Machine Learning Engineering Nano Degree

Capstone Final Report

Udacity, March 2020

Ali Rafieh

Domain Background

Starbuck is most famous chain store in the world in coffee industry. One big part of company development, in gaining new customers and keeping existing customers, comes from targeting advertisement. This ads should be subjective and attractive. For this matter, one of the way Starbucks using its mobile app, for customers whose using that.
Company every few days, send out an offer to its customers by mobile app. An offer can be merely an advertisement for a drink or an actual offer such as a discount or BOGO (buy one get one free).
In this project, we are going to help Starbucks in this targeting offer technics by using and exploring the datasets company provided.

Problem Statement

The data set that is going to be used for this project are simulated data that mimics customer behavior on the Starbucks rewards mobile app. As said, an offer can be merely an advertisement for a drink or an actual offer such as a discount or BOGO (buy one get one free). Some users might not receive any offer during certain weeks.
Not all users receive the same offer, or any offer.
The goal is determine which kind of offer, if any, to send to each customer based on their purchases and interaction with the previously sent offers. So, building a machine learning model that predicts how Starbucks customers will respond to an offer based on demographics and offer type, is subject that will followed.
For demographic data for each customer, three type of classification supervised machine learning models, feeding in the data from three combine data (portfolio, profile, transactional) will be used: GaussianNB, Decision Tree and Support Vector Machine (SVM). Finally a Logistic Regression apply on data.

Datasets and Inputs

The data consists of 3 files containing simulated data that mimics customer behavior on the Starbucks Rewards mobile app.
Portfolio.json contains info about the offers, profile.json contains info about the customers, and transcript.json contains info about customer purchases and interaction with the offers.
The data contain information about 10 offers: 4 BOGO, 4 discount, and 2 informational. It consist of 17,000 customers and a transcript containing 306,534 purchases and offer interactions.
A customer can interact with an offer by receiving it, viewing it, or completing it. It is possible for a customer to complete some offers without viewing them.
To split the customer data into training/validation/testing sets , a 60/20/20 split percentage, respectively for the customers will be used. So, 10.2k customers will be for training, 3.4k will be for validation and 3.4k for testing.
The dataset seems balanced. To determine this looked at following value counts for all events listed in transcript.json :

transaction	138953
offer received	76277
offer viewed	57725
offer completed	33579

Percentage customer who received an offer and complete it are 55.97% ((76,277 – 33,579) / 76,277). That means that 55.79% of the people completed their offers, while 44.03% received offers but did not complete. These percentages are close enough to consider this a balanced dataset.

Following describe the different datasets is contained in three files:

- portfolio.json - containing offer ids and meta data about each offer (duration, type, etc.)
- profile.json - demographic data for each customer
- transcript.json - records for transactions, offers received, offers viewed, and offers completed

portfolio.json

Range Index:(10, 6)

- id (string) - offer id
- offer_type (string) - type of offer ie BOGO, discount, informational

- difficulty (int) - minimum required spend to complete an offer
- reward (int) - reward given for completing an offer
- duration (int) - time for offer to be open, in days
- channels (list of strings)

	reward	channels	difficulty	duration	offer_type	id
0	10	[email, mobile, social]	10	7	bogo	ae264e3637204a6fb9bb56bc8210ddfd
1	10	[web, email, mobile, social]	10	5	bogo	4d5c57ea9a6940dd891ad53e9dbe8da0
2	0	[web, email, mobile]	0	4	informational	3f207df678b143eea3cee63160fa8bed
3	5	[web, email, mobile]	5	7	bogo	9b98b8c7a33c4b65b9aebfe6a799e6d9
4	5	[web, email]	20	10	discount	0b1e1539f2cc45b7b9fa7c272da2e1d7
5	3	[web, email, mobile, social]	7	7	discount	2298d6c36e964ae4a3e7e9706d1fb8c2
6	2	[web, email, mobile, social]	10	10	discount	fafdcd668e3743c1bb461111dcafc2a4
7	0	[email, mobile, social]	0	3	informational	5a8bc65990b245e5a138643cd4eb9837
8	5	[web, email, mobile, social]	5	5	bogo	f19421c1d4aa40978ebb69ca19b0e20d
9	2	[web, email, mobile]	10	7	discount	2906b810c7d4411798c6938adc9daaa5

PORTFOLIO DATASET

profile.json

Range Index: (17000, 5)

- age (int) - age of the customer
- became_member_on (int) - date when customer created an app account
- gender (str) - gender of the customer (note some entries contain 'O' for other rather than M or F)
- id (str) - customer id
- income (float) - customer's income

	gender	age	id	became_member_on	income
0	None	118	68be06ca386d4c31939f3a4f0e3dd783	20170212	NaN
1	F	55	0610b486422d4921ae7d2bf64640c50b	20170715	112000.0
2	None	118	38fe809add3b4fcf9315a9694bb96ff5	20180712	NaN
3	F	75	78afa995795e4d85b5d9ceeca43f5fef	20170509	100000.0
4	None	118	a03223e636434f42ac4c3df47e8bac43	20170804	NaN

PROFILE DATASET

transcript.json
Range Index: (306534, 4)

- event (str) - record description (ie transaction, offer received, offer viewed, etc.)
- person (str) - customer id
- time (int) - time in hours since start of test. The data begins at time t=0
- value - (dict of strings) - either an offer id or transaction amount depending on the record

	gender	age	customer_id	became_member_on	income
1	0	55	0610b486422d4921ae7d2bf64640c50b	2017	112000.0
3	0	75	78afa995795e4d85b5d9ceeca43f5fef	2017	100000.0
5	1	68	e2127556f4f64592b11af22de27a7932	2018	70000.0
8	1	65	389bc3fa690240e798340f5a15918d5c	2018	53000.0
12	1	58	2eeac8d8feae4a8cad5a6af0499a211d	2017	51000.0
...	...	...	...	...	...
16995	0	45	6d5f3a774f3d4714ab0c092238f3a1d7	2018	54000.0
16996	1	61	2cb4f97358b841b9a9773a7aa05a9d77	2018	72000.0
16997	1	49	01d26f638c274aa0b965d24cefe3183f	2017	73000.0
16998	0	83	9dc1421481194dcd9400aec7c9ae6366	2016	50000.0
16999	0	62	e4052622e5ba45a8b96b59aba68cf068	2017	82000.0

TRANSCRIPT DATASET

Data Wrangling and Preparation

After investigating all datasets, for each of them following steps had been done for cleaning and preparations.

Portfolio dataset:¶

1. Change column name id to offer_id.
2. Turn duration numbers from *day* to *hour*.
3. Pivot channels column to four columns web, email, mobile and social.

Profile dataset:¶

Here is changes on profile dataset:

1. It seems in profile dataset, all rows with NaN in gender and income, register with age ***118***. So, we can take it as outlier and drop these rows.

	reward	channels	difficulty	duration	offer_type	offer_id
0	10	[email, mobile, social]	10	168	bogo	ae264e3637204a6fb9bb56bc8210ddfd
1	10	[web, email, mobile, social]	10	120	bogo	4d5c57ea9a6940dd891ad53e9dbe8da0
2	0	[web, email, mobile]	0	96	informational	3f207df678b143eea3cee63160fa8bed
3	5	[web, email, mobile]	5	168	bogo	9b98b8c7a33c4b65b9aebfe6a799e6d9
4	5	[web, email]	20	240	discount	0b1e1539f2cc45b7b9fa7c272da2e1d7
5	3	[web, email, mobile, social]	7	168	discount	2298d6c36e964ae4a3e7e9706d1fb8c2
6	2	[web, email, mobile, social]	10	240	discount	fafdcd668e3743c1bb461111dcafc2a4
7	0	[email, mobile, social]	0	72	informational	5a8bc65990b245e5a138643cd4eb9837
8	5	[web, email, mobile, social]	5	120	bogo	f19421c1d4aa40978ebb69ca19b0e20d
9	2	[web, email, mobile]	10	168	discount	2906b810c7d4411798c6938adc9daaa5

PORTFOLIO AFTER CHANGES

2. Change column name id to customer_id.
3. Extract year part from become_member_on.
4. Numbering gender column as: F : *0*, M: *1* and O: *2* values.

	gender	age	customer_id	became_member_on	income
1	0	55	0610b486422d4921ae7d2bf64640c50b	2017	112000.0
3	0	75	78afa995795e4d85b5d9ceeca43f5fef	2017	100000.0
5	1	68	e2127556f4f64592b11af22de27a7932	2018	70000.0
8	1	65	389bc3fa690240e798340f5a15918d5c	2018	53000.0
12	1	58	2eeac8d8feae4a8cad5a6af0499a211d	2017	51000.0
...	...	...	...	...	...
16995	0	45	6d5f3a774f3d4714ab0c092238f3a1d7	2018	54000.0
16996	1	61	2cb4f97358b841b9a9773a7aa05a9d77	2018	72000.0
16997	1	49	01d26f638c274aa0b965d24cefe3183f	2017	73000.0
16998	0	83	9dc1421481194dcd9400aec7c9ae6366	2016	50000.0
16999	0	62	e4052622e5ba45a8b96b59aba68cf068	2017	82000.0

14825 rows × 5 columns

PROFILE AFTER CHANGES

Transcript dataset:¶

1. Change column person to costumer_id.
2. Create separate columns for amount, reward and offer_id from value column.
3. Pivot offer_id column to different type of offers by reading from portfolio dataset.
4. Select only transaction and offer completed from event column. Based on we want to decide how costumer response for an offer.
5. Pivot categorical event and offer_type columns by making dummies variables.
6. Drop unnecessary columns.
7. Group by dataset by customer_id.

	customer_id	time	amount	reward	offer completed	bogo	discount
0	0009655768c64bdeb2e877511632db8f	5862	127.60	9.0	3	1	2
1	00116118485d4dfda04fdbaba9a87b5c	1224	4.09	0.0	0	0	0
2	0011e0d4e6b944f998e987f904e8c1e5	3660	79.46	13.0	3	1	2
3	0020c2b971eb4e9188eac86d93036a77	3864	196.86	14.0	3	1	2
4	0020ccbbb6d84e358d3414a3ff76cffd	5700	154.05	13.0	3	2	1
...	...	...	...	...	...	...	...
16573	fff3ba4757bd42088c044ca26d73817a	3408	580.98	9.0	3	1	2
16574	fff7576017104bcc8677a8d63322b5e1	3732	29.94	9.0	3	1	2
16575	fff8957ea8b240a6b5e634b6ee8eafcf	1896	12.15	0.0	0	0	0
16576	fffad4f4828548d1b5583907f2e9906b	5022	88.83	15.0	3	3	0
16577	ffff82501cea40309d5fdd7edcca4a07	7236	226.07	18.0	6	1	5

16578 rows × 7 columns

TRANSCRIPT AFTER CHANGE

After these wrangling, we can merge **transcript** and **profile** datasets based on share **customer_id.** Here is result:

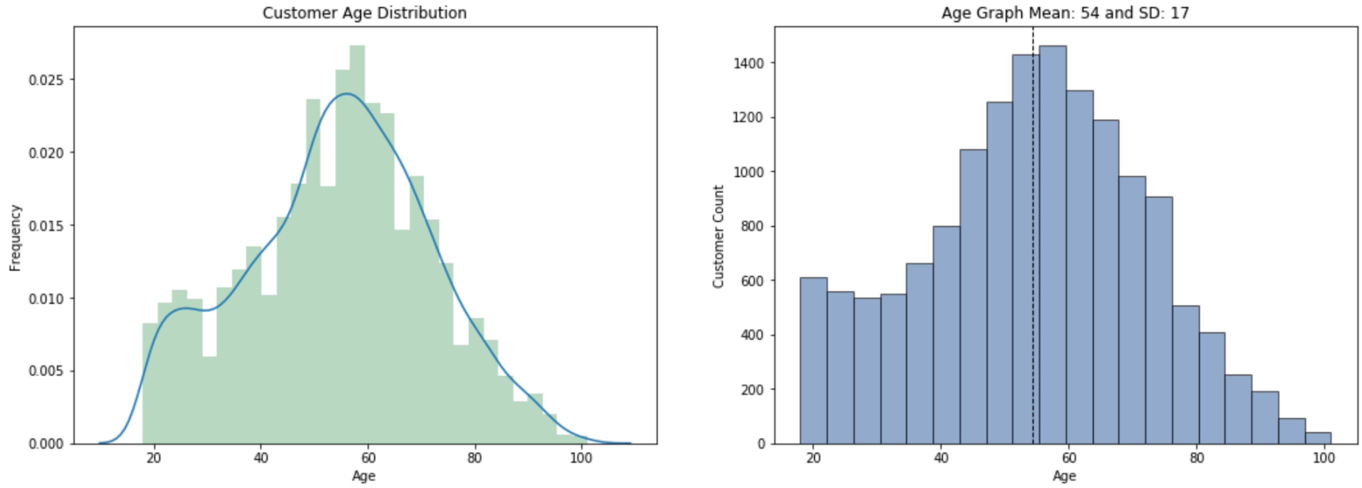
	customer_id	time	amount	reward	offer completed	bogo	discount	gender	age	became_member_on	income
0	0009655768c64bdeb2e877511632db8f	5862.0	127.60	9.0	3.0	1.0	2.0	1	33	2017	72000.0
1	0011e0d4e6b944f998e987f904e8c1e5	3660.0	79.46	13.0	3.0	1.0	2.0	2	40	2018	57000.0
2	0020c2b971eb4e9188eac86d93036a77	3864.0	196.86	14.0	3.0	1.0	2.0	0	59	2016	90000.0
3	0020ccbbb6d84e358d3414a3ff76cffd	5700.0	154.05	13.0	3.0	2.0	1.0	0	24	2016	60000.0
4	003d66b6608740288d6cc97a6903f4f0	9174.0	48.34	9.0	3.0	0.0	3.0	0	26	2017	73000.0

RESULT DATAFRAME MERGE PROFILE AND TRANSCRIPT

Data Visualization

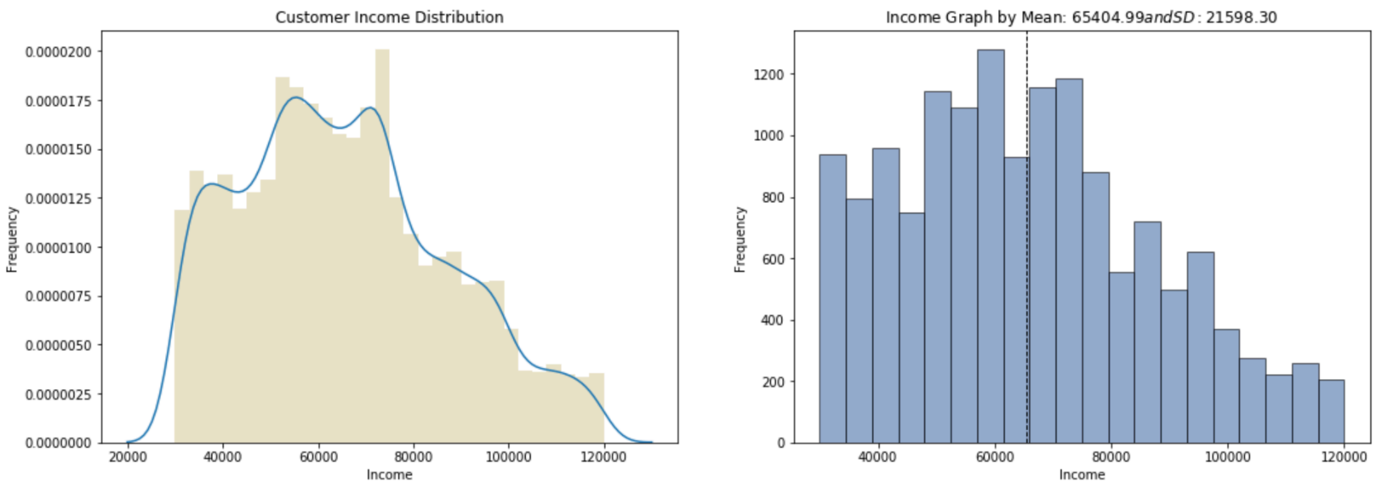
Visualizing data is one important part of showing data relation and exploratory data analysis phase. In this part, we review relation between different variables from datasets by graphs and dig into out datasets more, by plotting.

Age distribution and frequency of age between Starbucks customers, graph by following plots:



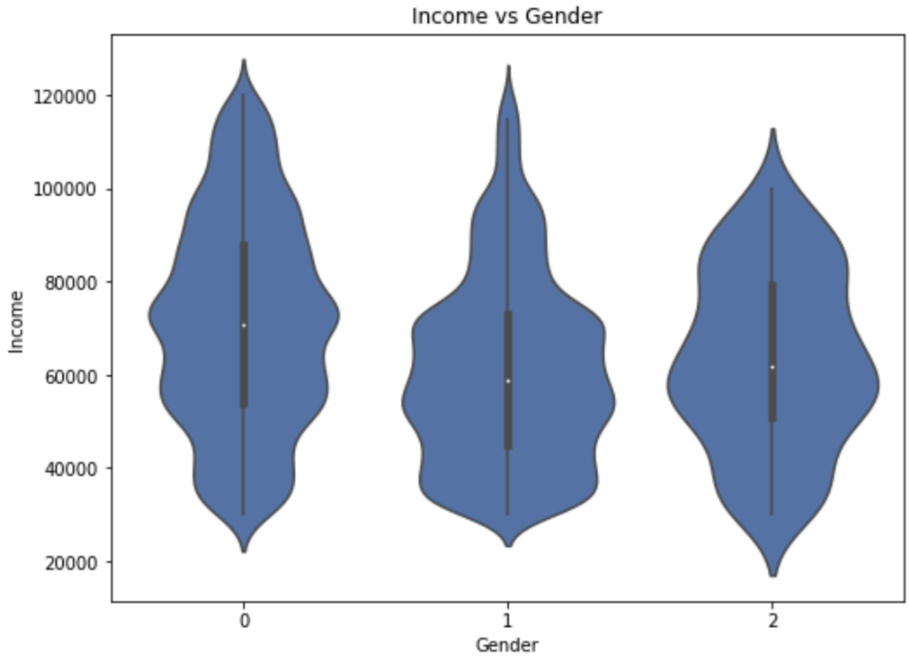
CUSTOMER AGE DISTRIBUTION AND COUNT OF THEM GRAPHS

Next one is **income** distribution and its frequency:



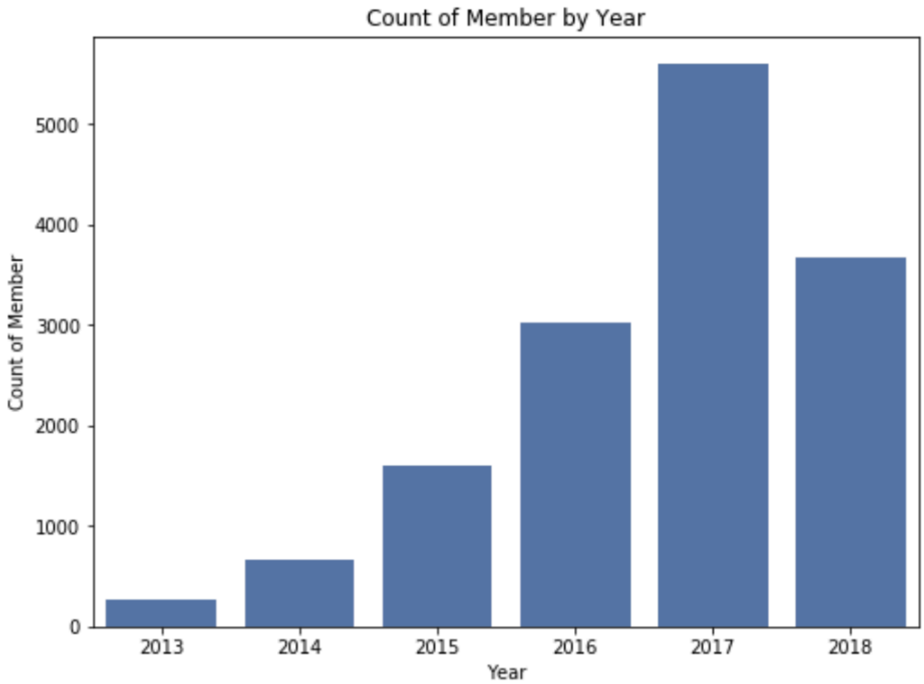
COSTUMER INCOME DISTRIBUTION AND COUNT OF THEM GRAPHS

Income vs **gender** is a bivariate graph:



CUSTOMER INCOME BY GENDER

Following plot shows how many member each year join Starbucks and install its mobile app:

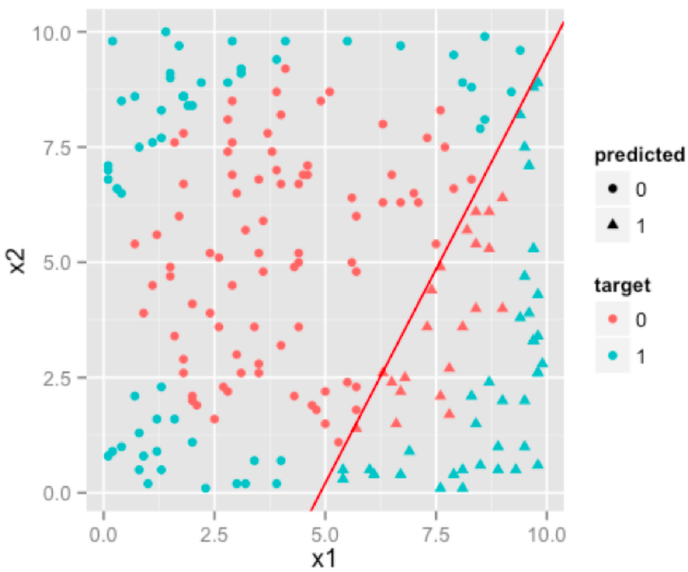


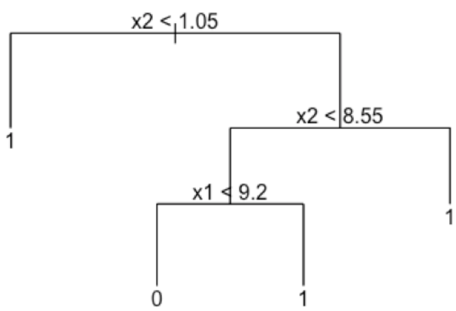
CUSTOMER FREQUENCY BY YEAR

Models Selection

Bias and variance are two characteristics of a machine learning model. Bias refers to inherent model assumptions regarding the decision boundary between different classes. On the other hand, variance refers a model's sensitivity to changes in its inputs. A logistic regression model constructs a linear decision boundary to separate successful and unsuccessful offers. Logistic Regression is great used here since we have few binomial outcomes. It also good here because we have a decent amount of data to work with. So, LR used in this model as **benchmark**. But the other models used are: Decision Tree, GaussianNB and SVM. In following we describe each these models and compare together to be clear why they chose.

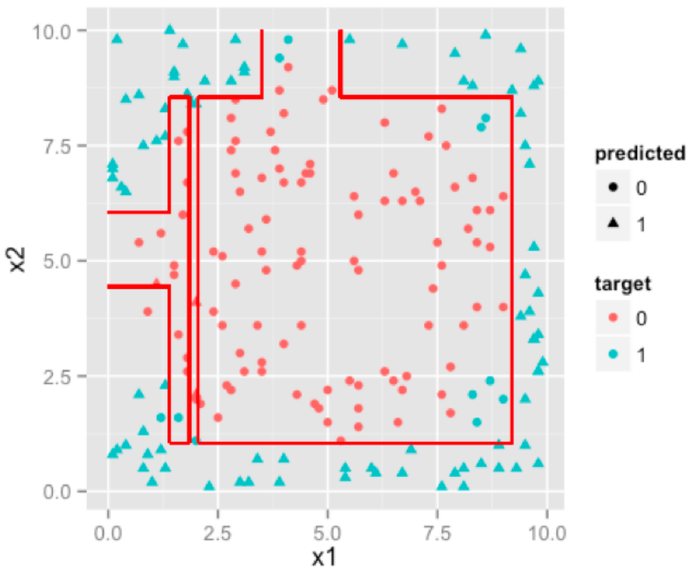
Logistic Regression is actually a classification model. In this method, decision boundary produced by logistic regression will always be linear , which can not emulate a circular decision boundary which is required. So, logistic regression will work for classification problems where classes are approximately linearly separable. (Although you can make classes linear separable in some cases through variable transformation, but we'll leave that discussion for some other day).





Decision Trees are made of hierarchical one variable rules . Such an example for our data is given. (in the left)

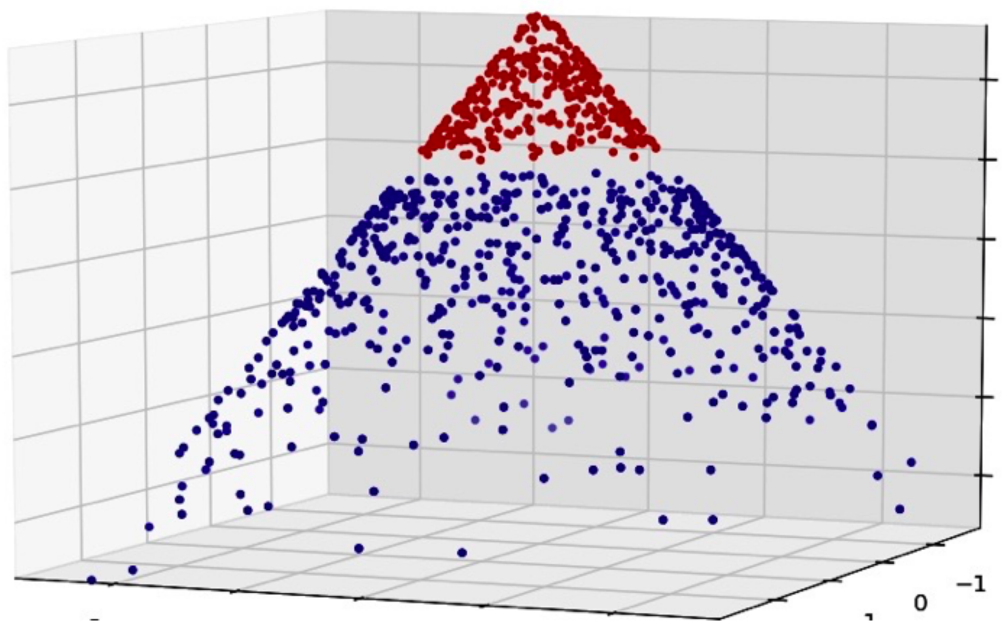
these decision rules
x2 |</>| const OR x1 |</>| const
do nothing but partition the feature space with lines parallel to each feature axis like the diagram given below:



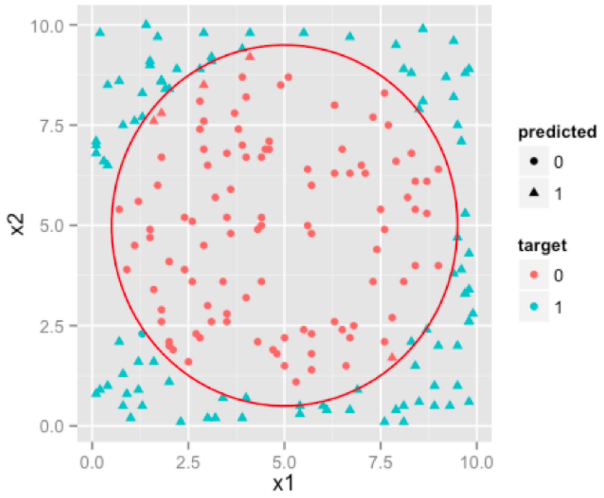
We can make our tree more complex by increasing its size , which will result in more and more partitions trying to emulate the circular boundary.

If you keep on increasing size of the tree , you'd notice that decision boundary will try to emulate circle as much as it can with parallel lines.So, if boundary is non-linear and can be approximated by cutting feature space into rectangles (or cuboids or hyper-cuboid for higher dimensions) then D-Trees are a better choice than logistic regression.

SVM (Support Vector Machines) works by projecting your feature space into kernel space and making the classes linearly separable. An easier explanation to that process would be that SVM adds an extra dimension to your feature space in a way that makes classes linearly separable. This planar decision boundary when projected back to original feature space emulates non linear decision boundary . Here this picture might explain better:



You can see that , once a third dimension in a special manner added to data , we can separate two classes with a plane (a linear separator), which once projected back onto the original 2-D feature space; becomes a circular boundary. see how



well SVM performs on our sample data.

Naïve Bayes assumes all the features to be conditionally independent. So, if some of the features are in fact dependent on each other (in case of a large feature space), the prediction might be poor.

Naive Bayes classifiers are built on Bayesian classification methods. These rely on Bayes's theorem, which is an equation describing the relationship of conditional probabilities of statistical quantities. In Bayesian classification, we're interested in finding the probability of a label given some observed features, Bayes's theorem tells us how to express this in terms of quantities we can compute more directly:

$$P(L\,|\,features) = \frac{P(features\,|\,L)P(L)}{P(features)}$$

This is where the "naive" in "naive Bayes" comes in: if we make very naive assumptions about the generative model for each label, we can find a rough approximation of the generative model for each class, and then proceed with the Bayesian classification. So, how about ***Gaussian Naive Bayes***? ¶Perhaps the easiest naive Bayes classifier to understand is Gaussian naive Bayes. In this classifier, the assumption is that data from each label is drawn from a simple Gaussian distribution.

One extremely fast way to create a simple model is to assume that the data is described by a Gaussian distribution with no covariance between dimensions. This model can be fit by simply finding the mean and standard deviation of the points within each label, which is all you need to define such a distribution.

So, based on description for each of models, they have some Pros an Cons which help us to select them for this project:

Logistic Regression Pros:

- Convenient probability scores for observations
- Efficient implementations available across tools
- Multi-collinearity is not really an issue and can be countered with L2 regularization to an extent

- Wide spread industry comfort for logistic regression solutions

Logistic Regression Cons:
- Doesn’t perform well when feature space is too large
- Doesn’t handle large number of categorical features/variables well
- Relies on transformations for non-linear features
- Relies on entire data

Decision Trees Pros:
- Intuitive Decision Rules
- Can handle non-linear features
- Take into account variable interactions

Decision Trees Cons:
- Highly biased to training set (Random Forests better)
- No ranking score as direct result

SVM Pros:
- Can handle large feature space
- Can handle non-linear feature interactions
- Do not rely on entire data

SVM Cons:
- Not very efficient with large number of observations
- It can be tricky to find appropriate kernel sometimes

Naive Bayes Pros:
- Computationally fast
- Simple to implement
- Works well with high dimensions

Naive BayesCons:
- Relies on independence assumption and will perform
- badly if this assumption is not met

Evaluation Metrics

The performance of models will be measured using two metrics, accuracy and F1 score. These measures coms from Confusion Matrix. This matrix simply shows for a model in supervised learning, how many time model correctly predict and how many time wrongly predict. Following table shows this matrix:
So based on this matrix, each cell for this project, apply following:

- **True Positive (TP)**: Send offer and customer will likely use it.

		True condition				
	Total population	Condition positive	Condition negative	Prevalence = $\frac{\Sigma \text{ Condition positive}}{\Sigma \text{ Total population}}$	Accuracy (ACC) = $\frac{\Sigma \text{ True positive} + \Sigma \text{ True negative}}{\Sigma \text{ Total population}}$	
Predicted condition	Predicted condition positive	**True positive**	**False positive**, Type I error	Positive predictive value (PPV), Precision = $\frac{\Sigma \text{ True positive}}{\Sigma \text{ Predicted condition positive}}$	False discovery rate (FDR) = $\frac{\Sigma \text{ False positive}}{\Sigma \text{ Predicted condition positive}}$	
	Predicted condition negative	**False negative**, Type II error	**True negative**	False omission rate (FOR) = $\frac{\Sigma \text{ False negative}}{\Sigma \text{ Predicted condition negative}}$	Negative predictive value (NPV) = $\frac{\Sigma \text{ True negative}}{\Sigma \text{ Predicted condition negative}}$	
		True positive rate (TPR), Recall, Sensitivity, probability of detection, Power $= \frac{\Sigma \text{ True positive}}{\Sigma \text{ Condition positive}}$	False positive rate (FPR), Fall-out, probability of false alarm $= \frac{\Sigma \text{ False positive}}{\Sigma \text{ Condition negative}}$	Positive likelihood ratio (LR+) = $\frac{\text{TPR}}{\text{FPR}}$	Diagnostic odds ratio (DOR) $= \frac{\text{LR+}}{\text{LR−}}$	F_1 score = $2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$
		False negative rate (FNR), Miss rate $= \frac{\Sigma \text{ False negative}}{\Sigma \text{ Condition positive}}$	Specificity (SPC), Selectivity, True negative rate (TNR) = $\frac{\Sigma \text{ True negative}}{\Sigma \text{ Condition negative}}$	Negative likelihood ratio (LR−) = $\frac{\text{FNR}}{\text{TNR}}$		

CONFUSION MATRIX

- **False Positive (FP)**: Send offer but customer doesn't want it or doesn't use it. (type II error)
- **True Negative (TN)**: Do not send offer and customer doesn't want it or doesn't use it.
- **False Negative (FN)**: Do not send offer but customer would have likely used it if we sent it. (type I error)

Regard of above description, **F1** and **accuracy** two measures are used as performance metrics for different models in this project, calculate based on following formulas:

$$Accuracy = \frac{tp + tn}{tp + tn + fp + fn}$$

$$F = 2.\frac{precision\ .\ recall}{precision + recall}$$

Precision and recall with use in F1, are calculated on:

$$precision = \frac{TP}{TP + fP}$$

$$recall = \frac{TP}{TP + fN}$$

So now can say a Confusion Matrix is a graph that shows the TP, TN, FP, and FN counts.

Benchmark Model

As describe in previous section, (Model Selection) Logistic Regression is classification solution in learning machine learning problem and categorized as linear solution.
As we are predicting that a person will/ not respond to a given offer, this will be binary classification problem.
A logistic regression model constructs a linear decision boundary to separate successful and unsuccessful offers. Logistic Regression is great used here since we have few binomial outcomes. It also good here because we have a decent amount of data to work with. So, LR used in this model as **benchmark**. The Lr in Scikit-learn package use 5 different solver. All 5 solver applied on data in this project. Following table compare these cover together:

Compare Table of Solver in LR Scikit-learn Package

	Solvers				
Penalties	**liblinear**	**lbfgs**	**newton-cg**	**sag**	**saga**
Multinomial + L2 penalty	no	yes	yes	yes	yes
OVR + L2 penalty	yes	yes	yes	yes	yes
Multinomial + L1 penalty	no	no	no	no	yes
OVR + L1 penalty	yes	no	no	no	yes
Elastic-Net	no	no	no	no	yes
No penalty ('none')	no	yes	yes	yes	yes
Behaviors					
Penalize the intercept (bad)	yes	no	no	no	no
Faster for large datasets	no	no	no	yes	yes
Robust to unscaled datasets	yes	yes	yes	no	no

Also as another option, results of one student of DSND program who work on this project and use RandomForrest and GradianBoosting methods, used as benchmark.
His blog post can reach here.

Modeling Results and Improvement

For running selected models on data, first need to determine which variables are features and which one is label, a very important question that form all data wangling and analyzing through a data project. In other meaning questions which

be asked lead to data project. Answer of these/this question/s are/is subject of project and identify these variable.
General goal in this project is *'how customers respond to offers'* and specifically in my assumption how time, amount, reward, age, gender and income effect on customer to complete an offer. So, **offer completed** select as label variable and **time, amount, reward, age, gender, income** as features:

Features	time, amount, reward, age, gender, income
Label	offer completed

In next step split data by X group as feature variables and y for label part in two train and test.
After run the models, following results achieved:

Final Results

Models	Accuracy	F1 Score	Improvement Model
Logistic Regression			
liblinear	0.5573239	0.5684475	
lbfgs	0.5090369	0.5102536	
newton-cg	0.6565956	0.6610901	
saga	0.3968168	0.3974635	
sag	0.4014027	0.3999820	
GaussianNB	0.6118154	0.6125202	
Decision Tree	**0.6679255**	1.0	**0. 7178**
Support Vector Machine	0.6679255	0.3630149	

Based on results, **Decision Tree Classifier** has the best performance and because of higher F1 score rather than **SVM** select for improvement.

During improvement process an instance of model create and by using GridSearchCV technic, parameters like min_samples_split and max_depth tuned to fit the training data. As result showed improvement method works great on **DTC** and 5% increase performance.

As **Final Results** table shows all Logistic Regression models are not behave as the same. Even **saga** and **sag** that was expected return great rust act very bad. But

best sober, **newton-cgb**, has quite near accuracy to best result models, **Decision Tree** and **SVM**.
Decision Tree after improvement return completely better performance than LG models (as benchmark model).

But when results compare to work of one other student, which reach following results:

Final Result of another Person for Benchmark

classifiertype	accuracy	f1score
randomforest	0.742205	0.735510
gradientboosting	0.735610	0.724685
logisticregression	0.721678	0.716066
naivepredictor	0.470916	0.640303

our model has less performance but not significant, after improvement around 3% less performance shows.
The benchmark shows, classification methods, which used decision tree have the best performance, because both random forest and gradient boosting models also are a combination of multiple decision trees.

Conclusion

In this project, three dataset used as input data. First step was looking inside of these datasets, clean and prepared them based on factors want to analyze. In this step, tried to use best built-in function in Pandas to increase performance and reduced time rather than writing custom code.
Two of these datasets, transcript and profile, had been joined on their share column, customer_id. Then explore little deeper on datasets by plotting between variables.
In modeling section, 4 classification model apply on data: Logistic Regression, GuassianNB, Decision Tree and SVM.
In Regression Model, all five Solver, which are in Scikit_learn, used and best result achieve by newton-cg although result by this solver are not converge.

Between the rest models, Decision Tree and SVM have same accuracy of 66.55% (very near to newton_cg with 65.66%) and because Decision Tree has better F1 score, this model selected for improvement by GridSearch technic and it increased to 71.78% accuracy which seems great.

Future Improvements

There are several improvement point for this project; following share some:

- Using other tree-based classification like Random forest` as result has been achieved.
- Using different strategy on data variable modeling, like classification customers age to find their response.
- Using PCA. Dimensionality reduction helps reduce the noise in the data by projecting it from high-dimensional into low-dimensional space, while retaining as much of the variation as possible. The ML algorithms thus can identify patterns in the data more effectively (because of less variability in data) and more efficiently (because of less dimensions hence less computational power needed to make calculations).
- Using different portion of train and test data, to see how different model response, specially Logistic Regression.
- Working more on data visualization and using various Univariate, Bivarian and Multivarient explorations.