Recharge Classification of Telecom Users based on their Usage Patterns

Domain Background

Nowaday, telecom industry is very competing and improving a lot. Telecom industry needs to keep customers with company and maintain their loyalty. In order to maintain the customers, it is important to make sure that customer do recharge for using company's products and services. This project intends to make analysis of possibility of recharge based on customer behaviors. It will predict the possibility of recharge based on customer usage patterns.

Problem Statement

In Telecom industry, customers staying with us mainly depend on the recharge. Moreover, customer state is defined by their recharge. Possibility of churn will also increase if the customer no longer recharge. Hence recharge analysis plays important role in Telecom industry. Hence, this project will predict the possibility of recharge based on customer behavior.

Datasets and Inputs

In this project, 17 features (inclusive of target value, Recharge_flag) are extracted. Ten days data of 40000 customers are collected from real world data. 38000 datasets are used for training and 2000 records have been used for Testing dataset.

Out of 17 features, RCHRG_FLAG is used as target values. VOICE_SBRP_TOT_RVN_AMT and VOICE_TOT_ISD_OG_RVN_AMT are excluded in building model since those two features has less influence on recharge.

Data value definition are shown in following table.

LABEL	**DEFINITION**
AON	Age of Network (How many days the customer is active in Network)
DORMANCY_DAY_CNT	Number of Inactive days (Number of days of anactivity)
UC_TOT_RVN_AMT	Usage Total Revenue Amount for Pay Go
VAS_TOT_RVN_AMT	VAS service total subscription revenue
SBRP_TOT_RVN_AMT	All Pack purchase subscription revenue
VOICE_SBRP_TOT_RVN_AMT	Voice pack subscription revenue amount
VOICE_TOT_ONNET_OG_RVN_AMT	Total Onnet (the same network) outgoing revenue
VOICE_TOT_OFFNET_OG_RVN_AMT	Total Offnet (other network) outgoing revenue
VOICE_TOT_ISD_OG_RVN_AMT	International outgoing revenue
INTERCONNECT_RVN	Interconnect revenue (Revenue of incoming calls from other operators)
DATA_SBRP_TOT_RVN_AMT	Data pack subscription revenue
DATA_UC_TOT_RVN_AMT	Data Total usage revenue for Paygo
VOICE_OG_TOT_USG	Voice Outgoing Total minutes
VOICE_IC_TOT_USG	Voice Incoming Total Minutes

DATA_TOT_USG_MB	Total Data usage MB
RVN_AMT	Total Usage Revenue
RCHRG_FLAG	1 when there is recharge in last 10 days and 0 if there is no recharge in last 10 days

Their detail exploratory is shown in following table.

	AON	DORMANCY_DAY_CNT	UC_TOT_RVN_AMT	VAS_TOT_RVN_AMT
count	38000	38000	38000	38000
mean	707.0065	3.363132	756.474572	145.484336
std	558.39966	6.617707	1646.903957	614.664652
min	0	0	0	0
25%	159	0	0	0
50%	604.5	0	133.545	0
75%	1208.25	3	895.24	0
max	1727	29	72290.55	47971

	SBRP_TOT_RVN_AMT	VOICE_SBRP_TOT_RVN_AMT	VOICE_TOT_ONNET_OG_RVN_AMT	VOICE_TOT_OFFNET_OG_RVN_AMT
count	38000	38000	38000	38000
mean	708.077226	82.642855	217.644417	240.144717
std	1681.191242	363.792234	595.246223	651.446427
min	0	0	0	0
25%	0	0	0	0
50%	0	0	0	0
75%	760.9524	0	180	203.9775
max	46588.5715	12980.0001	24897.24	27867.35

	VOICE_TOT_ISD_OG_RVN_AMT	INTERCONNECT_RVN	DATA_SBRP_TOT_RVN_AMT	DATA_UC_TOT_RVN_AMT
count	38000	38000	38000	38000
mean	14.585849	209.042857	622.497611	250.910806
std	219.963042	594.543711	1607.923271	973.695004
min	0	0	0	0
25%	0	0	0	0
50%	0	18	0	0
75%	0	185.5834	151.4286	41.405
max	14666.08	28219.4524	45352.381	72122.61

	VOICE_OG_TOT_USG	VOICE_IC_TOT_USG	DATA_TOT_USG_MB	RVN_AMT	RCHRG_FLAG
count	38000	38000	38000	38000	38000
mean	30.581021	31.470021	818.994131	1634.918736	0.504474
std	79.190316	73.427136	2604.197058	2842.899784	0.499987
min	0	0	0	0	0
25%	0	0	0	0	0
50%	3.616667	6.033333	0.000165	596.76	1
75%	28.65	32.15	435.449742	2005.135	1
max	2868	2210.866667	114136.6389	73847.69	1

Solution Statement

In this project, three classification algorithms are used to build the model.

- Decision Tree (Gini algorithm) – We need to import Scikit-Learn library. A decision tree is a flowchart-like tree structure where an internal node represents feature(or attribute), the branch represents a decision rule, and each leaf node represents the outcome. The topmost node in a decision tree is known as the root node. It learns to partition on the basis of the attribute value. It partitions the tree in recursively manner call recursive partitioning. This flowchart-like structure helps you in decision making. It's visualization like a flowchart diagram which easily mimics the human level thinking. That is why decision trees are easy to understand and interpret. Ref: https://www.datacamp.com/community/tutorials/decision-tree-classification-python
- AdaBoost Algorithm – We need to import Scikit-Learn library. An AdaBoost [1] classifier is a meta-estimator that begins by fitting a classifier on the original dataset and then fits additional copies of the classifier on the same dataset but where the weights of incorrectly classified instances are adjusted such that subsequent classifiers focus more on difficult cases. Ref: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.AdaBoostClassifier.html
- Random Forest – we need to import sklearn.ensembler package. Random forests is a supervised learning algorithm. It can be used both for classification and regression. It is also the most flexible and easy to use algorithm. A forest is comprised of trees. It is said that the more trees it has, the more robust a forest is. Random forests creates decision trees on randomly selected data samples, gets prediction from each tree and selects the best solution by means of voting. It also provides a pretty good indicator of the feature importance.
  Ref: https://www.datacamp.com/community/tutorials/random-forests-classifier-python

Benchmark Model

I have tested the data with above three classification algorithms and I have got the following accuracy scores.

Algorithm	16 Features	14 features
Decision Tree	88.5%	88.2%

AdaBoost	91.85%	91.8%
Random Forest	92.5%	92.75%

There is no much difference for different number of features because like explained above, VOICE_SBRP_TOT_RVN_AMT and VOICE_TOT_ISD_OG_RVN_AMT have less influence on likelihood of recharge.

Among three algorithms, Random Forest classifier got best performance while decision tree has the least.

Evaluation Metrics

Accuracy is the important metric in evaluation of classification algorithms. It is the ratio of number of correctly classified instances to total number of instances.

There are four important terms used for evaluation metrices:

True Positives : Predicted YES and the actual is also YES.

True Negatives : Predicted NO and the actual is also NO.

False Positives : Predicted YES and the actual is NO.

False Negatives : Predicted NO and the actual is YES.

	Predicted class		
Actual Class		Class = Yes	Class = No
	Class = Yes	True Positive	False Negative
	Class = No	False Positive	True Negative

Using above terms, other evaluation metrices such as F1 Score, Precision, Recall, Sensitivity can also be computed.

Project Design

In this project, data has been already processed since extraction time and hence no extra preprocessing is required. For example, AON is already computed in my dataset while extracting the data (today date – activation date). Recharged flag is also processed when there is recharge during last 10 days will be 1 and if there is no recharge RCHRG_FLAG is 0.

In the data exploration part, RCHRG_FLAG is used as target value. Distribution of RECHRG_FLAG in train and test is as below:

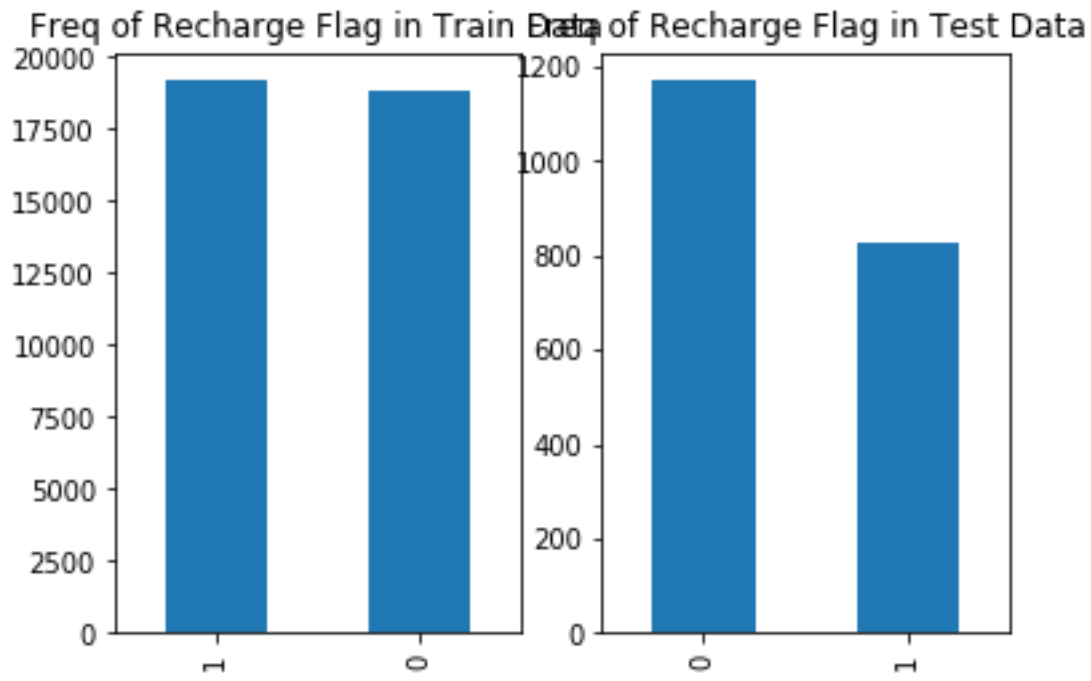

Figure 1: Distribution of Target value (RCHRG_FLG 1, 0) in Training and Testing datasts.

The correlation of whole dataset is shown in Figure 2.

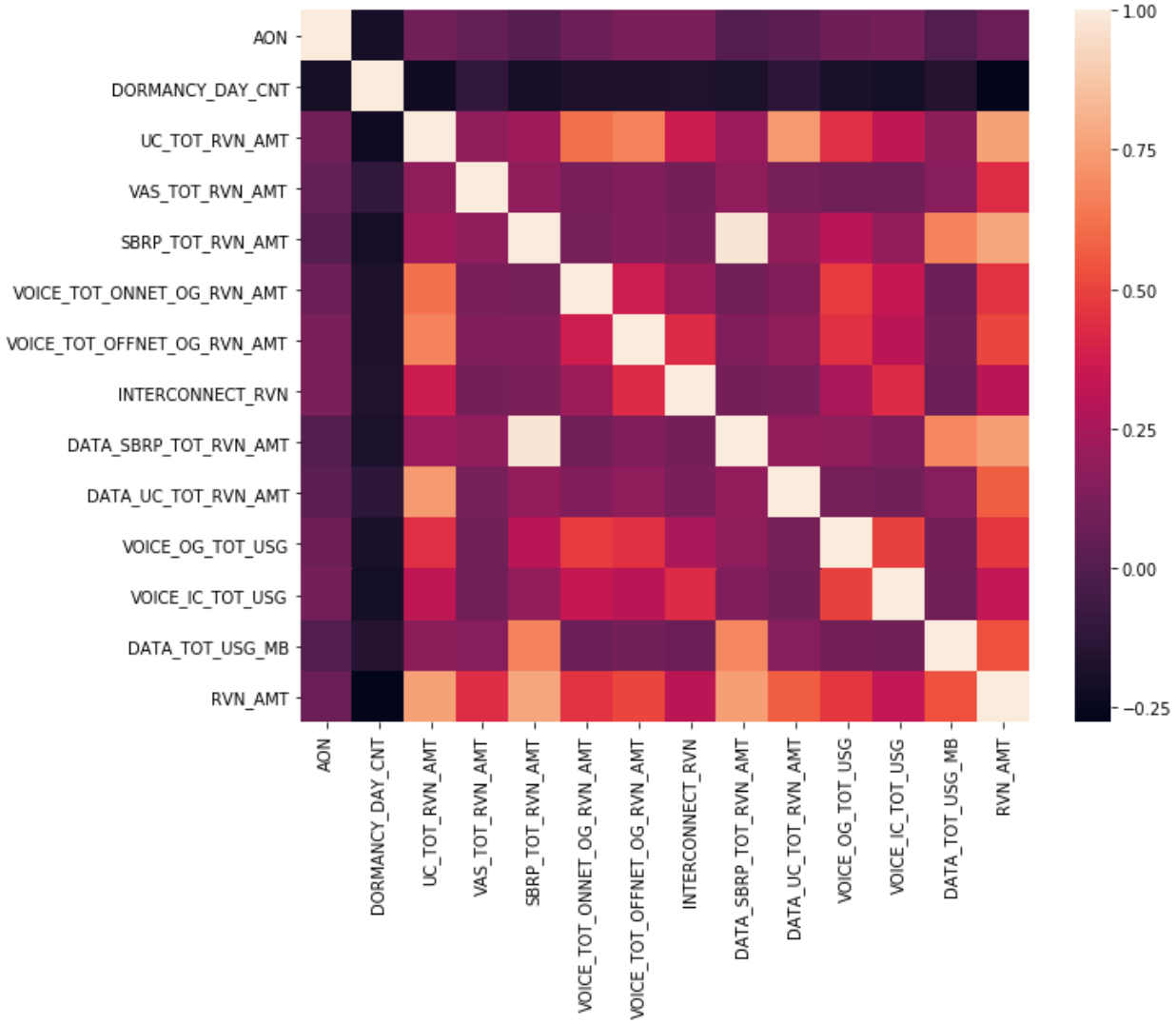

Figure 2: Correlation of Training Dataset for all 17 features

Correlation of dormancy day count and revenue amount with hue value RCHRG_FLAG is shown figure 3.

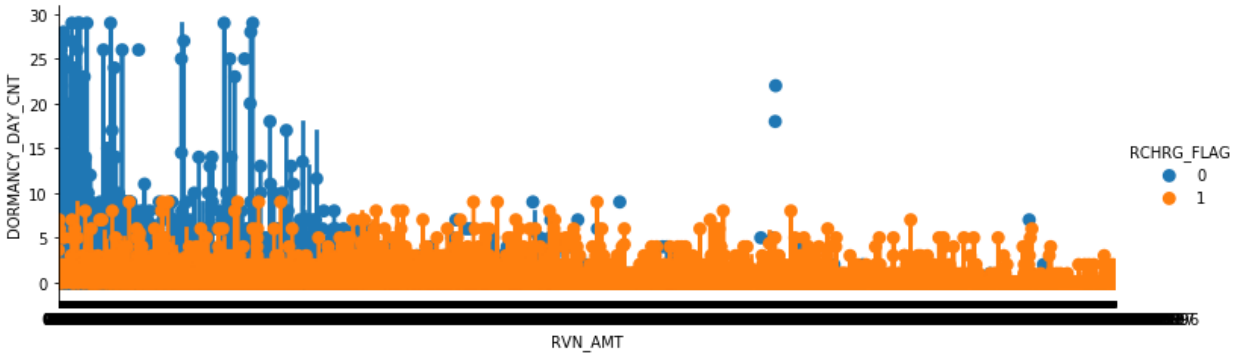

Figure 3: Correlation of Dormancy day count and Revenue Amount (based on RCHRG_FLAG)

The observation of above figure is that higher dormancy day count tends to non recharge with decrease revenue.

The detailed implementations of classifiers are shown in attached yaml file.