==0]
minority = upsample_data[upsample_data["Churn"]==1]

minority_upsampled = resample(minority,
replace=True,n_samples=5163,random_state=42)
del(upsample_data)
upsample_data = pd.concat([majority,minority_upsampled])
```

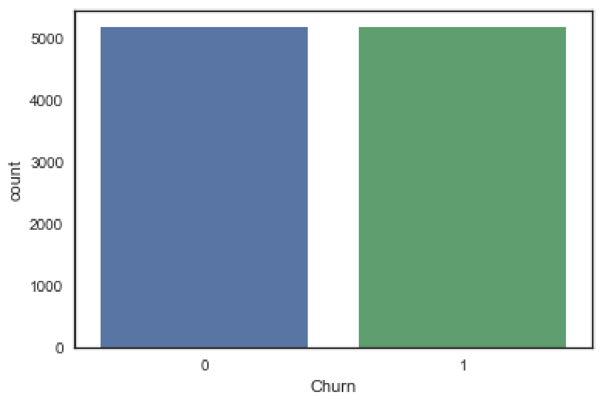


Figure 18: Churn columns after upsampling

4 Results

4.1 Model Evaluation and Validation

The best result before upsampling :

```
  Classification  report  :
                      precision      recall    f1-score     support


               0          0.91        0.71        0.80        1549
               1          0.51        0.81        0.62         561


avg  /  total             0.80        0.74        0.75        2110
```

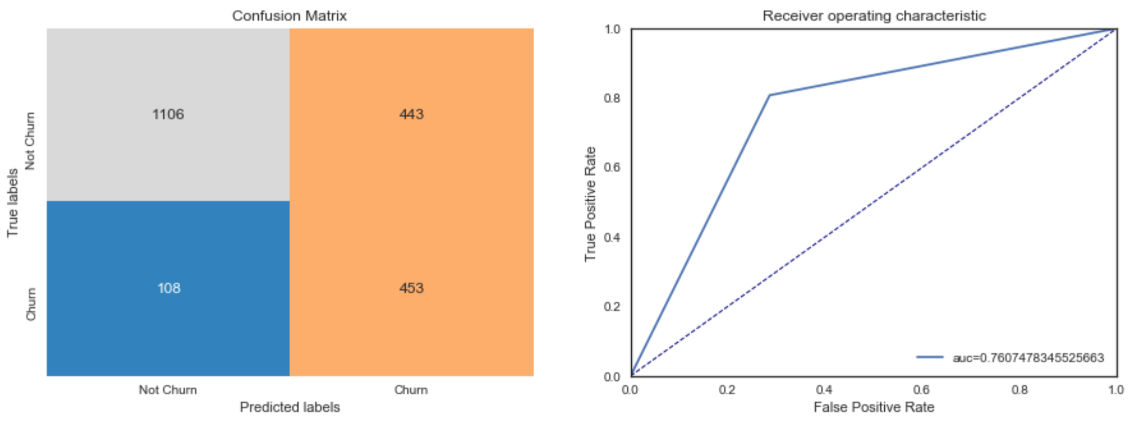


Figure 19: Adaboost

The best result after upsampling :

```
Classification report :
              precision    recall  f1-score   support


          0       0.95      0.85      0.90      1571
          1       0.86      0.95      0.90      1527


avg / total       0.91      0.90      0.90      3098
```

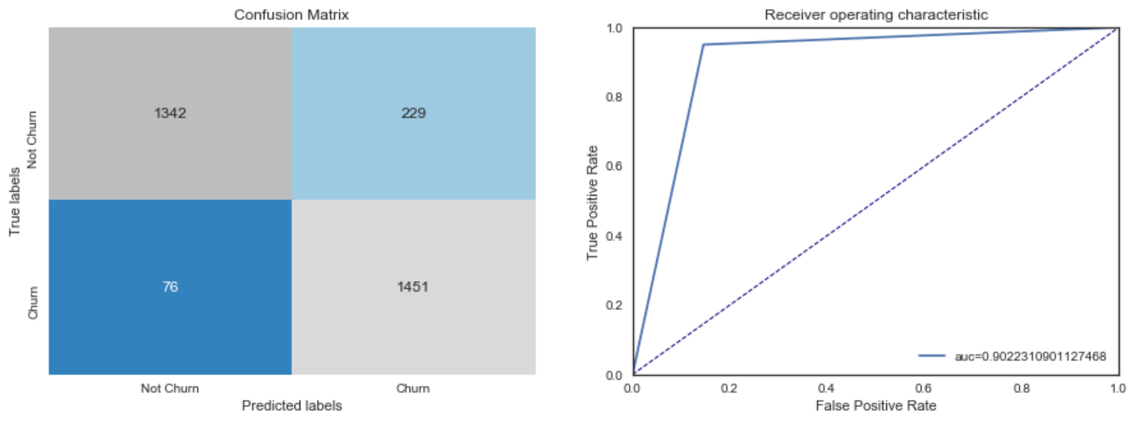


Figure 20: Adaboost

In both case Adaboost with random forest have best results. It can be beacause of Adaboost uses some weak learners and combined the result with a strong learner, which can give a better classification accuracy in this dataset. Also upsampling made the result better since it prevent the algorithms bias to majority class.

4.2 Justification

	precision	**recall**	**f1-score**	**AUC**
BenchMark Model	0.79	0.80	0.79	0.6621
Final Model	0.91	0.90	0.90	0.9022

Table 6: Comparing final model and benchmark

5 Conclusion

5.1 Free-Form Visualization

The seven most important feature for this dataset.

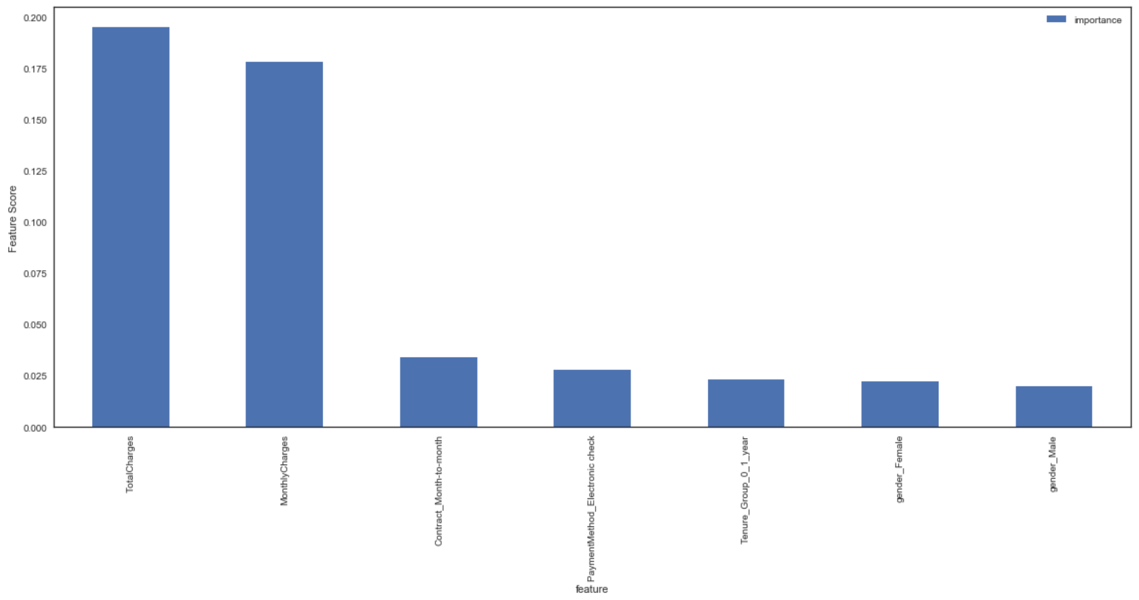


Figure 21: Feature Importance

5.2 Reflection

This project can be summarized as following steps:

- Searching for problem and deciding what kind of project I want to work on
- Finding database by searching at websites like kaggle.com
- Visualizing different aspect of data
- Prepossessing and cleaning data
- Choosing algorithms to apply on data
- Choosing the best metrics based on the data that I have
- Writing helper function for implementation of different algorithm and avoid redundant codes
- Deciding which parameter should be tuned
- Writing helper function for grid search
- Applying upsamling method on data

- Selecting the best model

I think number one and two were very interesting. At first it was very difficult for me to decide which kind of project I want to work on. Should it be a supervised or unsupervised problem!!. And finding the dataset was so challenging and time consuming. The next challenge for me was writing the proposal and especially selecting the right metrics. In part o implementation writing helper functions and data cleaning also were challenging.

5.3 Improvements

One of the improvements that we could do is tuning more parameter with grid search and maybe choosing the most effective parameter to tune. We also can apply some other ensembling method like bagging.

References

[1] K. B. Oseman, S.B.M. Shukor, N. A. Haris, F. Bakar, Data Mining in Churn Analysis Model for Telecommunication Industry, Journal of Statistical Modeling and Analytics, Vol. 1 No. 19-27, 2010.
[2] S.V. Nath, Customer Churn Analysis in the Wireless Industry: A Data Mining Approach, Technical Report, retrieved from http://download.oracle.com/owsf_2003/40332.pdf, April 14, 2014.
[3] https://www.optimove.com/learning-center/customer-churn-prediction-and-prevention
[4] https://towardsdatascience.com/understanding-confusion-matrix-a9ad42dcfd62
[5] https://developers.google.com/machine-learning/crash-course/classification/roc-and-auc
[6] https://medium.com/deep-math-machine-learning-ai/chapter-4-decision-trees-algorithms-b93975f7a1f1
[7] https://www.medcalc.org/manual/logistic_regression.php
[8] https://www.analyticsvidhya.com/blog/2017/09/understaing-support-vector-machine-example-code/
[9] https://en.wikipedia.org/wiki/Random_forest [10] https://machinelearningmastery.com/gentle-introduction-xgboost-applied-machine-learning/
[11] https://datascienceplus.com/predict-customer-churn-logistic-regression-decision-tree-and-random-forest/
[12] https://medium.com/deep-math-machine-learning-ai/chapter-2-0-logistic-regression-with-math-e9cbb3ec6077
[13] https://medium.com/deep-math-machine-learning-ai/chapter-2-0-logistic-regression-with-math-e9cbb3ec6077
[14] http://math.bu.edu/people/mkon/MA751/L19RandomForestMath.pdf
[15] https://www.analyticsvidhya.com/blog/2018/09/an-end-to-end-guide-to-understand-the-math-behind-xgboost/