Starbucks Capstone Challenge

Project Report

Sooyeon Won

09. February 2021

won.sooyeon@live.com

Content

1. Introduction
    - Current Business problem
    - Potential solutions and performance metrics

2. Data Exploration & Analysis
    - Understanding the current business situation through visualizations and exploration
    - Selection of proper algorithms and features to solve the problem

3. Algorithms Implementation
    - Implementation of algorithms
    - Computation of chosen metrics
    - Documentation the preprocessing, refinement and postprocessing

4. Performance Results
    - Collection of model performance results
    - Visualization of significant quantities
    - Validation and justification of the findings

5. Conclusion
    - Evaluation of the implemented solutions

6. References

1. Introduction

This project is derived from the field of Customer Relationship Management (CRM) in Starbucks. One main concern for CRM is to interact with not only current customers, but also previous and potential customers, so that the company can effectively maintain its business relationship with all customers.

In this business situation, Starbucks sent out offers to customers through various channels as a marketing promotion. Offers onsist of three types. It is either just an advertisement for a certain beverage (**informational offer**)[1], or a coupon-type offer such as a '**Discount**'[2] or 'buy 1, get 1 (**BOGO**)'[3]. Each offer is valid for certain number of days and requires different level of difficulties. In addition, the rewards that customers received after offer-completion are different from each other. Not all of customers obtain the offers, and different customers receive various the type of the offers.

In this analysis, I focused on the fact that whether the issued offers are desirably used by customers. To do so, I firstly define "desirably-used" offers. As you can the table in the project proposal, I defined Case 1 or Case 2 as desirably-used offer cases in this analysis.

A Possible Process of Using Offers

Case	Received	Viewed	Transaction	Completed	Reward	Desirable
1	√	√	√	√	√	Yes
2	√	√	√			Yes
3	√	√				No
4	√		√	√		No
5	√		√			No
6	√					No
7			√			No

Unlike Cases 3-7, customers in Case 1 and Case 2 viewed their received offers and purchased products during the offer valid period. Although customers used their offer desirably, not all customers successfully obtained the corresponding rewards, if their spending could not reach to the required difficulty (Case 2). Note that an appropriate way of using offers means when each stage („Received", „Viewed", „Transaction") occurs in consecutive order and within the terms of validity.

[1] **Informational offer:** Its main purpose is providing an (updated) information about products to customers. There is no reward, neither a required spending that a customer is expected to spend.

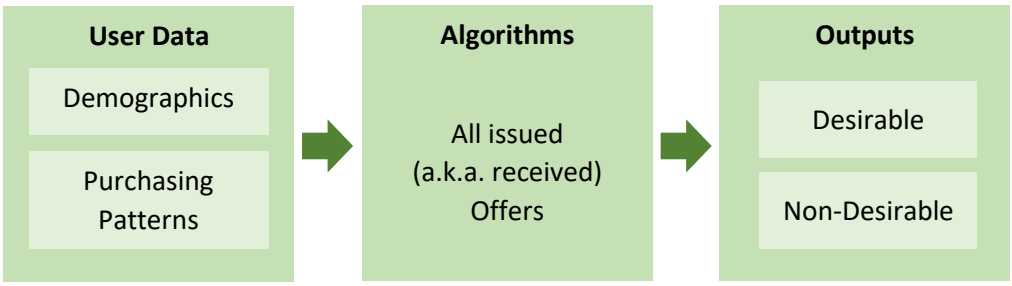
[2] **Discount offer:** Customers get monetary rewards which is equivalent to a certain proportion of the amount of spending.

[3] **Buy 1, Get 1 (BOGO):** Customers should spend certain amounts (threshold) so that they can receive monetary rewards, which is equal amount of the cut-off amount.

The prior stages are not necessarily required to complete the offers. For example, a customer who did not view the offer, but the customer could complete to use the offer then get the corresponding rewards, though. Since such customers' purchasing patterns are not influenced by offers, the company can optimize its marketing events by sending offers other customers whose purchasing patterns is highly dependent on the received offers.

In this analysis, I identified the "desirably-used" offers, among all issued offers, by assigning each event the date when the individual events occurred. This feature engineering process is detailed in the following section. Then I found out whether an offer is desirably-used, is related to the demographic factors of users and individual purchasing patterns.

The main task I encounter is to classify whether the offers are desirably-used, based on the demographic features and purchasing patterns of users.



I conducted the analysis using frequently used Classification models with Random Forest- , AdaBoost-, and Gradient Boosting Classifier. Additionally, XG Boost -, LightGBM and CatBoost Classification models will be applied as complimentary supervised learning approaches. The target variable in this analysis is "Desirability", indicating whether offers are desirably used by customers. In addition, customer demographic features and purchasing characteristics represented by RFM scores are used as input variables. This classification analysis is conducted for each type of offer separately.

Although the models which show less performance than the model with best performance could be considered as benchmark models, I designated a "logistic regression" as a benchmark model for this project to compare my solution objectively.

The performance of individual models can be measured according to various evaluation metrics. For this analysis, I computed Accuracy, Precision, Recall, and F1-Score. "Accuracy" however is not appropriate to be used as an evaluation metric for predictive models when classifying in predictive analytics. As "Accuracy Paradox" indicates, a simple model might be able to achieve a high score of accuracy but be too crude to be useful. On the other hand, F1-Score is computed with the prediction and recall of the test.

$$F1\ Score = 2 * \frac{Prediction * Recall}{Prediction + Recall}$$

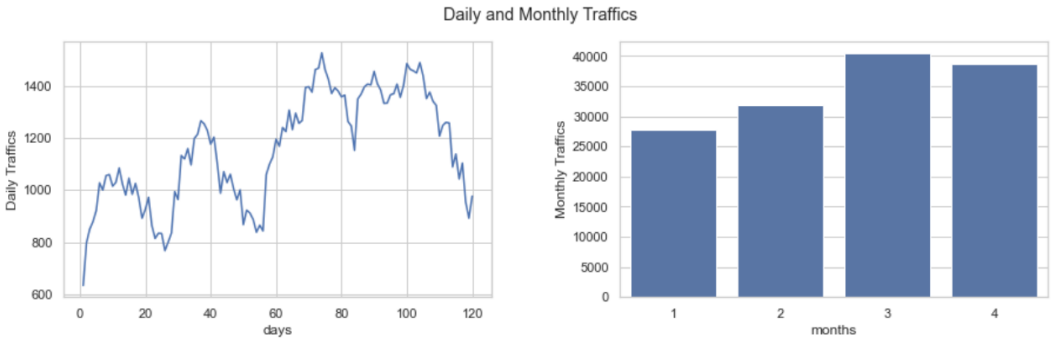
Since the F1-Score indicates a weighted average of the prediction and recall values, it takes false positive as well as false negatives into account. F1-Score reaches its maximum (best) value at 1 and its minimum (worst) value at 0. F1-Score is often useful in comparison to accuracy, especially if each class is not evenly distributed. Thus, I will select the best performance model according to its F1-Score. Additionally, I count the time spending of each model. So if the f1-score of each model is equivalent, the model with less time spending would be optimal for the analysis.

2. Data Exploration & Analysis

To begin with, I analyzed the business situation of Starbucks during the test period, based on the transcript datasets.

The Change of Traffics during the Test Period

```
Out[26]: Text(0.5, 0.98, 'Daily and Monthly Traffics')
```



The Sales Trend Amount during the Test Period

```
Out[27]: Text(0.5, 0.98, 'Daily and Monthly Sales Amount')
```

I found out the fact that the number of traffics varied in each month. It increased until the third month. In the fourth month, however, the total number of traffics slightly decreased. Note that the 4th month traffics are still almost as large as that in the third month.

The change of average sales amounts follows almost identical patterns with the change of traffics. We could interpret this situation that more traffics are likely to bring better sales performances. Interestingly, although the number of traffics and average monthly sales amount showed similar patterns across the months, the average spending per each purchase is not significantly different across the months.

Then I focused the offers which sent to customers and the potential relations between customer demographic factors and the sent offers. The table below (Out [31]) provides descriptive statistics. There are 17,000 customers in the customer dataset. All of the customers involved with purchasing during the test period, but not all of them received the offers. 6 customers (= 17,000 - 16,994) purchased products without received any types of offers.

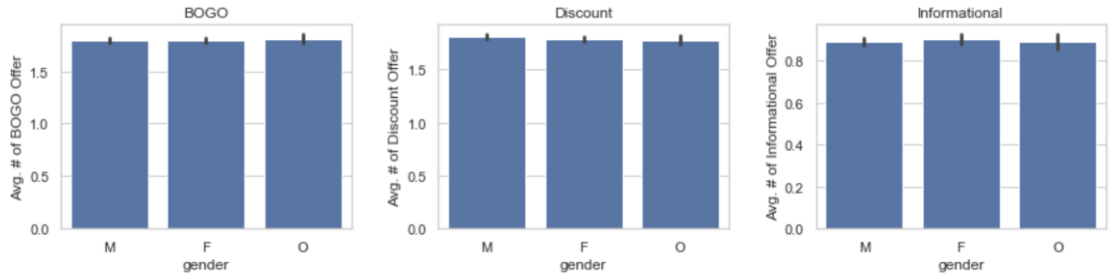
From the table we can expected that 'bogo' and 'discount' types offers are issued twice as much as 'informational' type offers[4]. Customers who purchased products during the test period obtained almost two offers of 'bogo' and 'discount' types and one offer of 'informational' type on average. The average value of customer age is 54.5, and that of reported customer income is 65,226.90 USD. When we compare the mean and median value of the feature: "days as member", we can expect that the data points are skewed to the right. Its mean value is larger than its median value.

Out[31]:

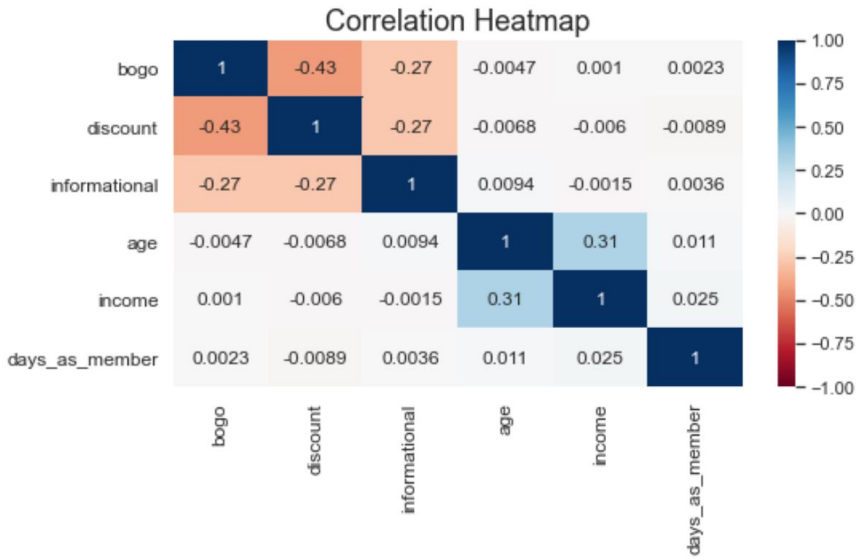
	bogo	discount	informational	age	income	days_as_member
count	16994.00	16994.00	16994.00	16994.00	16994.00	16994.00
mean	1.79	1.80	0.90	54.47	65226.90	522.44
std	1.12	1.13	0.87	16.23	20174.70	411.27
min	0.00	0.00	0.00	18.00	30000.00	5.00
25%	1.00	1.00	0.00	45.00	51000.00	213.00
50%	2.00	2.00	1.00	55.00	64000.00	363.00
75%	3.00	3.00	1.00	65.00	76000.00	796.00
max	6.00	6.00	5.00	101.00	120000.00	1828.00

Then I explored whether the number of offers customers received is related to customer demographic factors. As you can see in the output [32], there are no significant differences of average received offers on gender.

Out[32]: Text(0.5, 1.0, 'Informational')

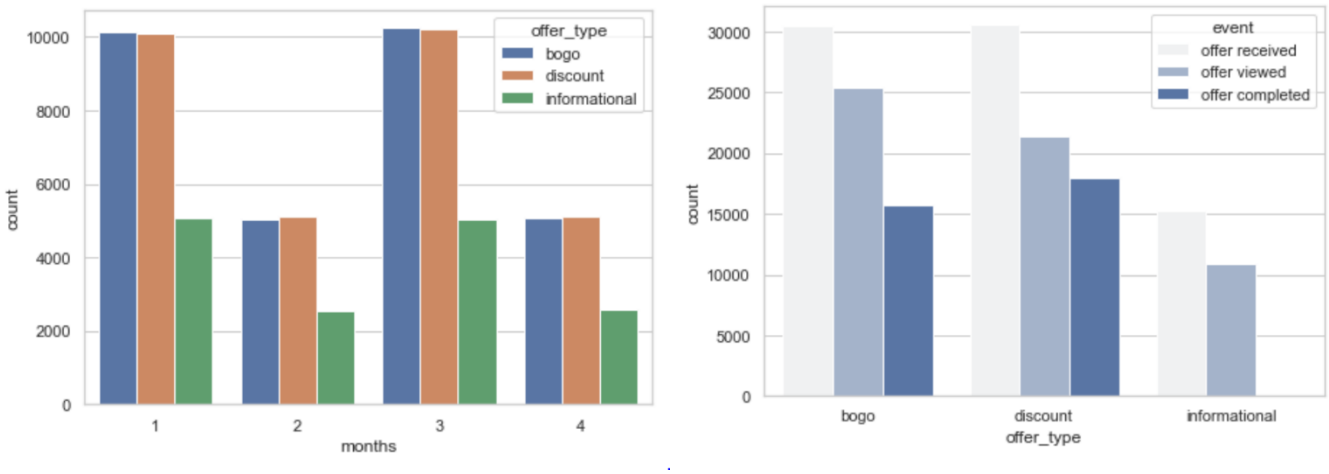


[4] According to the output [36] in the Jupyter notebook, 30543 'BOGO' offers, 30499 'Discount' offers, and 15235 'informational' offers are sent.



Furthermore, it turns out that the total number of offers and the number of each type of offers are not correlated with customer's ages, incomes, the number of days as a Starbucks member. In other words, Starbucks randomly sent out its offers during the test period. Next, I investigated the data by focusing on the offers.

```
Out[37]: <AxesSubplot:xlabel='offer_type', ylabel='count'>
```



The number of issued offers is not equal. 'BOGO' and 'Discount' types of offers are almost evenly distributed across the months. On the other hand, "Informational" type of offer is issued only the half of them over the months. Furthermore, the offers are twice as much as distributed in the first and third months in comparison to the second and fourth months.

Not all the issued offers are viewed. Only 75,68% of offers are noticed by customers. Almost half of issued 'BOGO' and 'Discount' offers are completed only. Note that the event whether the informational type of data is completed, is not collected. Like other offers, if a customer viewed the informational data and had at least one purchase history during the offer valid period, it is defined as a desirably used offer. Finally from the output [37], we can recognize that customers viewed more 'BOGO' offers than 'Discount' offers. On the other hand, more 'Discount' type of offers are completed.

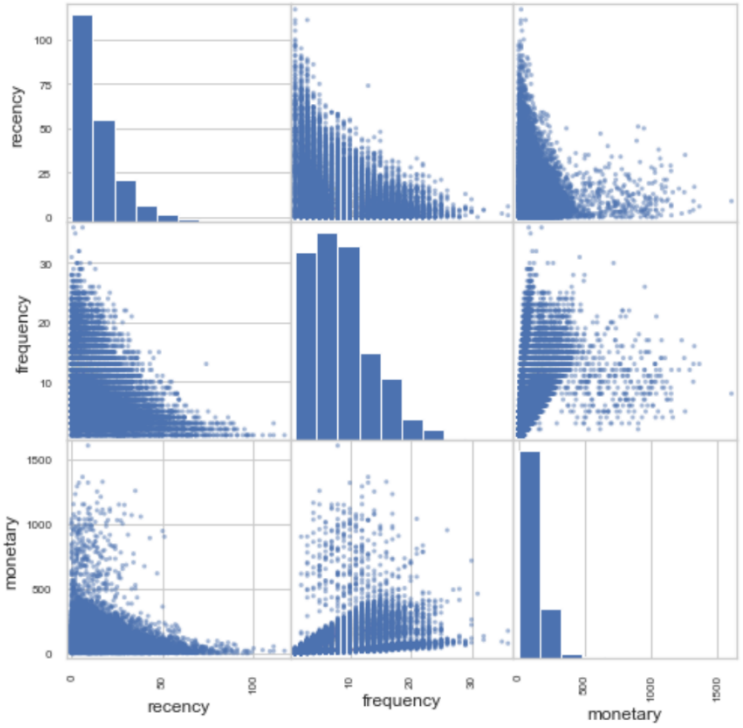
I explored purchasing patterns of customers according to RFM Analysis. RFM analysis evaluates customer purchasing patterns with 3 elements:

- How recently does a customer purchase a product at Starbucks (Recency)?
- How often does the customer purchase a product during the period (Frequency)?
- How much does the customer spend at Starbucks (Monetary Value)?

Out [42]:

	recency	frequency	monetary
count	16578.00	16578.00	16578.00
mean	14.24	8.38	107.10
std	13.79	5.01	126.39
min	0.00	1.00	0.05
25%	4.00	5.00	23.68
50%	10.00	7.00	72.41
75%	20.00	11.00	150.94
max	117.00	36.00	1608.69

According to descriptive statistics in the output [42], the latest purchase of customers is ca. 14 days ago on average. During the test period, customers bought products 8 times on average. In addition, the mean value of 'monetary' implies that their average spending during the test period is more than 100 dollars.



The scatter matrix shows notable purchase patterns of customers. First, the histograms of recency, frequency and monetary values (Plots on the main diagonal of the scatter matrix) are all skewed to the right. There are more customers who recently visit our stores than the customers who purchased beverages long time ago. Also there are a few customers who purchased products extremely often, spent extremely a lot during the test period, but most customers bought products less than 11 times and spent less than 151 dollars.

Recency is negatively correlated with frequency and monetary values, vice versa. On the other hand, frequency is positively related with monetary values. This indicates that the more recently customers purchase Starbucks products (lower recency value), the more frequently the customers buy our products, and the more they spend at Starbucks. Intuitively, the more frequently customers purchase, the more they spend at our store. In the Jupyter Notebook, the relations are visualized using a heatmap.

During the test period, customers are randomly received different offer(s). Some customers followed desirable processes to be rewarded. However, not all of them. Some other customers get rewarded without noticing that they received the offers. In this section, I will explain how the "desirably-used" offers are identified from the given dataset.

First, I restructured the transactions dataframe, using pandas ‚get_dummies'. Then I merged the dataframe with the offer information to compute the last valid day of the corresponding offers. The column 'max_offer_day' in output [49] indicates that all events of a certain offer should be completed before the last valid day to be defined as a desirably used offer. At the end, I grouped the transaction dataframe by 'customer_id', 'days', and 'offer_id'.[5]

Out[49]:

	customer_id	days	offer_id	max_offer_day	offer_received	offer_viewed	transaction	offer_completed
0	0009655768c64bdeb2e877511632db8f	29	5a8bc65990b245e5a138643cd4eb9837	32.00	29	0	0	0
1	0009655768c64bdeb2e877511632db8f	33	5a8bc65990b245e5a138643cd4eb9837	32.00	0	33	0	0
2	0009655768c64bdeb2e877511632db8f	39	5a8bc65990b245e5a138643cd4eb9837	32.00	0	0	39	0
3	0009655768c64bdeb2e877511632db8f	57	3f207df678b143eea3cee63160fa8bed	61.00	57	0	0	0
4	0009655768c64bdeb2e877511632db8f	63	3f207df678b143eea3cee63160fa8bed	61.00	0	63	0	0
5	0009655768c64bdeb2e877511632db8f	69	f19421c1d4aa40978ebb69ca19b0e20d	74.00	69	0	0	0
6	0009655768c64bdeb2e877511632db8f	70	f19421c1d4aa40978ebb69ca19b0e20d	74.00	0	0	70	0
7	0009655768c64bdeb2e877511632db8f	70	f19421c1d4aa40978ebb69ca19b0e20d	74.00	0	0	0	70
8	0009655768c64bdeb2e877511632db8f	77	f19421c1d4aa40978ebb69ca19b0e20d	74.00	0	77	0	0
9	0009655768c64bdeb2e877511632db8f	85	fafdcd668e3743c1bb461111dcafc2a4	95.00	85	0	0	0

For example, the customer with id (0009655768c64bdeb2e877511632db8f) in the table output [49], received an offer (offer ID: 5a8bc65990b245e5a138643cd4eb9837) on the 29th testing day and the last valid day of the offer is the 32nd testing day, since the offer is valid for 4 days. The customer viewed the offer on 33rd testing day and purchased product on the 39th testing day. Therefore, we can conclude that the customer didn't use the offer desirably.

This customer received a new offer on the 57th testing day, but viewed it after the last valid day. The customer received the third offer on 69th testing day, and then purchased products and completed the offers on the next day. However the user viewed the offer after the last valid day of the offer. As mentioned earlier, although the customers completed the offer and rewarded, it is not the desirable way of using offers that the company intended. In this way you can understand the whole dataframe (trans_per_event – in [50]).

As we previously defined, if the events: "offer_viewed" and "transaction" of each offers occurred prior to the last valid day of the corresponding offer, we can say that the offer is desirably used. So I filtered out such cases and ended up with the dataframe with the name of event_name_df (Out [53]) This dataframe listed the desirably used offers per customer, per day.

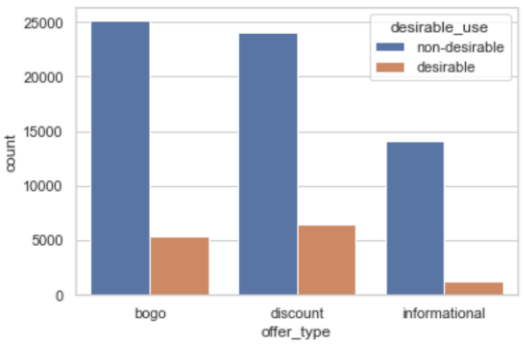
Starbucks randomly sent out 76,277 offers to their customers. Among them, 6468 'discount' offers, 5370 'bogo' offers and 1187 informational offers are desirably used. 9042 unique customers used the offers in a desirable way. Therefore I assigned those offers as 'desirable' and the rest as 'non-desirable'. This is used as a target variable in the following classification models.

[5] This feature engineering process is more detailed in the Jupyter Notebooks.

3. Algorithms Implementation[6]

Imbalanced data & Feature Selection

```
Out[8]: <AxesSubplot:xlabel='offer_type', ylabel='count'>
```




```
BOGO Offer:
non-desirable    25129
desirable         5370
Name: desirable_use, dtype: int64

Discount Offer:
non-desirable    24075
desirable         6468
Name: desirable_use, dtype: int64

Informational Offer:
non-desirable    14048
desirable         1187
Name: desirable_use, dtype: int64
```

As you can see in the above bar plot, I had come across imbalanced class distribution. Almost 21.5%, 27%, 8.5% of the provided BOGO offers, of the issued Discount offers, of all Informational offers are desirably used, respectively. In this circumstance, the predictive model developed using conventional supervised machine learning algorithms could be biased and inaccurate. This is because the algorithms do not consider the balance of classes, since in general they are designed to improve accuracy by reducing the error. Therefore, I alleviated the imbalance by applying Synthetic Minority Over-sampling Technique (SMOTE), after splitting the dataset into training and testing sets.

Feature selection is mainly focused on removing non-informative or redundant input variables from the model. When irrelevant predictors are included to the target variable, the performance of some models can be degraded. In addition, some predictive models with many variables can slow the development and training of models and might need a huge amount of system memory.

In this analysis, I selected subsets of features based on their relationship with the target. We have a classification predictive modelling problem with numerical and categorical input variables. Thus, two statistical measures are involved that may be used for filter-based feature selection with different input and output variable data types.

- Numerical Inputs: ANOVA correlation coefficient
- Categorical Inputs: Chi-Squared test (contingency tables)

Data Splitting & Balancing Strategy

Then I split the data into random train and test subsets. 20% of data points are assigned to the test dataset. As expected, training dataset is also imbalanced.

Imbalanced Training Set

```
          BOGO Training Data:  Counter({'non-desirable': 20108, 'desirable': 4291})
      Discount Training Data:  Counter({'non-desirable': 19243, 'desirable': 5191})
Informational Training Data:  Counter({'non-desirable': 11244, 'desirable': 944})
```

[6] Please refer to the second part of Jupyter Notebook from here.

Balanced Training Set - Out [18]

BOGO offer	Class=non-desirable, n=20108 (58.825%) Class=desirable, n=14075 (41.175%)
Discount offer	Class=non-desirable, n=19243 (58.824%) Class=desirable, n=13470 (41.176%)
Informational offer	Class=desirable, n=7870 (41.174%) Class=non-desirable, n=11244 (58.826%)

Thus, the minority class is oversampled according to SMOTE methods. This technique eases the overfitting problem driven by random oversampling as synthetic examples are generated instead of duplicating samples. In addition, we can expect no loss of useful information.

Model Comparisons

For this analysis, I set up several models, then compared individual performances, represented by f1-score and time duration. I set a logistic regression model as a benchmark.

Sklearn Ensemble Methods		**Benchmark**
Random Forest	XG Boost	Logistic Regression
AdaBoost	Light GBM	
Gradient Boosting	CatBoost	

Random Forest fits several decision tree classifiers on different sub-samples of the dataset and makes use of averaging to improve the predictive accuracy and to avoid severe over-fitting.

AdaBoost (Adaptive Boosting) is often used with many learning algorithms to improve performance. AdaBoost classifier starts by training a classifier on the initial dataset. Then the output of other learning algorithms on the same dataset (a. k. a. weak learners) is aggregated with a weighted sum. The sum represents the final output of the classifier.

Gradient Boosting generate a prediction model in the form of an ensemble of weak prediction models, usually decision trees. Like other boosting methods, it builds an additive model in a stage-wise way. This allows to optimize an arbitrary differentiable loss function.

XGBoost (e**X**treme **G**radient Boosting) is a recently dominated algorithm for structured or tabular data. XGBoost is introduced for execution speed and model performance. It implemvaluents the gradient boosting decision tree algorithm.

LightGBM (Light Gradient Boosting Machine) is also based on decision tree algorithms and often used for ranking, classification and so on. It has most XGBoost's advantages, including sparse optimization, parallel training, multiple loss functions, regularization, bagging, and early stopping. The difference between XGBoost and LightGBM how the algorithms construct the trees. Instead of growing trees leaf-wise, LightGBM selects the leaf which is believed to generate the largest decrease in loss. In addition, LightGBM uses a highly optimized histogram-based decision tree learning algorithm, instead of sorted-based decision tree learning algorithm, which is widely-used in XGBoost or other algorithms.

CatBoost is a framework for gradient boosting on decision trees. It is primarily used for ranking, and forecasting such as recommendation systems.

Finally, as a benchmark model, I chose a logistic regression model. Logistic regression is most used when the data in question has binary output, so when it belongs to one class or another, or in this analysis is either a 'desirable' or 'non-desirable' for each type of offers.

I analysed each offer type datasets based on all the classification algorithms, and then compared individual model performance according to the f1-score and time duration.

4. Performance Results

BOGO

Out[24]:

	accuracy	prediction	recall	f1_score	run_time
LogisticRegression	0.628525	0.776846	0.628525	0.694859	0.286197
RandomForestClassifier	0.723443	0.732937	0.723443	0.728159	4.289023
AdaBoostClassifier	0.719180	0.766604	0.719180	0.742135	21.096846
GradientBoostingClassifier	0.805902	0.738656	0.805902	0.770815	23.922598
XGBClassifier	0.800984	0.741303	0.800984	0.769989	0.595745
LGBMClassifier	0.810820	0.740694	0.810820	0.774172	0.419876
CatBoostClassifier	0.727869	0.766007	0.727869	0.746451	1.850489

Discount

Out[25]:

	accuracy	prediction	recall	f1_score	run_time
LogisticRegression	0.623343	0.743510	0.623343	0.678144	0.323135
RandomForestClassifier	0.686201	0.692868	0.686201	0.689518	4.415531
AdaBoostClassifier	0.706499	0.732201	0.706499	0.719120	19.978014
GradientBoostingClassifier	0.772303	0.706182	0.772303	0.737764	23.486044
XGBClassifier	0.764119	0.700871	0.764119	0.731129	0.455747
LGBMClassifier	0.779342	0.702285	0.779342	0.738810	0.257313
CatBoostClassifier	0.713537	0.734273	0.713537	0.723757	4.544156

Informational

Out[26]:

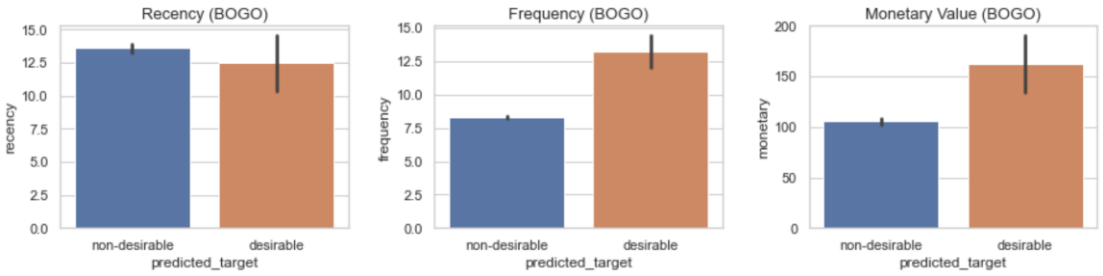
	accuracy	prediction	recall	f1_score	run_time
LogisticRegression	0.638005	0.883252	0.638005	0.740860	0.184477
RandomForestClassifier	0.861503	0.862537	0.861503	0.862020	2.764857
AdaBoostClassifier	0.782737	0.868796	0.782737	0.823524	11.787836
GradientBoostingClassifier	0.905481	0.859478	0.905481	0.881880	15.608478
XGBClassifier	0.898589	0.854029	0.898589	0.875743	0.457454
LGBMClassifier	0.911388	0.856343	0.911388	0.883008	0.403919
CatBoostClassifier	0.790286	0.868964	0.790286	0.827759	8.431468

LGBMClassifier

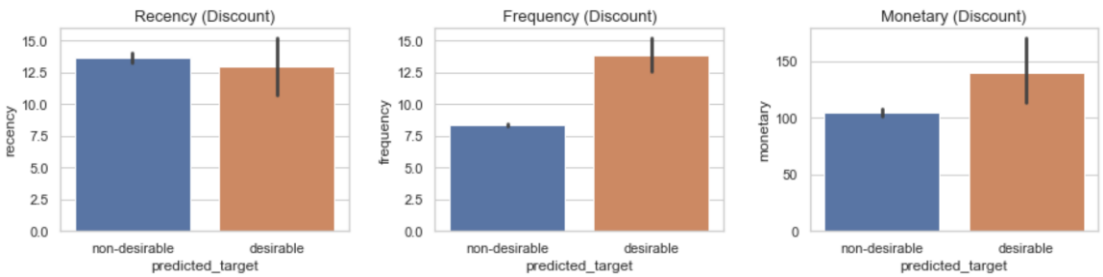
For all type of offers, “LGBMClassifier" shows the outstanding model performance in terms of f1-score. Note that Gradient Boosting Classifier showed the second-best performance. However, the LGBMClassifier shows slightly better performance within much shorter period of time.

I additionally compared RFM scores between customers who used offers in a desirable way and customers who did not. I found out that the users who desirably used offers share something common regardless of offer types. The average recency score of the two groups is not significantly different. However, the desirable users a lot more frequently purchased products and much more spent at Starbucks in comparison to the customer group who didn’t used the received offers properly.

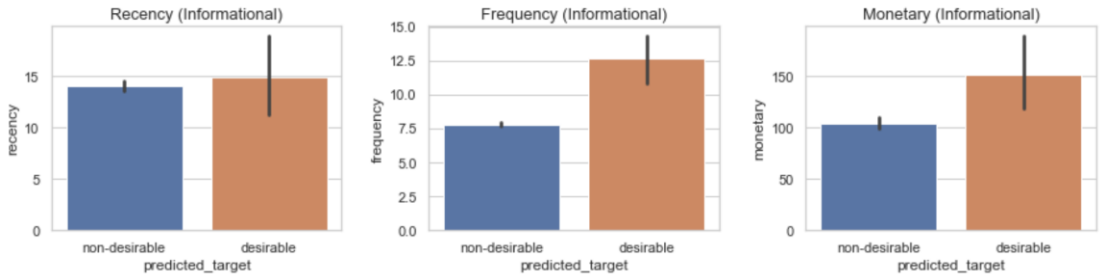
Out[34]: Text(0.5, 1.0, 'Monetary Value (BOGO)')



Out[32]: Text(0.5, 1.0, 'Monetary (Discount)')



```
Out[33]: Text(0.5, 1.0, 'Monetary (Informational)')
```



5. Conclusion

In this analysis, I mainly analysed how customer profiles and their purchasing patterns affect whether the customer use their received offers desirably. To begin with, by exploring current business situations, I found out that providing offers to customers brings increases on total sales amounts. Note that the average spending per each purchase is remarkably similar across the months.

All customers in the profile dataset purchased products at Starbucks, although not all of them received offers. Starbucks sent out all offers randomly. There are no significant relationships between the number of each type offers and customer demographic factors. Note that 'BOGO' and 'Discount' types of offers are sent out almost twice as many as "Informational" type of offer. Not all issued offers are viewed by users and only half of issued 'BOGO' and 'discount' offers are completed. In addition, I computed individual Recency, Frequency, Monetary scores which represent customer purchasing patterns.

I defined the desirably used offers by both Case 1 and Case 2. Based on the definition, I identified all offer usages into 2 groups: 'desirable', 'non-desirable' per each offer type. It turns out that all three datasets are highly imbalanced. Therefore, I alleviated the imbalance by applying Synthetic Minority Oversampling Technique (SMOTE). Then I trained each dataset with various classification models.

For all type of offers, "LGBMClassifier" showed the optimal model performances. It achieved 0.7742, 0.7388, 0.8830 of f1-score for bogo, discount, informational datasets, respectively, within the shorter period of time. The f1-score is considerably larger than that of the benchmark model. Also the time duration is much shorter than that of the benchmark model. The model with LGBMClassifier is more efficient than the benchmark model.

6. References

https://scikit-learn.org/stable/modules/classes.html#module-sklearn.ensemble

https://en.wikipedia.org/wiki/AdaBoost

https://en.wikipedia.org/wiki/Gradient_boosting

https://de.wikipedia.org/wiki/Random_Forest

https://towardsdatascience.com/catboost-vs-light-gbm-vs-xgboost-5f93620723db

https://machinelearningmastery.com/gentle-introduction-xgboost-applied-machine-learning/

https://en.wikipedia.org/wiki/LightGBM

https://affine.medium.com/catboost-a-new-game-of-machine-learning-72a7dcea0ac4

https://kambria.io/blog/logistic-regression-for-machine-learning/