Business Data Analytics Project 1
Telecom Churn Prediction

Delton M Antony, 18MCMI05
Garima Jain, 18MCMI14
Mtech Artificial Intelligence

ABSTRACT

Churning or customer attrition is the situation where a customer belonging to a company will switch to its competition. We are given a dataset that contains the details of the customers of a telecommunications company that is experiencing a high churn ratio. The goal is to find out insights as to why the churn is happening and also to build a prediction model for the same. We did exploratory data analysis on the dataset and found out the trends and possible reasons to the churn ratio. Furthermore, we trained eleven models on the data and evaluated their performance to find out the best one suited for deployment. We finally built a decision tree classifier which is trained on the entire dataset to be deployed by the company to predict unseen data.

CONTENTS

1. INTRODUCTION

In this project, we will apply descriptive and predictive analytics techniques on the dataset provided to us by the customer. The customer is a telecommunications company that is experiencing an abnormal ratio of churners among its customers. The company wants to know why the customers are churning and also, the company wants a solution which will enable them to predict churners so that they can tackle the attrition before it happens.

The company has collected the data from the customers and has got a good quality dataset with itself. The aim of this project is to exploit the dataset and find insights into why this abnormal churn ratio is happening and make predictive models based on this dataset which can be used to predict churn on unseen data corresponding to both new and existing customers.

The workflow of this project is split into two parts. The first part deals with exploring the data provided by the customer and finding trends in the data which can be utilized to explain why this churn is happening. This is called exploratory data analysis. The second part is where we are building different models that can predict the churn of customers. Then we evaluate the models and will choose the best one as the solution to be deployed and monitored by the company.

The end product we deliver to the telecommunications company is also divided into two. Firstly, the insights we got form the data which can be used by the customer to find out why this churn is happening. Next, the final model which can be used by the customer to predict future churners.

2. PROBLEM DEFINITION AND DATA-MINING TASK IDENTIFICATION

We are given a dataset by the company with all the details of the company's customers like gender, whether the customer is a senior citizen, whether he/she is a partner. Along with the demographics data, the dataset also includes the details of the telecom plan opted by the customer like whether he has an internet connection, if yes, of what type, if he has online security and backup protection. The dataset also contains details about the billing options chosen by the customer as well as the amount paid by the customer.

With this data, our aim is to provide insights into the churns that have already happened so that it can be stopped at the source. Also, we need to make models that can identify future churners based on their data.

As we are identifying who all the churners are as opposed to who all the non-churners, this project can be identified as a classification problem. As we have only two outcomes, it is a binary classification problem with two classes - Churn and Not-Churn.

3. METHODS AND TECHNIQUES APPLIED

3.1. Principal Component Analysis:

Principal Component Analysis, commonly abbreviated as PCA, is a method to reduce the input dimensions without performing feature selection. PCA merely performs a linear transformation of the input data from one form to another. Then generally we only consider the first few resultant principal components as most of the variance of the data will be covered by them. Each of these principal components contains the essence of all the input variables. This means selecting the first few principal components after doing PCA will essentially result in dimension reduction without feature selection.

3.2. Logistic Regression:

The logistic model (or logit model) is a widely used statistical model that uses the logistic function to model a binary dependent variable. In regression analysis, logistic regression is estimating the parameters of a logistic model; it is a form of binomial regression. Logistic Regression is a discriminative classifier. For a single input variable, the Logistic Regression function looks like this

$$p(x) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 x)}}$$

3.3. Naive Bayes Classifier:

Naive Bayes classifiers are a family of simple probabilistic classifiers based on applying Bayes' Theorem with strong (naive) independence assumptions between the features. Naive Bayes is a popular method for text classification. Naive Bayes is preferred on smaller datasets. It is a generative classifier.

$$\begin{aligned} p(C_k \mid x_1, \ldots, x_n) &\propto p(C_k, x_1, \ldots, x_n) = \\ &= p(C_k)\, p(x_1 \mid C_k)\, p(x_2 \mid C_k)\, p(x_3 \mid C_k)\, \cdots \\ &= p(C_k) \prod_{i=1}^{n} p(x_i \mid C_k)\,, \end{aligned}$$

Here α means directly proportional.

3.4. K - Nearest Neighbors Classifier:

KNN can be used for both classification and regression predictive problems. However, it is more widely used in classification problems. The input consists of the k closest training examples in the feature space. The output is a class membership. An object is

classified by a majority vote of its neighbors, with the object being assigned to the class most common among its k nearest neighbors. Nearness is calculated by using distance measure like Euclidean distance. This is the Euclidean distance between points p and q.

$$d(\mathbf{p},\mathbf{q}) = d(\mathbf{q},\mathbf{p}) = \sqrt{(q_1 - p_1)^2 + (q_2 - p_2)^2 + \cdots + (q_n - p_n)^2}$$

$$= \sqrt{\sum_{i=1}^{n} (q_i - p_i)^2}.$$

3.5. Support Vector Classifiers:

Unlike other models, SVC does not use the entire dataset to train itself. It uses the tipping points - also known as support vectors. These are those points that define the boundary of the two classes. This is a rare case of dimension reduction with respect to the records as opposed to features. An SVM model is a representation of the records as points in space, mapped so that the records of the separate categories are divided by a clear gap that is as wide as possible. New examples are then mapped into that same space and predicted to belong to a category based on which side of the gap they fall.

$$x = b + \sum \alpha i \; a(i).a \;,\; i \text{ is support vector}$$

3.6. Kernel Support Vector Machine Classifiers:

Kernel Support Vector Machine will introduce the data to a higher dimension and then try to classify them. The essence of the procedure is to make a non-linearly-separable data linearly separable. Here, the kernel function used is set as rbf which is "Radial Basis Function". The function is as follows:

$$K(\mathbf{x},\mathbf{x}') = \exp\left(-\frac{\|\mathbf{x}-\mathbf{x}'\|^2}{2\sigma^2}\right)$$

Here, |x - x'|^2 is the squared Euclidean distance.

3.7. Decision Tree Classifier:

Decision Tree is the go to method if you want clear insights on how the classifier predicts and under what criteria. This gives clear rules followed by the classifier. Hence, it is not considered as a black box. We are using information gain as the criteria ie criteria = entropy. In the tree structure, leaves represent class labels and branches represent conjunctions of features that lead to those class labels.
The formula for entropy is given below:

Entropy = $-\sum p(x) log(p(x))$

3.8. Random Forest Classifier:

Random forest comes under a category called ensemble algorithms. Here, instead of one single decision tree, an army of decision trees are used and the results learned by them are then converged together to create the model. The hyperparameter n_estimators can be used to set the number of individual decision trees used in the forest. We are using information gain as criteria as usual.

3.9. Xtreme Gradient Boosting:

Gradient boosting is a machine learning technique for regression and classification problems, which produces a prediction model in the form of an ensemble of weak prediction models, typically decision trees. It builds the model in a stage-wise fashion like other boosting methods do, and it generalizes them by allowing optimization of an arbitrary differentiable loss function. This is done sequentially. As the number of iterations goes by, the error will steadily decrease.

3.10. Multi Layer Perceptron:

A multilayer perceptron is a class of feedforward artificial neural network. An MLP consists of, at least, three layers of nodes: an input layer, a hidden layer and an output layer. Except for the input nodes, each node is a neuron that uses a nonlinear activation function. MLP utilizes a supervised learning technique called backpropagation for training. MLP can work with non-linearly-separable data.

3.11. Model Evaluation Methods:

The methods used for evaluating the models include confusion matrix, 10 - fold cross validation and Area under Receiver Operating Characteristic Curve. Model Comparison is done by t-stat test.

4. DATASET DESCRIPTION

The dataset given by the telecommunications company contains 21 columns and 7043 rows. It is of the format csv - comma separated values where each of the records have the attributes separated by commas. The header, which mentions the attribute name for each columns is present by default in the dataset. These 21 columns correspond to the attributes of the customer. One of these - the Churn column can be considered as our target variable which says whether the customer is going to churn or not. The 7043 rows correspond to each of the 7043 customers.

Out of the total number of columns, three are numeric values while the rest are nominal categorical. Categorical variables like gender and SeniorCitizen have two values while those like MultipleLines have three values each of which is equally weighted among each other.

5. EXPLORATORY DATA ANALYSIS

The following is what the info method, which is a method to describe the dataset in terms of Attribute Names and data type, returned after loading the dataset:

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

While listing the number of unique values in each of the variables, the following was the result:

customerID	7043
gender	2
SeniorCitizen	2
Partner	2
Dependents	2
tenure	73
PhoneService	2
MultipleLines	3
InternetService	3
OnlineSecurity	3
OnlineBackup	3
DeviceProtection	3
TechSupport	3
StreamingTV	3

```
StreamingMovies     3
Contract            3
PaperlessBilling    2
PaymentMethod       4
MonthlyCharges      1585
TotalCharges        6531
Churn               2
dtype: int64
```

What this means is that there are both continuous as well as categorical variables in this dataset. Continuous variables are those that can take a real number value and categorical variables are those that are only able to take a value that belongs to a set of predetermined values. We can see that those variables that has the number of unique values as 2, 3 or 4 are in fact categorical variables. On preprocessing, these have to be encoded.

The following is the list of categorical values:

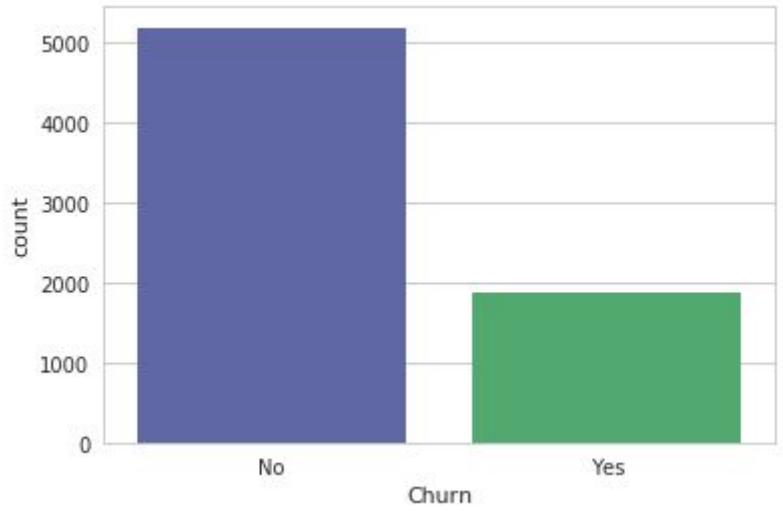
1. Gender
2. SeniorCitizen
3. Partner
4. Dependents
5. PhoneService
6. MultipleLines
7. InternetService
8. OnlineSecurity
9. OnlineBackup
10. DeviceProtection
11. TechSupport
12. StreamingTV
13. StreamingMovies
14. Contract
15. PaperlessBilling
16. PaymentMethod

Here is the list of numerical values:

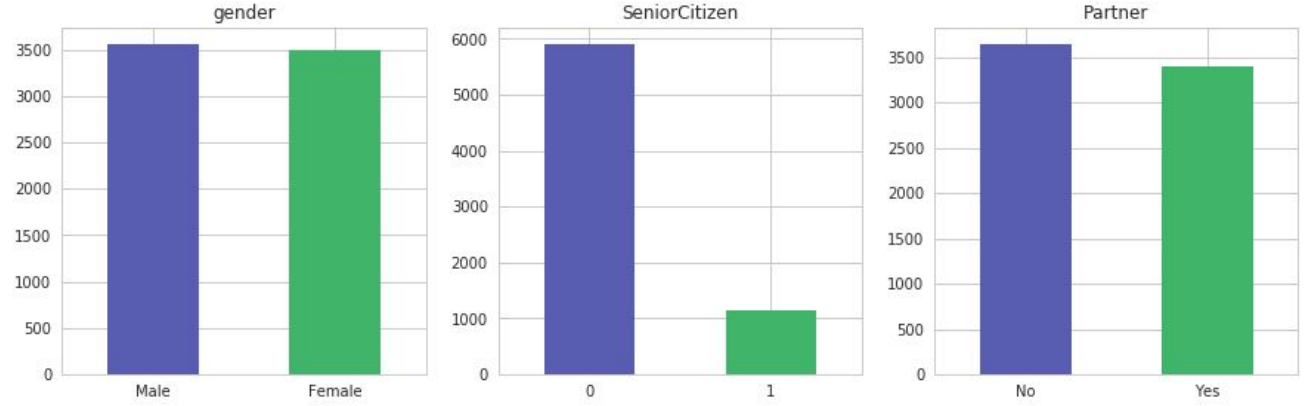
1. Tenure
2. MonthlyCharges
3. TotalCharges

Also, in terms of database management, the primary key is the CustomerId as it has got 7043 distinct values which is the same as the number customers. As in almost all cases, the primary key is not important in training a model.

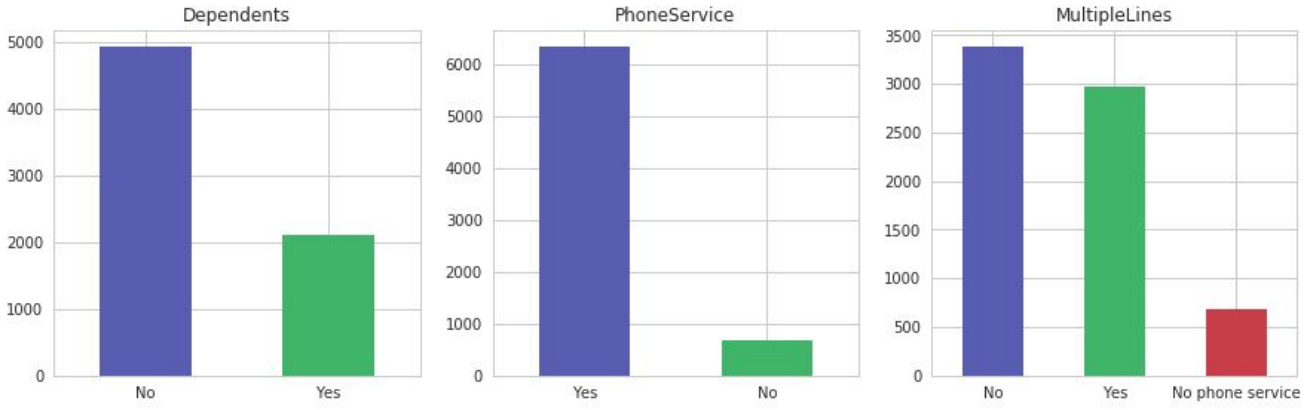
In univariate analysis, let us take the categorical variables first and see how they are distributed. For that we are visualizing the data using histograms. The following graphs are plotted using the seaborn and matplotlib.pyplot libraries of python3.



From the above graph we can see that the Non-Churners and Churners are distributed in the approximate ratio 5:2. This is a serious problem as customer acquisition is costlier than customer retention.

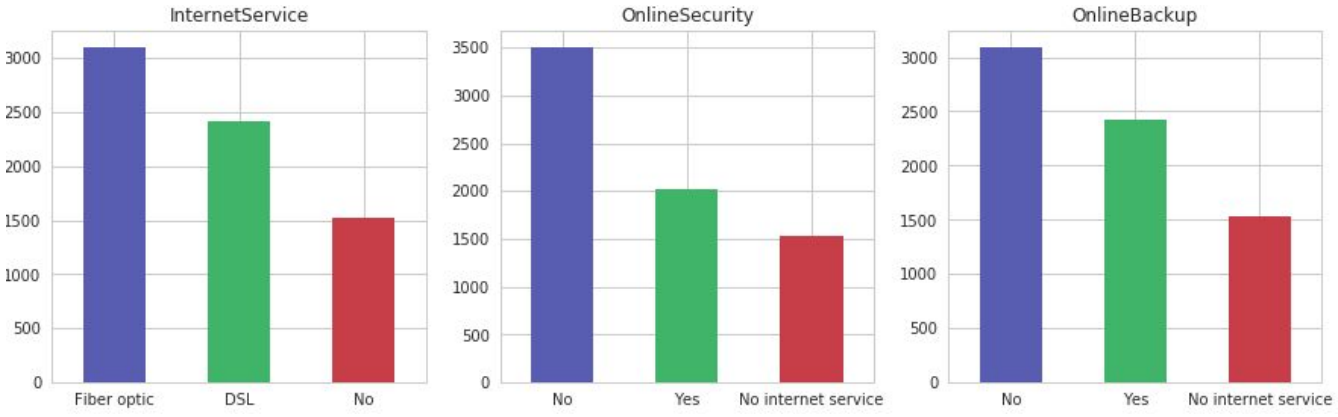


As you can see from above, distribution of gender is balanced. There are equal number of male customers and female customers. The number of senior citizen customers is very less though. The number of customers having partners is also fairly balanced.

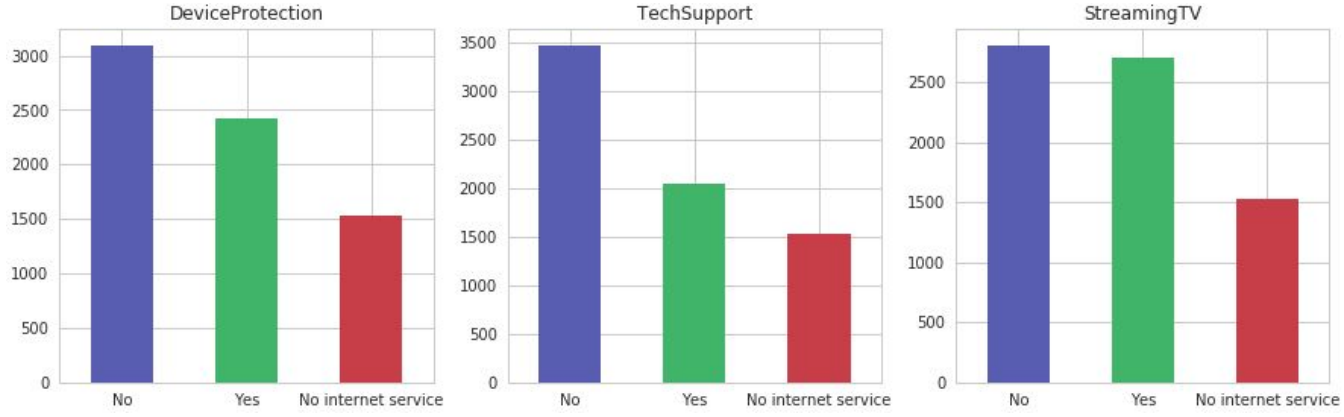


The customers without dependents is more in number. Almost all customers opted for a phone service with a very less number of customers opting out of it. From the MultipleLines graph, we

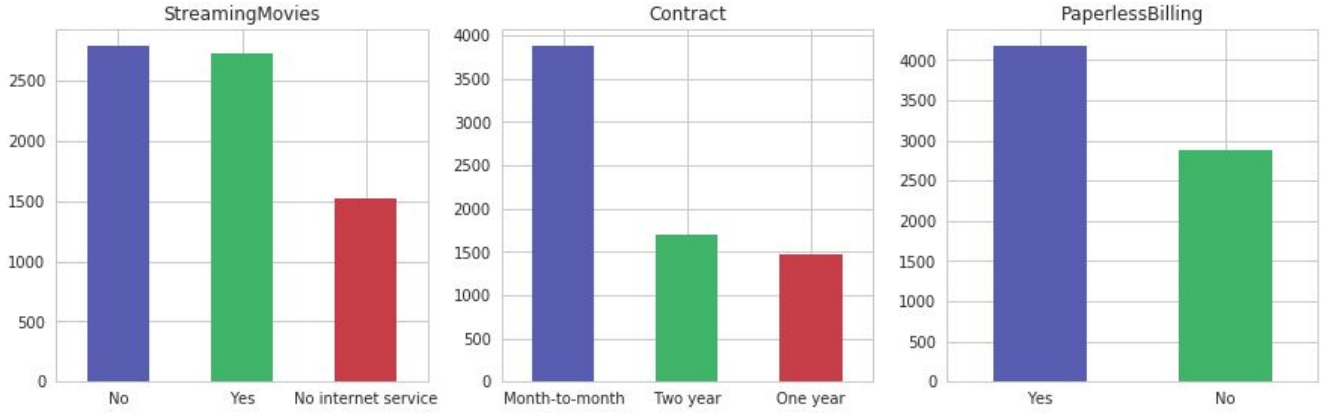
can see that more people prefer a single line connection but a slightly less number of people have opted for multiple lines. Of course, you cannot say anything for the customers who did not opt for phone service.



From the above graphs, we can see that more number of people are choosing an OFC connection compared to ordinary broadband using DSL. Also you can see that some people are not subscribed to either of their internet plans. The online security feature provided by the company is rejected by most of the customers. Also, Online Backup facility is also comparatively underutilized.

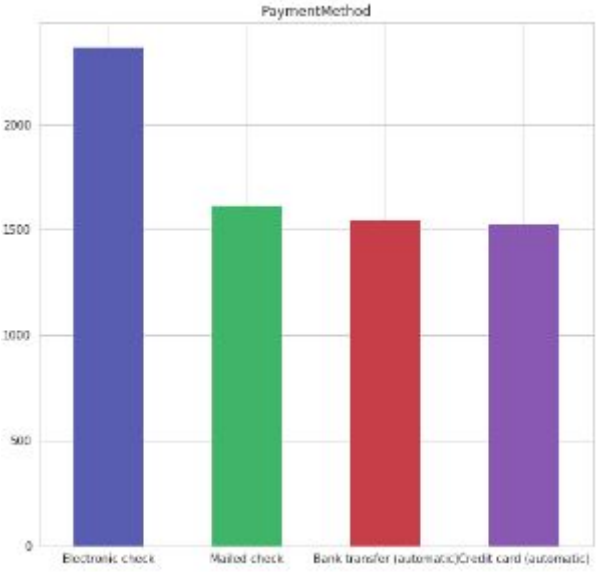


Most of the customers are also not opting for Device Protection and Tech support. One thing to be noted here is that the Streaming TV feature is opted by more customers compared to other features like Tech Support.



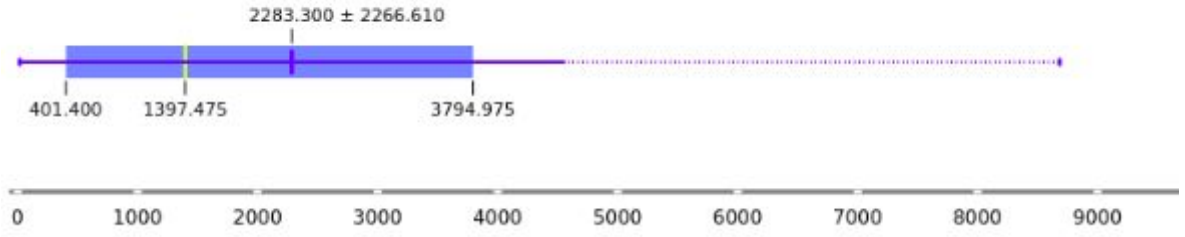
Streaming Movies service is also a preferred feature as it is also in par with Streaming TV. When it comes to Contract, most of the customers prefer the monthly plan. Interestingly, the number of

customers opting for two year plan is much higher than those opting for a one year plan. Most of the customers are opting for paperless billing.

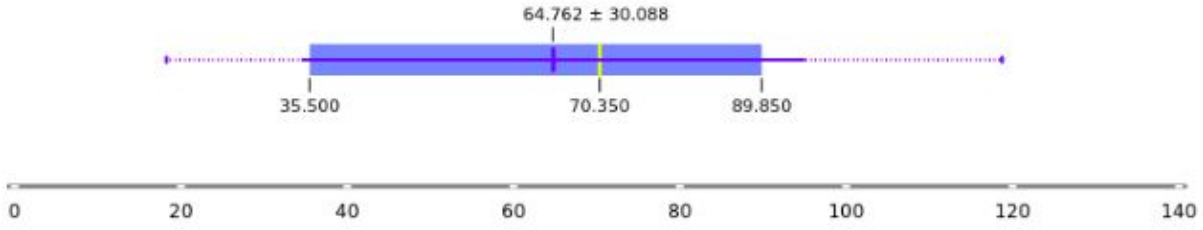


This is the graph for payment method.
Legend:
Electronic Check
Mailed Check
Bank Transfer (automatic)
Credit Card (automatic)

For understanding the numerical variables, we are constructing Box Plots. We are making box plots for monthly charges and total charges.
The following is the boxplot for MonthlyCharges:

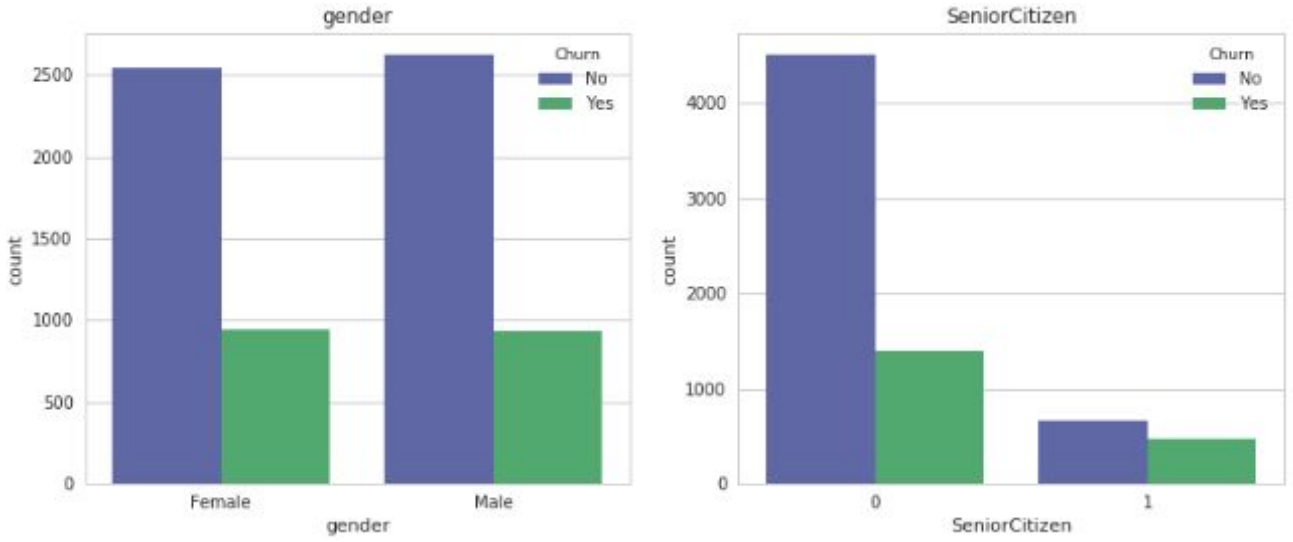


Given below is the boxplot for TotalCharges:

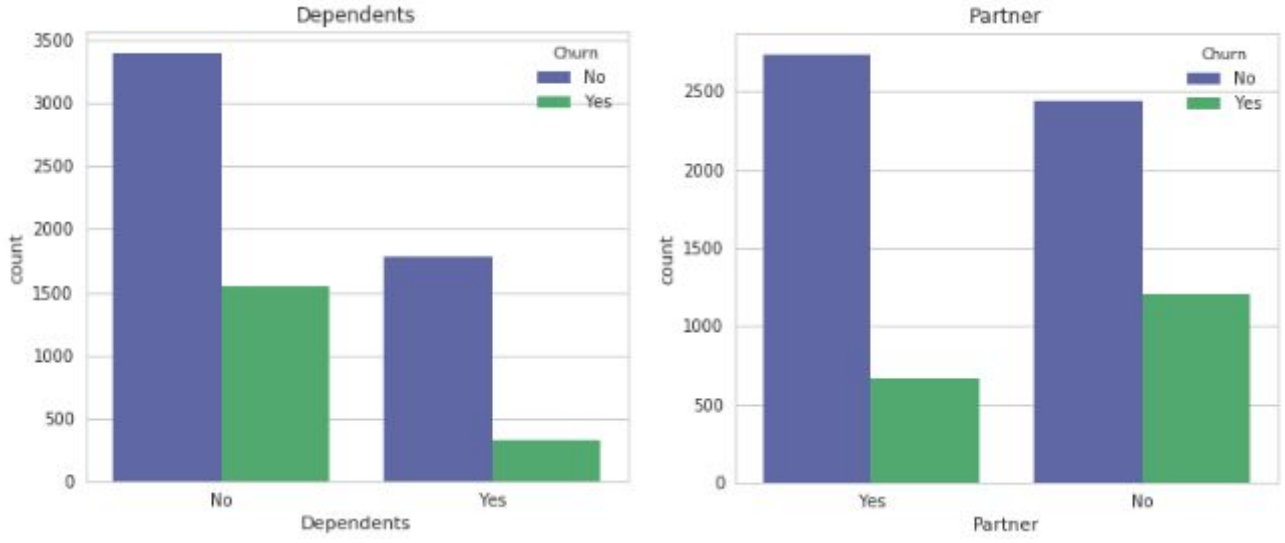


Let's move on now to multivariate exploratory data analysis.
Here, we will be again using histograms to visualize data with respect to Churners and Non-Churners. Also, we will identify the correlation between the numerical values by using Correlation Matrix.

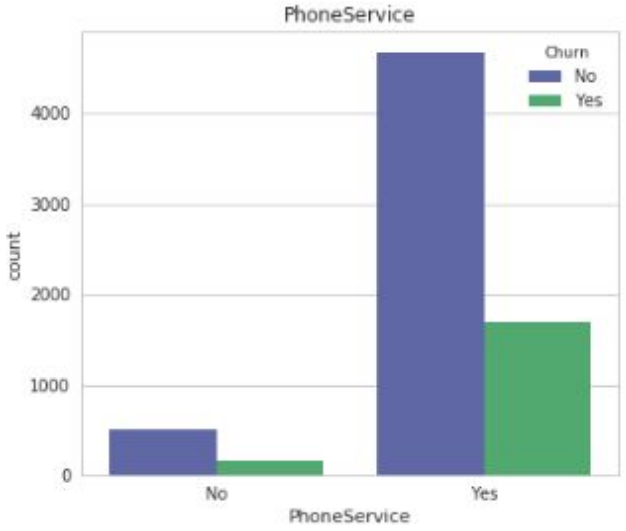
Firstly, plotting the histograms of the above categorical features with respect to if the customer is churning or not.

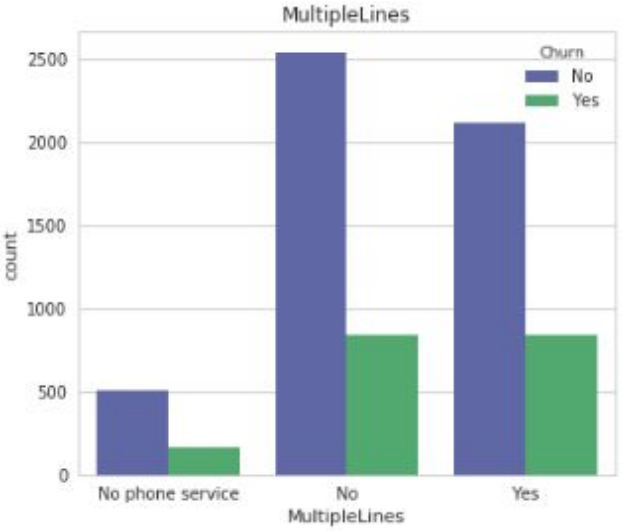


It can be seen from this that both male and female customers are equally likely to churn. The churn ratio is similar for both the genders. From the SeniorCitizen graph, you can see that the churn ratio is very high for senior citizens.

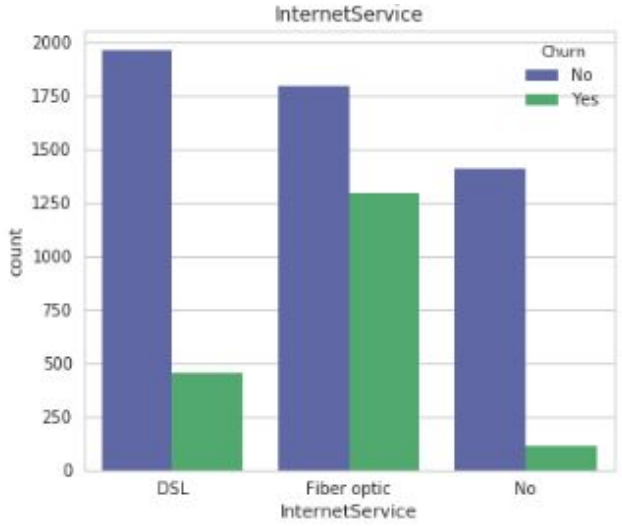


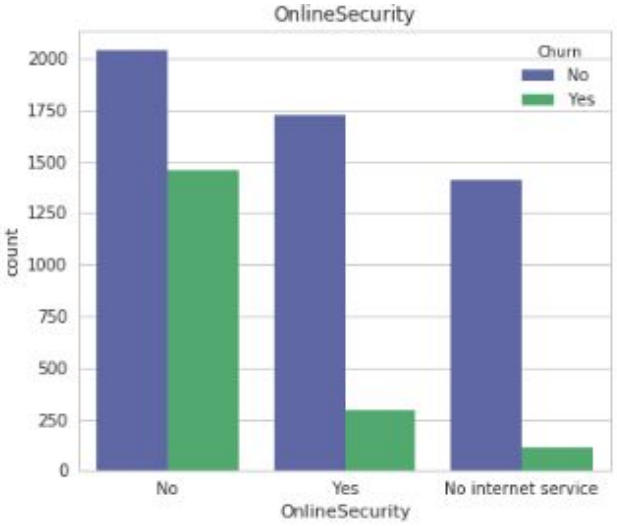
There is a higher churning seen in people without dependents. If the customers do not have partners, then the churn ratio is high for them.



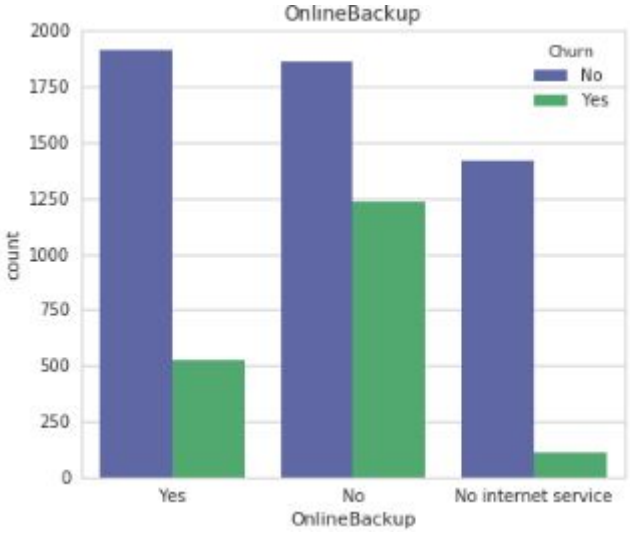


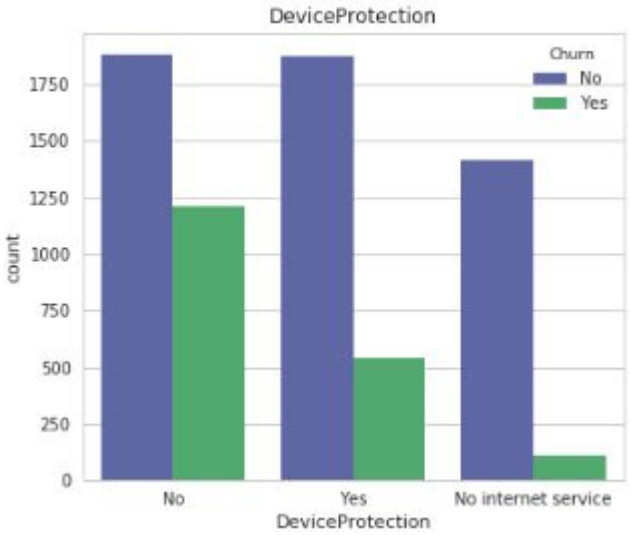
The churn ratio is slightly higher for people who are having multiple lines connection compared to those with a single line connection.





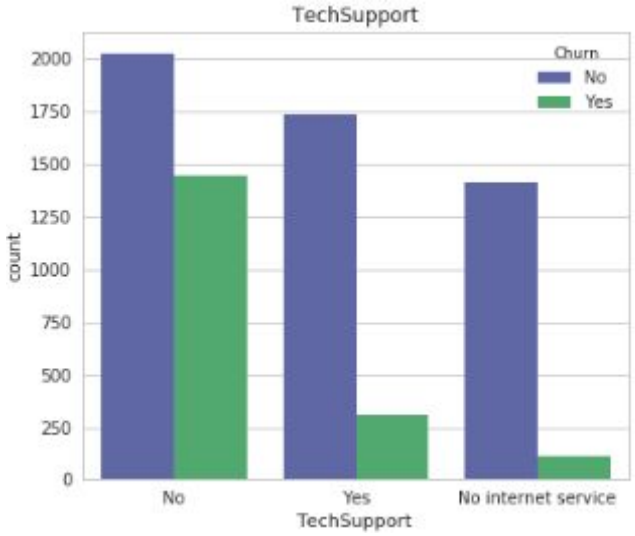
We can see that people with a fibre optics connection are churning much more compared to the customers opting for DSL broadband. Another important trend noticed here is that the customers who did not opt for online security is churning in large numbers in comparison with those with the online security product.

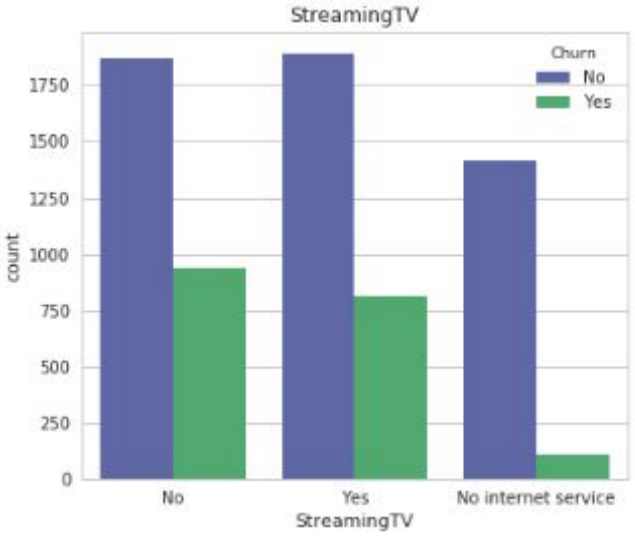




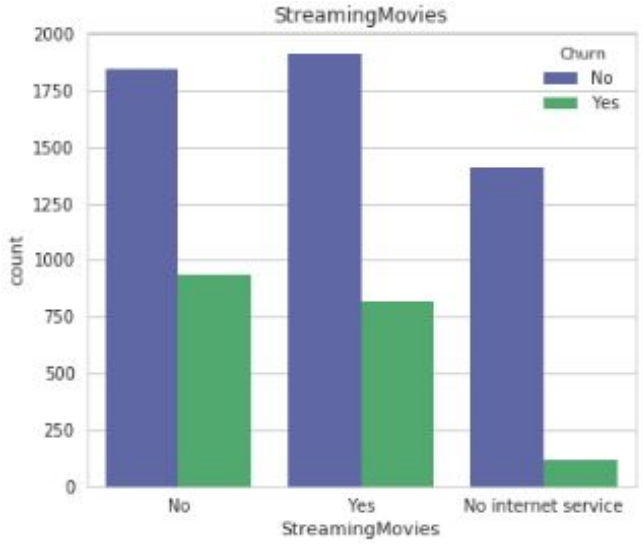
Here, in both OnlineBackup and DeviceProtection, you can see that it is following the same trend of OnlineSecurity. The customers who have not opted online backup and device protection are churning more.

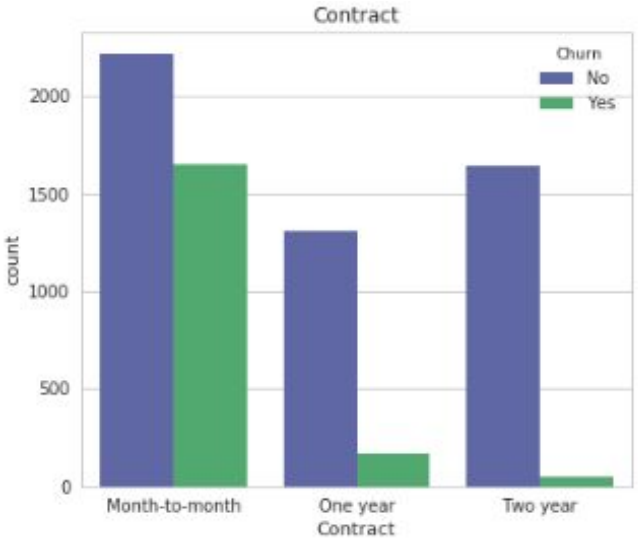
It seems that some of the important products from the company like Backup, Security and Protection are missed out by the customer and this may be a reason why they are churning. Let us explore the data further and see if the trend follows for other products of the company as well.



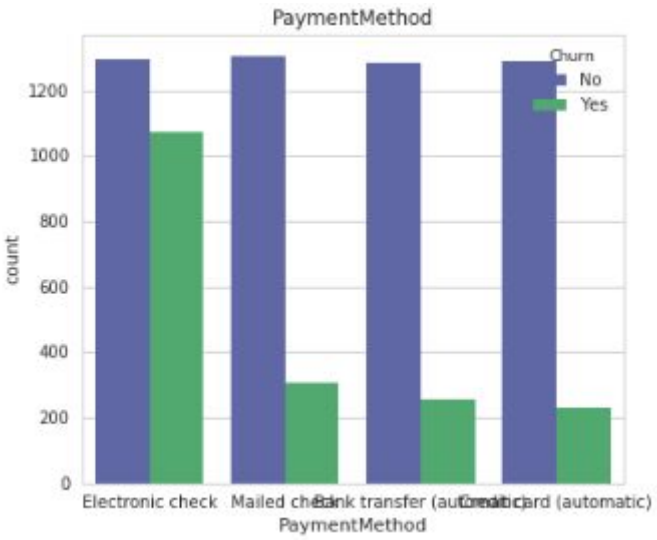


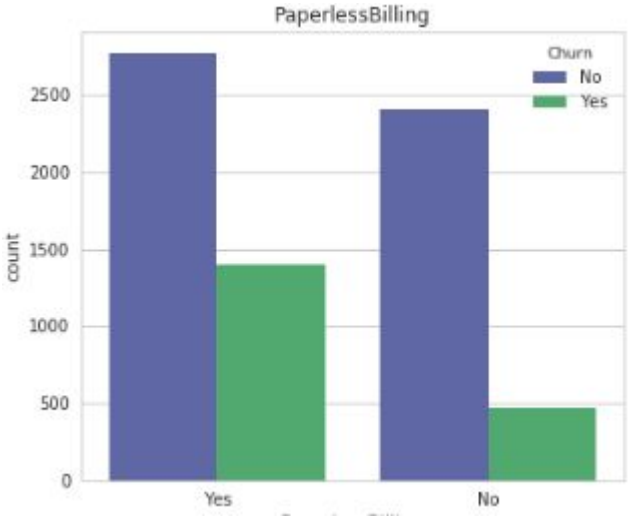
We can see that the customers without TechSupport are churning in large numbers.





Another important trend is that the customers whose contract is monthly tend to churn more.





Customers paying their bills through paperless billing are having a higher churn ratio. Also, the customers paying the bills via Electronic check have a very high churn ratio.

Now, for numeric values, let us make a correlation matrix and see the results.

We can see that, as expected, the correlations are positive. Moreover, obviously TotalCharges and tenure are highly positively correlated. So are TotalCharges and MonthlyCharges.

One more thing that we noted from the data is that the Fibre Optic Customers have a very high monthly bill.

What we can understand from the exploratory data analysis is that:

1. Gender does not matter in customer attrition in this case.
2. Senior Citizens are more likely to churn.
3. If a customer has a partner, then he/she is less likely to churn.
4. Customers without dependents are more likely to churn than settled people with dependents.
5. Customers with an Optical Fibre Internet Connection are highly probable to Churn. Will need to check this for possible reasons.
6. Customers without Online Security is highly likely to churn. Possible reason could be security breaches and frequent connectivity issues due to it.
7. Customers without online backup are also likely to Churn.
8. Customers without device protection are likely to Churn.
9. Customers without Tech Support are likely to Churn.
10. Customers with a month-to-month contract are the most frequent churners.
11. Customers using paperless billing churn more that those who transact in cash.
12. Customers paying their bills using electronic checks are highly likely to churn.

6. DATA PREPROCESSING

Data Preprocessing is the most important part of the CRISP-DM methodology which we are following here. Here, in this case, we need to correct the datatype of the input features, check and impute the missing values, handle categorical data, perform feature selection and feature scaling then, finally do the data partitioning for hold out method.

6.1. Converting Data type:

The TotalCharges feature was read as an object instead of a numerical value. So we needed to parse it into a numeric value for each of the records. The pandas.to_numeric() method did this in a single line. The objects got converted into float values. Once it became numeric, we are ready for further preprocessing.

6.2. Checking for null values and imputing:

We can see that on converting the datatype, we got eleven null values in the TotalCharges feature. These null values have to be either imputed with mean or median or these records should be dropped. However, on further probing by sorting the TotalCharges, we realized that the null values are present in only those eleven cases where the tenure is also zero. This means that the customers have not completed their tenure, hence have not been billed. This is the reason why we decided to impute them with zeroes - to preserve this meaning.

6.3. Separating the Input Variables and the Dependent Variable:

Now, we split the dataset vertically so that we can separate the independent variable - Churn, from the rest of the data.

6.4. Handling Categorical Variables:

Categorical variables are those variables that can only take values from a finite set of values called its domain. Here, all of the categorical variables are nominal as opposed to ordinal. Moreover, the categorical values of the variables are represented as “Yes”, “No”, “No internet service”. That is, they are represented as character strings. So firstly, they have to be converted to numbers and then, since they are nominal variables, we have to do one hot encoding on them. After converting to numbers, you will get values like 0, 1, 2 for the variables. After one hot encoding, you will get multiple variables in place of one, with one out of the two binary values for each.

6.5. Feature Selection:

Here, we are discarding the unimportant features. A feature can be considered as unimportant if it either does not contribute any meaningful data to the classifier so that the classifier can associate it with the target (for example, the primary key CustomerId))or if the contribution is so less that the added complexity is a worse trade-off which can be sacrificed without much loss in accuracy. We use the $\chi 2$ value to perform feature selection. After feature selection, we discarded features like PhoneService, Gender, StreamingTV, StreamingMovies, MultipleLines and InternetService.

6.6. Feature Scaling:

The features may have different ranges. Feature scaling can be used to standardize the range of input variables. Some classifiers uses Euclidean distance which requires the data to be scaled. This is highly sensitive to the magnitude of the data. Moreover, methods like Gradient Descent converges much faster with scaled data. We are using the sklearn StandardScaler to scale the data. This transforms the data such that it will have a zero mean and unit variance. Standard Scaler does the following

$$x_{stand}=x_i\text{-mean(f)/std(f)}$$

6.7. Data Partitioning:

Data partitioning means how we are splitting the data into training set and testing set. We are doing both the regular hold out method with the sklearn train_test_split and also we are doing 10 fold cross validation for model selection. In hold out, our split ratio is 7:3 with a fixed seed so that we get the same split on each trial.

7. RESULTS AND DISCUSSIONS

The results of this project can be divided into two parts - the results of exploratory data analysis and their discussion and the results of data modeling, their accuracies and their discussion.

Let's start with the discussion of the results of EDA which was given in the end of EDA.

We performed EDA and found certain trends in the data. Now the question is - based on those trends, what can we conclude? What insights can be used to prescribe to the company, which is our customer, the possible ways to prevent customer attrition? The answer to this is given below:

1. Give senior citizens more incentive to stay on as customers.
2. For customers without dependents, they might move places often. Make connection relocation cheaper and easier.
3. Reduce the cost of Optical Fibre Internet Connection.
4. Make online security a default feature. It will reduce churn and will make customers more secure.
5. Similarly make online backup also accessible to customers.
6. Device Protection should be a default feature as damaged devices will lead to a possible churn.
7. Make Tech Support available to each customer.
8. Try to convert month-to-month contract customers to yearly contract customers. Provide more year-based payment plans and offers.
9. Find out the reason why the customers who opt for paperless billing and electronic checks churn more. Maybe a fault in the company's online billing process.

Now, with EDA and the corresponding prescriptions done, let's move on to the results we got in data modeling. Furthermore, we will decide on a final model which will be used for deployment using the t - stat test.

The classification models we built here for this Customer Churn Prediction Problem are the following:

1. Logistic Regression
2. Naive Bayes Classifier
3. K - Nearest Neighbor Classification
4. Support Vector Classifier
5. Logistic Regression Classifier in Principal Component Space
6. Support Vector Classifier in Principal Component Space
7. Support Vector Classifier with RBF Kernel
8. Decision Tree Classifier
9. Random Forest Classifier
10. Xtreme Gradient Boosting Classifier
11. Multi Layer Perceptron Classifier

For selecting the hyperparameters, we used grid search in python. Grid search is a brute force based technique where you initialize a list with all the parameter values you want to try out for your model instances. Then you pass that to the function that calls the models with different permutations of all the parameters you chose. Note that you should keep the seed or random_state as a constant value. It finds a tuple of optimal parameters for you to fit on the

model. Grid search can be considered as a kind of methodical trial and error. Interestingly, we noticed that in some cases, the best results were given by the default parameters itself. It seems that they are set as default in sklearn for a reason.

We are using Confusion Matrix and Receiver Operating Characteristic Curve to evaluate the models.

Confusion Matrix:

Confusion Matrix is one of the preferred method to evaluate a classifier - especially the outcome of a binary classifier. A typical confusion matrix looks like this:

TN	FP
FN	TP

True positives (TP): These are cases in which we predicted a churn, and they do churn.
True negatives (TN): We predicted a non-churn, and they don't churn.
False positives (FP): We predicted a churn, but they don't actually churn.
False negatives (FN): We predicted a no-Churn, but they actually do churn.

Receiver Operating Characteristic Curve:

A receiver operating characteristic curve, i.e., ROC curve, is a graphical plot that illustrates the classifying ability of a binary classifier system. The ROC curve is created by plotting the true positive rate (TPR) against the false positive rate (FPR) at a single or various threshold settings. As accuracy cannot be a foolproof metric to evaluate the classifying power, we use another metric - AUC, the Area Under receiver operating characteristic Curve. The higher the area, the better the classifier is.

The models used and their results are listed below

7.1. Logistic Regression:

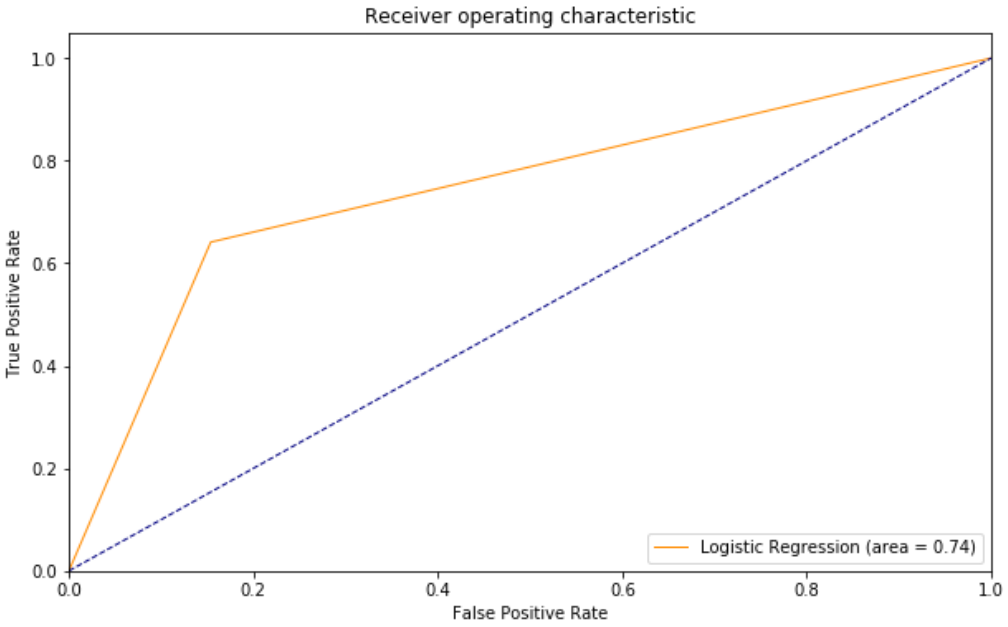
We used the sklearn LogisticRegression from linear_model and fitted it with our training data in hold out method as well as 10 fold cross validation partitioning. We got two accuracies - the first one is the accuracy we got from the hold out method. Secondly, we got a set of 10 different accuracies - one for each fold in 10 fold cross validation.

The Confusion Matrix for Hold out is:

1392	168
253	300

Here, the accuracy is 80.075 %

For this, the ROC is given below:



AUC for logistic regression model is 0.74

After 10-Fold Cross Validation, the accuracies are:
0.8097166, 0.80364372, 0.82388664, 0.78340081, 0.81135903, 0.79716024, 0.79268293, 0.79674797, 0.79878049, 0.84552846.
The mean accuracy is 0.8062906880454601 ie 80.629 %
The mean true positive rate is 0.6624278647287172 ie 66.24 %
Logistic Regression, as expected, is giving us a very good accuracy.

7.2. Naive Bayes Classifier:

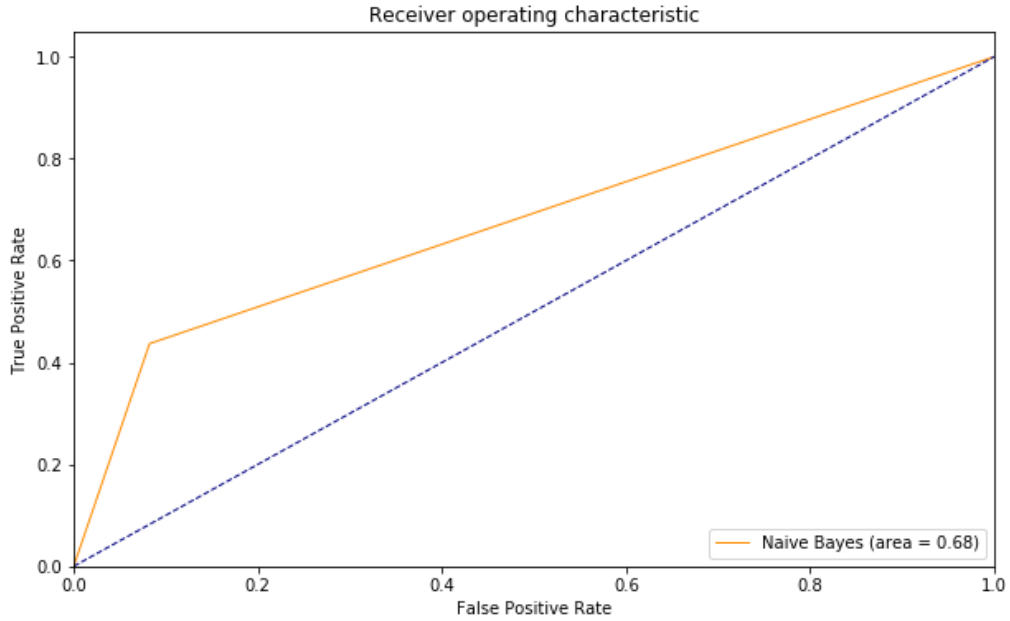
We used the sklearn GaussianNB from naive_bayes in classifying the data. As usual, we are performing it on hold out as well as cross-validated partitions that results in one accuracy and one set of ten accuracies. We kept the priors parameter as None.

The confusion matrix for hold out is:

958	602
86	467

We can see that the naive bayes model is performing the worst here with an accuracy of 67.43%

The ROC for naive bayes model is:



AUC for naive bayes model is 0.68

After 10 - fold cross validation, the accuracies are:
0.68016194, 0.69230769, 0.71052632, 0.64979757, 0.68356998, 0.65922921, 0.68495935, 0.65650407, 0.70934959, 0.7195122

The mean accuracy is 0.6845917914160212 ie 68.459 %
The mean true positive rate is 0.43907420455579127 ie 43.90 %
Our explanation for the poor performance of naive bayes model is that it does not take into account the obvious interaction effects between the attributes.

7.3. K Nearest Neighbors Classifier:

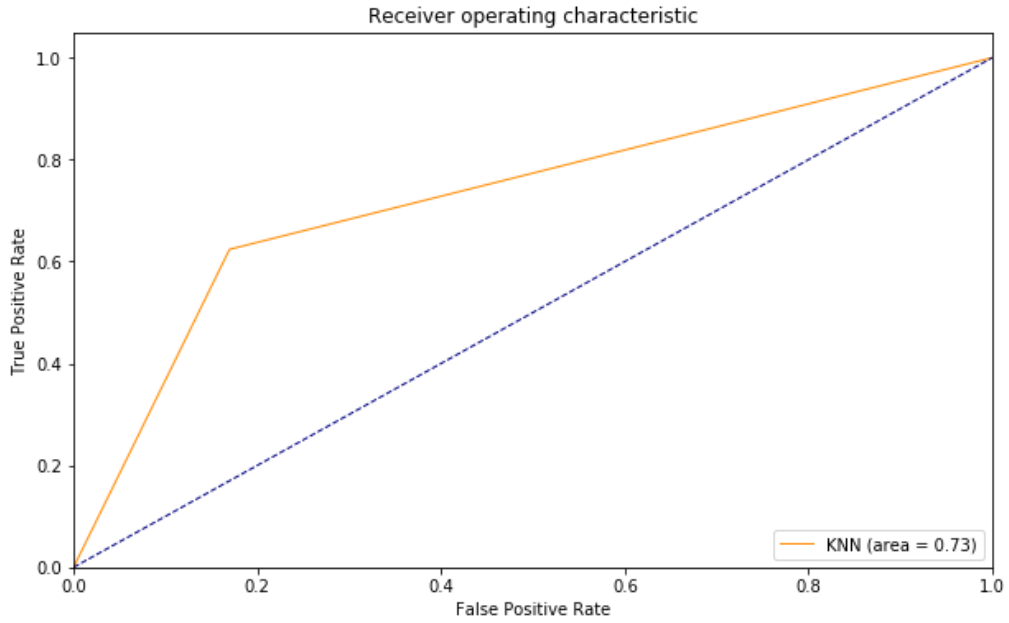
We used the sklearn KNeighborsClassifier from neighbors package. The best parameters returned by grid search were Euclidean distance (Minkowski with p value = 2) with 20 neighbors. As usual, we did modeling for both hold out and 10 fold cross validation. The results are summarized below.

The confusion matrix for hold out is:

1399	161
286	267

Here, the accuracy is 78.845 %

The ROC for K Nearest Neighbors Model is:



The AUC is 0.73

After 10 fold cross validation, the accuracies are:
0.79757085, 0.78744939, 0.80769231, 0.78340081, 0.79310345, 0.75862069, 0.77845528, 0.7804878 , 0.77235772, 0.80894309
The mean accuracy is 0.7868081400693945 ie 78.680%
The mean true positive rate is 0.6287008186469414 ie 62.87 %

7.4. Support Vector Classifier:

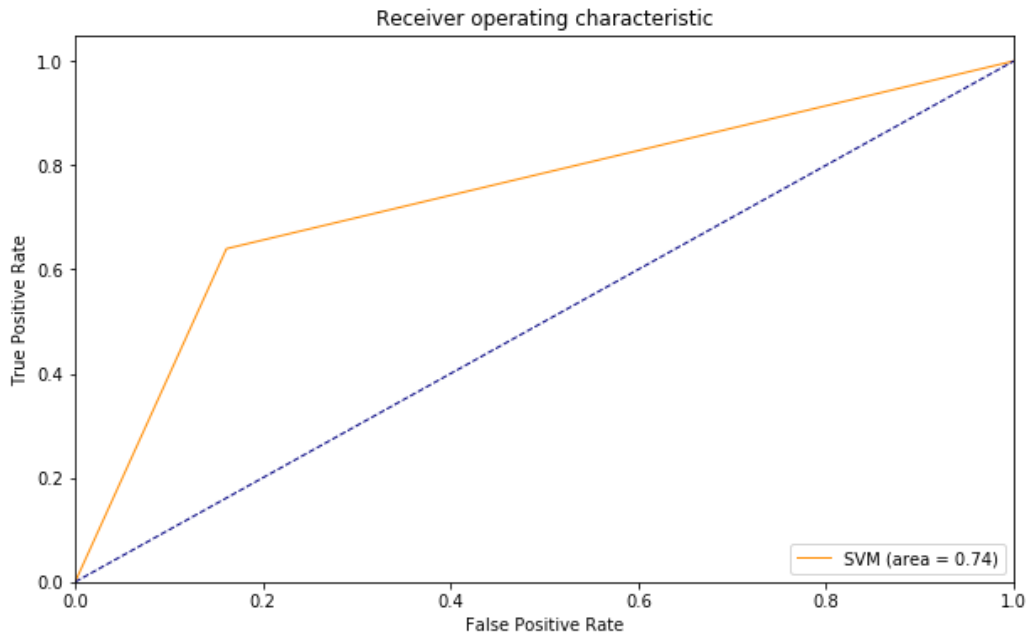
We are using the SVC class from sklearn.svm. In this classifier instance of SVM, we set the kernel parameter to linear. As usual, we did modeling for both hold out and 10 fold cross validation. The results are summarized below.

The confusion matrix that resulted from hold out partitioning is given below:

1400	160
269	284

Here, the hold out accuracy is 79.697%

The ROC for Support Vector Classifier is:



After performing 10 fold cross validation, the accuracies from each fold are:
0.80364372, 0.77732794, 0.81174089, 0.76315789, 0.80121704, 0.77079108, 0.78658537, 0.79471545, 0.77845528, 0.83943089
The mean accuracy is 0.7927065550804312, ie 79.27 %
The mean true positive rate is 0.6409408809261288 ie 64.09 %
One thing we noted here is that SVM comparatively took a much longer time compared to other models. Grid search and cross validation took much longer as we are fitting the model multiple times with different parameters and data respectively.

7.5. Logistic Regression in the Principal Component Space:

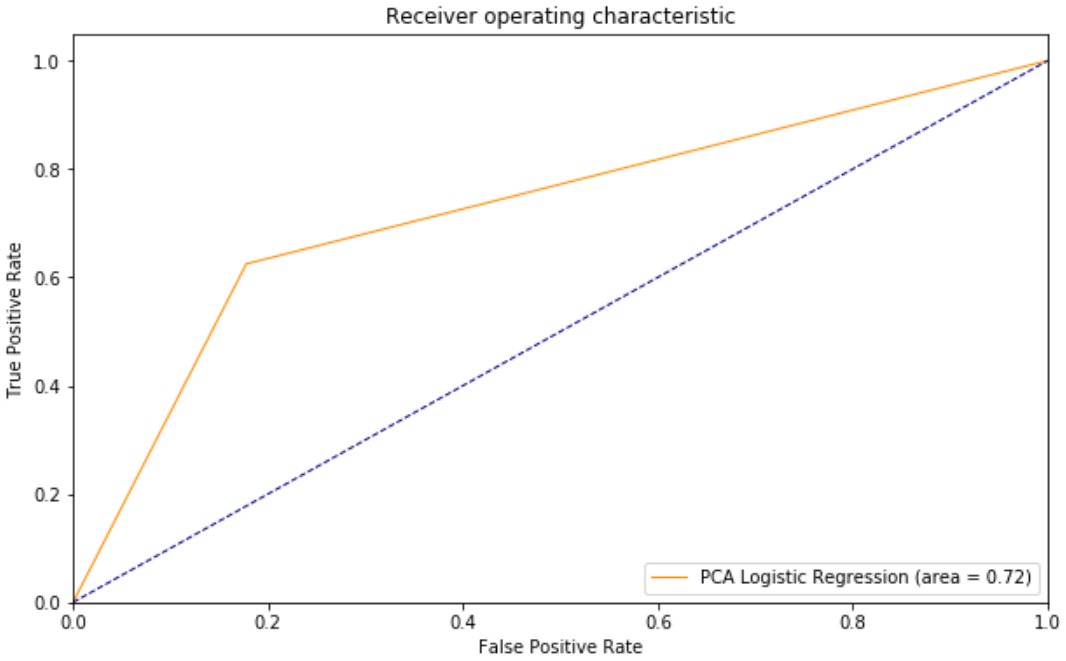
We used the sklearn decomposition to perform PCA using the PCA class. Then applied that principal component space as input to a LogisticRegression instance to fit it. Then, as usual, we did modeling for both hold out and 10 fold cross validation. The results are summarized below.

The confusion matrix for Hold Out is given below:

1411	149
305	248

Here, the accuracy is 78.513%

The ROC is given below:



After performing 10 fold cross validation, the resulted accuracies are listed below:
0.77935223, 0.78340081, 0.80566802, 0.78137652, 0.79918864, 0.76876268, 0.77845528, 0.78861789, 0.78252033, 0.83333333
The mean accuracy is 0.7900675718576912, ie 79 %
The mean True Positive Rate is 0.6344944067157344 ie 63.44 %
We noticed that the accuracy of Logistic Regression in fact decreased slightly after doing PCA. This could be because the features may be non-linearly related. PCA works best only when the data is multicollinear.

7.6. Support Vector Classifier on Principal Component Space:

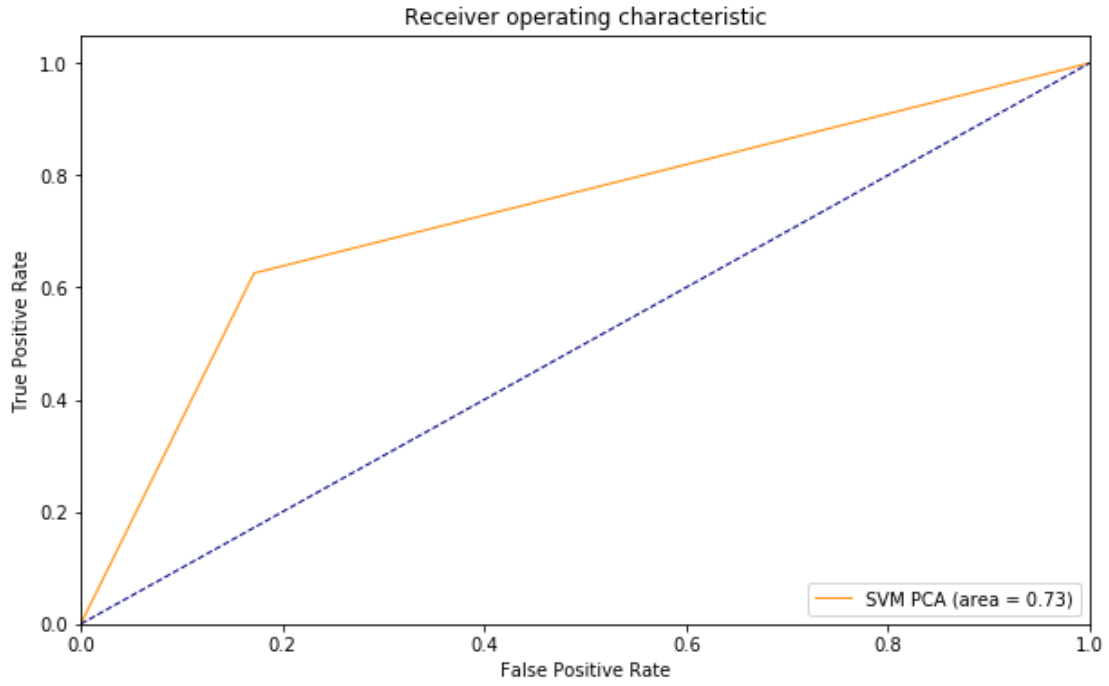
We used the same Support Vector Classifier used early with the same parameters so that we can compare again how PCA works with it. Here is what we got:

The confusion matrix for Hold out partition is given here:

1403	157
291	262

Here, the accuracy is 78.79 %. Here also the accuracy decreased.

The ROC is given below



The accuracies from 10 fold cross validation are listed below:
0.78137652, 0.78137652, 0.81174089, 0.77935223, 0.79107505, 0.76673428, 0.77642276, 0.79065041, 0.77845528, 0.83739837
The mean accuracy is 0.789458231374325 ie 78.94 % which is still less than the accuracy of the SVM without applying PCA.
The mean true positive rate is 0.6287510118549975
With the above two models we are concluding that applying PCA here is useless.

7.7. Support Vector Classifier with RBF kernel:

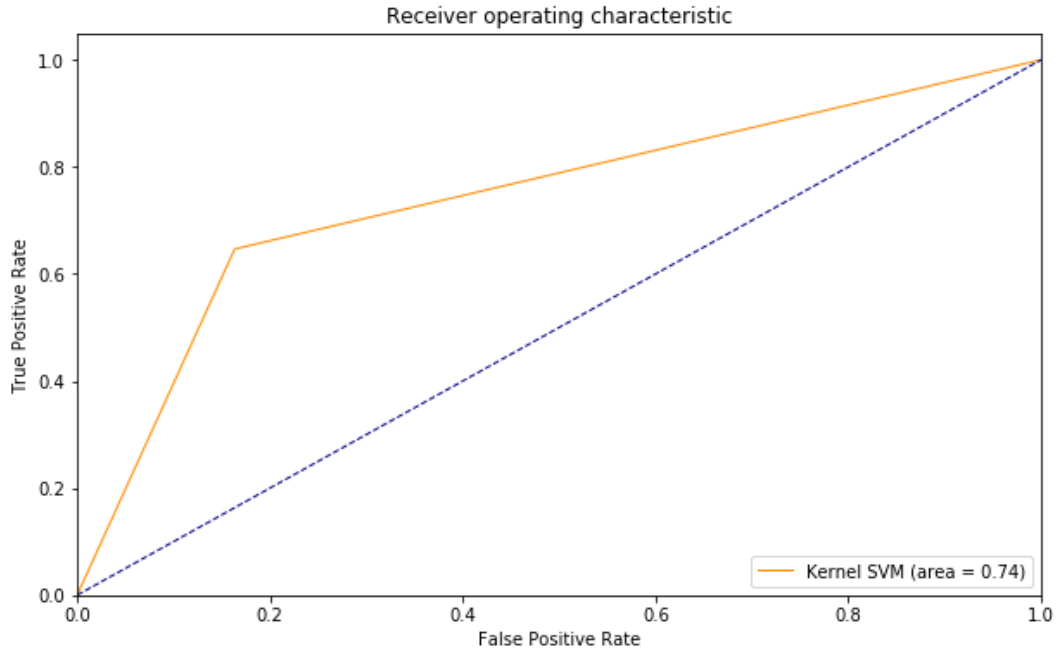
We can use the same scikit package and class for kernel svm as well. The only difference is that we have to explicitly mention which kernel and degree we are using for making an instance of the class. We are using the RBF kernel function with a degree of 3, which was found out by grid search.

The confusion matrix for hold out is given by:

1408	152
275	278

The accuracy here is 79.791%

The ROC is given below:



The list of accuracies after 10 fold cross validation is:
0.81174089, 0.79959514, 0.81983806, 0.78340081, 0.81135903, 0.77890467, 0.78252033, 0.78861789, 0.7804878 , 0.82723577
The mean accuracy is 0.7983700379086882 ie. 79.83%.
Although there is a very small increase in accuracy, we will not prefer this as this is computationally more complex than linear SVM.
The mean true positive rate is 0.6719098493604446 ie 67.19 %

7.8. Decision Tree Classifier:

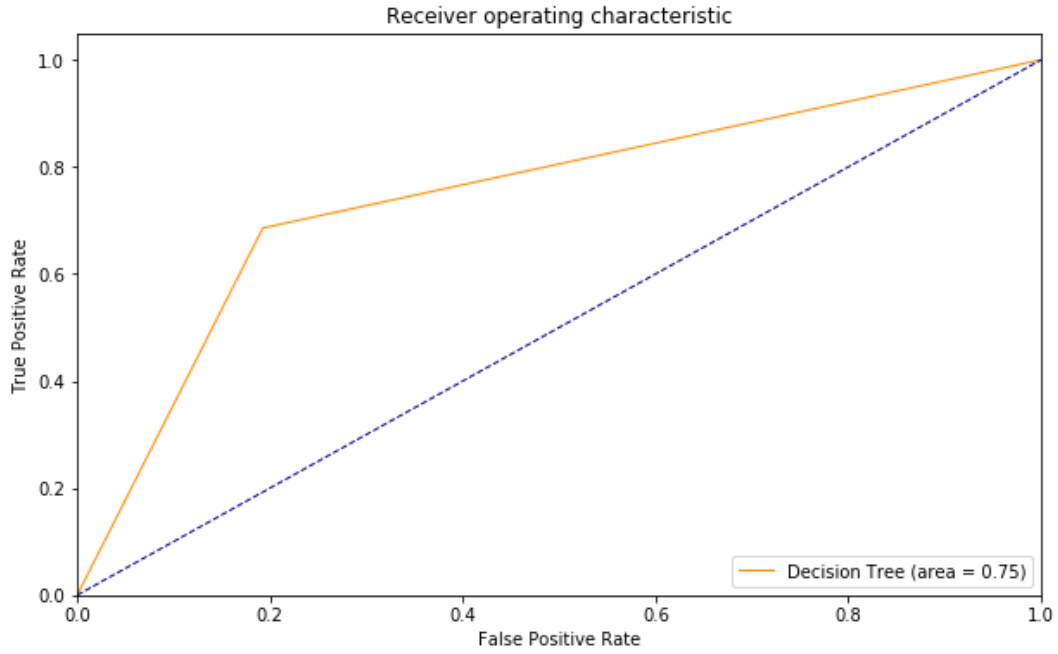
We are using the sklearn DecisionTreeClassifier from tree. The criterion we finalized which gives better results is entropy. As usual, we are doing hold out as well as 10 fold cross validation.

The confusion matrix for hold out partitioning is:

1467	93
350	203

The accuracy here is 79.03%.

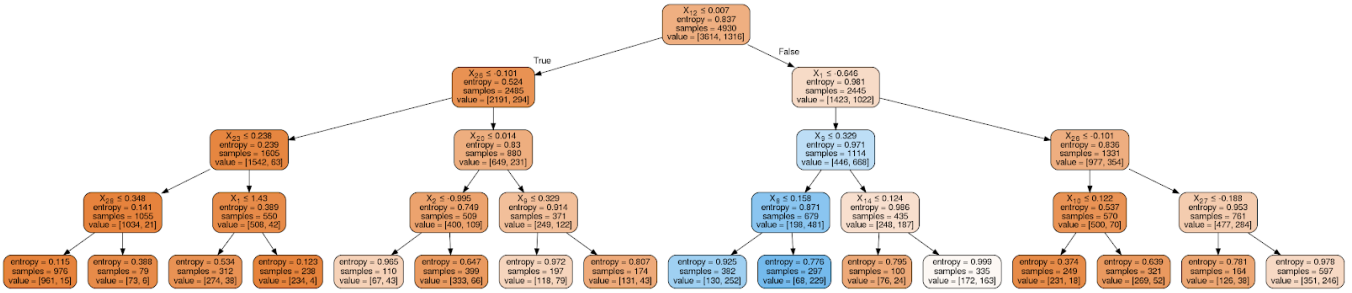
The ROC is :



After cross validation, the accuracies we are getting are:
0.78947368, 0.78340081, 0.81174089, 0.76923077, 0.79513185, 0.77484787, 0.76422764, 0.78455285, 0.77845528, 0.82520325

The mean accuracy is 0.727580808919421 ie 78.76%
The mean true positive rate is 0.495477550790957 ie 49.54 %

We were able to visualize the decision tree using export_graphviz and pydotplus. This is what we got:Contract, InternetService, Tenure, TechSupport, TotalCharges and MonthlyCharges



The top features are

7.9. Random Forest Classifier:

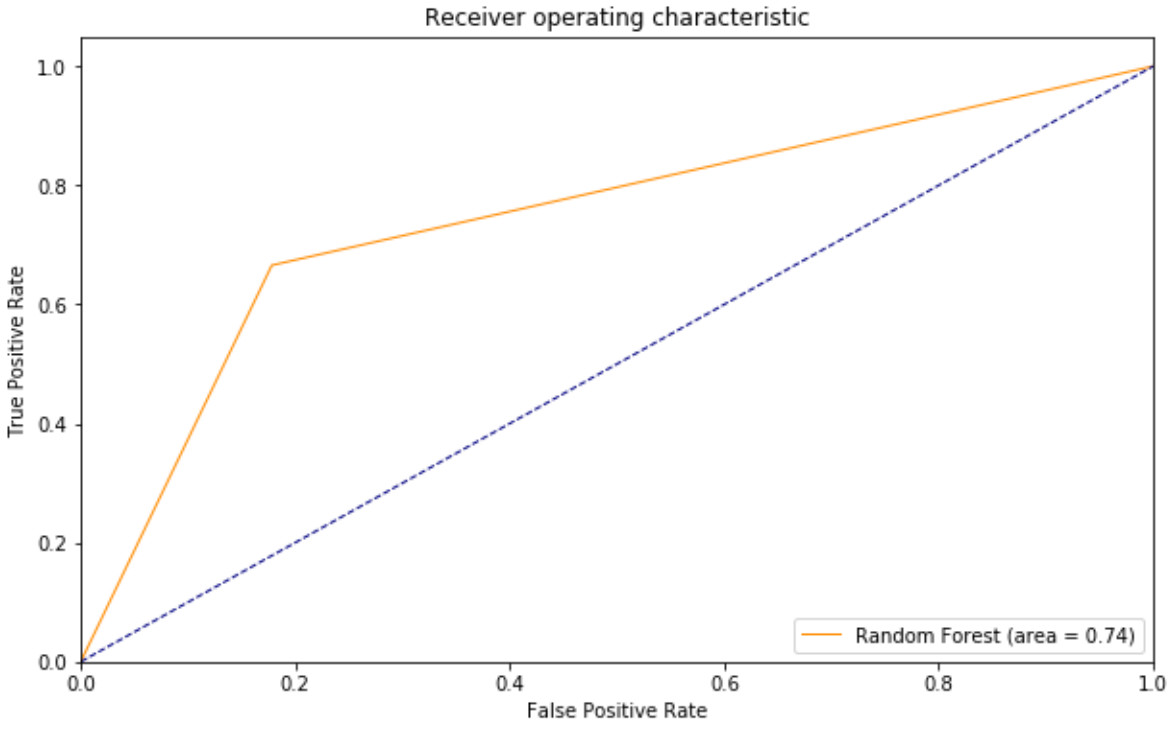
Random forest is one of the ensemble algorithms we tried out on the dataset. We used the sklearn RandomForestClassifier found in sklearn.ensemble. We used 10 individual and independent trees as n_estimator parameter value. The criterion used here is the same as that used in a single decision tree, which is entropy.

The confusion matrix for hold out method is:

1439	121
312	241

Here, the accuracy is 79.50%

The ROC is :



After cross validation, the list of accuracies are:
0.80566802, 0.78947368, 0.79757085, 0.77530364, 0.81135903, 0.77890467, 0.77845528, 0.77642276, 0.78252033, 0.83739837
The mean accuracy is 0.0.7933076633983033 ie. 79.33 %
The mean true positive rate is 0.6232301050218866 ie 62.32 %

7.10. Xtreme Gradient Boosting Classifier:

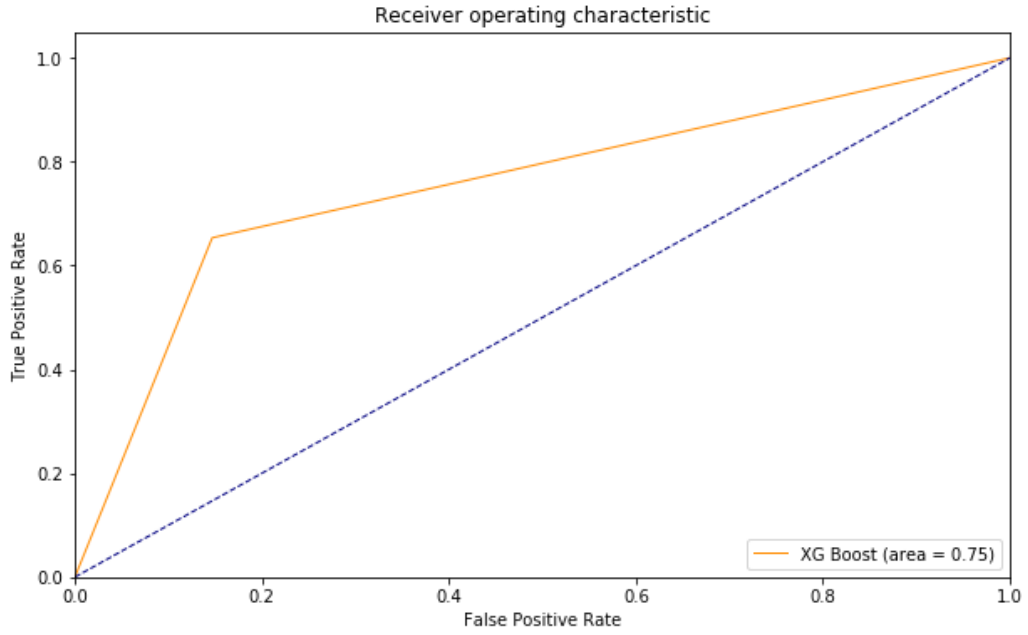
XGBoost is an ensemble learning technique which uses multiple decision trees. However, instead of using them in parallel as in random forests, here, it is used sequentially. The output of one decision tree becomes the input for the next with one goal - to minimize the error. Both the execution speed and model performance is very good for xgbClassifier. We had to install xgboost package in our conda virtual environment to import it at runtime. The chosen parameters are a max_depth of 4 and learning rate of 0.1 to prevent overfitting.

The confusion matrix for the hold out method is:

1394	166
240	313

The accuracy here is 80.785 %

The ROC for xgbClassifier is :



After cross validation, the list of accuracies are:
0.80769231, 0.80364372, 0.82186235, 0.79959514, 0.82150101, 0.79107505, 0.78455285, 0.79065041, 0.79471545, 0.83739837
The mean accuracy is 0.8052686660346661 ie 80.52 %
The mean true positive rate is 0.6641198835494618 ie 66.41%
As expected, xgb is another ensemble learning method that gave us a high accuracy. Along with a high accuracy, xgb runs very fast despite being a sequential ensemble learning algorithm.

7.11. Multi-Layer Perceptron:

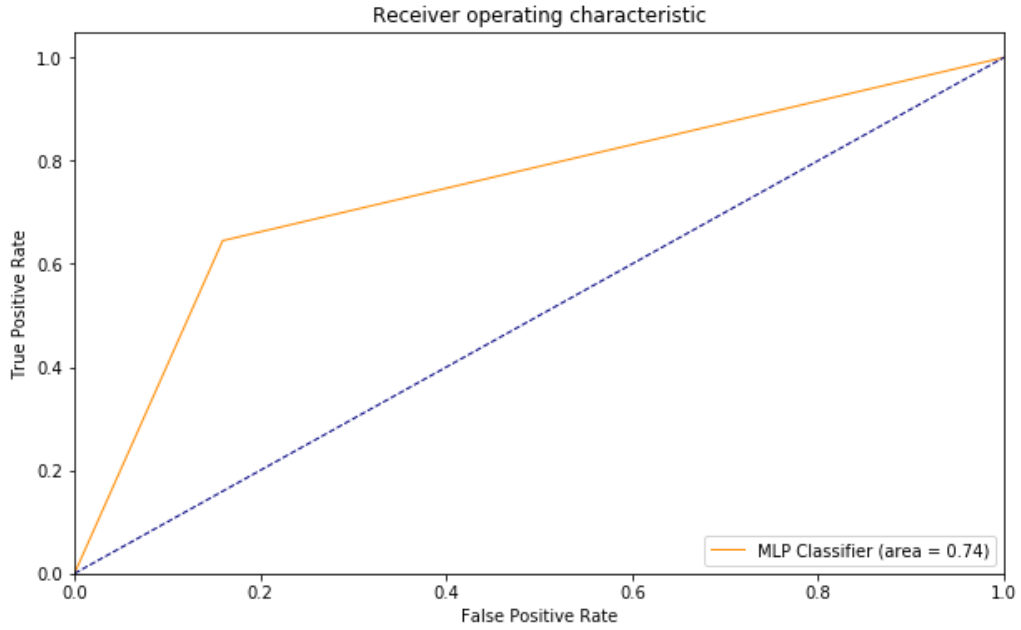
This is the only neural-network solution we used. We used the sklearn MLPClassifier located in the neural_network package. After grid search, the chosen parameters were relu function for activation function, stochastic gradient descent solver, and a maximum iterations of 85 as there was not much change after that. We used the verbose switch to visualize the training epochs in the initial build, then later turned it off after finalizing the model.

Here is the confusion matrix for the hold out method:

1402	158
266	287

The accuracy here is 79.93%.

The ROC is given below:



After performing 10 fold cross validation, the list of accuracies are given below:
0.80769231, 0.80161943, 0.81376518, 0.79149798, 0.79107505, 0.78701826, 0.79065041, 0.77845528, 0.80894309, 0.83739837
The mean accuracy is 0.8008115359545002 ie. 80.08 %
The mean true positive rate is 0.6574884155991074 ie 65.74 %

Model Selection Using t - test:

Student's t-test, in statistics, a method of testing hypotheses about the mean of a small sample drawn from a normally distributed population when the population standard deviation is unknown. If we have the mean and the variance of a list, in this case, the list of ten sensitivities from 10 fold cross validation, then t is given as

$$t = \frac{\mu}{\sqrt{\frac{\sigma^2}{k}}}$$

The following are the t values we got when comparing different classifiers:

SINo	Classifier 1	Classifier 2	t value
1	Logistic Regression	Logistic Regression PCA	3.85
2	XG Boost	Naive Bayes	22.44
3	Logistic Regression	K Nearest Neighbors	4.994
4	Logistic Regression	Kernel Support Vector	3.688
5	XG Boost	Logistic Regression	7.694
6	XG Boost	Multi-Layer Perceptron	3.772
7	Decision Tree	Random Forest	6.60
8	Decision Tree	Logistic Regression	6.137

8. CONCLUSIONS

We are able to draw out inferences from the data by performing Exploratory Data Analysis. It showed us how the lack of certain products like Online security and Device Protection caused customers to churn. Also we are able to identify that the cost of optic fibre internet connection is too high and that have contributed to the churn ratio as well. EDA helped us to understand what is causing the churn and what should be changed about the company's functioning such that we can avoid churn.

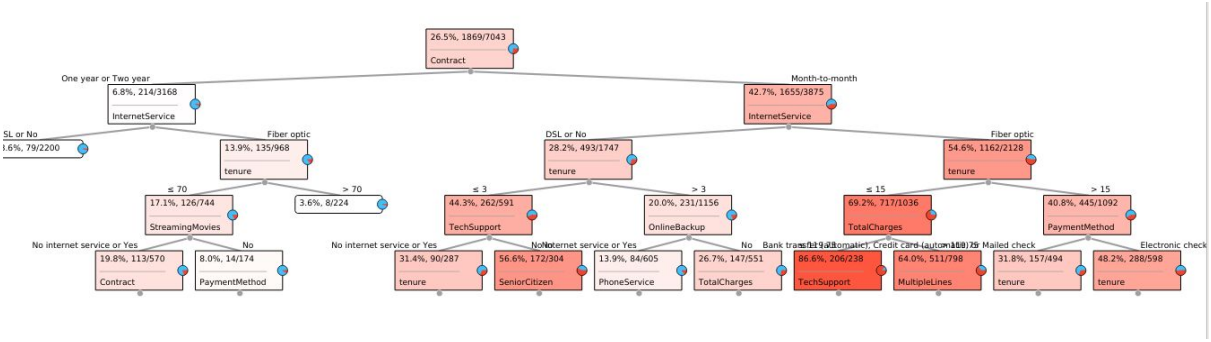
When it comes to predictive analytics, we built a total of eleven models with varying parameters and chose the best parameters for each model. We evaluated those models using both hold out and k fold cross validation partitioned data. We also proved using t-test that these models are dissimilar in terms of predicting power.

We noticed that performing PCA reduced the accuracy. Also feature selection did not really impact the predictive accuracy. Therefore we discarded some features to reduce the complexity without sacrificing much of the accuracy. Naive Bayes did not give good results for this dataset as the variables were not conditionally independent.

Despite the eleven models, the goal of this project is to find the one model which can be used to predict unseen data. That model should be computationally less complex and should be accurate in prediction. Moreover, it is desirable that the model should be easily able to explain its prediction process.

Final Model to be used for building the classification solution:

After doing t test, we found out how the models are different from each other. The most promising models are Logistic Regression, XG Boost and Decision tree. Among the top performing models, we decided to choose Decision tree classifier for the final build of the classifier as it is giving high accuracy and AUC. Moreover, decision trees will give you clear knowledge - if then rules, which you can use to identify the features that contribute the most to churn. Once the model is fixed as Decision Tree with max_depth=4 and criterion=entropy, we train it on the entire dataset. Now it is ready to predict for unseen data.



Now that the model is built, a predictive solution can be built on top of it so that the company can predict the possible churners so that they can prevent future churns.

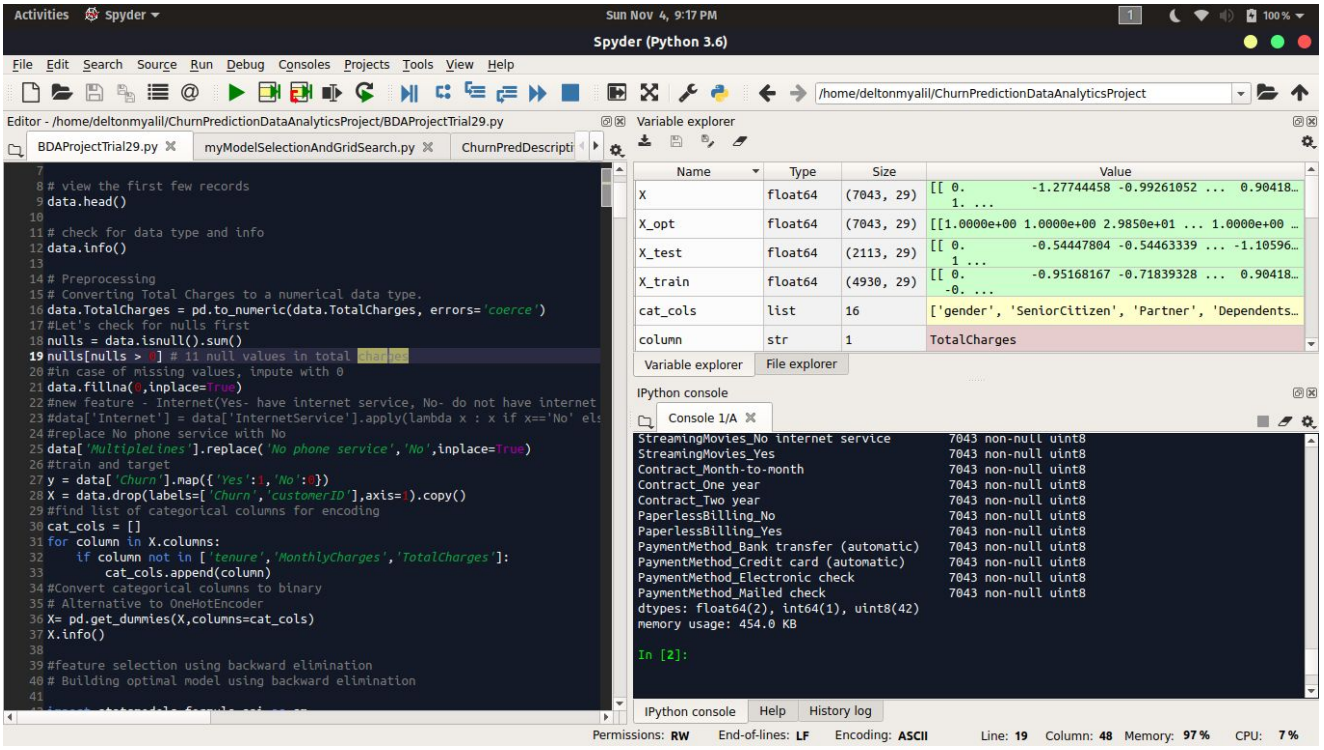
On a final note to the company, it is necessary to monitor the model to check that it does not get outdated.

9. SCREENSHOTS, TABLES AND FIGURES

9.1 Tools and Software Used

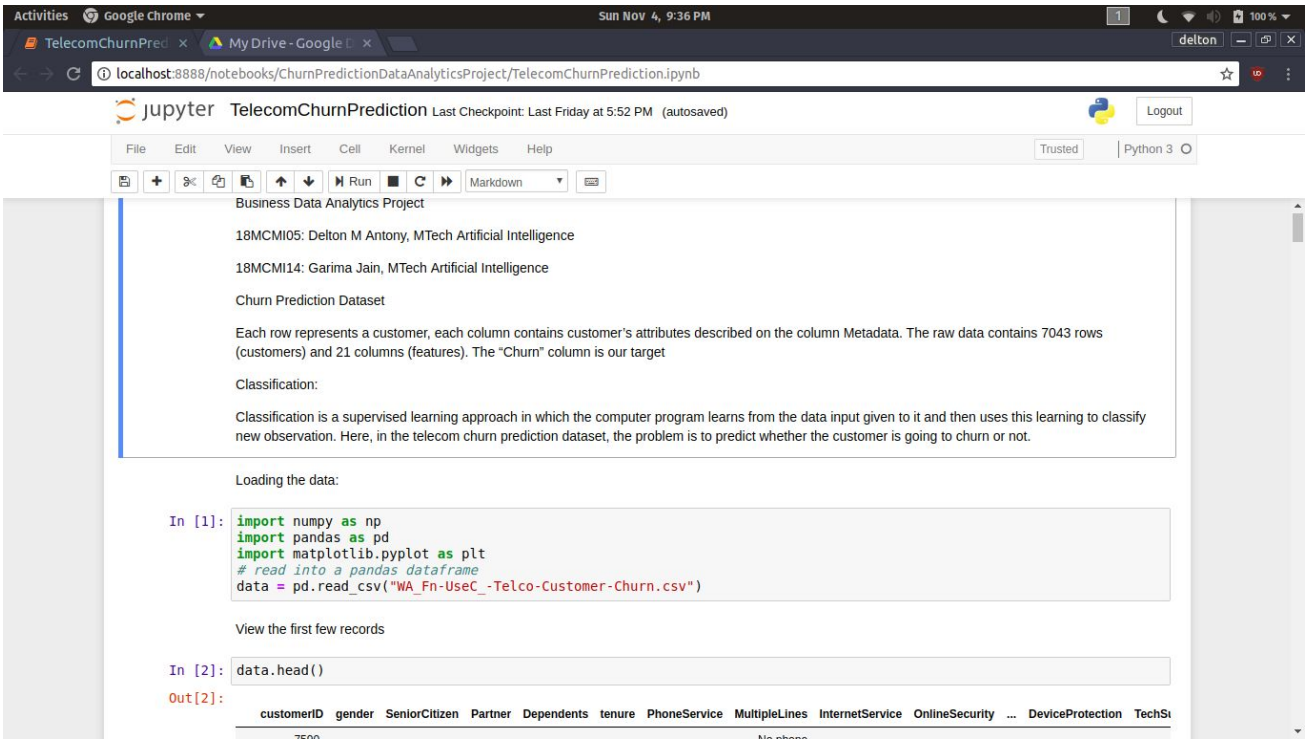
9.1.1 Spyder:

We decided to code this project in python3. One of the best editors out there for python which is suited for data science is spyder. The features of spyder include separate in window panes for editor, console and file explorer. But what sets spyder apart from other Integrated Development Environments like Eclipse and Pycharm is that Spyder also comes with a handy Variable Explorer which can be used to see the intermediate variables which you get. Spyder also supports selective code running. You can select the part of the code you want to run. Spyder comes as a default software in Anaconda for WIndows, Mac and Linux.



9.1.2 Jupyter Notebook:

In order to make a notebook file which contains both computer readable code, graphs and tables and text which describes the code and the graphs, we used Jupyter Notebook. Once the coding was complete in Spyder, we took the code block by block and executed them in Jupyter Notebook. After each logical block of code, we described the code and the output generated by it - whether it be returns or tables and graphs. You can easily save the code, its results and descriptions as an executable ipynb file or a static html, tex or pdf file. Jupyter Notebook also comes preinstalled with Anaconda. Jupyter Notebook is available for Windows, Mac and Linux.



9.1.3 Development Environment:

We developed this project in two laptops running two latest Linux distributions - Ubuntu and ElementaryOS. The hardware used are i5 and i7 processors with 4GB and 8GB RAMs respectively.

9.2 Tables and Figures

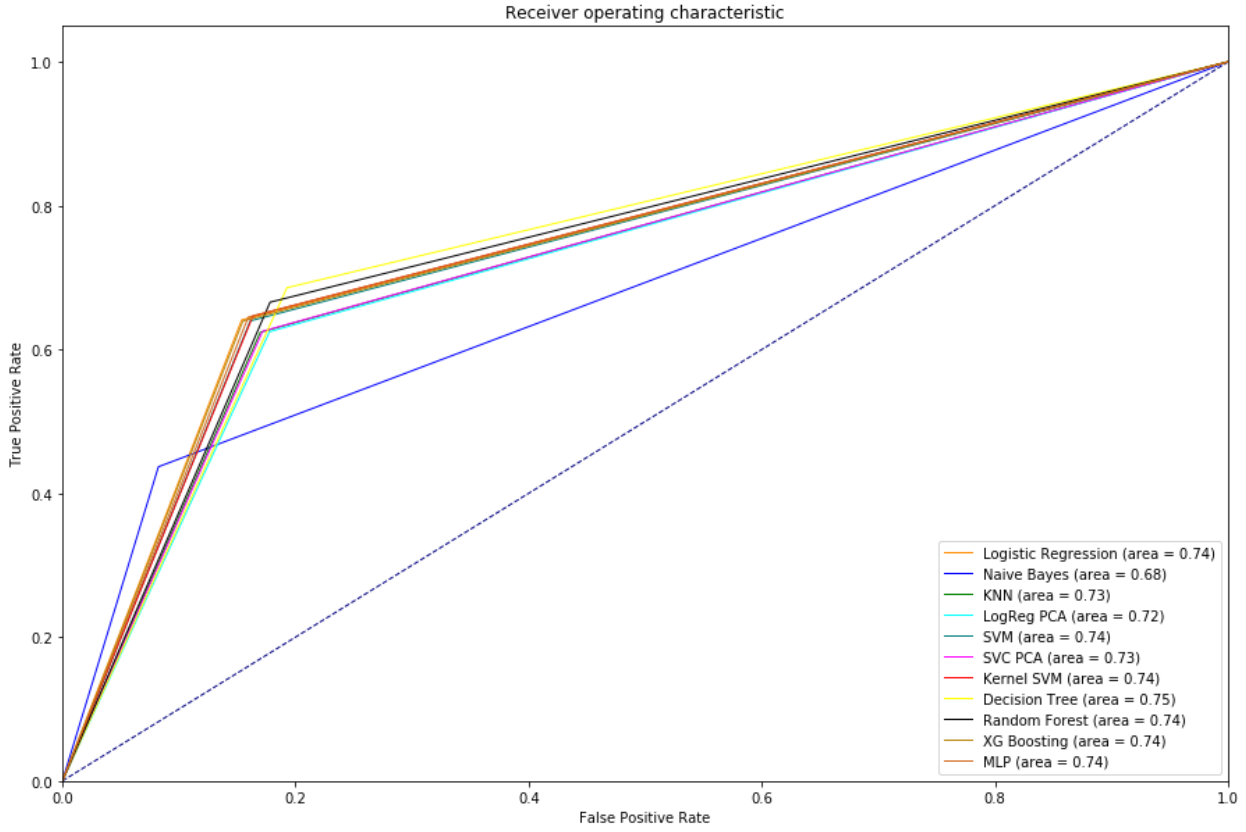
9.2.1 Summary of all the classifiers:

Here I am summarizing the accuracies of both hold out and cross validated data, sensitivity and AUC for cross validated data for each model.

SINo	Modelname	HoldOut Accuracy	Cross Validated Mean Accuracy	Hold Out Sensitivity	AUC	Cross Validated Mean TPR
1	Logistic Regression	80.075	80.63	64.10	0.74	66.24
2	Naive Bayes	67.43	68.46	43.68	0.68	43.90
3	K Nearest Neighbors	78.85	78.68	62.38	0.73	62.87
4	Support Vector Classifier	79.70	79.27	63.96	0.74	64.09
5	Logistic Regression PCA	78.52	79.0	62.46	0.72	63.44
6	Support Vector PCA	78.79	78.94	62.52	0.73	62.87
7	Kernel SVM	79.79	79.83	64.65	0.74	67.19
8	Decision Trees	72.22	72.75	68.58	0.75	66.86
9	Random Forest	78.75	77.68	62.62	0.73	62.32
10	XG-Boost	80.78	80.52	63.94	0.75	66.41
11	Multi-Layer Perceptron	79.93	80.08	65.34	0.74	65.74

8.2.2 Combined ROC graph

The following is an ROC graph that contains the curve of all the eleven models we made.



We can see from the graph that Logistic Regression and XG Boost are the best models in terms of AUC.

8.2.3 Screenshots of Python3 Code:

```
from sklearn.tree import DecisionTreeClassifier
decisionTreeClassifierEntropy = DecisionTreeClassifier(criterion="entropy", max_features=10, max_depth=4, random_sta
decisionTreeClassifierEntropy.fit(X_train, y_train)
y_predDecTreeEntropy = decisionTreeClassifierEntropy.predict(X_test)
```

Let's see the confusion matrix

```
decTreeEntropyConfusionMatrix = confusion_matrix(y_test, y_predDecTreeEntropy)
decTreeEntropyConfusionMatrix
```

```
array([[1467,   93],
       [ 350,  203]])
```

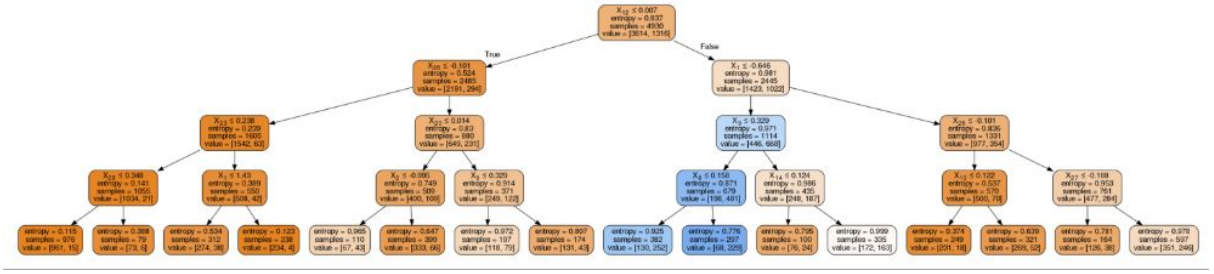
```
Here, the accuracy is 79.03%
```

Plotting the roc and finding the auc

```
fpr, tpr, thresholds = roc_curve(y_predDecTreeEntropy, y_test)
roc_auc = auc(fpr, tpr)
plt.figure(figsize=(10,6))
plt.plot(fpr, tpr, color='darkorange', lw=1, label='Decision Tree (area = %0.2f)' % roc_auc)
plt.plot([0, 1], [0, 1], color='navy', lw=1, linestyle='--')
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Receiver operating characteristic')
plt.legend(loc="lower right")
plt.show()
```

The screenshot given above shows how make an instance of the DecisionTreeClassifier class of sklearn. Setting the parameters is done during this time. Once the instance is made, you can fit it with the training data after which the classifier is ready to predict. You can see that the confusion matrix is returned as a two dimensional python3 list. The last block seen in the screenshot is how we plotted the ROC using matplotlib.pyplot.

```
from sklearn.externals.six import StringIO
from IPython.display import Image
from sklearn.tree import export_graphviz
import pydotplus
dotData = StringIO()
export_graphviz(decisionTreeClassifierEntropy, out_file=dotData, filled=True, rounded=True, special_characters=True)
decTreeEntropyGraph = pydotplus.graph_from_dot_data(dotData.getvalue())
Image(decTreeEntropyGraph.create_png())
```



The above screencap shows how we visualized the decision tree.

```
def findT(accuracy1, accuracy2):
    differences = []
    for i in range(10):
        differences.append(abs(accuracy1[i]-accuracy2[i])) #(1-x) - (1-y) = x-y
    meanDiff = mean(differences)
    varDiff = pvariance(differences)
    t = meanDiff/sqrt(varDiff/10)
    if t < 2.82: # 1% confidence level
        return False, t
    return True, t
```

The above screencap is the function defined to perform the t-test. In this, accuracy1 and accuracy2 are just the identifiers. We are calculating t value based on sensitivity.

10. REFERENCES

[1] http://scikit-learn.org/stable/documentation.html

[2] https://matplotlib.org/contents.html

[3] https://xgboost.readthedocs.io/en/latest/parameter.html

[4] https://docs.anaconda.com/anaconda/install/linux/

[5] https://docs.python.org/3/library/statistics.html

[6] https://www.youtube.com/watch?v=ErDgauqnTHk

[7] https://en.wikipedia.org/wiki/Stochastic_gradient_descent

[8] http://scikit-learn.org/stable/modules/grid_search.html