Capstone Report

I. Definition

Project Overview

This project is the final project of Udacity Machine Learning Engineer Nanodegree Program. The data provided by Udacity are simulated data that mimics behaviors of customers of Starbucks rewards app. Once a customer registered this app, he/she will receive offers such as discounts, buy-one-get-one-free, or advertisements.

Problem Statement

Customers response differently to offers. Some complete after viewing the offers, and some doesn't; some complete without viewing the offers, and some view after completing the offers; some complete offer A but not offer B, and some does the opposite; some complete soon after receiving the offers, and some wait till the last second to complete the offers.

The goal is to build a model that finds the best offer for each customer. Since this project aims at maximizing completion rate, I define the "best" offer as the one that has highest completion rate.

Metrics

This is a multiclassification problem, and the target could be any offer, so I will use accuracy score and F1 score to measure the performance of model. Accuracy score focuses on the average per-class effectiveness of a classifier, and F1 score focuses on relations between data's positive labels and those given by a classifier based on a per-class average[1]. I will draw a confusion matrix and start with calculating precision and recall scores, and then calculate the F1 scores.

II. Analysis

Data Exploration

The 3 datasets are provided here in json format.

The portfolio.json file contains basic information about 10 different offers and it has 6 columns:

1. The "reward" column shows the reward points a customer will obtain once he or she completes the offer, and reward points range from 0 to 10
2. The "channels" column shows different channels that the offers are distributed, including web, email, mobile, and social. The values are in list type, so later I will separate the list into 4 columns.

3. The “difficulty” columns shows the minimum amount of money a customer has to spend in order to complete the offer, and difficulty ranges from 0 to 10
4. The “duration” column shows the valid period of the offer, and duration ranges from 3 days to 10 days.
5. The “offer_type” column shows the type of the offer, including bogo, discount, and informational.
6. The “id” column shows the unique id of each offer. The type of id is string and each id contains meaningless characters, so later I will simplify them.

The profile.json file contains basic information about 17000 customers of Starbucks Rewards Program, and it has 5 columns:

1. The “gender” column shows the gender of customers, including F, M, O, and unknown values.
2. The “age” column shows the age of customers, ranging from 18 to 118.
3. The “id” column shows the customer id
4. The “became_member_on” column shows the day each customer registered the program
5. The “income” column shows the annual income of each customer, ranging from $29999.999 to $120000

There are 2175 customers with no gender and income information, and an age of 118, so it is reasonable to treat age 118 as unknown values as well.

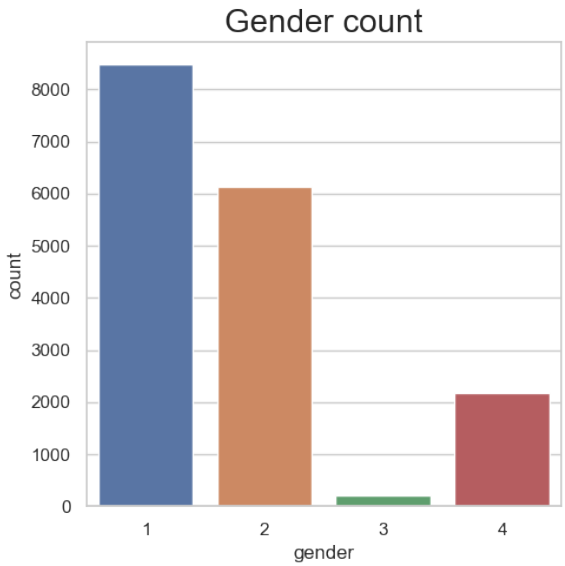
The transcript.json contains 306534 record of events, and it has 4 columns:

1. The “person” column shows customers ids
2. The “event” column shows different events, including “offer received”, “offer viewed”, “offer completed”, and “transactions”
3. The “value” column shows details of events, including offer_id, transaction amount, and reward points. The values are in dictionary format, so later I will separate the dictionary into several columns.
4. The “time” column shows hours between the event and the first event.

Exploratory Visualization

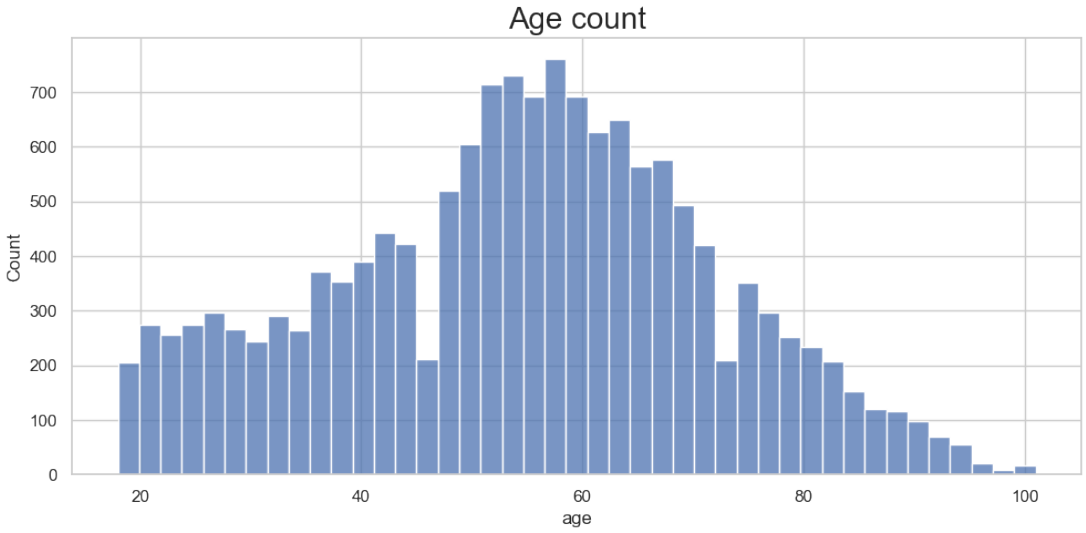
In order to analyze this data, I asked the following questions:

1. **What are the gender distribution?**

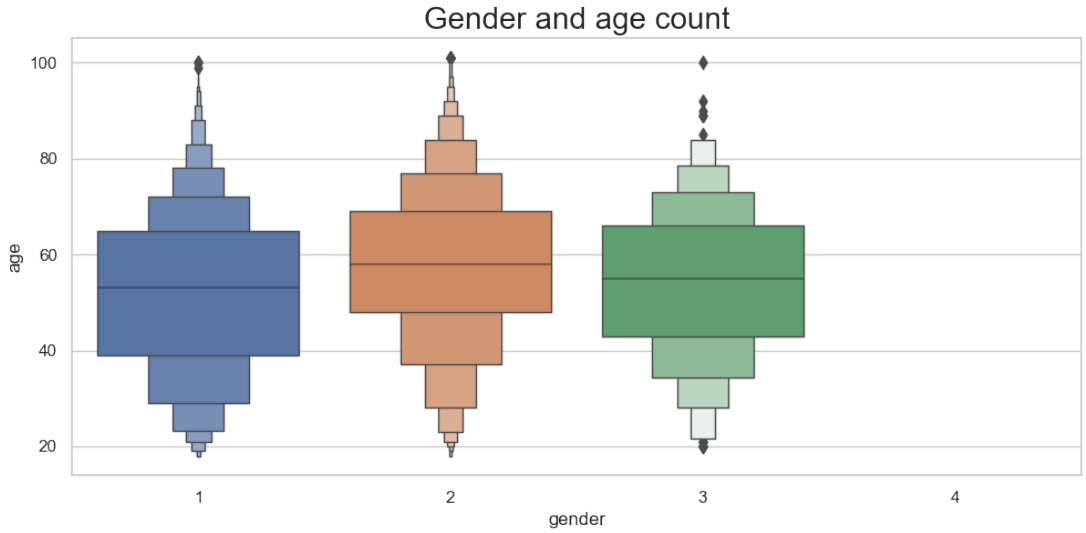


The nearly half of population is male, 36% is female, 1% is other gender, and 13% is unknown gender.

2. How old are the customers?

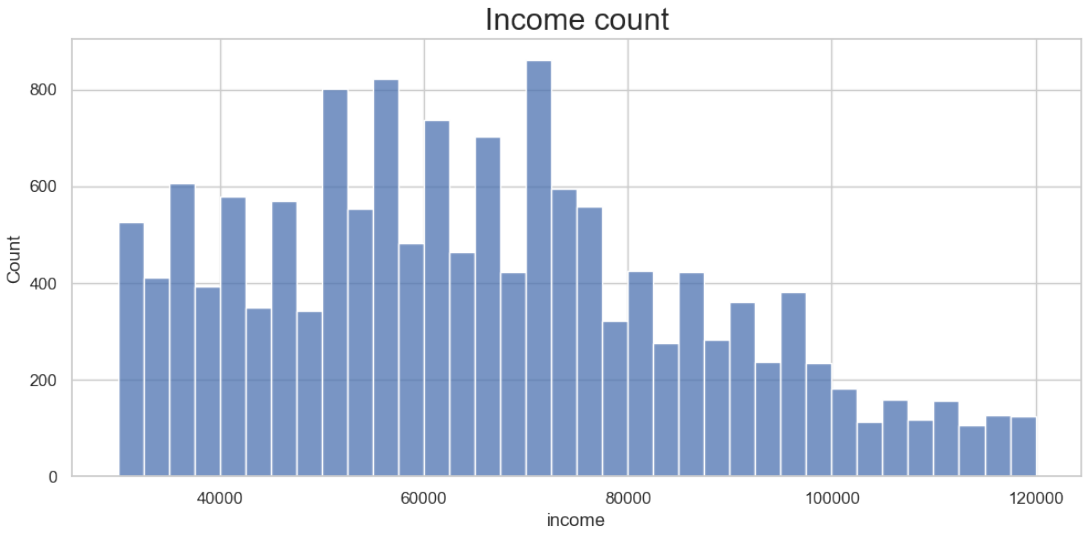


The majority of customers are between age 50 and 70, so this dataset consists mostly of old people.



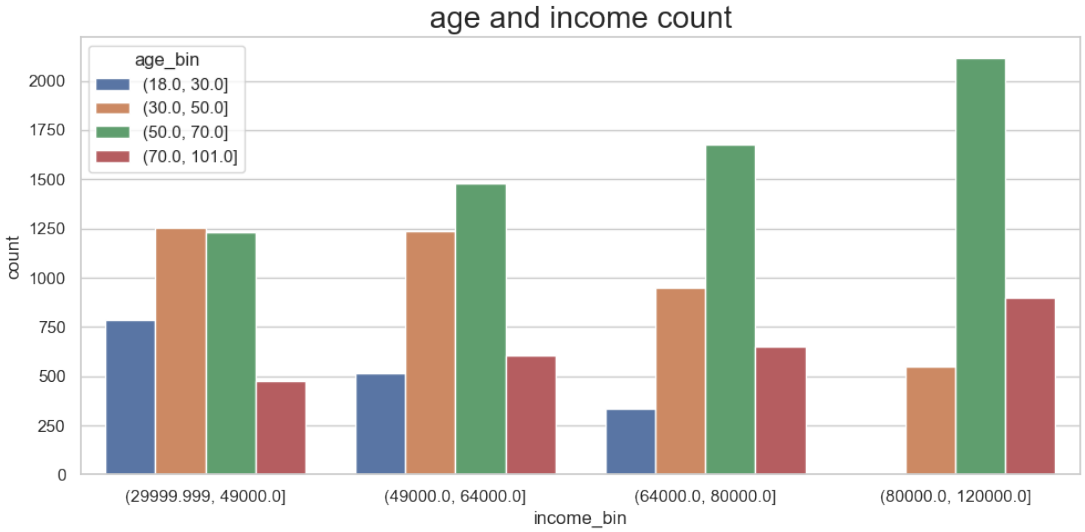
The percentage of old people in female group is higher than others.

3. How much do they earn each year?



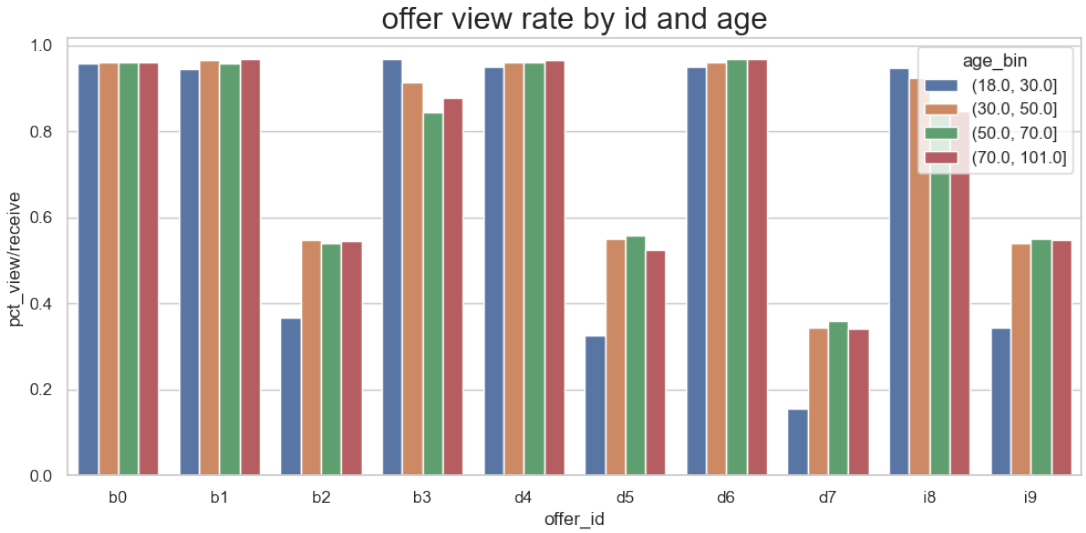
Majority of customers have annual income of $49000 to $80000.

Female customers earn more than other genders.



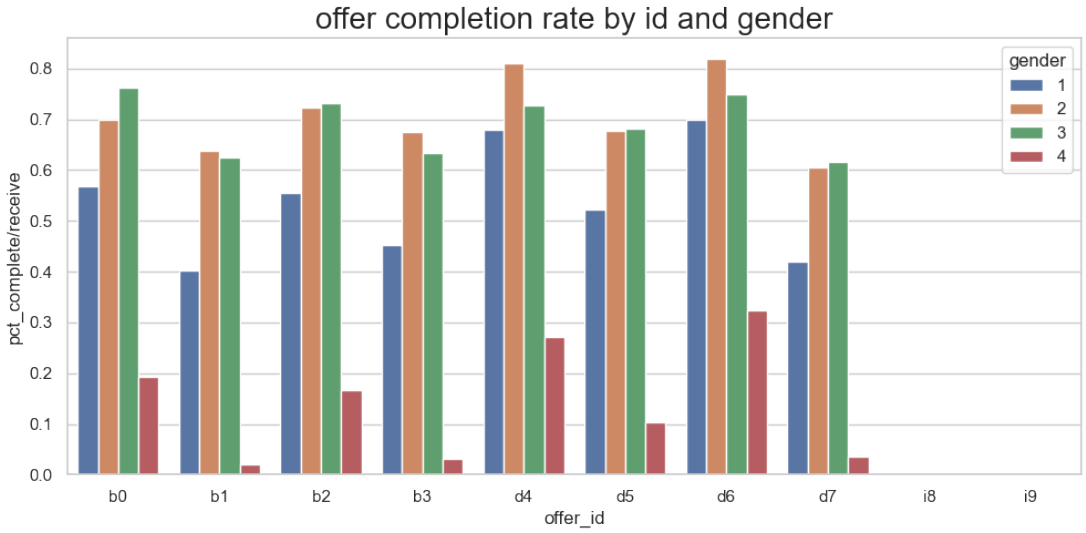
Age and income are positively correlated, and the poorest group is below age 30, the richest group is between age 50 and 70.

4. which type of offers are most attractive?

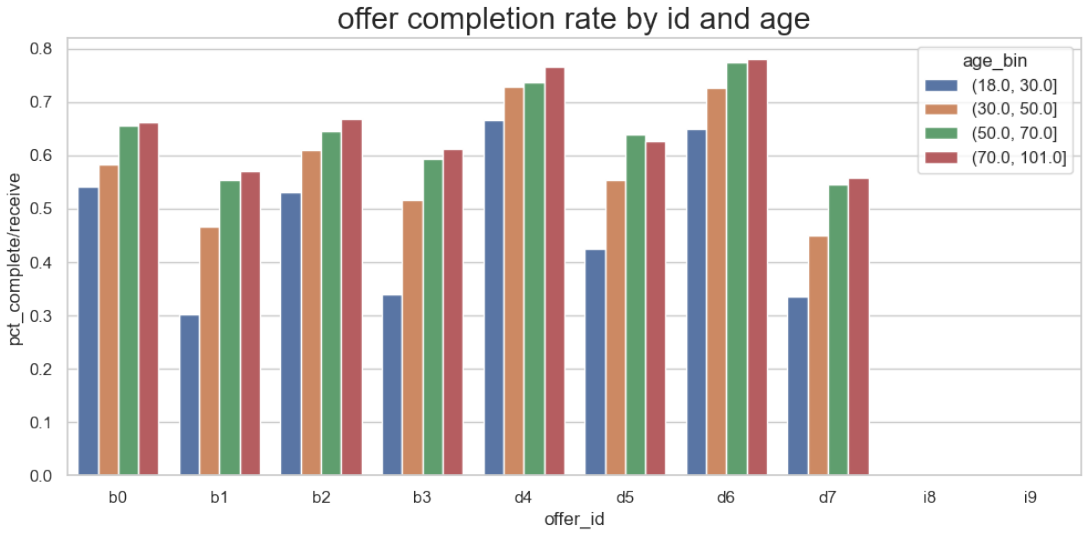


The offers with highest view rate are b0, b1, d4, and d6, because these 4 offers are distributed via all 4 channels (email, mobile, social, and web). And those offers with low view rate are not distributed via social channel.

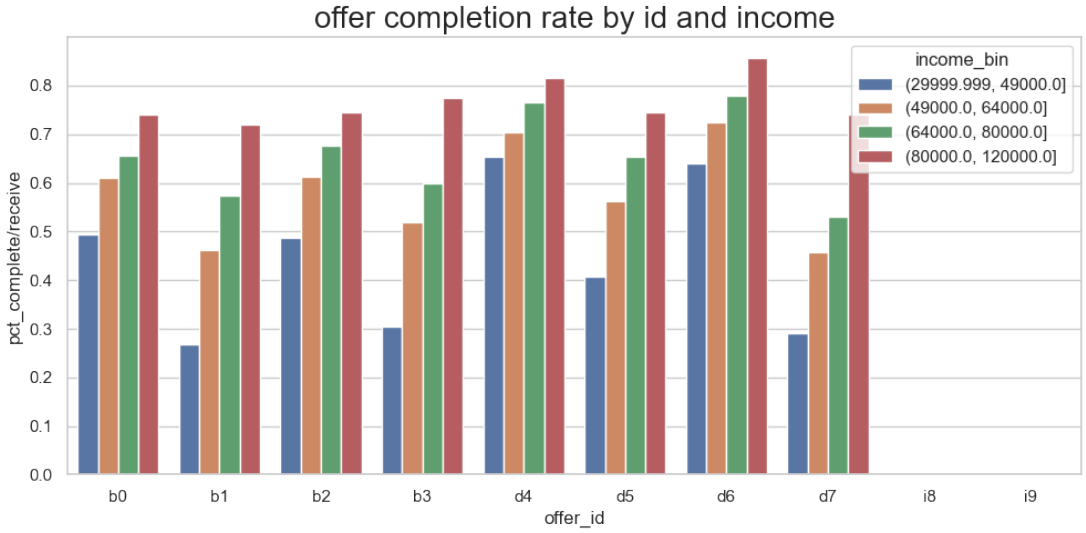
5. Which offer has highest completion rate?



Completion rate of d4 and d6 are slightly higher than others, and female customers have higher completion rate.

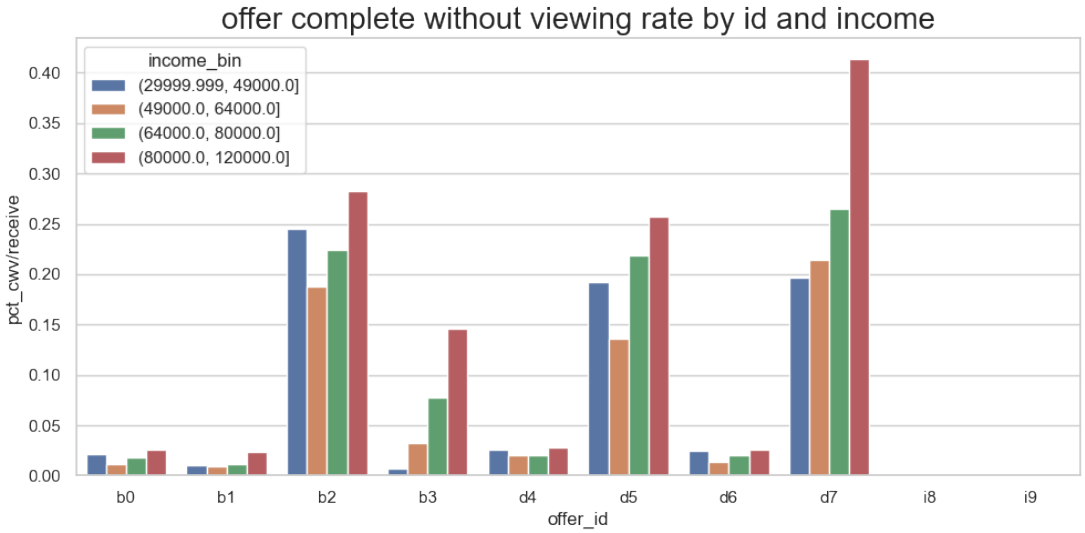


No matter which offer, age is positively correlated with completion rate



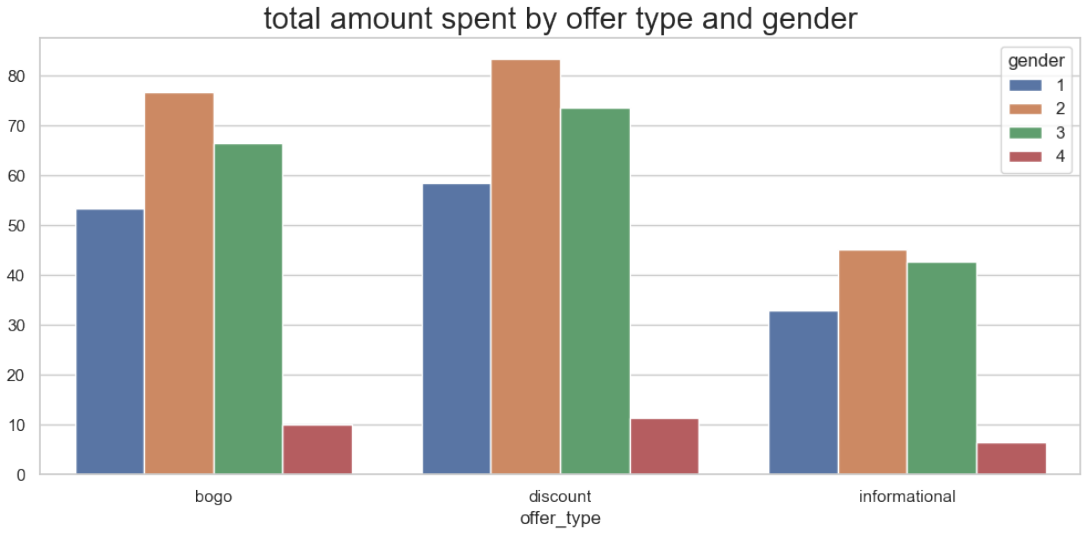
Income is also positively correlated with completion rate, because age is positively correlated with income as well.

6. Who is most likely to complete an offer without viewing it?

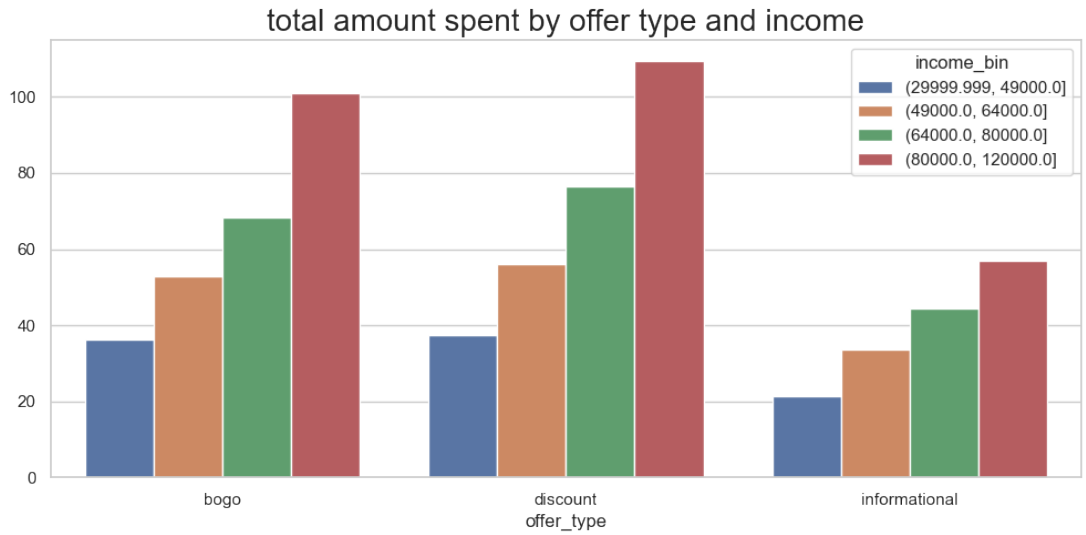


Offer b2, d5, and d7 have the highest complete_without_view_rate, because they are not distributed via social channel, and customers in highest income group have the highest compelte_without_view_rate, I guess it's because they are not sensitive to these offers.

7. How much did they spend after receiving each offer?

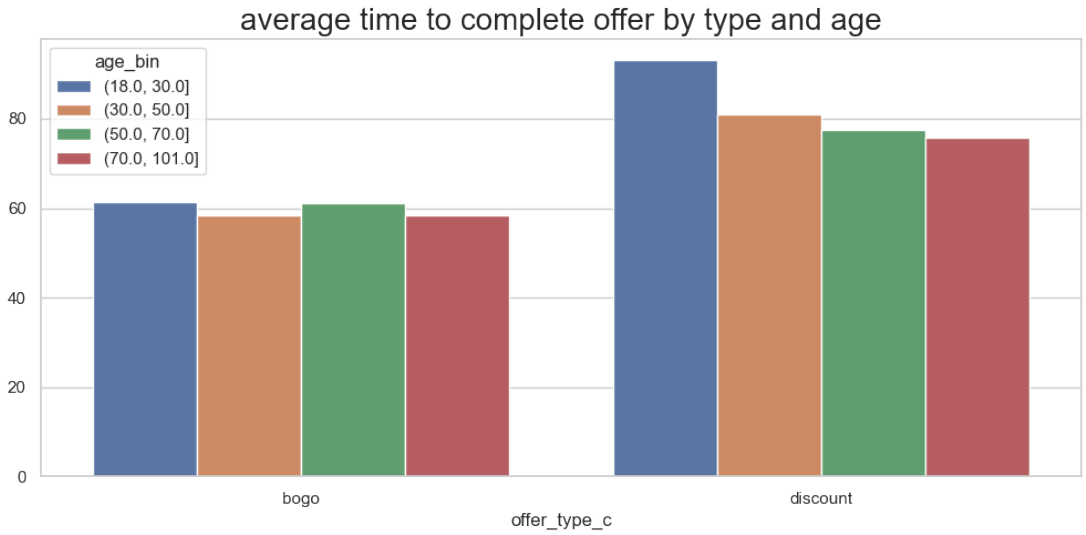


No matter which type of offers, females spent the most, and the unknown gender customers spent the least. And no matter which gender, customers spent most after viewing discount offers.



The amount of money spent is also positively correlated with income.

8. How long did it take a customer to complete an offer after receiving it?



It generally takes longer time for customers to complete discount offers than bogo, and younger people especially acted faster to complete the discount offers.

Algorithms and Techniques

Since this is a classification problem, I intend to use Naïve Bayes first because it works fast, and then use Decision Tree because the result produced by tree algorithms are easy to interpret, and if the accuracy score is still low, I will use ensemble algorithms such as Gradient Boosting Decision Tree. Next I will tune the hyperparameters to improve the scores.

Benchmark

I will use accuracy score of Naïve Bayes without hyperparameter tuning as the benchmark, because this algorithm is suitable for solving multiclass prediction problem, and it doesn't require much training data. However, Naïve Bayes assumes that all the attributes are mutually independent, which is impossible in real life, so the accuracy score might not be very high and is good to be a benchmark algorithm.

III. Methodology

Data Preprocessing

1. portfolio.json
    a) I split the "channels" column into 4 columns using pandas.explode(), and pandas.get_dummies(), so that if an offer is distributed by a specific channel (eg. web), then the value of that column is 1, otherwise 0.
    b) I change the measurement of the "duration" column from days to hours, because the time in transcript.json is in hours
    c) I delete the original "duration" column and name the new column as "duration_h"
    d) I simplify the "id" column as first character of values in "offer_type" column plus index (eg. b0, b1, d5, i9) and rename the column as "offer_id"
2. profile.json
    a) I replace genders with numbers because most algorithms don't work with string (M->1, Female->2, O->3, unknown->4)
    b) I treat age 118 as unknown
    c) I use pd.cut() to separate age into 4 bins: (18,30], (30,50], (50,70], (70,101] and name the column as "age_bin"
    d) I rename the "id" column as "member_id"
    e) I change the data type of "became_member_on" column to datetime, and then extract year, month, and calculate membership days and name the columns as year, month, and membership_days respectively
    f) I use pd.qcut() to separate income into 4 bins: (29999.99,49000], (49000,64000], (64000,80000], (80000,120000] and name the column as "income_bin"
3. transcript.json
    a) I rename the "person" column as member_id
    b) I separate the "value" column into offer_id, amount, rewards columns
4. I create the following features:
    a) number of offers received
    b) number of offers viewed
    c) number of offers completed
    d) number of offers completed without viewing
    e) number of offers viewed after completing
    f) percentage of offers viewed (view/receive)
    g) percentage of offers completed (complete/receive)

h) percentage of offers completed without viewing (complete without view/complete)
i) percentage of offers viewed after completing (view after complete/view)
j) hours between receive and view offers
k) hours between receive and complete offers
l) number of transactions made
m) number of transactions made after viewing offers
n) amount of money spent
o) amount of money spent after viewing offers

Implementation

I calculate the completion rate of each offer for each customer, and the one with highest rate is the final pick for that customer. If the customer hasn't completed any offer, then I will randomly choose one that hasn't been sent to him/her as the final pick.

1. My first model is naïve bayes (sklearn.naive_bayes.BeroulliNB) and get a precision score of 0.66, recall score of 0.5, and f1 score of 0.50 without hyperparameter tuning. The parameters are:
   a) 'alpha': 1.0
   b) 'binarize': 0.0
   c) 'class_prior': None
   d) 'fit_prior': True

2. My second model is decision tree (sklearn.tree.DecisionTreeClassifier) and get precision score of 0.77, recall score of 0.76 and f1 score of 0.77 without hyperparameter tuning. The parameters are:
   a) 'ccp_alpha': 0.0
   b) 'class_weight': None
   c) 'criterion': 'gini'
   d) 'max_depth': None
   e) 'max_features': None
   f) 'max_leaf_nodes': None
   g) 'min_impurity_decrease': 0.0
   h) 'min_impurity_split': None
   i) 'min_samples_leaf': 1
   j) 'min_samples_split': 2
   k) 'min_weight_fraction_leaf': 0.0
   l) 'random_state': 0
   m) 'splitter': 'best'

3. My third model is random forest (sklearn.ensemble.RandomForestClassifier) an d get precision score of 0.80, recall score of 0.80 and f1 score of 0.80 witho ut hyperparameter tuning. The parameters are:

a) 'bootstrap': True
b) 'ccp_alpha': 0.0
c) 'class_weight': None
d) 'criterion': 'gini'
e) 'max_depth': None
f) 'max_features': 'auto'
g) 'max_leaf_nodes': None
h) 'max_samples': None
i) 'min_impurity_decrease': 0.0
j) 'min_impurity_split': None
k) 'min_samples_leaf': 1
l) 'min_samples_split': 2
m) 'min_weight_fraction_leaf': 0.0
n) 'n_estimators': 100
o) 'n_jobs': None
p) 'oob_score': False
q) 'random_state': 7
r) 'verbose': 0
s) 'warm_start': False

4. My fourth model is catboost (catboost.CatBoostClassifier) and get f1 score of 0.824 without hyperparameter tuning. The parameters are:
   a) 'nan_mode': 'Min'
   b) 'eval_metric': 'MultiClass'
   c) 'iterations': 1000
   d) 'sampling_frequency': 'PerTree'
   e) 'leaf_estimation_method': 'Newton'
   f) 'grow_policy': 'SymmetricTree'
   g) 'penalties_coefficient': 1
   h) 'boosting_type': 'Plain'
   i) 'model_shrink_mode': 'Constant'
   j) 'feature_border_type': 'GreedyLogSum'
   k) 'bayesian_matrix_reg': 0.10000000149011612
   l) 'l2_leaf_reg': 3
   m) 'random_strength': 1
   n) 'rsm': 1
   o) 'boost_from_average': False
   p) 'model_size_reg': 0.5
   q) 'use_best_model': True
   r) 'class_names': ['b0', 'b1', 'b2', 'b3', 'd4', 'd5', 'd6', 'd7']
   s) 'random_seed': 0
   t) 'depth': 6
   u) 'posterior_sampling': False
   v) 'border_count': 254

w) 'bagging_temperature': 1
x) 'classes_count': 0
y) 'auto_class_weights': 'None'
z) 'sparse_features_conflict_fraction': 0
aa) 'custom_metric': ['TotalF1']
bb) 'leaf_estimation_backtracking': 'AnyImprovement'
cc) 'best_model_min_trees': 1
dd) 'model_shrink_rate': 0
ee) 'min_data_in_leaf': 1
ff) 'loss_function': 'MultiClass'
gg) 'learning_rate': 0.1134589985013008
hh) 'score_function': 'Cosine'
ii) 'task_type': 'CPU'
jj) 'leaf_estimation_iterations': 1
kk) 'bootstrap_type': 'Bayesian'
ll) 'max_leaves': 64

Refinement

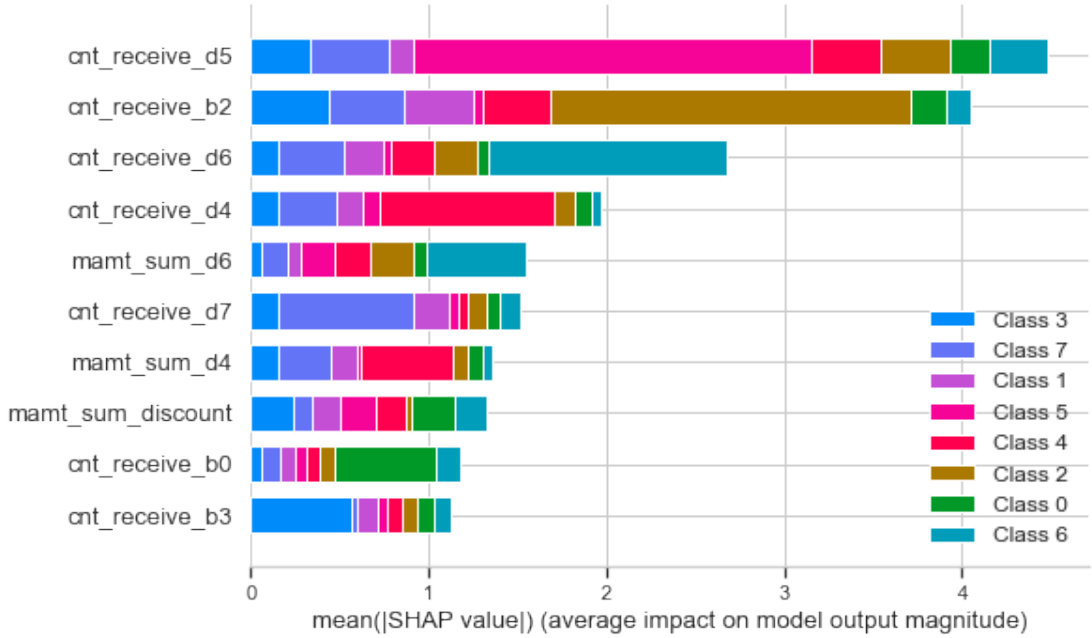
I tune the parameters on my fifth model using randomize_search() of catboost module. The parameters dictionary I set is {'learning_rate': [0.1,0.2,0.3], 'depth': [4,6,8,10], 'l2_leaf_reg': [1,3,5,7], 'custom_metric': ['TotalF1'], 'iterations': [1000,1500,2000]}.

The highest f1 score for test set is 0.83. The best parameters are {'depth': 4, 'l2 _leaf_reg': 3, 'iterations': 2000, 'learning_rate': 0.1}.

IV. Results

Free-Form Visualization

The most important features are number of offers received and amount of money spent after viewing offers.



Model Evaluation and Validation

I choose the last model built on catboost algorithm as the final model and the reasons are:

1. The score is highest among all models.
2. CatBoost is much faster than GBDT module in sklearn.
3. The randomized search function of CatBoost does cross validation, so this model is proved to be robust.
4. The most important features are , which is reasonable and easy to interpret.

Justification

The f1 score of the final model is 0.83, and the score of the benchmark using Naïve Bayes algorithm is 0.51. The final model is far better than benchmark.

V. Conclusion

Reflection

In this project I first clean the raw data, and then create a few new features, finally I build a model that recommends the best offer for each customer.

For the data cleaning part, I conclude the following:

1. I split the column values in list or dictionary type into several columns
2. I unify the time measurement into hours
3. I simplify offer_id
4. I use label encoder for gender column
5. I group income and age into bins

For the EDA part, I conclude the following:

1. Gender matters: although this dataset consists of mainly male customers, female customers are generally richer and have more spending power and higher offer completion rate
2. Channels matter: the offers that are distributed via social channel have higher view rate than those aren't
3. Age matters: the older the age, the higher the income, and the higher the offer completion rate, but also the higher the complete without view rate
4. Offer types matter: although customers complete bogo faster than discount offers, they spend more after viewing discount offers than bogo

For the model part, I conclude the following:

1. I create several features
2. I try several algorithms
3. I tune hyperparameters of CatBoost Algorithm

Improvement

Firstly, since this data is simplified with limited time duration and number of transactions, my analysis and the model I built may not be useful in the real world. Secondly, I set the rule by merely ranking the completion rate and pick the highest one for each customer, which may not be most appropriate. Finally, CatBoost algorithm supports sophisticated categorical features and NaN values, but I preprocessed those issues ahead (creating dummy variables for categorical features and filling NaN values with zero), which may not be good for CatBoost because it has better ways dealing with those issues.

Reference

[1] https://www.sciencedirect.com/science/article/abs/pii/S0306457309000259