MLND CAPSTONE PROPOSAL

Udacity Machine Learning Nanodegree

Gross Addition Customer Survival Prediction Model

Khin Ohnmar Swe

March 2020

Domain Background

In our country, Telecom industry is getting growth and highly competitive market since last few years. So, all operators are struggling to maintain existing customers and to obtain new acquisition. Our company also trying to maintain the existing customers in various ways: by offering the attractive packs, making loyalty programs, propose high end customers to use postpaid plan, etc. To get the gross add customers, we are offering attractive newbies benefits such as free data usage, free voice.

In here, some customers have used free benefit and churned out. So, we need know the behavior of GA customers who will be mature customers and still attach in our network. In this project, I will emphasize the quality GA who are still active and using our services in 3rd month after activation date. So that, our business team can do planning the appropriate program/campaigns to get the quality GA customers.

Problem Statement

Since the size of these data sources is impossible for a human analyst to come up with interesting information that will help in the decision-making process. Machine learning models are completely helping in this identifying those quality GA customers. With the experimental results I will demonstrate the performance of the models by statistical metrics like accuracy, sensitivity, precision, recall, etc. Regarding the higher scoring of these metrics, we will be able to judge the success of these models in predicting the quality GA customers.

Datasets and Inputs

Monthly average gross addition count is ~ 0.8 Mn. And I will use the Jan-9 Gross add customers and predict who will be the quality GA in Mar-19.

Since this data is our company data, I have already taken the prior approval from our company's management. And I got the right to use since the customer's details information is not included.

I will use the first 7 days after GA date behavior of those customers. Below are the attributes.

1) REC DOU - Revenue generated days count in first 7 days after GA date
2) OG REVENUE – Outgoing revenue given in first 7 days after GA date
3) IC REVENUE – Incoming revenue given in first 7 days after GA date
4) ISD_REVENUE – International voice revenue given in first 7 days after GA date
5) ROAMING REVENUE – Roaming revenue given in first 7 days after GA date
6) PACK REVENUE – Data/Voice/SMS pack subscription revenue given in first 7 days after GA date
7) TOPUP AMOUNT – Topup amount in first 7 days after GA date
8) TOPUP COUNT – Number of topup times in first 7 days after GA date
9) DATA USAGE (MB) – Data usage in MB
10) VOICE OG USAGE (MIN) – Outgoing voice usage in minutes
11) VOICE IC USAGE (MIN) – Incoming voice usage in minutes
12) VOICE OG CALL COUNT – Outgoing voice call count
13) VOICE IC CALL COUNT – Incoming voice call count
14) SMS USAGE (COUNT) – SMS usage in count
15) IS VAS USER - Is using VAS service (Y/N)
16) BALANCE_TRANSFER AMOUNT – Main balance transfer amount to others
17) ECB_COUNT - Loan service taking count
18) HANDSET TYPE – Handset type (2G/3G/4G..)

Target

1) Is Survived – Is Quality GA customer in 3rd month (Y/N)

Solution Statement

We will prepare the data by splitting feature and target/label columns and also check for quality of given data and perform data cleaning. To check if the model I created is any good, I will split the data into training and validation sets to check the accuracy of the best model. We will split the given training data in two, 80% of which will be used to train our models and 20% we will hold back as a validation set.

As described in above section, there are several non-numeric columns that need to be converted. Many of them are simply yes/no, e.g. USIM. These can be reasonably converted into 1/0 (binary) values. Other columns, like device type and weekly average recharge amount, have more than two values, and are known as categorical variables. The recommended way to handle such a column is to create as many columns as possible values and assign a 1 to one of them and 0 to all others. These generated columns are sometimes called dummy variables, and we will use the pandas.get_dummies() function to perform this transformation.

We can also make subsets of original data using feature scaling techniques to normalize and scale data to try various iterations on the chosen models just to see if we see any differences in performance. Since the range of values of raw data varies widely, in some machine learning algorithms, objective functions will not work properly without normalization. For example, the majority of classifiers calculate the distance between two points by the Euclidean distance. If one of the features has a broad range of values, the distance will be governed by this particular feature. Therefore, the range of all features should be normalized so that each feature contributes approximately proportionately to the final distance.

To assess which algorithms would be fit for this problem or what configurations to use, we will try on a few algorithms to evaluate.

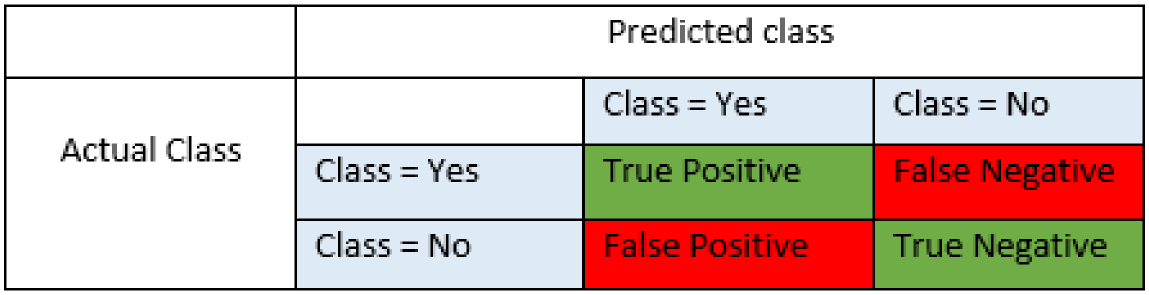
- ✓ Logistic Regression
- ✓ Decision Tree Classification
- ✓ Adaboost Classification
- ✓ Random Forests

We are using 10-fold cross validation to estimate accuracy. This will split our dataset 10 parts, train on 9 and test on 1 and repeat for all combinations of train-test splits.

Also, we are using the metric of accuracy to evaluate models. This is a ratio of the number of correctly predicted instances in divided by the total number of instances in the dataset multiplied by 100 to give a percentage (e.g. 95% accurate). We will be using the scoring variable when we run build and evaluate each model next.

Evaluation Metrics

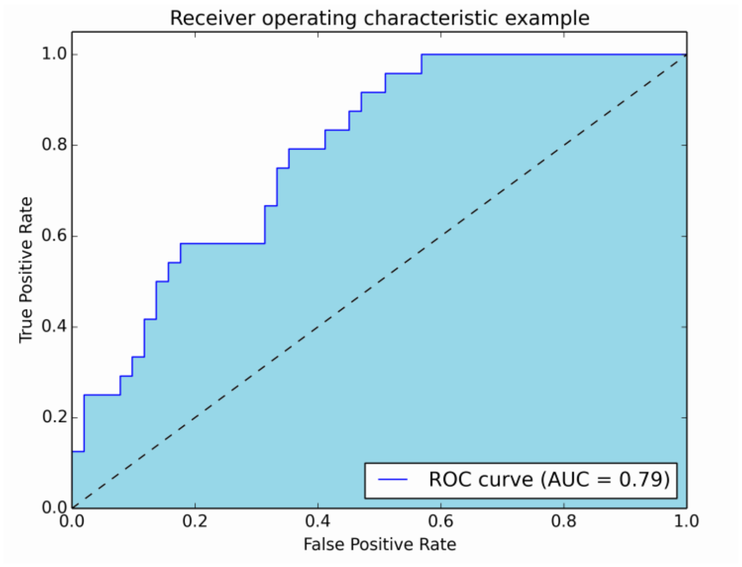
The performance of each classification model is evaluated using three statistical measures; classification accuracy, sensitivity and specificity. It is using true positive (TP), true negative (TN), false positive (FP) and false negative (FN). The percentage of Correct/Incorrect classification is the difference between the actual and predicted values of variables. True Positive (TP) is the number of correct predictions that an instance is true, or in other words; it is occurring when the positive prediction of the classifier coincided with a positive prediction of target attribute. True Negative (TN) is presenting a number of correct predictions that an instance is false, (i.e.) it occurs when both the classifier, and the target attribute suggests the absence of a positive prediction. The False Positive (FP) is the number of incorrect predictions that an instance is true. Finally, False Negative (FN) is the number of incorrect predictions that an instance is false. Table below shows the confusion matrix for a two-class classifier.



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

AUC - ROC curve is a performance measurement for classification problem at various thresholds settings. ROC is a probability curve and AUC represents degree or measure of separability. It tells how much model is capable of distinguishing between classes. Higher the AUC, better the model is at predicting 0s as 0s and 1s as 1s. By analogy, Higher the AUC, better the model is at distinguishing between patients with disease and no disease.

Here is the sample of AUROC curve:



Analysis

Basic Statistic

	REC_DOU	MOBILE_REV	OG_REVENUE	IC_REVENUE	ISD_REVENUE	PACK_REVENUE	TOPUP_AMOUNT	TOPUP_COUNT	DATA_MB
count	143,101	143,101	143,101	143,101	143,101	143,101	143,101	143,101	143,101
mean	2	1,385	1,171	44	27	445	1,589	2	398
std	2	1,961	1,856	187	279	1,092	1,922	1	1,007
min	1	135	-	-	-	-	-	-	-
25%	1	244	109	-	-	-	997	2	-
50%	2	306	151	-	-	94	997	2	-
75%	5	1,870	1,619	6	-	151	1,994	2	171
max	8	12,999	12,856	9,358	10,533	12,377	15,000	19	15,041

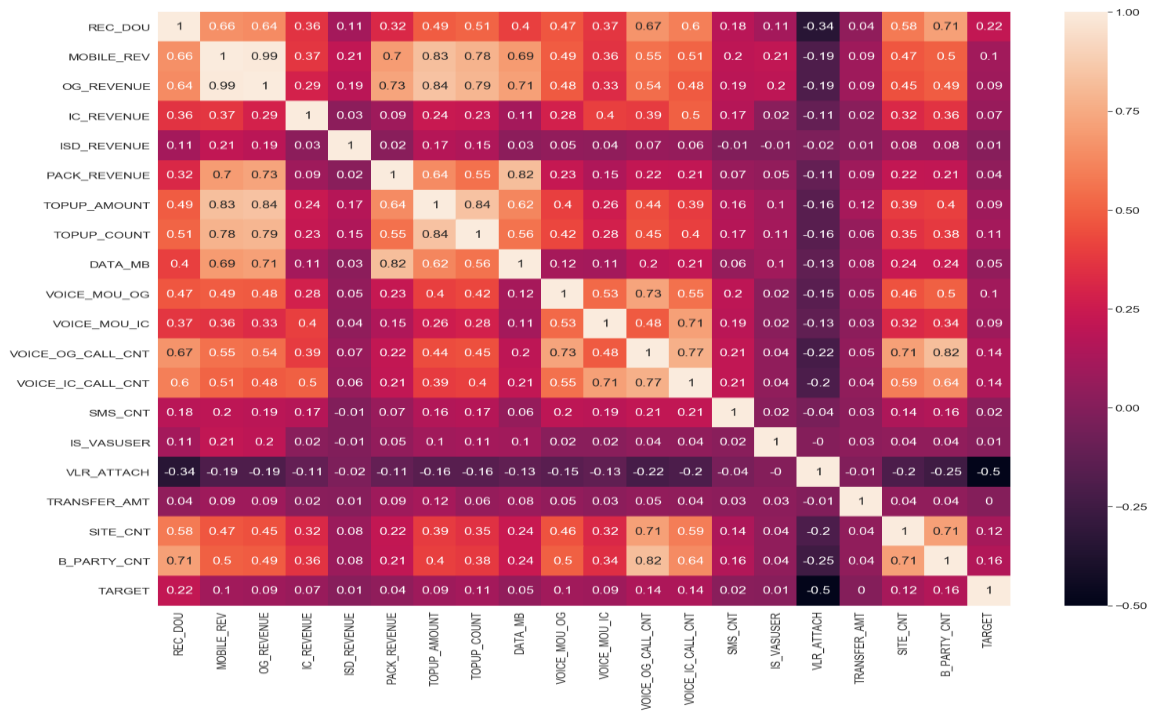
	VOICE_MOU_OG	VOICE_MOU_IC	VOICE_OG_CALL_CNT	VOICE_IC_CALL_CNT	SMS_CNT	IS_VASUSER	VLR_ATTACH	TRANSFER_AMT	Ste Count	B_Party_Cnt
count	143,101	143,101	143,101	143,101	143,101	143,101	143,101	143,101	143,101	143,101
mean	17	11	8	4	2	-	-	3	1	2
std	49	41	15	11	8	-	-	104	3	4
min	-	-	-	-	-	-	-	-	-	-
25%	-	-	1	-	-	-	-	-	-	-
50%	-	-	1	-	-	-	-	-	1	1
75%	14	4	10	5	1	-	-	-	2	4
max	996	994	200	198	199	1	1	13,065	67	163

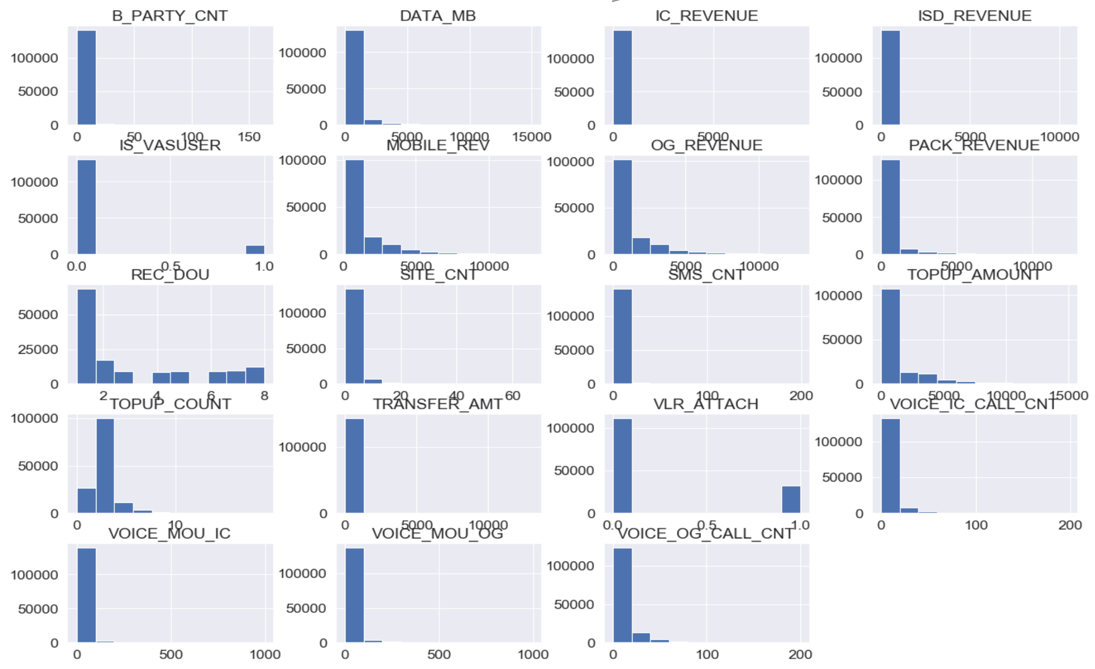
	TARGET – 0	TARGET – 1
REC_DOU	2	3
MOBILE_REV	1,109	1,515
OG_REVENUE	921	1,289
IC_REVENUE	26	53
ISD_REVENUE	24	28
PACK_REVENUE	388	473
TOPUP_AMOUNT	1,324	1,714
TOPUP_COUNT	2	2
DATA_MB	322	433
VOICE_MOU_OG	10	21
VOICE_MOU_IC	5	13
VOICE_OG_CALL_CNT	5	9
VOICE_IC_CALL_CNT	2	5
SMS_CNT	1	2
IS_VASUSER	-	-
VLR_ATTACH	-	-
TRANSFER_AMT	2	3
SiteCount	1	2
B_Party_Cnt	1	3

Observation

- Average revenue earning day count for Target1 group is more than 1 day compare with Target2.
- Voice usage/ Attach site count of Target1 group is double of Target2.
- Target1 group is ~ 29% better in Topup.

Visualization





Observation

- According to the correlation matrix plot, OG_Revenue & Mobile_REV , Voice_call_cnt & B_Party_Cnt and Voice_OG_Call_Cnt &Voice_IC_Call_Cnt are highly correlated.
- Target & REC_DOU,B_Party_Cnt,Voice_OG_Call_Cnt and Voice_IC_Call_Cnt have the positive correlation.
- Target & VLR_Attach have the negative correlation.

Summary

As per above observation, I considered below features will be the top 5 most relevant for prediction.

- REC DOU: GA customers who were more active are high chance to be the quality GA.
- B Party Count: The more interactive with other customers the higher the chance to be quality GA.
- Site Attach count: The customers who were attached in more than one BTS site seem to be the quality GA.
- Mobile_REV: The GA customers who generated more revenue are likely to be the quality GA.
- Topup Amount: There is the high chance to be quality GA for the customers who made topup.

Algorithms and Technique

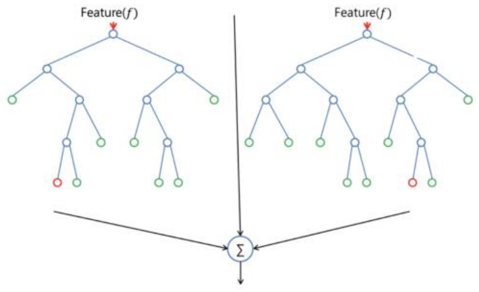
I will be using below algorithms in my project.

Algorithms	Pros	Cons
Random Forest	❖ Decorrelates trees (relative to bagged trees) ❖ Important when dealing with multiple features which may be correlated ❖ Reduced variance (relative to regular trees)	❖ Not as easy to visually interpret
Logistic Regression	❖ Low variance ❖ Provides probabilities for outcomes ❖ Works well with diagonal (feature) decision boundaries ❖ NOTE: logistic regression can also be used with kernel methods	❖ High bias
Decision Tree Classifier	❖ Easy to understand and interpret, perfect for visual representation. This is an example of a white box model, which closely mimics the human decision-making process. ❖ Can work with numerical and categorical features. ❖ Requires little data preprocessing: no need for one-hot encoding, dummy variables, and so on. ❖ Non-parametric model: no assumptions about the shape of data. ❖ Fast for inference. ❖ Feature selection happens automatically: unimportant features will not influence the result. The presence of features that depend on each other (multicollinearity) also doesn't affect the quality.	❖ Cannot handle complicated relationship between features. ❖ Simple decision boundaries. ❖ Problems with lots of missing data.
AdaBoost classifier (Benchmark)	❖ Good algorithm ❖ Corrects its mistakes.	❖ Not good for noisy data

How selected models work

RandomForest

To say it in simple words: Random forest builds multiple decision trees and merges them together to get a more accurate and stable prediction.

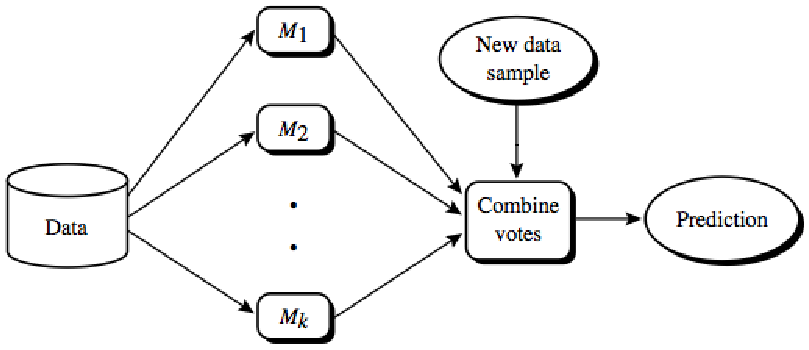


Adaboost

Adaboost is a boosting type ensemble learner. This method works by combining multiple individual "weak" learning hypotheses to create one strong model. Each weak hypothesis used is better at classifying the data than random chance. However, it's the combination of all of these independent weak learning hypotheses what makes the model more capable of predicting accurately on unseen data than each of the individual hypothesis would.

This algorithm is trained iteratively. During each training iteration, the model attempts to correctly classify the training data. Based on how well it performed during the iteration, the algorithm assigns higher weights to the observations that it was unable to classify correctly. On the next training iteration, the algorithm focuses on those more complex, initially misclassified observations, and tries to classify them correctly. Again based on the results of this training iteration Adaboost assigns higher weights to those observations that were misclassified. This process is repeated for a number of iterations defined by the user in an attempt to find the set of weighted hypotheses that combined will perform best on unseen data.

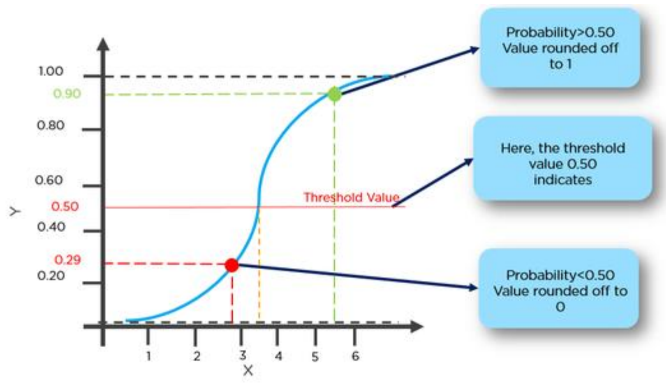
Finally, to make predictions, Adaboost uses the weak independent hypotheses on which it was trained during each iteration and develops individual predictions that are then weighted accordingly to come up with the most probable single strong prediction.



Logistic Regression

Logistic Regression, falls under Supervised Machine Learning. It solves the problems of Classification (to make predictions or take decisions based on past data). It is used to predict binary outcomes for a given set of independent variables. The dependent variable's outcome is discrete.

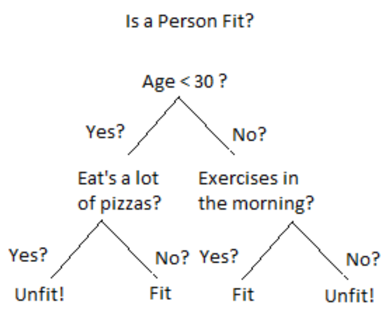
The output of Logistic Regression is a sigmoid curve(or more popularly known as S-curve). Where the value on the x-axis(independent variable) would determine the dependent variable on the y-axis. In logistic regression there are only two possible outcomes. 0 and 1. That something occurs, or it doesn't. We use a threshold value to make our prediction easier. If the x-axis' corresponding y-value (probability) is lesser than the threshold value, the outcome is taken as 0. If it is greater than the value, the outcome is taken as 1.



Decision Tree

Decision Trees (DTs) are a non-parametric supervised learning method used for classification and regression. Decision trees learn from data to approximate a sine curve with a set of if-then-else decision rules. The deeper the tree, the more complex the decision rules and the fitter the model.

Decision tree builds classification or regression models in the form of a tree structure. It breaks down a data set into smaller and smaller subsets while at the same time an associated decision tree is incrementally developed. The final result is a tree with decision nodes and leaf nodes. A decision node has two or more branches. Leaf node represents a classification or decision. The topmost decision node in a tree which corresponds to the best predictor called root node. Decision trees can handle both categorical and numerical data.



Benchmark

- As per naive approach, if we considered all of the customers are Target 1, we will get the accuracy 0.6801 and FScore 0.7265
- My benchmark, Adaboost Classifier with default parameters is given the below result.
  - AUC 0.7127 , Accuracy 0.7925 and F Score 0.8234

Methodology

Data Preprocessing

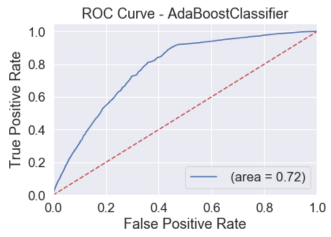
- I have replaced the Null value to default value & remove extreme value since data extracting process.
- I transformed label encode the Region and HS_Type.
- I transformed the following features by MinMaxScalar() –
    - MOBILE_REV
    - OG_REVENUE
    - IC_REVENUE
    - ISD_REVENUE
    - PACK_REVENUE
    - TOPUP_AMOUNT
    - TOPUP_COUNT
    - DATA_MB
    - VOICE_MOU_OG
    - VOICE_MOU_IC
    - VOICE_OG_CALL_CNT
    - VOICE_IC_CALL_CNT
    - SMS_CNT
    - TRANSFER_AMT

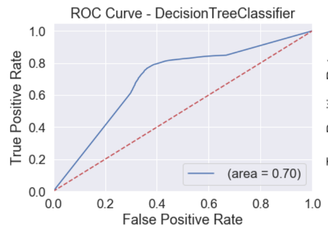
I used MinMaxScalar() to transform numerical features. And I believe that there will not much difference for the algorithms that I will be using if I used StandardScalar().

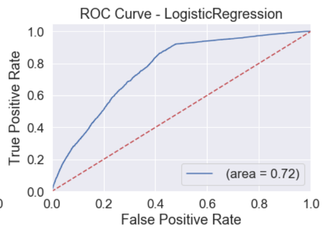
Implementation

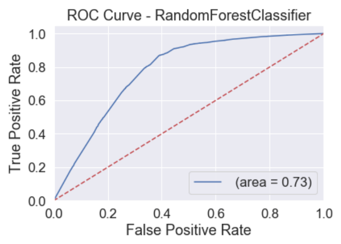
- Firstly I split up the dataset to training and validation set with 80:20 ratio.
- Then I trained the models without giving any parameters on training set (80%) and validated on the testing set (20%)
- Below given result is the validation scores. RandomForest model is leading.

	Adaboost (Benchmark)	Decision Tree	Logistic Regression	RandomForest
AUC	0.7217	0.6968	0.7207	0.7322
Accuracy	0.7925	0.7324	0.7915	0.7865
Fbeta Score	0.8234	0.8041	0.8228	0.8286





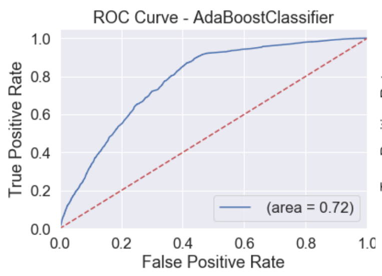


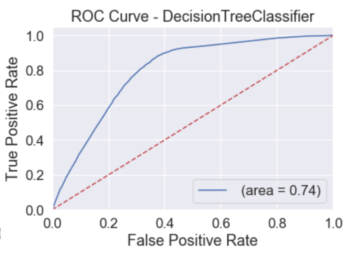


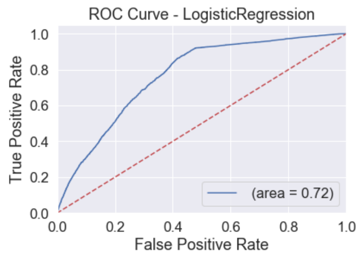
Refinement

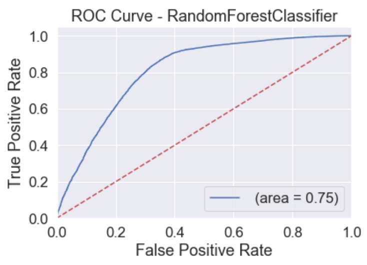
After optimized all models, the result of DecisionTree model is slightly increased. But RandomForest model is still leading in score. So, I choose RandomForest as my final model.

	Adaboost (Benchmark)	**Decision Tree**	**Logistic Regression**	**RandomForest**
Parameter grid	{'n_estimators': [47,48,49,50,100], 'learning_rate': [1.0,1.3,1.5,2]}	{'criterion': ['gini', 'entropy'],'splitter': ['best', 'random'],'max_depth': [7,8,9,10, 11], 'min_samples_split': [1.0,2, 3, 4,5]}	{'C': [1,2,3, 10, 100, 1000] }	{'n_estimators': [45,50,55,100], 'max_depth': [8,9,10] , 'max_features': [15,16]}
AUC	0.7225	0.7446	0.7207	0.7456
Accuracy	0.7926	0.8041	0.7915	0.8076
Fbeta Score	0.8238	0.8363	0.8228	0.8370







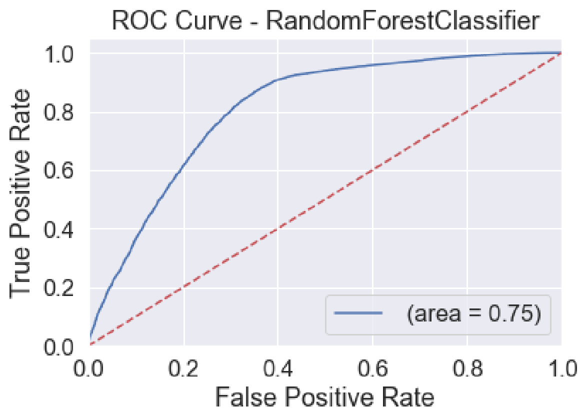


Result

Model Evaluation and Validation

Final model result is -

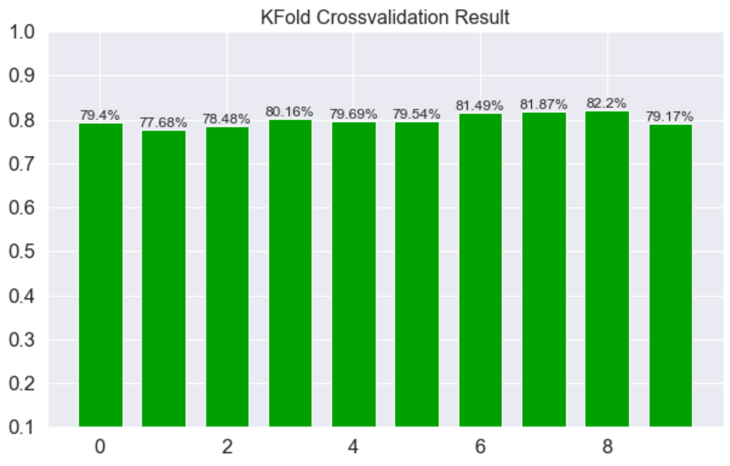
- AUC : 0.7456
- Accuracy : 0.8076
- Fbeta Score : 0.8370



Again I have check with the 10 folds cross validation and got the below result and it is better than train/test split (20:80) -

- Mean AUC : 0.7997
- Max AUC: 0.8220
- Min AUC: 0.7768

So, I can trust my final model result.



Justification

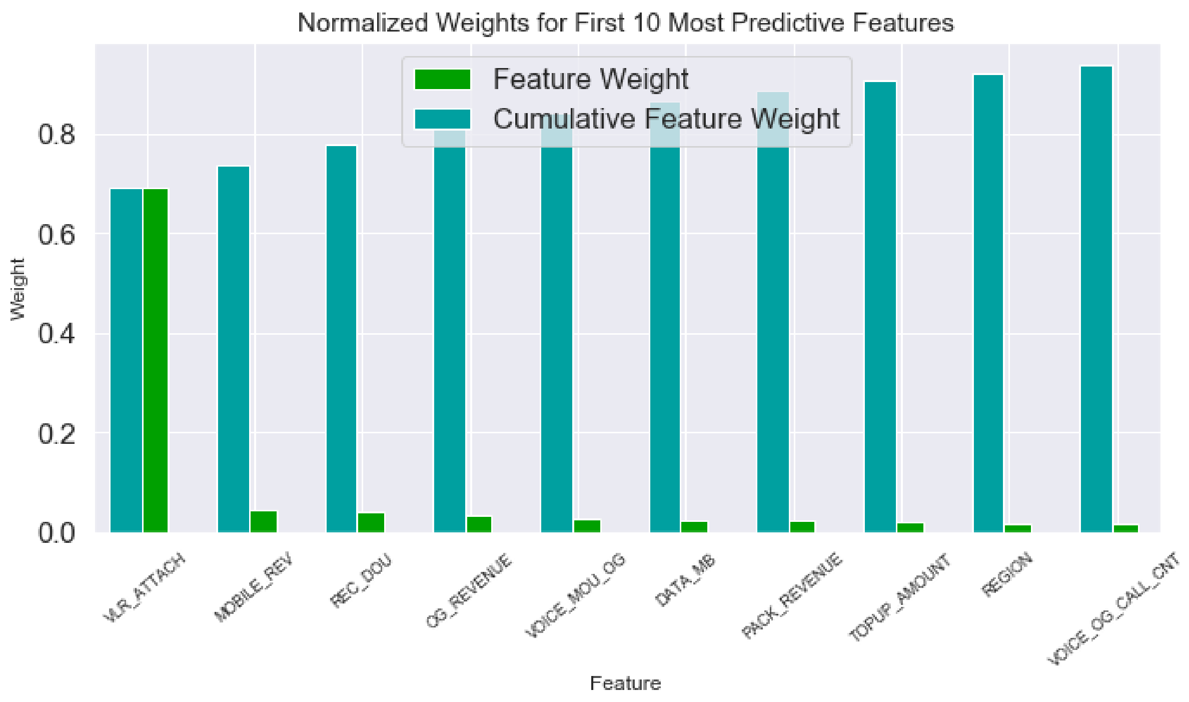
Final model result is slightly better than Naive predictor and also ~2 % better than Benchmark model.

	Naive Approach	Benchmark Model	Final Model
AUC		0.7225	0.7456
Accuracy	0.6801	0.7926	0.8076
Fbeta Score	0.7265	0.8238	0.8370

Conclusion

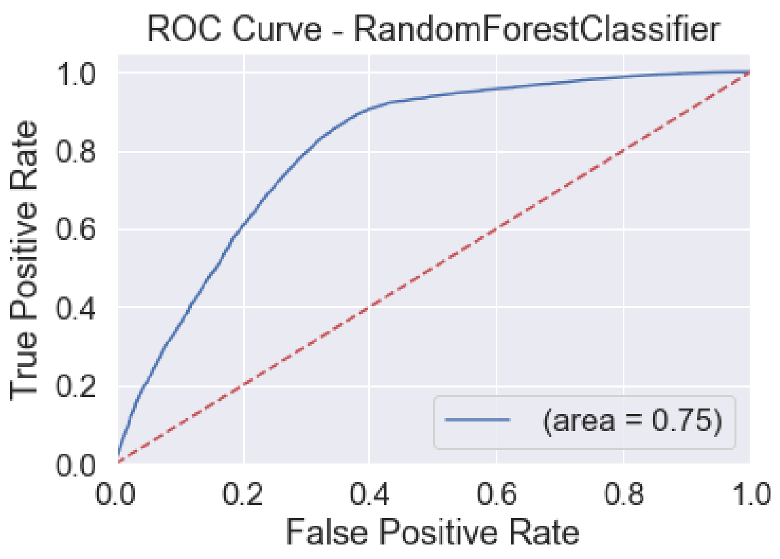
Free-Form Visualization

It is surprising for me that VLR_ATTACH is the most important feature for final model. The rest features are the same as I guessed.



I got the similar result with only top 10 features.

- AUC : 0.7454
- Accuracy : 0.8078
- Fbeta Score : 0.8369



Reflection

Every single step is difficult and interesting for me. Major interesting difficulty I have found in implementing this model is within feature selection part, as there are some difficulties to identify the customer behavior to select the right parameter. Since I was facing the system capacity issue, I reduced the dataset size to 7 days GA from 1 month GA.

- Data extraction process: I manual extracted the features from data warehouse and did clearing the data such as replace default value instead of null value. The challenge is to get the relevant features among a lot of data columns.
- Choosing the algorithms: It is difficult to select the algorithms since many algorithms are good to use for my dataset. So, I have read the documentation and research about the supervised learning algorithms and then I tested these algorithms. Finally I choose Radomforest and Adaboost classifier from ensemble models , Decision Tree classifier from Tree models and Logistic Regression from linear models that are the most popular/ recommended models.
- Tuning the models: I haven't seen any difficult in this part except waiting long time to finish the tuning. I have created the parameter grid for the important parameter for each model and find the best one by gridsearch().
- Validation process: I was so happy when I saw the validation result that is the similar to my final model result.

When I have done this capstone project, I have gained the following benefit.

- I have done one real world end to end machine learning project
- More understanding on the function of the supervised models.
- Got valuable feedbacks and learning resources from reviewer for further study to improve my ML skill.
- I am now more clear the areas of improvement for ML journey and many more benefits.

Improvement

Even the final model can predict correctly ~ 81%, it still need to improve. I thought that the following way will bring better result.

- Performing advance features engineering. I am currently using the traditional manual features engineering with less than 20 features. I believe that my model result will be improve, if I can apply the automate feature engineering technique with thousands of features.

Ref: https://towardsdatascience.com/why-automated-feature-engineering-will-change-the-way-you-do-machine-learning-5c15bf188b96

-- End of Document --