Customer Churn- Binomial Classification

1. **Problem Statement:**

A bank like to identify the customers who are likely to churn ( move out of services ) based on the various features.

Challenge: Need to develop and train the classification model and on testing with the new data , it need to predict whether the customer will Churn or not with good accuracy .

1. **Import Data & Review**

* **Data – 27000 instances \* 31 features/ columns**
* **Target variable is Churn [1] , not Churn[0]**
* **Number of variables 31**
* **Number of observations 27000**
* **Missing cells 11262 (1.3%)**
* **Numeric 13**
* **Categorical 3**
* **Boolean 12**

1. **EDA**

**Observations**

* Removing the missing data i.e rows where age is null as the data is very small and complete credit\_score, rewards\_earned columns are the data is small.
* Observation of Categorical columns distribution by Hist & pie chars
* **Following columns have uneven data  
  waiting\_4\_loan,  
  cancelled\_loan,  
  received\_loan,  
  rejected\_loan,  
  left\_for\_one\_month**

**Check any bias of respective columns with target column**

* **The above feaures are not bias to the target**

for example waiting\_4\_loan == 1 (waiting\_4\_loan – Yes) has the target of both 1 & 0

**Correlation Data Analysis**

Correlation with target variable

* Below variables are positively correlated with Churn . I.e Higher the count in below variables then more prone to Churn

'cc\_taken', 'cc\_disliked', 'cc\_liked', 'web\_user', 'app\_web\_user', 'ios\_user', 'cancelled\_loan', 'received\_loan', 'rejected\_loan', 'left\_for\_two\_month\_plus', 'left\_for\_one\_month'

* Below variables are Negitively correlated with Churn . I.e Lower the count in below variables then more prone to Churn

'age', 'deposits', 'withdrawal', 'purchases\_partners', 'purchases', 'cc\_recommended', 'cc\_application\_begin', 'app\_downloaded', 'android\_user', 'waiting\_4\_loan', 'reward\_rate', 'is\_referred'

Total correlation

* **The Scale shows that +ve value to 0.2 which is very small when compared to -ve value to 0.8. So we can ignore the red square columns**
* ios\_user shows strong correlation with android user
* By name we can say that ios\_user, web\_user, app\_web\_user has a relation and these columns are not independent to each other. so we need to remove one column

1. **Machine Learning model**

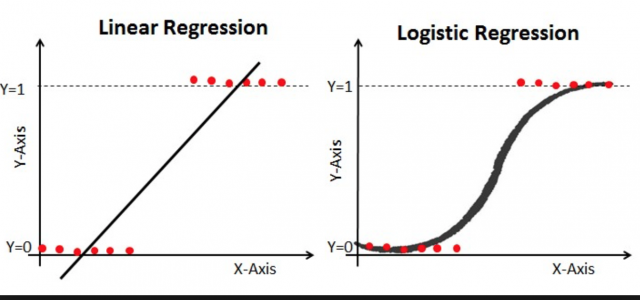
* **Convert the categorical column to numerical by get\_dummy**
* **Split the train\_test with stratify with target**
* check distribution of target in both train & test
* Balancing the Training Dataset - Training data has 50% - 0 and 50% - 1
* Standard scaling of all the independent columns. As StandardScaler looses index & column names we saving the results in other df and later pass it back to X\_train & X\_test
* **Model Building** : LogisticRegression
* **Model Evaluation** : confusion\_matrix, classification\_report, accuracy\_score, f1\_score, precision\_score, recall\_score

**Logistic Regression:**

 Logistic Regression is used when the dependent variable(target) is categorical.

For example,

* To predict whether an email is spam (1) or (0)
* Whether the tumor is malignant (1) or not (0)



* **Accuracy of the raw model with cross validation is 64.3% with St. Deviation +/- 0.033**

1. **Feature Selection :**

* **model.coef\_ gives the coeff of each variable. Higher the coeff important the variable is.**
* “The **coefficient** value represents the mean change in the response given a one-unit **increase** in the predictor. Consequently, it's easy to think that **variables** with **larger coefficients** are **more important** because they represent a **larger** change in the response”
* Select the top 20 feaures by rfe. Train the rfe with X\_train & y\_train and get the top 20 feaures.

1. **Further Improving the Model - Parameter tuning by Grid Search & important parameters**

* **Important parameters of Logistic Regression is are C, and penalty [ L1, L1]**

**C: Penalty parameter C of the error term. It also controls the trade off between smooth decision boundary and classifying the training points correctly.**

**Penalty : ['l1', 'l2']**

**L1 regularization - *Lasso Regression***

**L2 regularization -  *Ridge Regression*.**

**The key difference between these techniques is that Lasso shrinks the less important feature’s coefficient to zero thus, removing some feature altogether. So, this works well for feature selection in case we have a huge number of features.**

**Results :**

* **Model Accuracy is 62%. It wasn't changed much even after application of GridSearch and Best 20 features**
* **This shows that other 20 features are not adding any value to the model**
* **Compare the results of y\_test vs y\_pred wrt to user**