Machine Learning Engineer Nanodegree

Capstone Project

Hang Guo

Project Overview

The coffee retail giant company Starbucks owns a consumer mobile application, which sent out rewards and offers for customers occasionally. The business team in Starbucks want to find out the pattern of a customer that takes the offer, providing us the data sets that include customer profiles, transaction history and offer type data.

This project is designed to analyze the application data and designed and develop a machine learning model to classify that a particular group of customers will accept the offer. The model should test properly by using the testing data after deployed the machine learning model.

Problem Statement

From the given data we receive, Starbucks provides customer profile info, including age, number of days become members and income, transaction info including offer receive and complete, offer type info including communication channels and reward. To better understand which group of customers are more likely to take which type of the offer is the goal of this project. There is a lot of machine learning model that could be use to do the classifications, including, but not limited to, logistic regression and Decision Tree model.

Metrics

The metrics that we generally use for classification problems are Accuracy scores, recall and precisions. In this case, we are going to focusing on high recall score as we want to capture the number of customers who are willing to take Starbucks offers as much as possible(low false negative), assuming that the additional cost incur on those customers that actually didn't accept the offer but recommended by the model(False positive) is low.

Data Exploration

The input data set is given as following:

- 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

For a classification problem, first we need to look at each table and remove those column that are unique and descriptive, such as offer_id and email so on and so forth. Second, we need to make one-hot

encoding for those “type” column, for example, the channel and offer_type in portfolio table. After the transformation, the table look as following:

	reward	difficulty	duration	offer_id	bogo	discount	informational	email	mobile	social	web
0	10	10	7	ae264e3637204a6fb9bb56bc8210ddfd	1	0	0	1.0	1.0	1.0	0.0
1	10	10	5	4d5c57ea9a6940dd891ad53e9dbe8da0	1	0	0	1.0	1.0	1.0	1.0
2	0	0	4	3f207df678b143eea3cee63160fa8bed	0	0	1	1.0	1.0	0.0	1.0
3	5	5	7	9b98b8c7a33c4b65b9aebfe6a799e6d9	1	0	0	1.0	1.0	0.0	1.0
4	5	20	10	0b1e1539f2cc45b7b9fa7c272da2e1d7	0	1	0	1.0	0.0	0.0	1.0
5	3	7	7	2298d6c36e964ae4a3e7e9706d1fb8c2	0	1	0	1.0	1.0	1.0	1.0
6	2	10	10	fafdcd668e3743c1bb461111dcafc2a4	0	1	0	1.0	1.0	1.0	1.0
7	0	0	3	5a8bc65990b245e5a138643cd4eb9837	0	0	1	1.0	1.0	1.0	0.0

We will need to keep the offer_id at the moment as we need to use it to join with other tables.

For profile table, the useful information should be age, days become member, income and gender. These are all logically can be the candidate feature to classify for different group of customer. After we handle the missing data by converting missing income to 0 and missing gender for another column, the table is shown as following:

	age	person	became_member_on	income	missing_data	F	M	N	O
0	118	68be06ca386d4c31939f3a4f0e3dd783	20170212	0.0	1	0	0	1	0
1	55	0610b486422d4921ae7d2bf64640c50b	20170715	112000.0	0	1	0	0	0
2	118	38fe809add3b4fcf9315a9694bb96ff5	20180712	0.0	1	0	0	1	0
3	75	78afa995795e4d85b5d9ceeca43f5fef	20170509	100000.0	0	1	0	0	0
4	118	a03223e636434f42ac4c3df47e8bac43	20170804	0.0	1	0	0	1	0
...	...	...	...	...	...	...	...	...	...
16995	45	6d5f3a774f3d4714ab0c092238f3a1d7	20180604	54000.0	0	1	0	0	0
16996	61	2cb4f97358b841b9a9773a7aa05a9d77	20180713	72000.0	0	0	1	0	0
16997	49	01d26f638c274aa0b965d24cefe3183f	20170126	73000.0	0	0	1	0	0
16998	83	9dc1421481194dcd9400aec7c9ae6366	20160307	50000.0	0	1	0	0	0
16999	62	e4052622e5ba45a8b96b59aba68cf068	20170722	82000.0	0	1	0	0	0

We will keep the person id as we need to join with other table as the key.

For transcript, we will keep all of the offer received data and offer completed data and then make a self join to get the column “IsAccept”, which tells us whether the offer that has been sent out gets accepted by customer.

	person	time_x	offer_id	isAccept
0	78afa995795e4d85b5d9ceeca43f5fef	0	9b98b8c7a33c4b65b9aebfe6a799e6d9	1
1	a03223e636434f42ac4c3df47e8bac43	0	0b1e1539f2cc45b7b9fa7c272da2e1d7	0
2	e2127556f4f64592b11af22de27a7932	0	2906b810c7d4411798c6938adc9daaa5	0
3	8ec6ce2a7e7949b1bf142def7d0e0586	0	fafdcd668e3743c1bb461111dcafc2a4	0
4	68617ca6246f4fbc85e91a2a49552598	0	4d5c57ea9a6940dd891ad53e9dbe8da0	0
...	...	...	...	...
86427	d087c473b4d247ccb0abfef59ba12b0e	576	ae264e3637204a6fb9bb56bc8210ddfd	1
86428	cb23b66c56f64b109d673d5e56574529	576	2906b810c7d4411798c6938adc9daaa5	1
86429	6d5f3a774f3d4714ab0c092238f3a1d7	576	2298d6c36e964ae4a3e7e9706d1fb8c2	0
86430	9dc1421481194dcd9400aec7c9ae6366	576	ae264e3637204a6fb9bb56bc8210ddfd	1
86431	e4052622e5ba45a8b96b59aba68cf068	576	3f207df678b143eea3cee63160fa8bed	0

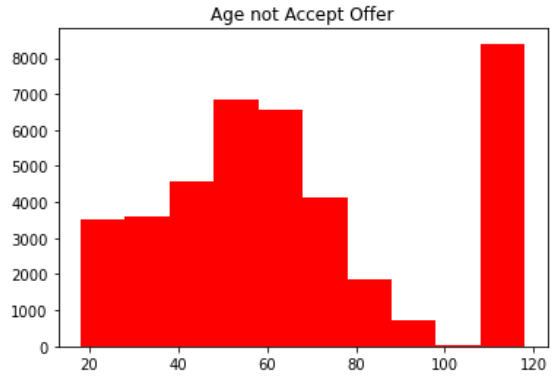
The other transaction will be ignored, as we only focusing on whether the offer is accepted or not.

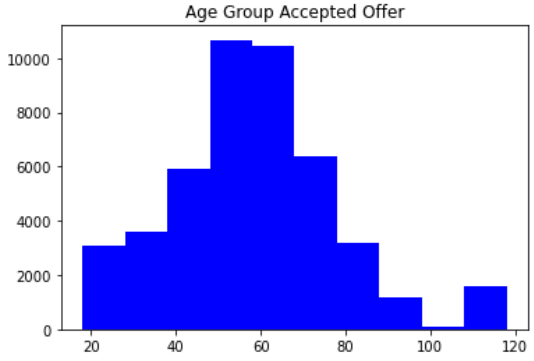
Finally, we will need to join the transcript offer data with the profile and portfolio:

	age	became_member_on	income	F	M	N	O	time_x	isAccept	reward	difficulty	duration	bogo	discount	informational	email	mobile	social	web
0	118	2017	0.0	0	0	1	0	168.0	0.0	2.0	10.0	7.0	0.0	1.0	0.0	1.0	1.0	0.0	1.0
1	118	2017	0.0	0	0	1	0	336.0	0.0	5.0	20.0	10.0	0.0	1.0	0.0	1.0	0.0	0.0	1.0
2	118	2017	0.0	0	0	1	0	408.0	1.0	2.0	10.0	10.0	0.0	1.0	0.0	1.0	1.0	1.0	1.0
3	118	2017	0.0	0	0	1	0	504.0	1.0	3.0	7.0	7.0	0.0	1.0	0.0	1.0	1.0	1.0	1.0
4	118	2017	0.0	0	0	1	0	576.0	1.0	2.0	10.0	10.0	0.0	1.0	0.0	1.0	1.0	1.0	1.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
36433	83	2016	50000.0	1	0	0	0	576.0	1.0	10.0	10.0	7.0	1.0	0.0	0.0	1.0	1.0	1.0	0.0
36434	62	2017	82000.0	1	0	0	0	0.0	1.0	3.0	7.0	7.0	0.0	1.0	0.0	1.0	1.0	1.0	1.0
36435	62	2017	82000.0	1	0	0	0	336.0	0.0	0.0	0.0	4.0	0.0	0.0	1.0	1.0	1.0	0.0	1.0
36436	62	2017	82000.0	1	0	0	0	408.0	1.0	5.0	5.0	5.0	1.0	0.0	0.0	1.0	1.0	1.0	1.0

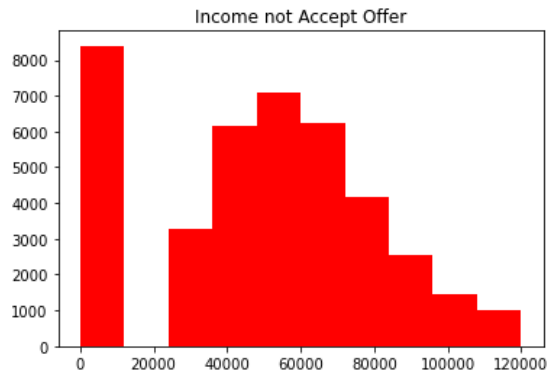
Exploratory Visualization

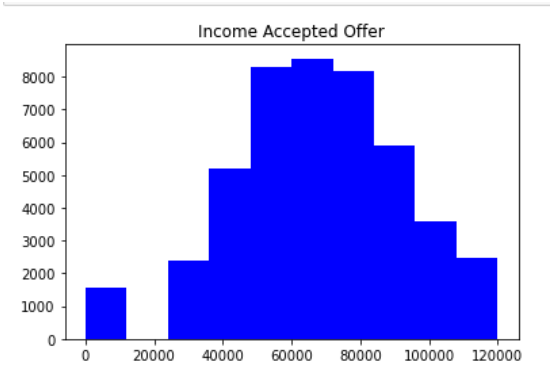
Age in terms of offer acceptance analysis:



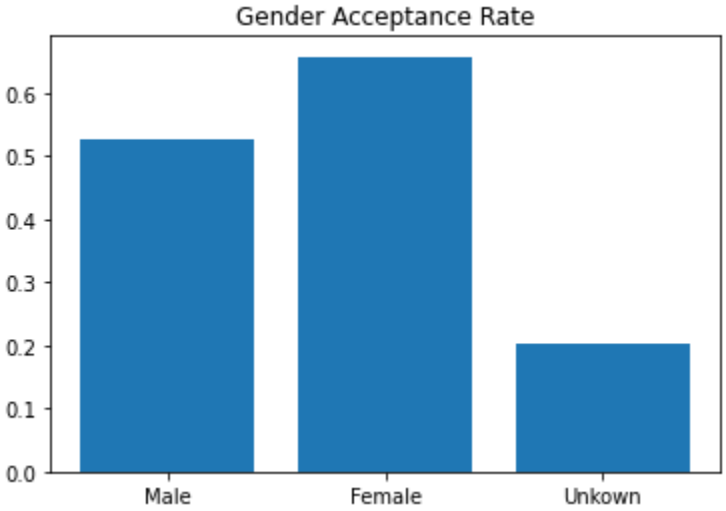


From the chart the most significant and useful information is the age group from 100 to 120 are more likely to accept the offer.





The customer group's income range from 0 – 20000 and income higher than 90000 shows less interest in picking up any type of offers.



Female is the group that are more likely to accept an offer.

Offer that have Reward less than one that are not attractive to any customers.

Algorithms and Techniques

Two general techniques that can be use for classifications are logistic regression and Decision Trees. Base on our visualization and data exploration, there is no clear linear relationship found between features and the outcome. However, decision trees might sometimes tend to overfitting so we need to spend some time to tune the model by trying different hyperparameters. In this project, I will use both Logistic Regression and Decision Tree algorithm to train the model and compare the Recall score between the two models.

- Decision Tree
  Advantages:
  1. Data scaling and normalization is not required.
  2. Missing data wouldn't affect the process of training the model.
  3. It is good to handle non-linear relationship data.
  4. Easy to understand.

  Disadvantage:

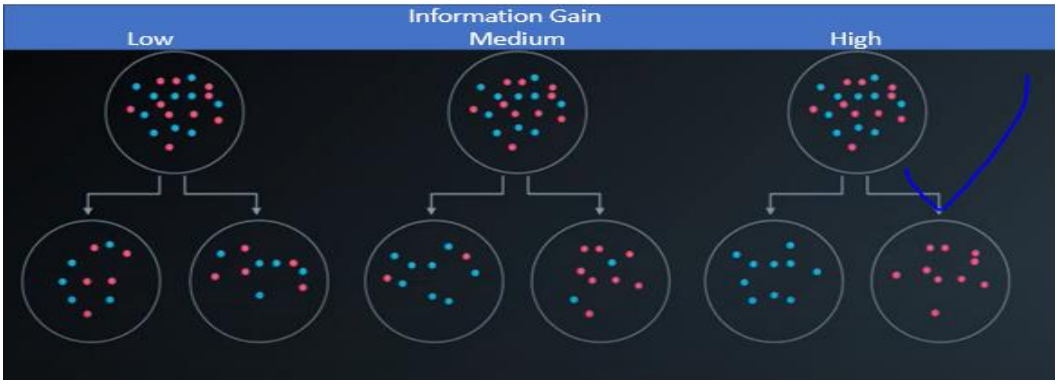
  1. Decision tree model training process may take long time along with the data size increase.
  2. Is not applicable for continuous value prediction.
  3. Highly likely to overfit the model.

- Logistic Regression
  Advantages:
  1. Very efficient to train.
  2. Performs well when the dataset is linearly separable.
  3. Less likely to over-fitting.

  Disadvantage:

  1. Performs bad when the data is not linearly separable.
  2. Tough to obtain complex relationships using logistic regression.

Because there is no clear linear relationship found in the data set and the size of the data set is small, so decision tree would be selected as the main model. The decision tree algorithm is to splitting the data into binary tree, each split will determine by the largest information gain of the split. Data splitting will stop until the data is not splitable anymore or reach the max limit we set in the hyper-parameter.



Benchmark Model

Benchmark Model would be Logistic Regression model by sklearn, and the training data set would be completely the same as the Decision Tree model.

Data Preprocessing

The data preprocessing step is being done in the feature engineering step:

- One hot encoding has been done to the fields like channels, offer_type, and gender.
- Dropping unique values such as offer_id, email, person_id as it is not helpful on classifying data.
- Extract year values from became_member_on in order to better classify data.

Implementation

- Machine Learning Algorithms

Logistic Regression: Group by data using linear method.

Decision Tree: Group by data using trees until enough information gain. Expected to be the winner of two models.

- Metrics

Recall Score: True Positive / (True Positive + False Negative)

We focusing on capturing customers who is likely to accept offers as many as possible, assume that we do not consider about the cost of advertisement.

- Implementation of Algorithms

Created custom scripts by using sklean module and use AWS Sagemaker SKLearn estimator object to train the model. After training the model, the model would be deployed to an endpoint and do a metric calculation.

Parameters for Decision Tree Model:

{max_depth=10, min_samples_leaf=6, min_samples_split=4)}.

This parameter yields the highest recall and precision score.

Parameters for Benchmark Model: Default

- Difficulties

The general coding and training model in my local laptop is a lot easier, but if I use the AWS Sagemaker as the interface to train the model, there is a little syntax different which we need to use the object in the sagemaker and follow it's technical requirement. Also, the job training process is taking longer time

for sagemaker around 7-10 minutes to train a simple model, compare to do so in local environment using sklearn which tooks around 0-1 second.

Refinement

The Decision models can be tune in order to avoid overfitting. Initially, if I don't set the hyper-parameter, the model is going to classify each different record into a single group, which will overtrain the model. The first training without setting hyper-parameter turns out the 0.80 recall score and 0.73 precision. After tunning the model by setting the max depth, minimum samples per leaf and minimum samples per split, the recall score go up to 0.89.

Model Evaluation and Validation

Before training the model, we split the data randomly by 80/20 into training data set, the final matrics are generated by predicting the testing feature and compare with the actual label. From the data visualization part, we could see from the histogram that the data set is covering wide range of data, so the test result of this model is trustable.

Justification

Decision Tree Model Score:

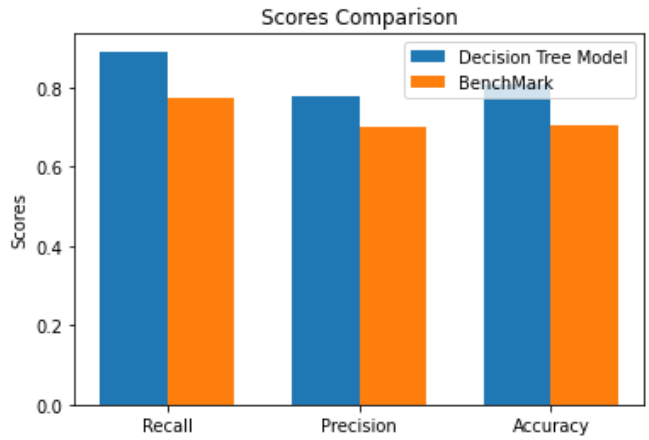
```
{'recall': 0.8903532461920709, 'precision': 0.7782079123784346}
```

Logistics Regression Model Score:

```
{'recall': 0.8032840012963163, 'precision': 0.7192880634552138}
```

The decision tree model's Accuracy score and recall score is higher than Logistics Model's, which is under our expectation.

Free-Form Visualization



From above bar chart, we could see that all the decision tree metrics are higher than the benchmark model.

Reflection

Initially, when I looked at the data set and I found that the age and income in profile has the linear relationship between the mean possibility of accepting the offer, and so I decided to use the linear regression algorithm to train the model. However, the outcome turns out that there is a huge difference between prediction result and the true value. When I looked back into the training and testing data, the training data is not aggregated, but the linear relationship I found before was after the data being aggregated to mean. In this case, I changed the strategy to use classification by using decision tree model.

The Decision tree algorithm is to split the data into binary trees base on the largest information gains per split, this is a good fit for those classification problems that do not have a linear relationship. The final recall score is 0.89 which makes the model a good one to predict whether they would likely to accept the offer.

Improvement

The final model solution we choose for is the decision tree model, which seems a better candidate as compare to logistic regression. There is another potential improvement which we can use the Random forest algorithm to avoid overfitting the model. The random forest is to take portion of the features and generate multiple decision trees and then combine it together to get the final output.