Starbucks capstone project

AWS Machine Learning Engineer Nanodegree Program

Dec 2022

Mina Nessim

Project Overview

The project analyzes data of 30 days related to 17K customers and 10 offers provided by Starbucks, the main goal is to find a good machine learning model to predict either the customer will buy a product to use the offer or not.

On a large scale this algorithm is very effective to optimize marketing budget by sending only campaigns that will most likely lead to sales.

Problem Statement

While working on the project I faced many problems, but the main one was to convert the data into clean data that machine learning model can use effectively to produce a meaningful result.

The data has a lot of variabilities, many data frames that need to be merged in the right way, also getting the dummies of variables without keeping redundant information,

One example was the offer id and offer details, is it better to get dummies for the 10 offers, or remove the offer id and keep the details only, while the offer id will be more direct way to do that, using it's specifications would help in anticipating the behavior of new promotions later on.

Also the problem of having transactions in separate lines than offers, without offer id to show which transaction is related to which offer, but that also was solved during processing.

The strategy to solve this problem is to merge all lines into 1 kind of rows which is related to customer/offer, and include if offer is viewed, completed or not, then drop all non-relevant rows

From that we will start to run machine learning algorithm with all relevant columns as features, and if the offer will be viewed and completed as the target,

Final result should be a model that tells us if the sending the offer to specific customer will lead to offer completion or not, provided customer demographics

Metrics

We will use accuracy for this problem to define if the machine learning model is effective or not.

After defining the best model, we will test accuracy, fbeta, recall ,precision

Data Exploration

After importing the data, I started with basic exploration for the data, and find the simple relation in the row data, there are 17K customers and 10 offers, each customer has his age, gender, income mentioned, with some customers with no data,

Also for the offers there are the offer type and broadcasting method

Transaction file is the most complicated, as it has 4 types of events (transaction, offer received, offer viewed, and offer completed), offers and transactions are not connected with offer ID, that makes it hard to connect together.

Out[457]:

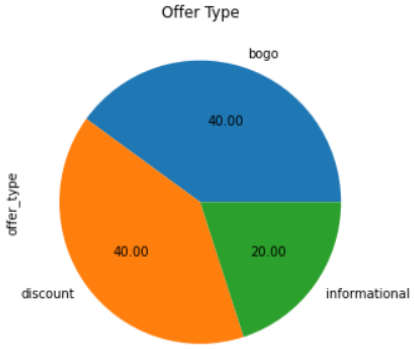
	offer_reward	offer_difficulty	offer_duration	offer_type	offer_id	channel_email	channel_mobile	channel_social	channel_web
0	10	10	7	bogo	ae264e3637204a6fb9bb56bc8210ddfd	1	1	1	0
1	10	10	5	bogo	4d5c57ea9a6940dd891ad53e9dbe8da0	1	1	1	1
2	0	0	4	informational	3f207df678b143eea3cee63160fa8bed	1	1	0	1
3	5	5	7	bogo	9b98b8c7a33c4b65b9aebfe6a799e6d9	1	1	0	1
4	5	20	10	discount	0b1e1539f2cc45b7b9fa7c272da2e1d7	1	0	0	1
5	3	7	7	discount	2298d6c36e964ae4a3e7e9706d1fb8c2	1	1	1	1
6	2	10	10	discount	fafdcd668e3743c1bb461111dcafc2a4	1	1	1	1
7	0	0	3	informational	5a8bc65990b245e5a138643cd4eb9837	1	1	1	0
8	5	5	5	bogo	f19421c1d4aa40978ebb69ca19b0e20d	1	1	1	1
9	2	10	7	discount	2906b810c7d4411798c6938adc9daaa5	1	1	0	1

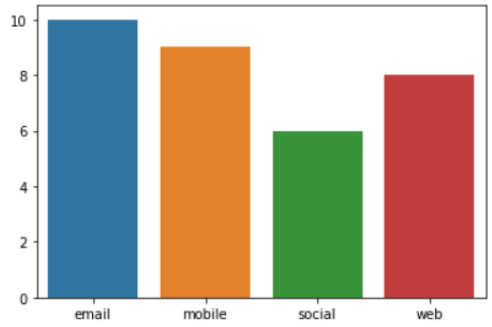
]:

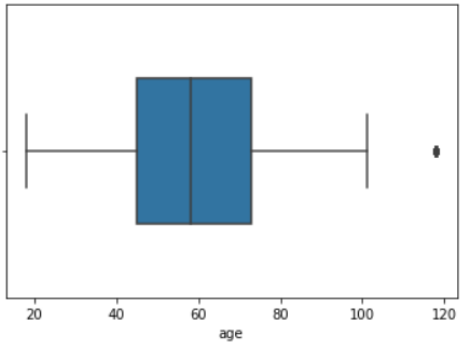
	gender	age	id	became_member_on	income
0	None	118	68be06ca386d4c31939f3a4f0e3dd783	20170212	NaN
1	F	55	0610b486422d4921ae7d2bf64640c50b	20170715	112000.0
2	None	118	38fe809add3b4fcf9315a9694bb96ff5	20180712	NaN
3	F	75	78afa995795e4d85b5d9ceeca43f5fef	20170509	100000.0
4	None	118	a03223e636434f42ac4c3df47e8bac43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

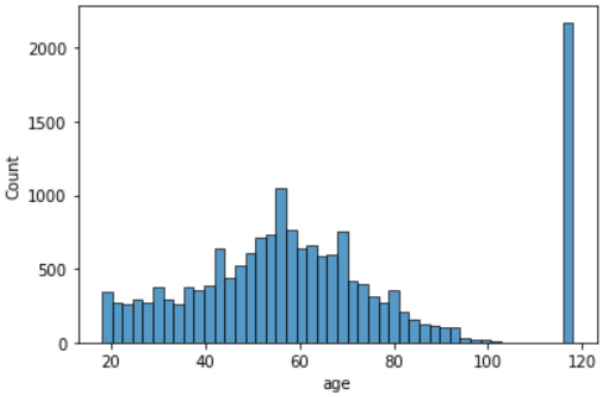
	person	event	value	time
0	78afa995795e4d85b5d9ceeca43f5fef	offer received	{'offer id': '9b98b8c7a33c4b65b9aebfe6a799e6d9'}	0
1	a03223e636434f42ac4c3df47e8bac43	offer received	{'offer id': '0b1e1539f2cc45b7b9fa7c272da2e1d7'}	0
2	e2127556f4f64592b11af22de27a7932	offer received	{'offer id': '2906b810c7d4411798c6938adc9daaa5'}	0
3	8ec6ce2a7e7949b1bf142def7d0e0586	offer received	{'offer id': 'fafdcd668e3743c1bb461111dcafc2a4'}	0
4	68617ca6246f4fbc85e91a2a49552598	offer received	{'offer id': '4d5c57ea9a6940dd891ad53e9dbe8da0'}	0

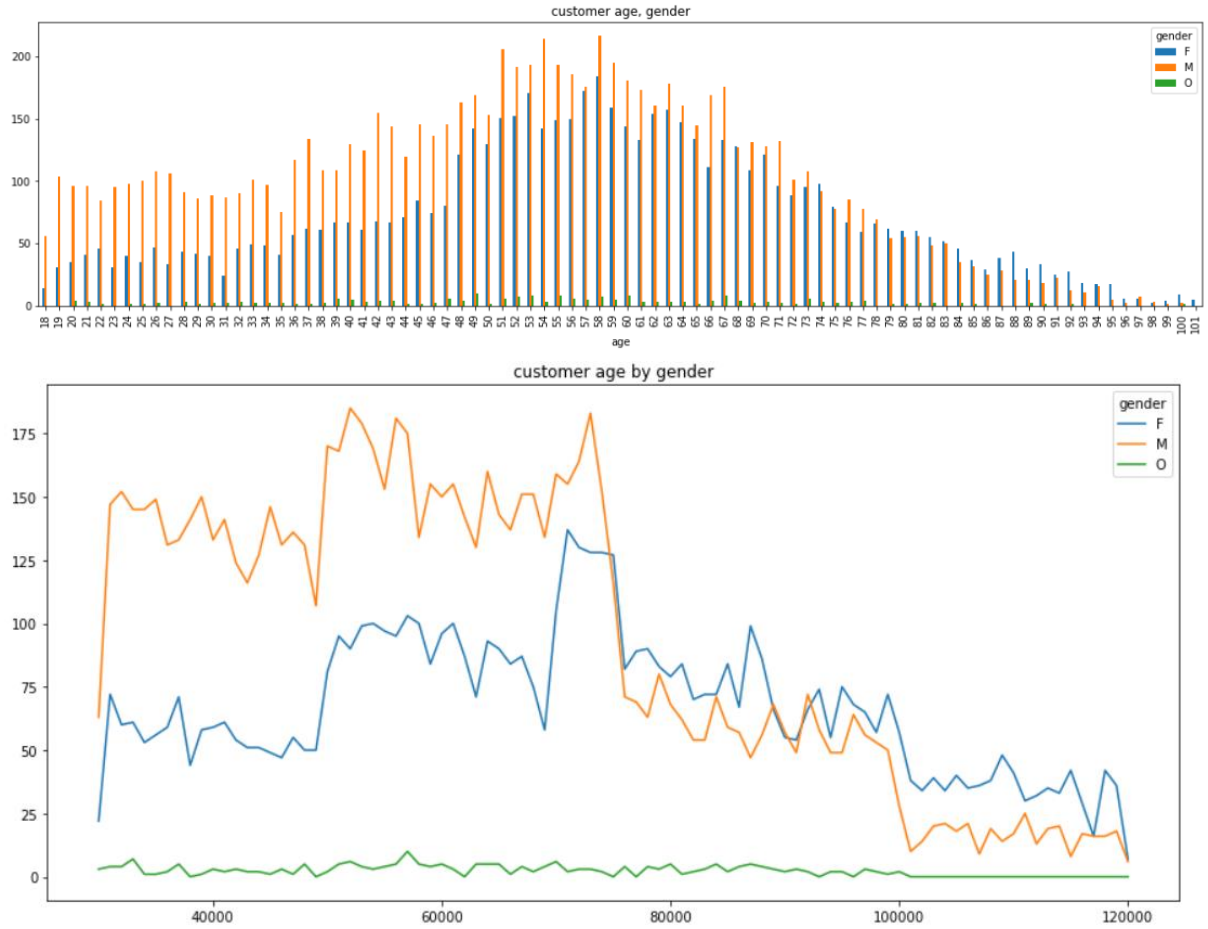
Exploratory Visualization











Data Processing

After data exploration, I started to work on merging the 3 data frames to get one data frame containing all important information

Then I started to merged transactions with offer completed events, as they should share same customer and same time and be on ordered,

Then creating a for loop to check per customer, and mark the received transaction either it's received only, or received, viewed but not done, received viewed and done, received and done but not viewed, or received, done but viewed after the transaction was done

This was the longest step in the processing

After that I dropped unneeded rows. So the machine learning model would work only on related events

Then I created dummy variables for all categorial variables, standardized the continues variables , then dropped all unneeded columns to avoid redundant information,

After that I started to apply and try the machine leaning techniques to get the best one

Algorithms and Techniques

This problem is a classification supervised problem, for that I needed to find relative models that can help in this problem, so I choose the below 4 models

AdaBoostClassifier, GaussianNB, DecisionTreeClassifier, RandomForestClassifier

After trying each of them with default parameters the results for training and testing were as below

Training with AdaBoostClassifier
Training Accuracy: 0.8777619621154057
Testing Accuracy : 0.8753120483510708
Training with GaussianNB
Training Accuracy: 0.7298149567502463
Testing Accuracy : 0.7285507817632374
Training with DecisionTreeClassifier
Training Accuracy: 1.0
Testing Accuracy : 0.8266325055840231
Training with RandomForestClassifier
Training Accuracy: 0.9999781013905616
Testing Accuracy : 0.867691499145973

Random forest classifier was chosen as the one to continue with, another good option would be the Ada Boost Classifier, I choose random forest as the testing accuracy is too close and the training accuracy is much better.

For hyperparameter tuning, using GridSearchCV would take hours and too much computational power, so instead I I ran RandomizedSearchCV for the hyper parameter tuning.

These are the hyperparameters

```
params = {
   'n_estimators': [75, 100, 125, 150],
   'max_features': ['sqrt', 'log2', None],
   'max_depth' : [10,20,30,40,50],
   'min_samples_split' : [2,10,15,20],
   'min_samples_leaf' :[1,2,5,10]}
```

Then I chose number of iterations to be 200

After running the randomized search, it gave the best parameters

{'n_estimators': 75,
 'min_samples_split': 15,
 'min_samples_leaf': 10,
 'max_features': None,
 'max_depth': 10}

So after running the model again the new accuracy for the model is
Training Accuracy: 0.8854264754188109
Testing Accuracy : 0.8790566285639206

Benchmark

To be able to compare the result, I ran hyperparameter tuning to Ada boost classifier to get another res ult to be able to benchmark with it

After running hyperparameter tuning
params = {
   'n_estimators': [40, 45, 50, 55, 60],
   'learning_rate' : [0.8,0.9,1,1.1,1.2],
   'algorithm':['SAMME', 'SAMME.R']
 }

The best parameters are
{'n_estimators': 60, 'learning_rate': 1.1, 'algorithm': 'SAMME.R'}

And the results are
Training Accuracy: 0.8772363954888864
Testing Accuracy : 0.8745237156746813

The random forest classifier is slightly better than Ada boost classifier with about .5%

Implementing process

To be able to implement the model, data needed to be precise for the machine learning model, having separate lines with different events like offer received and offer viewed doesn't provide a clear out put f or the model to work on, there should be a clear features to test and clear target to predict,

That's why merging the data into on line per offer, dropping non related transactions was a good option for the last form of the data.

After that I needed to find a good model for supervised classification, as mentioned in the previous step I tried some of them, but random forest provided a good result.

Improving process

Improving the results has two parts

Finding the best model and find the best hyperparameters through hyperparameter tuning

Fist I tried 4 different models, and got the best 2 out of them

To determine which of them is better, I tried hyperparameters tuning for both models

With random grid CV, I got the best hyperparameters for both models, as mentioned before

And then I created a model for both and got the accuracy scores

Results

After I got the two models, I created another test to get the accuracy score, fbeta, recall and precision sc ores and compare both models to each other

Results were

For random forest

Testing Accuracy : 0.8789252397845224
Testing Fbeta : 0.8749219248728783
Testing Recall : 0.9262532133676092
Testing Precision : 0.8064064904182403

For ada boost

Testing Accuracy : 0.8745237156746813

Testing Fbeta : 0.8698258799425693
Testing Recall : 0.8921915167095116
Testing Precision : 0.8175795053003534

Random forest is better for overall scores, for 3 out of 4 scores, yet the only one is for precision, yet the difference is only 1%

To make sure of the results, tried same model with different seeds and got the below results

Seed : 42

Testing Accuracy : 0.8788595453948232
Testing Fbeta : 0.8747750101656269
Testing Recall : 0.9282605191036595
Testing Precision : 0.8045270364677938

Seed :789

Testing Accuracy : 0.8775456576008409
Testing Fbeta : 0.8739222733527837
Testing Recall : 0.9311338885329917
Testing Precision : 0.8021523178807947

Scores are similar for different random seeds. For the 4 scores, after testing on 2 models