Starbucks Capstone Project Report

Youssef Medhat | 11 May 2023

Udacity – AWS Machine Learning Engineer Nanodegree

Problem Definition:

- **Project Overview**
  - **Domain Background:**

    Starbucks is an enthusiastic retailer of coffee and other beverages with its corporate headquarters in Seattle, Washington. Registered users of their mobile application can order coffee for pickup while on the go, pay in-store using the app, and get rewards points. Also, this app provides these users with promos for extra points that could simply be a drink marketing or it could be a real deal, like a discount or a BOGO (buy one, get one free) deal. The goal of this project is to identify the customers who are most likely to respond to an offer by personalizing promotional offers for them based on their answers to prior offers.

  - **Datasets and Inputs:**

    The simulated data in this data set closely resembles consumer activity on the Starbucks Rewards mobile app. Starbucks delivers offers to customers who use its mobile app every few days. The data set is provided in form of three JSON files:

    - portfolio.json - containing offer ids and metadata offer (duration, type, etc.)

    - profile.json - demographic data for each customer.

    - transcript.json - records for transactions, offers received, offers viewed, and offers completed.

```
In [3]: portfolio.head()
```

Out[3]:

	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

```
In [9]: profile.head()
```

Out[9]:

	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

```
In [13]: transcript.head()
```

Out[13]:

	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

▪ Problem Statement

My objective is to evaluate which kind of offer to deliver to each user based on their response to the offers that have already been made to them. The objective is to utilize the data set provided by Starbucks, which was collected over 30 days, to address the fact that not all users receive the same offer. Also, I'll create a machine-learning model that predicts how a customer will react to an offer.

▪ Evaluation Metrics

To evaluate the effectiveness of the strategy and identify which model produces the best outcomes, I will use the F1 score as the model metric. It can be understood as the weighted average of recall and precision. The traditional or balanced F-score (F1 score), which has a greatest value of 1 and a worst value of 0, is the harmonic mean of precision and recall.

Problem Analysis:

▪ Data Exploration

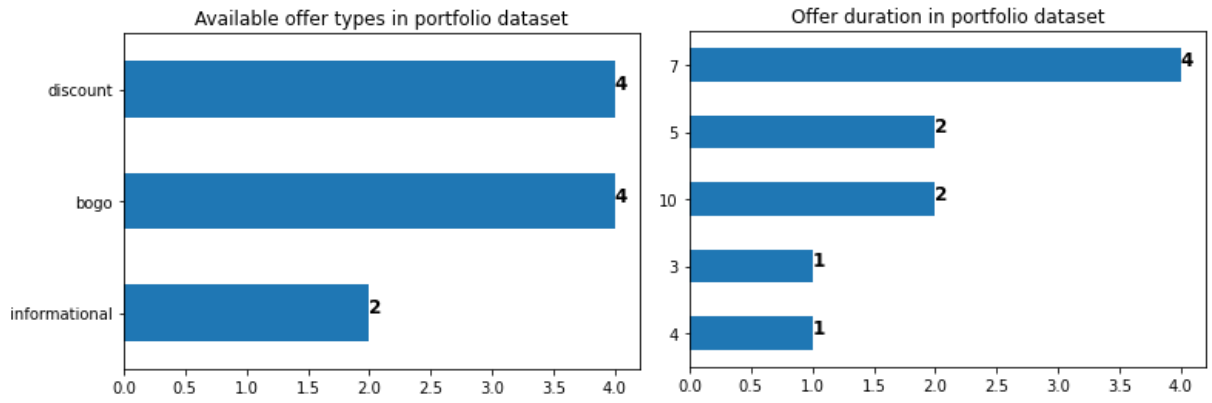
Here three dataset parts were explored individually to gain the major information. These parts were – portfolio.json, profile.json, and transcript.json.

Let's see what were the important insights for me to explore from the data and preferably build the ML models to solve the problem.

Portfolio.json:

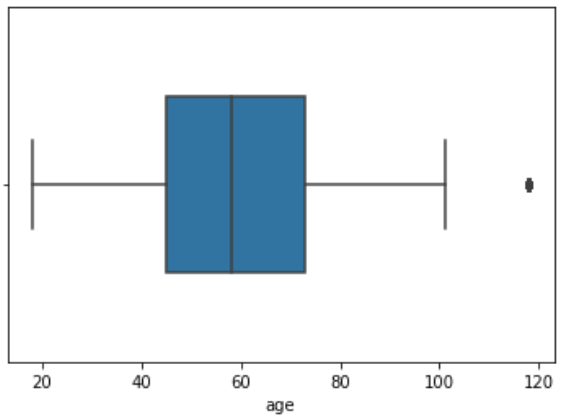
As it's just the exploration part, there was no cleaning done here. Just the data was explored to find out the offer types and duration. This dataset contains 10 rows and 6 columns, and there were 3 integer attributes and 3 object attributes.

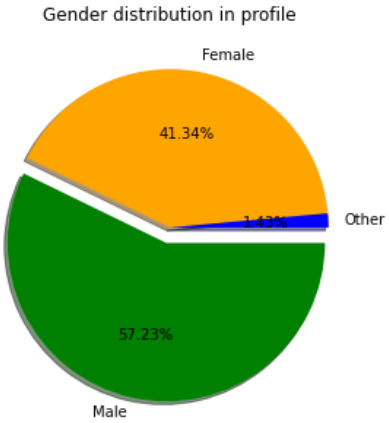
You can see the types of offers present in the dataset through the below bar graph representation.



Profile.json:

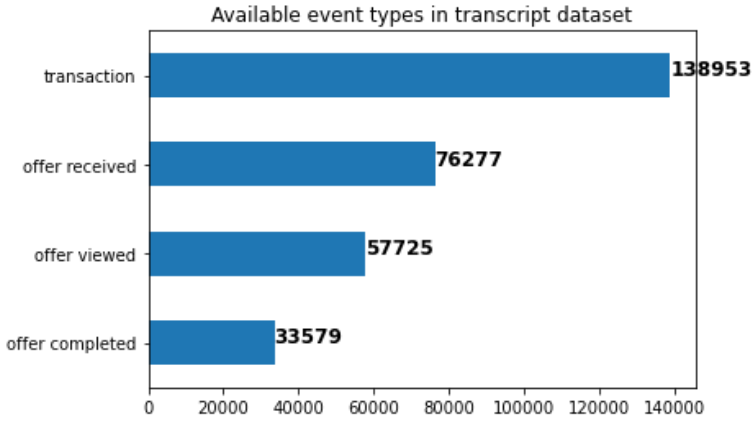
In this dataset, there were 17000 rows and 5 columns having datatypes as float64(1), int64(2), and object(2). According to the profile data frame and checking null values, it was explored that 2175 values of gender & income both were missing where age is 118. Therefore, all the missing age values were encoded as 118. Also, an outlier was explored where people with an age greater than 80 don't use the app much or they may not drink many beverages. Gender distribution in the profile dataset was also explored.





Transcript.json:

In this dataset, there were 306534 rows and 4 columns having datatypes as int64(1), and object(3). There was not much enough to explore in this part. Only the types of events that were present were explored which were measured to train and test the ML models.



▪ Data Cleaning

Here all the unnecessary data, renaming of columns for ease of training, or encoding of the categorical values was implemented. In the portfolio data frame, only the column names were renamed as there was nothing much to clean. In the profile dataset, the following tasks were implemented to clean the data – renaming some column names, imputing missing age & income values with mean and missing gender values with mode, removing outliers from the dataset i.e. people with an age above 80, and classifying ages for EDA into categories: Under 20, 20-45, 46-60, 61-80. In the transcript dataset, the following tasks were implemented– renaming some

column names, and expanding the keys of the 'value' column into new columns. Finally, all these three cleaned dataset parts were combined into one whole dataset for performing the EDA and training and testing the ML models. Here's a glimpse of the cleaned profile data –

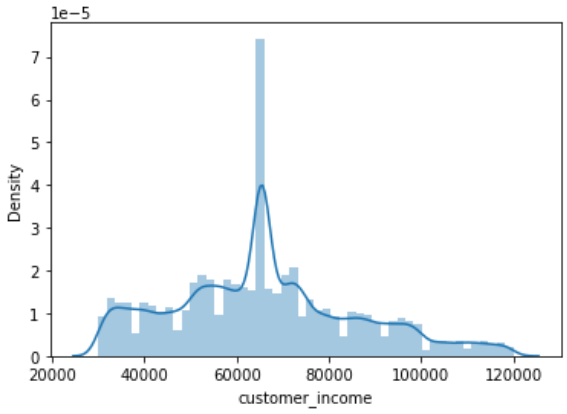
```
In [36]: cleaned_profile.head()
```

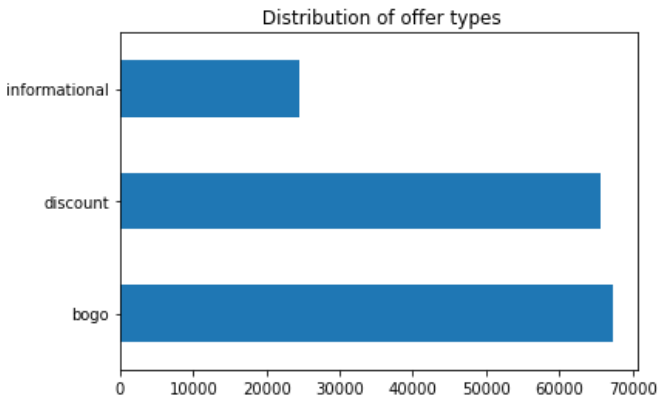
Out[36]:

	gender	customer_id	became_member_on	customer_income	Age_group
0	M	68be06ca386d4c31939f3a4f0e3dd783	20170212	65404.991568	46-60
1	F	0610b486422d4921ae7d2bf64640c50b	20170715	112000.000000	46-60
2	M	38fe809add3b4fcf9315a9694bb96ff5	20180712	65404.991568	46-60
3	F	78afa995795e4d85b5d9ceeca43f5fef	20170509	100000.000000	61-80
4	M	a03223e636434f42ac4c3df47e8bac43	20170804	65404.991568	46-60

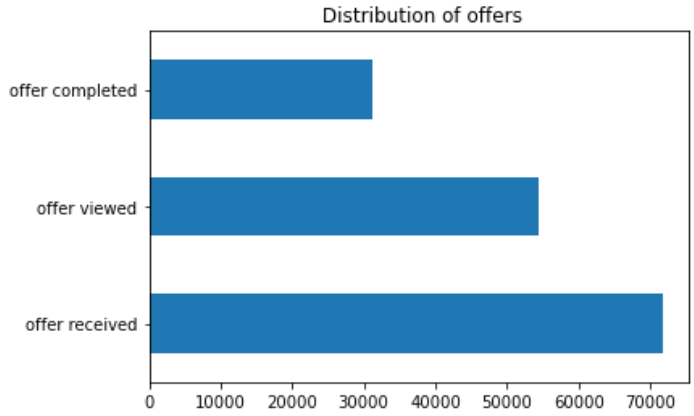
▪ Performing EDA

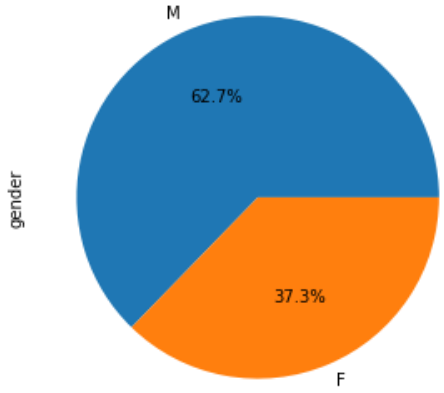
The average customer income that used Starbucks App lies between the range of 30000-120000. Here, the most used offer by the customers was explored which were the BOGO and discount offers that had nearly the same distributions. Customers within the age group of 46-60 used the Starbucks application frequently while those between the age group of 61-80 ranked second in using the App frequently.



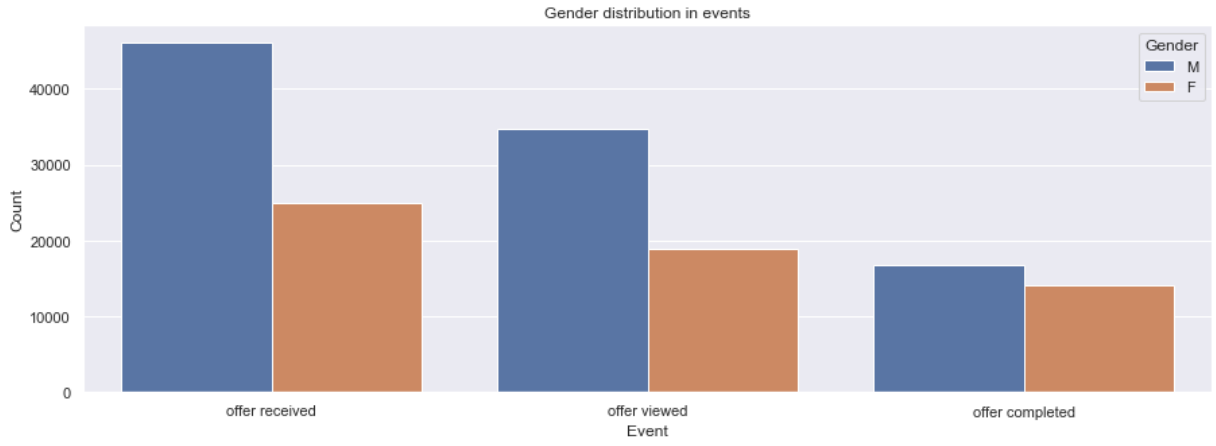


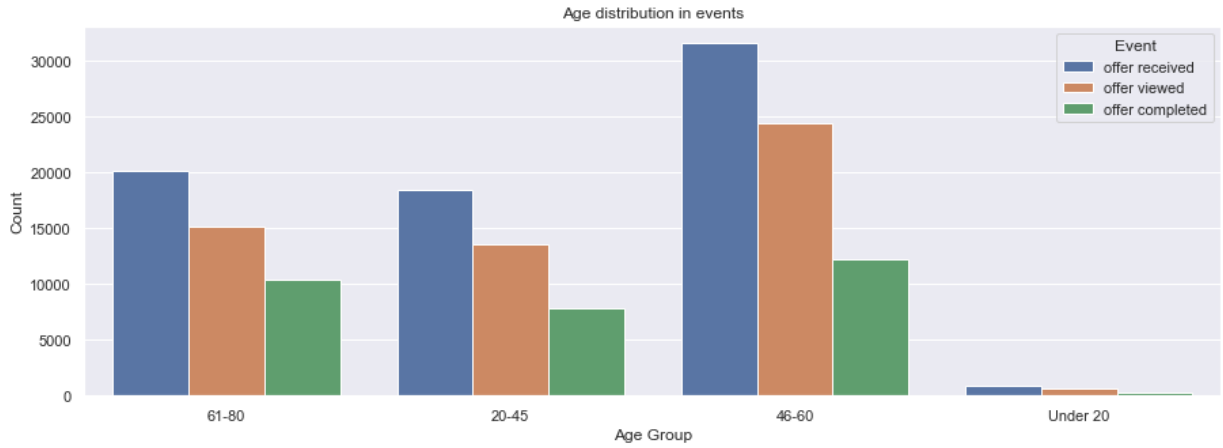
Actions to what these groups of customers do with the received offer were explored. Most of the customers don't pay attention to the offer and don't even have a look at it. Also, more several customers just view & ignore the offer than the ones who complete the offer. For better insights, the gender analysis was done to find out which group of customers most likely received the offers. It was observed that 63% of the male customers were engaging in these offers more compared to 37% of the female customers.





There were more male customers than female customers, and in general, people bought the discount offer then followed by the BOGO offer, and then the informational. Also, it was revealed that the age group of 41-60 was receiving more offers compared to the other groups.





Project Implementation:

- **Building and Evaluating the ML models and the benchmark model**

Before training the data some more cleaning on the data was performed to ease the model fitting process. The following tasks were implemented –

1. Encoding categorical data such as gender, offer type, channel, and age groups.

2. Encoding the 'event' data to numerical values: offer received -> 1, offer viewed -> 2, offer completed -> 3.

3. Encoding offer_id and customer_id because it was irrelevant in the training process.

4. Dropping the column 'became_member_on' and adding separate columns for month and year.

5. Scaling and normalizing the numerical data.

Final data is ready after tasks 1-5. Now the data was split (both features and their labels) into training and test sets, taking 60% of the data for training and 40% for testing. It contained – Training set: of 94501 rows, and a Testing set: of 63002 rows. Here, the F1 score was considered as the model metric to assess the quality of the approach and determine which model gives the best results. It can be interpreted as the weighted average of precision and recall. The traditional or balanced F-score (F1 score) is the harmonic mean of precision and recall, where an F1 score reaches its best value at 100 and worst at 0.

The three ML models – KNeighborsClassifier (also benchmark model), RandomForestClassifier, and DecisionTreeClassifier were trained and tested on the training and testing data respectively. No special parameters were provided to the models except for the 'random_state' parameter and the benchmark model: 'n_neighbors=5' parameter. A function was built to evaluate the train and test F1 scores of the models. After evaluating the models on training and testing data the following results were obtained –

```
In [75]: benchmark
```

Out[75]:

	KNN Benchmark Model	train F1 score	test F1 score
0	KNeighborsClassifier	54.35498	32.859274

```
In [78]: model_r
```

Out[78]:

	RandomForestClassifier Model	train F1 score	test F1 score
0	RandomForestClassifier	95.455075	72.783404

```
In [81]: model_d
```

Out[81]:

	RandomForestClassifier Model	train F1 score	test F1 score
0	DecisionTreeClassifier	95.455075	85.079839

- ## Model Refinement and Validation

As we can see the models RandomForestClassifier, and DecisionTreeClassifier gave a sufficiently better result compared to the benchmark model therefore I performed hyperparameter tuning on the benchmark model – KNeighborsClassifier to refine the model. To validate the model and tune the hyperparameters, I used the RandomizedSearchCV validation method. The reason behind this was the GridSearchCV was consuming more computational power and time and wasn't providing the results. I refined the benchmark model on the parameter – 'n_neighbors' list and 'metric' list. I also used fewer parameters to avoid the greater usage of CPU/GPU of my laptop as it was not providing results. Also, I lessened the number of iterations and the folds in the RandomizedSearchCV validation process for the same reason mentioned previously.

```
In [83]: # fit the cv model
         cv.fit(X_train, y_train)

         Fitting 2 folds for each of 1 candidates, totalling 2 fits

Out[83]: RandomizedSearchCV(cv=2, estimator=KNeighborsClassifier(), n_iter=1, n_jobs=-1,
                            param_distributions={'metric': ['minkowski', 'manhattan'],
                                                 'n_neighbors': [7, 9]},
                            random_state=42, verbose=1)

In [84]: cv.best_params_

Out[84]: {'n_neighbors': 9, 'metric': 'minkowski'}
```

After validation and tuning, the best parameters were obtained which were n_neighbors=9 and metrics= 'minkowski'. Later, I again trained the KNeighborsClassifier benchmark model on these parameters to improve the model F1 score. There wasn't much difference obtained, the training score was fair and similar to the previous model but the testing score improved from 33% to 36%. The refined benchmark model was evaluated on the f1 score metrics giving the results as –

```
In [87]: new_benchmark
```

Out[87]:

	Refined KNN Benchmark Model	train F1 score	test F1 score
0	KNeighborsClassifier	51.82379	35.902035

Project Results:

- ## Model Evaluation Report

The ML models as well as the refined model were evaluated on the F1 score metrics. The training and testing F1 scores of these models are already mentioned in the Model Refinement and Validation section of the report. Let us see a detailed classification report and the accuracy distribution of all these models.

KNeighborsClassifier

Train Report

	precision	recall	f1-score	support
1	0.54	0.84	0.66	43232
2	0.52	0.38	0.44	32553
3	0.82	0.15	0.25	18716
accuracy			0.54	94501
macro avg	0.62	0.46	0.45	94501
weighted avg	0.59	0.54	0.50	94501

Test Report

	precision	recall	f1-score	support
1	0.39	0.60	0.47	28631
2	0.19	0.15	0.17	21843
3	0.19	0.03	0.05	12528
accuracy			0.33	63002
macro avg	0.26	0.26	0.23	63002
weighted avg	0.28	0.33	0.28	63002

KNeighborsClassifier

Train Report

	precision	recall	f1-score	support
1	0.51	0.83	0.63	43232
2	0.51	0.31	0.39	32553
3	0.62	0.17	0.26	18716
accuracy			0.52	94501
macro avg	0.55	0.43	0.43	94501
weighted avg	0.53	0.52	0.47	94501

Test Report

	precision	recall	f1-score	support
1	0.41	0.66	0.51	28631
2	0.22	0.14	0.17	21843
3	0.22	0.05	0.09	12528
accuracy			0.36	63002
macro avg	0.28	0.28	0.25	63002
weighted avg	0.31	0.36	0.31	63002

RandomForestClassifier

Train Report

	precision	recall	f1-score	support
1	0.93	0.97	0.95	43232
2	0.96	0.90	0.93	32553
3	1.00	1.00	1.00	18716
accuracy			0.95	94501
macro avg	0.96	0.96	0.96	94501
weighted avg	0.96	0.95	0.95	94501

Test Report

	precision	recall	f1-score	support
1	0.68	0.75	0.72	28631
2	0.63	0.54	0.58	21843
3	1.00	1.00	1.00	12528
accuracy			0.73	63002
macro avg	0.77	0.76	0.76	63002
weighted avg	0.73	0.73	0.72	63002

DecisionTreeClassifier

Train Report

	precision	recall	f1-score	support
1	0.91	1.00	0.95	43232
2	1.00	0.87	0.93	32553
3	1.00	1.00	1.00	18716
accuracy			0.95	94501
macro avg	0.97	0.96	0.96	94501
weighted avg	0.96	0.95	0.95	94501

Test Report

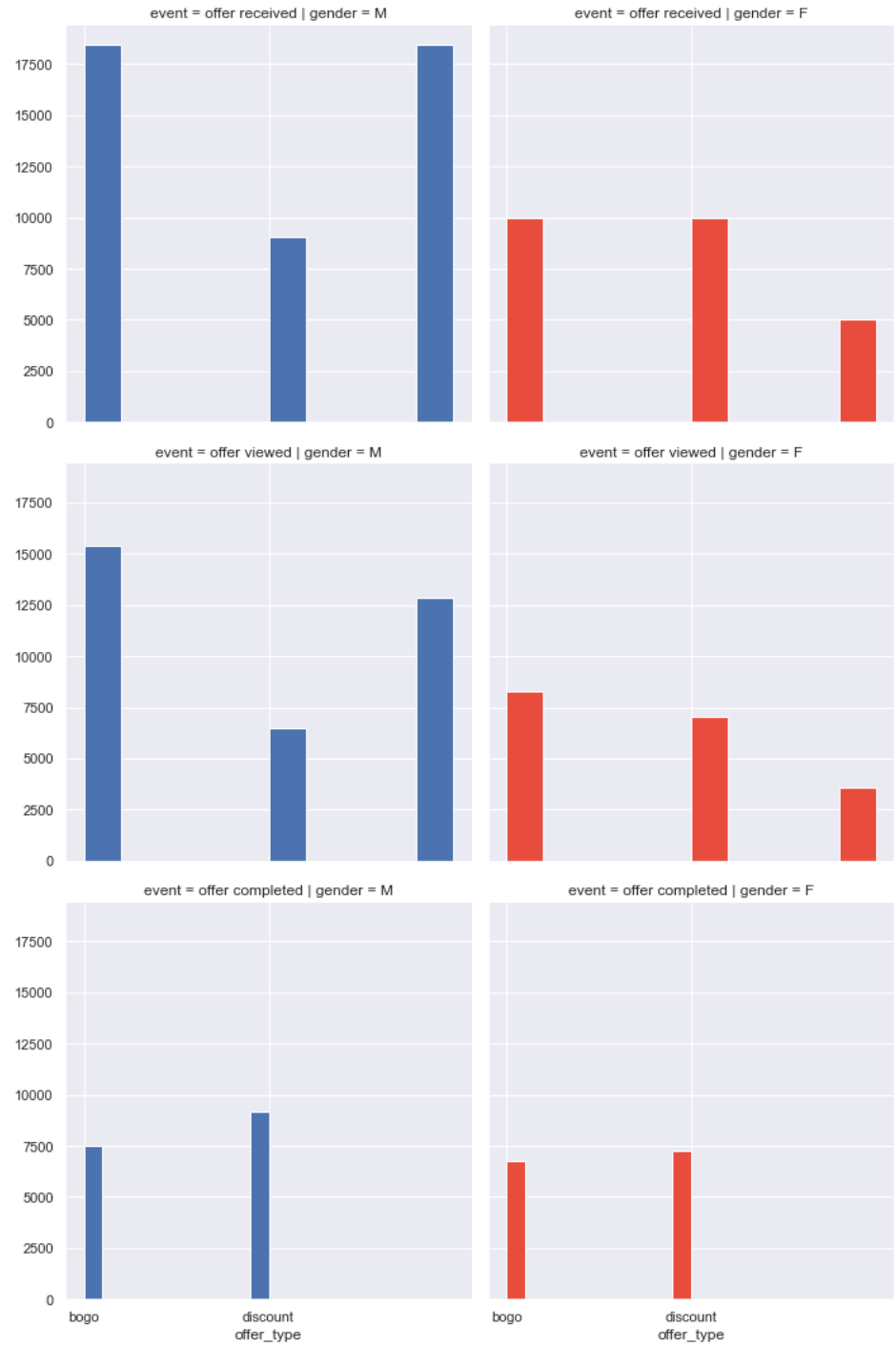
	precision	recall	f1-score	support
1	0.84	0.83	0.84	28631
2	0.78	0.79	0.79	21843
3	1.00	1.00	1.00	12528
accuracy			0.85	63002
macro avg	0.87	0.87	0.87	63002
weighted avg	0.85	0.85	0.85	63002

Conclusions:

- ### Conclusion and Justification

The males represent 62.7% of the data and use the Starbucks app more than the females. Specifically, both males & females in the age group 46-60 use the App the most. Discount offers are preferred by the customers. Also, there is less number of customers who complete the offer

as compared to the ones who just view & ignore it. We can look more at the figures & information in the Exploratory Data Analysis section to best determine which kind of offers to send to the customers.



The validation set (test data set) is used to evaluate the model. Both models are better than the benchmark. The best score is created by the DecisionTreeClassifier model, as its validation F1 score is 85.07, which is much higher than the benchmark. The RandomForestClassifier model scores well as well compared to the benchmark, with a test F1 score of 72.78. The Refined KNeighborsClassifier after hyperparameter tuning didn't give much improvement but its validation was 3% higher than the benchmark model. Our problem to solve is not that sensitive which requires a very high F1 score, so the scores are good & sufficient and can be used for the classification purpose to predict whether a customer will respond to an offer. The comparison of all the models is given below –

```
In [93]: comp
```

Out[93]:

	Model	train F1 score	test F1 score
0	KNeighborsClassifier (Benchmark)	54.354980	32.859274
1	Refined KNeighborsClassifier	51.823790	35.902035
2	RandomForestClassifier	95.455075	72.783404
3	DecisionTreeClassifier	95.455075	85.079839