Starbucks Capstone Challenge

1 Introduction

This data set contains simulated data that mimics customer behavior on the Starbucks rewards mobile app. Once every few days, Starbucks sends out an offer to users of the mobile app. An offer can be merely an advertisement for a drink or an actual offer such as a discount or BOGO (buy one get one free). Some users might not receive any offer during certain weeks.

Not all users receive the same offer, and that is the challenge to solve with this data set.

The task is to combine transaction, demographic and offer data to determine which customer respond best to which offer. This data set is a simplified version of the real Starbucks app because the underlying simulator only has one product whereas Starbucks actually sells dozens of products.

Every offer has a validity period before the offer expires. As an example, a BOGO offer might be valid for only 5 days. You'll see in the data set that informational offers have a validity period even though these ads are merely providing information about a product; for example, if an informational offer has 7 days of validity, you can assume the customer is feeling the influence of the offer for 7 days after receiving the advertisement.

There are given transactional data showing customer purchases made on the app including the timestamp of purchase and the amount of money spent on a purchase. This transactional data also has a record for each offer that a user receives as well as a record for when a user actually views the offer. There are also records for when a user completes an offer.

Example
To give an example, a user could receive a discount offer buy 10 dollars get 2 off on Monday. The offer is valid for 10 days from receipt. If the customer accumulates at least 10 dollars in purchases during the validity period, the customer completes the offer.

However, there are a few things to watch out for in this data set. Customers do not opt into the offers that they receive; in other words, a user can receive an offer, never actually view the offer, and still complete the offer. For example, a user might receive the "buy 10 dollars get 2 dollars off offer", but the user never opens the offer during the 10 day validity period. The customer spends 15 dollars during those ten days. There will be an offer completion record in the data set; however, the customer was not influenced by the offer because the customer never viewed the offer.

1.1 Domain Background

General article of advert targeting using ML algorithm, that describes how important personalized advertising is.
https://www.sciencedirect.com/science/article/pii/S2405959520301090

Specific article for the same data set where an unsupervised learning with clustering is used.
https://medium.com/@jeffrisandy/investigating-starbucks-customers-segmentation-using-unsupervised-machine-learning-10b2ac0cfd3b

The targeted control of offers to customers is a prime example of methods of machine learning and data science.
Which offer can I make to which customers. Which one is the customer most likely to look at and accept?
Is it possible from successfully made offers to develop other offers that are promising? These are questions of e-commerce that can be applied to many business areas.

1.2 Problem Statement

There are many questions to answer in this capstone project
* Is it possible with a combination of demographic and customers features to predict the offer completed value? This is part of a supervised machine learning algorithm.
* Same question is for viewed value.
* There are customers who make up 80% of sales. Let's call them power customer. If we label the customers as power or no power customer, is it possible to predict this label, based only on the customer demoraphic features?

1.3 Datasets and Inputs

The data is contained in three files:

- 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

Here is the schema and explanation of each variable in the files:

portfolio.json
- id (string) - offer id
- offer_type (string) - type of offer ie BOGO, discount, informational
- difficulty (int) - minimum required spend to complete an offer
- reward (int) - reward given for completing an offer
- duration (int) - time for offer to be open, in days
- channels (list of strings)

profile.json

- age (int) - age of the customer
- became_member_on (int) - date when customer created an app account
- gender (str) - gender of the customer (note some entries contain 'O' for other rather than M or F)
- id (str) - customer id
- income (float) - customer's income

transcript.json

- event (str) - record description (ie transaction, offer received, offer viewed, etc.)
- person (str) - customer id
- time (int) - time in hours since start of test. The data begins at time t=0
- value - (dict of strings) - either an offer id or transaction amount depending on the record

1.4 Solution Statement

To create a machine learning model, it is necessary to get the labels for each received offer. In the transcript file are all received offers, all viewed and all completed offers. But there is no link between the received, viewed and completed person offer combinations. Therefor it is necessary to create a dataset where for each received offer the information will be added about viewed and completed. The columns for viewed and completed are the label columns for the machine learning models.
After labeling the data an explorative data analysis is performed to decide which customer offer combinations will be part of the machine learning models.

1.5 Benchmark Model

The feature input vector for this dataset is with around 25 input features in csv file format relative less complex. I decide to use aws xgboost as the benchmark model the aws linear learner for comparison.

1.6 Evaluation Metrics

As evaluation metrics first the accuracy score will be defined, additional the confusion matrix will be plotted, and the precision and recall will be calculated. With the linear learner it is easy possible to optimize these values.

2 Data Cleaning

The project provides 3 three files as described in the dataset section. The files are formated json, I use the pandas read_json method to create dataframes.

```
# read in the json files
dataframe = pd.read_json('data/file.json', orient='records', lines=True)
```

After reading the files I start with the cleaning process for each dataframe

2.1 Portfolio

Let me display the first rows of portfolio dataframe

index	reward	channels	difficulty	duration	offer_type	id
0	10	[email, mobile, social]	10	7	bogo	ae264e3637204a6fb9bb56
1	10	[web, email, mobile, social]	10	5	bogo	4d5c57ea9a6940dd891ad5
2	0	[web, email, mobile]	0	4	informational	3f207df678b143eea3cee63

Tabelle 2-1: Raw Portfolio Data

The columns channels, offer_type and id need some attention.

2.1.1 Column specific cleaning steps

channels

The channels column contains list as data type. Each list contains different values. I will do a one hot encoding for this column and create for each value an own variable column. Is the value in the list, the variable column is one otherwise 0.

index	email	mobile	social	web
0	1	1	1	0
1	1	1	1	1
2	1	1	0	1

Tabelle 2-2: One hot encoded channels

offer_type

this column contains a categorical variable. For a later machine learning dataset, I need a numerical value. Alternatively an one hot encoding is here also possible. I will use the one hot encoding, but for reasons of explanatory data analysis I keep the original column.

id

The id column is an alpha numerical unique identification. In the other datasets also id columns will be available. To avoid name conflicts I rename this column to **offer_id**.
One more point is, that the id values are totally cryptic. I decide to add an additional identifying column called **ticks**. This column is combined as offer_type_reward_difficulty_duration. The corresponding value i.e. is bogo_10_10_7.

index	email	mobile	social	web	ticks
0	1	1	1	0	bogo_10_10_7
1	1	1	1	1	bogo_10_10_5
2	1	1	0	1	informational_0_0_4

Tabelle 2-3: One hot encoded offer_type and additional ticks column

duration

The duration column values are given in hours. For later preprocessing steps it is usefull to convert these values to hours by multiplicate the values with 24.

2.2 Profile

Let me first display the first rows of portfolio dataframe

index	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

Tabelle 2-4: Raw portfolio

Directly in the first three rows I see none values, combined with age = 118. When I investigating the dataframe more detailed I find that missing values for gender, income and age = 118 comes 2175 times together. This are customer who not give any detailed demographic information about themselves.

2.2.1 Missing values:

I decide to keep the 2175 customers for the first time. Of course the columns with the missing values need an update

gender
I decide to create a new gender called Unknown. The corresponding value is ‘U’. I replace all none values with ‘U’.

income
I decide to fill the none values with the mean income of all customers. Finally the value itself is not so important. For statistically investigations or machine learning models all customers have the same features.

age
I decide to use the same process as for income. Of course the mean age I calculate based on the customers exclude the customers with age = 118.

2.2.2 Column specific cleaning steps

gender
This column is a categorical variable. To use it in a machine learning dataset it’s necessary to map the value to numeric, alternatively a one hot encoding with columns for each gender is created.

index	F	M	O	U
0	0	0	0	1
1	1	0	0	0
2	0	0	0	1

Tabelle 2-5: One hot encoded gender columns

became_member_on
the column contains a simple string with the date of customer registration. This value will be converted to a date time format.

```
pd.to_datetime(clean_profile.became_member_on, format='%Y%m%d')
```

Additionally I create columns for year, month, day and member_since_days which I extract from the converted column

index	became_member_on	year	month	day	member_since_days
0	2017-02-12	2017	2	6	1698
1	2017-07-15	2017	7	5	1545
2	2018-07-12	2018	7	3	1183

id
The id column is an alpha numerical unique identification. In the other datasets also id columns will be available. To avoid name conflicts I rename this column to **person_id**.

2.3 Transcript

Let me first display the first rows of transcript dataframe

index	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

Tabelle 2-6: Raw transcript data

In this dataframe are again several cleaning steps necessary. Let me go on step by step.

2.3.1 Columns specific

person
The person column is the same as in the profile dataset. For an easy merge process I will rename this column to **person_id**.

event
The event column contains categorical values. To use this values in a machine learning model I will do a one hot encoding and add the columns. The event column itself I keep also.

index	person_id	time	completed	received	viewed	transaction
1	a03223e636434f42ac4c3df47e8bac43	0	0	1	0	0
2	e2127556f4f64592b11af22de27a7932	0	0	1	0	0
12657	9fa9ae8f57894cc9a3b8a9bbe0fc1b2f	0	0	0	0	1

Tabelle 2-7: Part oft he transcript dataframe after one hot encoding

value
This column contains json data. The content of the value depends on the corresponding event. Is the event offer received, viewed or completed the value contain the keys, **reward**, **offer id** or **offer_id** and the value for the reward and the offer_id itself. This is a case of messy data, where same variables are stored in columns with different names. In case that the event is transaction the value contain the key amount and an amount value.
To clean this column, I use the pandas json_normalize method. The columns offer id and offer_id must be combined in one column. The reward column will be droped. This column is part of the portfolio dataframe and will be merged later inside. The output of the value column cleaning is displayed in the next column.

amount	offer_id
NaN	0b1e1539f2cc45b7b9fa7c272da2e1d7
NaN	2906b810c7d4411798c6938adc9daaa5
34.56	NaN

Tabelle 2-8: value column after normalizing and cleaning

2.4 Full Dataframe

This dataframe is not available yet. I will create this dataframe by merging the profile and the portfolio dataframe with the transcript. The merging columns are **person_id** and **offer_id**.

2.4.1 Column specific

validity
This column is added to the full dataframe by adding time with duration. This column is needed for a later labeling process.

index	time	duration	validity
0	168	168.0	336.0
1	216	168.0	384.0
2	0	168.0	168.0
3	18	168.0	186.0

Tabelle 2-9: Columns time, duration and validity in full dataframe

3 Labeling Process

In the labeling process I want to figure out, how is the viewed and completed status for each received offer. In the currently dataset each offer event is an own row.

index	person_id	ticks	time	offer_type	received	viewed	completed
168338	68be06ca386d4c31939f3a	discount_2_10_10	408	discount	1	0	0
168339	68be06ca386d4c31939f3a	discount_2_10_10	408	discount	0	1	0
168340	68be06ca386d4c31939f3a	discount_2_10_10	552	discount	0	0	1

Tabelle 3-1: Raw situation, with one row for each event status

The goal is to transfer the events viewed and completed to the first received event.

index	person_id	ticks	tim	offer_typ	receive	viewe	complete
16833	68be06ca386d4c31939f	discount_2_10_1	408	discount	1	1	1

Tabelle 3-2: The received event with information about viewed and completed.

The process to do this labeling is the following.

- First I create a received dataframe only with the received offer events.
- For each event in the received dataframe I must find a match, with same person_id, same offer_id and finally the time for the corresponding viewed or completed event must within the validity.
- When I find a corresponding event the value in the column for the event become 1. Otherwise, when I do not find a corresponding event the column the value become -1.

For the informational offers are no completed events tracked. In this case I have to find in the regular transactions a corresponding match. It is assumed that the customer is influenced about the informational offer for the duration. In this case a match is found if person_id and time of regular transaction is within the validity of informational offer.

After the labeling process I export the updated received dataframe to a csv file to use the data in further notebooks.

In total I have 76277 received offers.

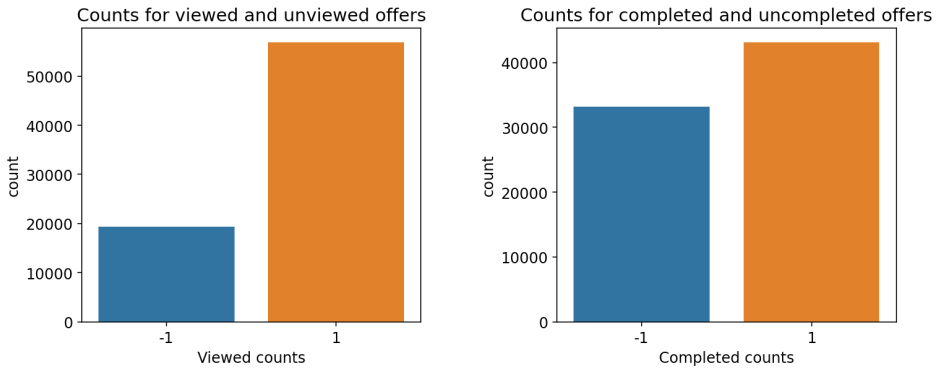


Figure 3-1: Counts for viewed and completed events

From these received offers are around 20000 unviewed and 32000 uncompleted.

The detailed process can be viewed in the jupyter notebook Labeling.ipynb.

4 Exploratory Data Analysis

For the exploratory analysis I go first through the portfolio and profile data seperatly

4.1 Portfolio

First plot how many offers for each offer type are available.

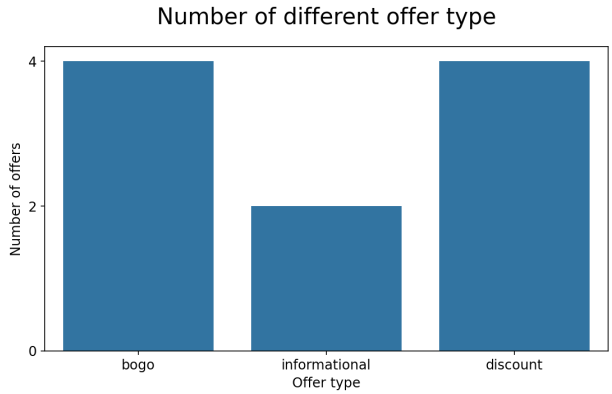


Figure 4-1: Offers per offer type

In total there are 10 different offers based on three offer types. From type bogo and discount we have respectively 4 from informational we have 2 different offers.

ticks	offer_type	reward	difficulty	duration	email	mobile	social	web
bogo_10_10_7	bogo	10	10	168	1	1	1	0
bogo_10_10_5	bogo	10	10	120	1	1	1	1
bogo_5_5_7	bogo	5	5	168	1	1	0	1
bogo_5_5_5	bogo	5	5	120	1	1	1	1
discount_5_20_10	discount	5	20	240	1	0	0	1
discount_3_7_7	discount	3	7	168	1	1	1	1
discount_2_10_10	discount	2	10	240	1	1	1	1
discount_2_10_7	discount	2	10	168	1	1	0	1
informational_0_0_4	informational	0	0	96	1	1	0	1
informational_0_0_3	informational	0	0	72	1	1	1	0

When we take a view to the sorted table we can see that for bogo rewards with 10$ and 5$ are available, the discounts have rewards 5$, 3$ and 2$ dependent on the difficulty.

4.2 Profile

4.2.1 Univariate Analysis

In total we have 17000 customers. Let's have first a view on the single demographic features.

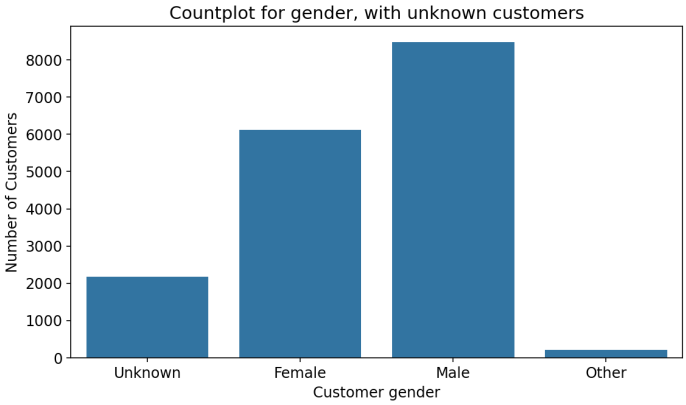


Figure 4-2: Customer count for gender

gender	gender_count
F	6129
M	8484
O	212
U	2175

Table 4-1: Customer Count for gender.

We have more male as female, 212 customer with gender other and 2175 customer how did not give demographic information.

In the customer gender count the customers without demographic information are inside. For the following plots I remove this customers, because they will falsify the plots.

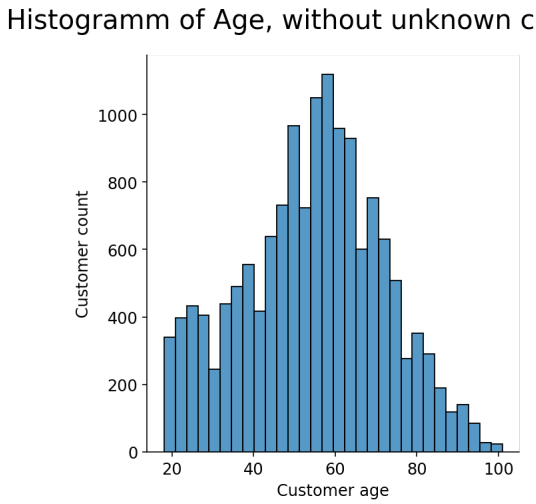


Figure 4-3:Histogram of customer age distribution

The age is reasonably normally distributed. This is the case when we removed the customers who not provided any informations and where the age in the raw data was set to 118.
The range of age is between 18 and 101 years old customers. The mean and median are close to each other at around 55 years.

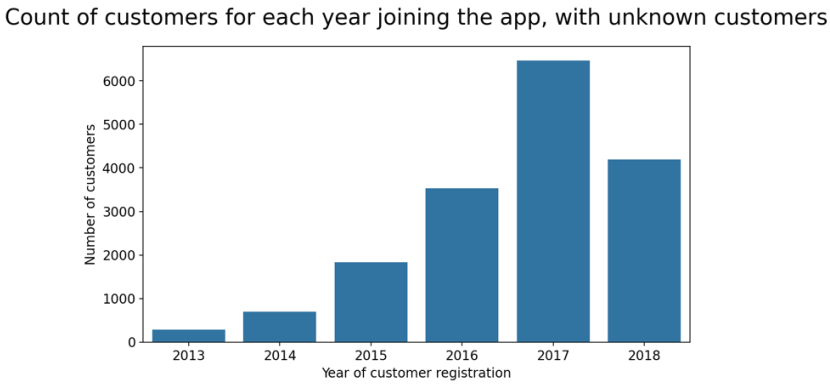


Figure 4-4: Year of registration

The number of registered customers is increasing every year until the maximum customer growth is reached in 2017. In 2018, the number of new customers will decrease again.

4.2.2 Bivariate Analysis

Let's check if there are any correlations between the customer features

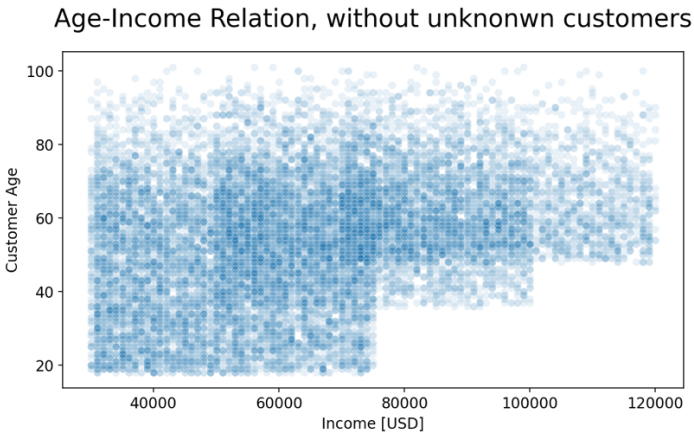


Figure 4-5: Age income relation of customers.

There are clear income blocks. i.e. more than 100.000$ only for people, who are older than 50 years. The general correlation is with a value from 0.3 small.

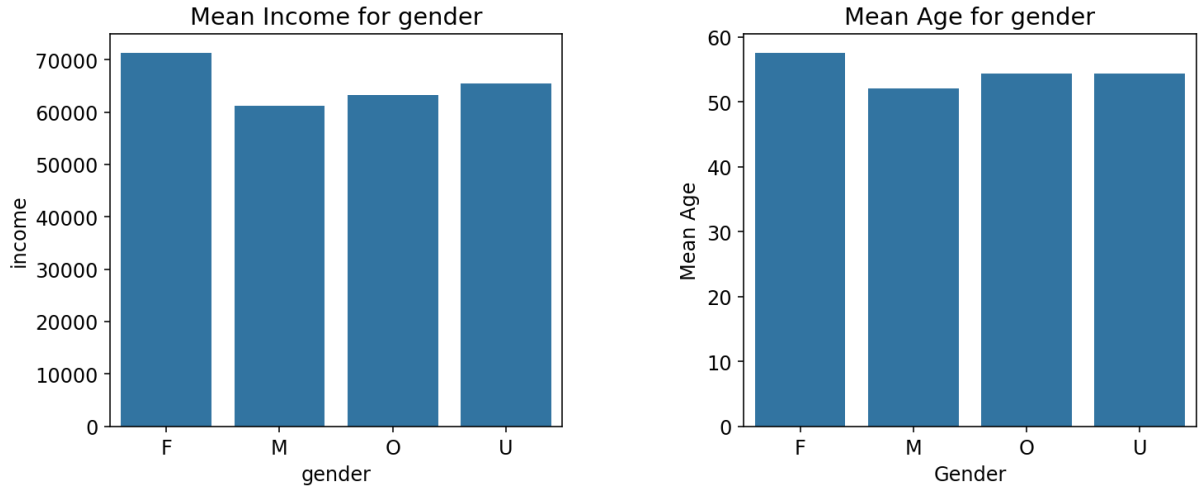


Figure 4-6: Mean Income and mean age for each gender

As we can see the mean income of female customer is around 10000$ higher than from male customer. The mean age of female customer is also higher. Therefore it might makes sense.

4.3 Offer based analysis.

Based on the received dataframe with the information a received offer is viewer an d completed or not we can check how are the viewed ratios and completed ratios for each offer.

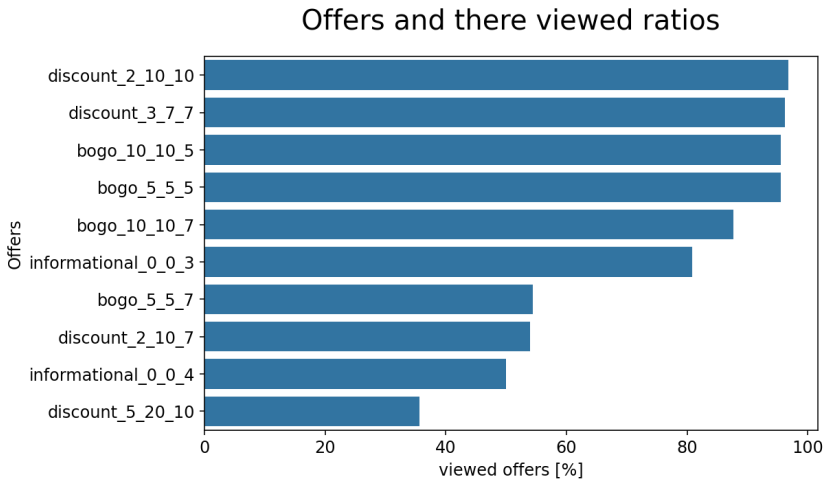


Figure 4-7: Viewed rate for offers

The viewed ratios for offers have high differences. There are offers with viewed rates around 95% and offers with viewed rates below 60%.
Let's check the channels of the offers.

ticks	unviewed	viewed	email	mobile	web	social
discount_5_20_10	64.4	35.6	1	0	1	0
informational_0_0_4	50.0	50.0	1	1	1	0
discount_2_10_7	46.0	54.0	1	1	1	0
bogo_5_5_7	45.6	54.4	1	1	1	0
informational_0_0_3	19.2	80.8	1	1	0	1
bogo_10_10_7	12.3	87.7	1	1	0	1
bogo_10_10_5	4.5	95.5	1	1	1	1
bogo_5_5_5	4.5	95.5	1	1	1	1
discount_3_7_7	3.8	96.2	1	1	1	1
discount_2_10_10	3.2	96.8	1	1	1	1

We can see that the viewed ratio depends directly on the used channels.
The missing social channel has a very big influence, also the missing mobile and web channels will move down the viewed ratio.

How is the situation for the completed ratios?

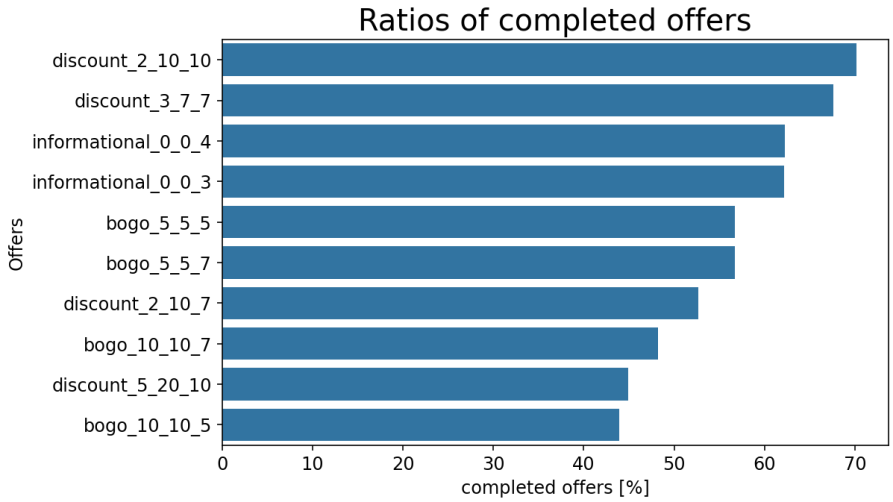


Figure 4-8: Completion rate for offers.

The completion ratio moves down more continuously. It seems that the popularity depends first on the offer type, followed by the features reward and difficulty.

Let's display the viewed ratios and completed ratios for the offers in one plot.

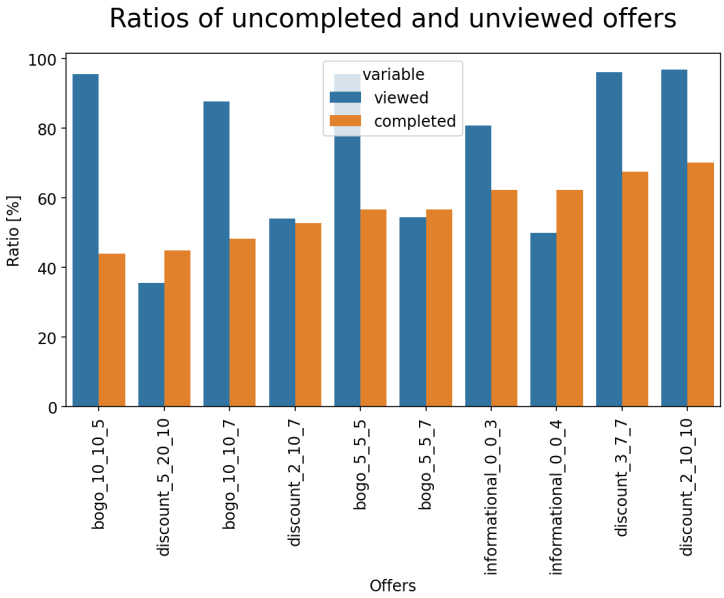


Figure 4-9: Ratios of viewed and completed based on offers

index	unviewed	uncompleted	viewed	completed
unviewed	1.000000	0.306593	-1.000000	-0.306593
uncompleted	0.306593	1.000000	-0.306593	-1.000000
viewed	-1.000000	-0.306593	1.000000	0.306593
completed	-0.306593	-1.000000	0.306593	1.000000

Table 4-2: Correaltion table for viewed and completed ratios.

Based on this bar plot it is not possible to see a correlation between the viewed ratio and the completed ratio. When we calculate the correlation, we get a value of 0.3.

4.4 Customer based analysis

Up to now we have worked with the offer-based counts. Let's implement the customers and check how is the customer behavior for the offers.

How many offers receive the customers?

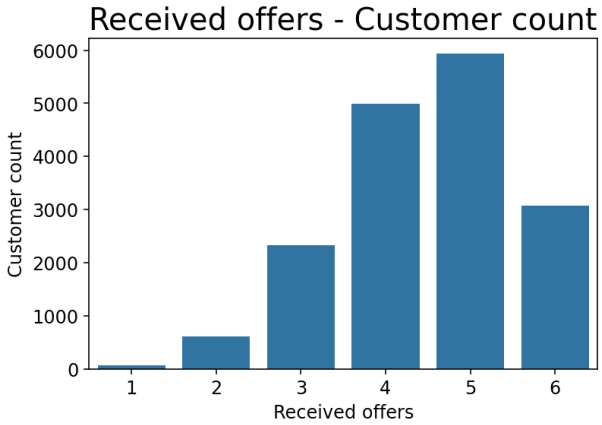


Figure 4-10: Received offers per customer.

Most customer have received 4 and 5 offers. How many offers were different viewed or completed from this offers we will analyze.

Let's create a table with the received, viewed and completed counts per customer.

person_id	received	viewed	completed
0009655768c64bdeb2e877511632db8f	5	4.0	5.0
00116118485d4dfda04fdbaba9a87b5c	2	2.0	0.0
0011e0d4e6b944f998e987f904e8c1e5	5	5.0	3.0

Table 4-3: received viewed and completed count for each customer

How many from the received offers the customer viewed? In the following plot we can see, that for customer who received i.e. 5 offers the customer count decreases with lower viewd counts.

Figure 4-11: Customer viewed counts based on received offer counts

Received / Viewed	0.0	1.0	2.0	3.0	4.0	5.0	6.0
1	12.0	61.0	-	-	-	-	-
2	50.0	184.0	376	-	-	-	-
3	53.0	360.0	797	1115	-	-	-
4	52.0	339.0	1100	1632	1865	-	-
5	29.0	180.0	700	1366	1753	1903	-
6	5.0	47.0	226	512	727	744	806

Table 4-4: Customer viewed counts based on received offer counts

Let's do the same plot for completed counts.

Figure 4-12: Customer completed counts based on viewed counts.

The completed counts show a similar behavior as the viewed counts before.

When there are customers who received i.e. 5 offers and many customer viewed 5 and 4 and 3 etc.
How is the individual viewed ratio for customer between received and viewed offers.

person_id	unviewed	viewed	viewed_ratio
0009655768c64bdeb2e877511632db8f	1.0	4.0	80.0
00116118485d4dfda04fdbaba9a87b5c	0.0	2.0	100.0
0011e0d4e6b944f998e987f904e8c1e5	0.0	5.0	100.0

Table 4-5: Viewed ratio for each customer

Let’s plot a diagram what shows the customer counts for each viewed ratio.

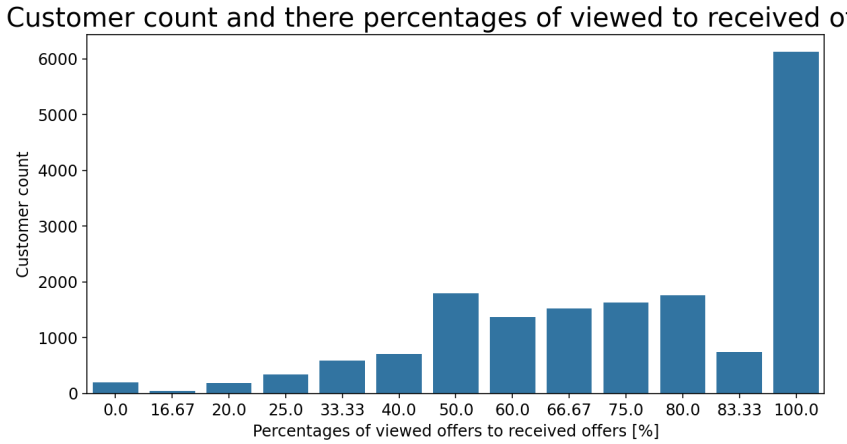


Figure 4-13: Customer count over viewed ratios

There are above 6000 customers who viewed every offer they received. And 87.9% of customer viewed more than 50% of their received offers.

Let’s do repeat this for the completed ratio, but as baseline we use no the already viewed offers. Completed offers which are not viewed are not considered.

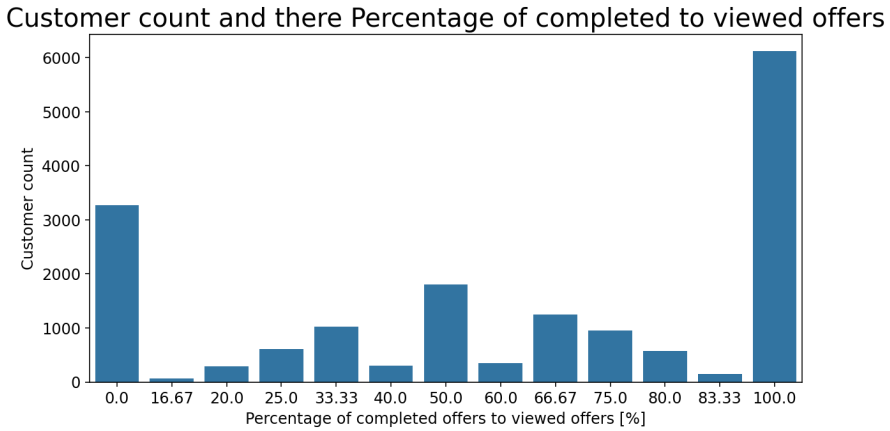


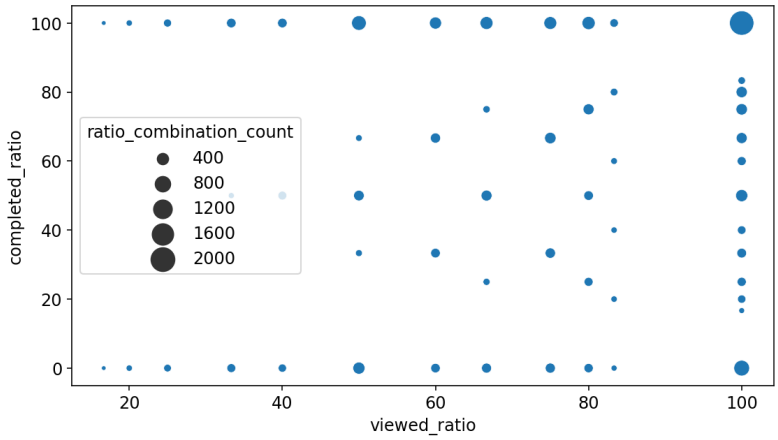
Figure 4-14: Customer count over completed ratios

Again we have 6000 customer who completed each viewed offer. But additional there are more than 3000 customerwho never completed an offer they received and viewed. Over 66% of customers completed more than 50% of received and viewed offers.

For the 6000 customer who viewed every offer and the 6000 customer who completed every viewed offer I want to know, whether or not this are the same customers.

When we check this, we found that only 2146 customers are in both groups. That is less than I expected.

Let's check how is the correlation between. Viewed and completed ratios.
As we can see there are around 2000 customer who have viewed and completed ratios of 100%. But the plot is relatively symmetric. It seems that there is no really a correlation between the viewed and the completed ratio.



4.5 Customers to exclude from discount offers

In the udacity intruduction test is one point that there are customers who might complete the offer before they viewed it. And more complex, that there are customers who made an amount summation within 7 or 10 days which is enough to complete a discount offer and it makes no sense to provide customers discounts, when they spent the money already without discount offers.

To simplify this situation I decide to search only for customers who completed discount offers without viewed this offer.

This group contains in total 2889 customers. Of course it is possible that some of these customers sometimes also viewed and completed an offer.
So I find out additionally customers who viewed and completed discount offers. For this group I have 9246 customers.
The intersection of the 2889 customer group and the 9246 customer group gives me 1846 customers.
When I exclude these customers from my unviewed completed group I have in total 1043 customers where it makes no sense to send discount offers.

4.6 Summary of explorative data analysis

Based on this explorative data analysis it is not possible to define who customers will prefer which offers. Of course we can make a detailed customer feature analysis for each provided and completed and uncumpleted offer. From my point of view that makes not much sense.

We have also seen that there are offers with really bad viewed ratios and we have seen that this depends definetely on the channels which used to receive the offer.

Also we have found customers to exclude from discount offers, because these customers have completed unviewed discount offers.

To go on and find out which offers we can provide to our customer I will go on with the following process.
I will create a dataframe where I have the viewed rate for each customer and offer. Is this viewed rate above or equel to 50% The viewed label for this customer offer combination is set to 1. Is the rate below 50% the label will be set to 0.

This dataset from customer offer combinations with the corresponding viewed labels is the baseline for a machine learning model.

The same process I will do based on the completed offers. With this machine learning models I predict the labels for the missing customer offer combinations.
Based on these predictions we can go in a detailed analysis of customer offer combinations.

5 Machine Learning

5.1 Preprocessing

Baseline for the machine learning process is the received dataframe with the viewed and completed information about each offer.

person_id	ticks	received	viewed	completed	bogo	discount	informational
68be06ca3...	discount_2_10_7	1	1	-1	0.0	1.0	0.0
68be06ca3...	discount_2_10_7	1	1	-1	0.0	1.0	0.0
68be06ca3...	discount_2_10_7	1	1	-1	0.0	1.0	0.0
68be06ca3...	discount_2_10_7	1	1	-1	0.0	1.0	0.0
68be06ca3...	discount_2_10_7	1	1	1	0.0	1.0	0.0

Table 5-1: Received dataframe

Based on this dataframe I created a dataframe where each customer offer combination is unique. If a customer received the same offer two or more times than the viewed and completed counts are a summation from the single events.

From the uncompleted, completed counts I calculate the ratio. These ratios are the baseline for the machine learning target. Is the ratio above or equal 0.5 the target value becomes 1, is the ratio below 0.5 the target values becomes null.

This step I do twice for completed und viewed separately.
My machine learning datasets currently looks like this table.

person_id	ticks	completed_coun	uncompleted_coun	tota	rati	binary_targe
0009655768c6	bogo_5_5_5	1.0	0.0	1.0	1.0	1
0009655768c6	discount_2_10_1	1.0	0.0	1.0	1.0	1
0009655768c6	discount_2_10_7	1.0	0.0	1.0	1.0	1

Table 5-2: Machine learning baseline table

Now the customers features and offer features will be merged in this dataframe. The binaray target column are the targets for the machine learning models.

One relevant question is which features are to use and which customer offer combinations. Remember I have 2175 customers with equal features because I filled these features with basic values. For the first time I decide to keep this customers in my machine learning models, because I think it is one group and possible that this groups have similar behavior of different offers.

As features I decide to use the following features

Customer features:
Age, income, gender (as one hot encoded vector), member_since_days

Offer features:
Offer_type (as one hot encoded vector), reward, difficulty, duration, channels (as one hot encoded vector)

After the feature selection I do a normalizing step on the age and the income feature.

The preprocessed machine learning data I write to csv files for features and targets.

5.2 Xgboost completed

First, I use the xgboost algorithm with an aws sagemaker build-in estimator. I want to predict, whether a customer completed an offer or not.

The machine learning model data is loaded from the csv files for features and targets created before in the last preprocessing step.

The data is not sorted in any way, so there is no shuffling necessary. I can directly start to split my data into a training and a test set. The size of the test set is set to 10%

Additionally, I create from the train set a validation set. The xgboost algorithm gives the possibility for a validation loop to avoid overfitting. The size of the validation set is again set to 20% of the train set.

In total we have 10% test set size, 72% train set size and 18% validation set size.

When I train the model I get the following confusion matrix.

prediction (col)	0.0	1.0
actual (row)		
0	1560	1043
1	788	2938

Table 5-3: Confusion matrix of xgboost completed output

Here are the values for Recall, Precision, Accuracy.

Recall: 0.789
Precision: 0.738
Accuracy: 0.711

For the first shot we get an accuracy of 71.1%. Let's go on with the same test size for a linear learner algorithm.

5.3 Linear Learner

Second , I use the Linear Learner algorith. I want to predict, whether a customer completed an offer or not.

The machine learning model data is loaded from the csv files for features and targets created before in the last preprocessing step.

The data is not sorted in any way, so there is no shuffling necessary. I can directly start to split my data into a training and a test set. The size of the test set is set to 10%. The linear Learner does not use an validation set to avoid overvitting

prediction (col)	0.0	1.0
actual (row)		
0	1396	1193
1	760	2980

Table 5-4: Confusion matrix of Linear Learner completed output.

Recall: 0.797
Precision: 0.714
Accuracy: 0.691

6 Results

The xboost algorithm gives slightly better results as the linear learner.
With the given machine learning model it is possible to predict for each customer offer combination, whether the customer will complete the offer or not.
Also new customer offer combinations can be predicted, whether the customer will complete or not an offer.

6.1 Improvements

- Maybe it makes sense to drop the customer without demographic features.
- The feature member_since_days does not work for new customers. An update of the machine learning model would maybe make sense.
- The machine learning model could also trained on the viewed predictions. But I think this makes not so much sense, because we have high viewed rates on offers with all channels. I expect when the offers received by all available channels a total viewed rate around 90%. A machine learning model which predict a viewed or unviewed will give a high accuracy based on the channel features.