DAT 554 Final Project: Starbucks Customer Rewards Offer Segmentation

Nathan Krasinski

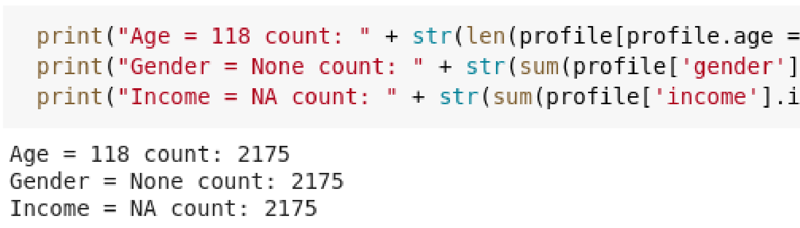
***Abstract*— This project uses a k-means to segment a simulated dataset of Starbucks reward users by the offers they most responded to. Starbucks reward users have the ability to add gender, income and age information to their profile; approximately 87% of the dataset has added this key demographic data. The 13% that did not have demographic data is treated as a demographic that doesn't wish to share data. The offers are separated into 3 types: discount, informational and buy one get one free. By analyzing which customers viewed an offer and ultimately redeemed the offer, I was able to determine the customer profiles that could be best utilized to maximize effectiveness of the offers. 5 clusters were found and identifiable by specific demographics; that allow more concise and direct marketing.**

I. Introduction

1) *Background:*
   Companies strive to effectively market to their customers. Sending irrelevant offers to a customer can erode customer loyalty, lower ad view rates, and ultimately waste money. As a data scientist, being able to maximize outcomes and minimize waste is an effective and worthwhile endeavor in the marketing field.
2) *The Problem Statement:*
   The problem that is sought to answer in this project is: How can Starbucks most effectively use their customer reward program offers to maximize redemption and minimize wasted offers to their customer base?

II. Dataset

This dataset was obtained from Kaggle. The dataset is simulated data from the Starbucks customer rewards program[3]. The dataset includes 3 files: portfolio.json, profile.json, and transcript.json. The portfolio file has the following information: id (string), offertype (string), difficulty (int), reward (int), duration (int), and channel (list of strings). The profile file contains the following: age (int), became_member_on (int), gender (string), id (string), income (float). The transcript file has: event (str), person (string), time (int), value (dict of strings). The dataset as whole contains 17,000 users that represent 306,534 recorded events that include: offer received, offer viewed, offer completed, or monetary transaction. 2,175 users, approximately 13% of the data, did not disclose their demographic data (age, income, gender).



```
print("Age = 118 count: " + str(len(profile[profile.age =
print("Gender = None count: " + str(sum(profile['gender']
print("Income = NA count: " + str(sum(profile['income'].i
```

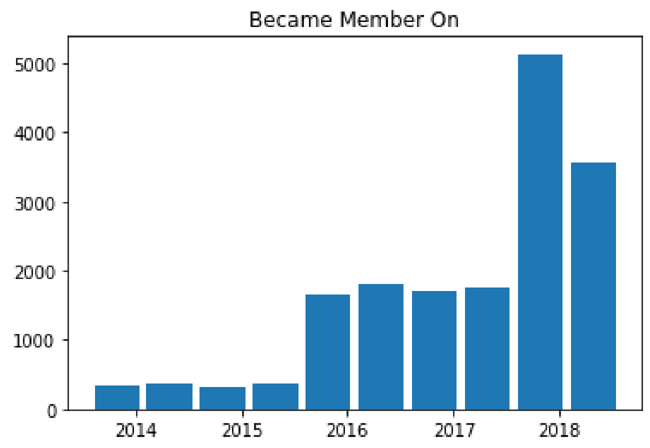
```
Age = 118 count: 2175
Gender = None count: 2175
Income = NA count: 2175
```

These users were treated as a separate subset, with the idea that customers who do not wish to share their data may share similarities in the way to effectively market to them.

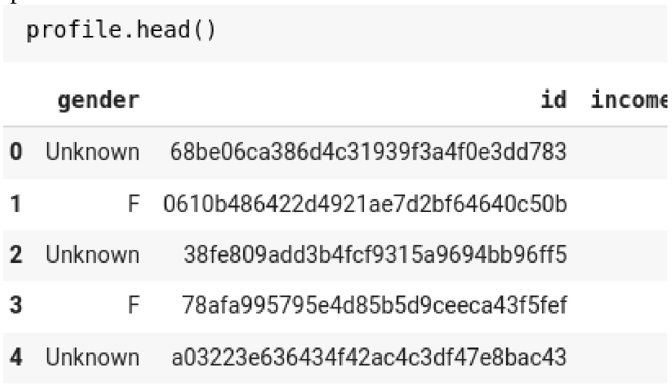
Preprocessing:

Preprocessing the profile data included making some unique decisions. As previously stated, I made the decision to treat the 13% of users that did not provide their age, income and gender as unknowns rather than imputing or dropping the data. This decision was made primarily because this group could constitute a specific customer type that may share values in how they interact with offers. This choice led me to change the age and income variables from numerical categories to categorical variables by binning certain groups. For the income variable it was categorized by "unknown", under "$50k", "$50k-$80k" and "$80k+". For the age variable it was categorized as "unknown", "<25", "25-40", "61-80", and "81 and over". Gender had the categories: "unknown", "male", "female", "other".

The became_member_on variable skews heavily toward new users and does not contain enough variability for segmentation.



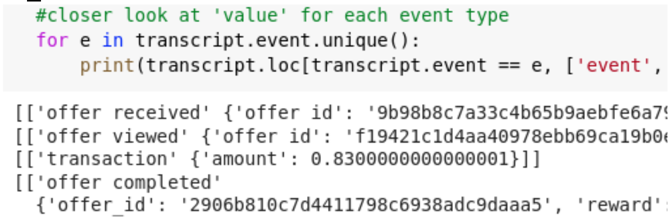
This variable was dropped from the dataframe. The id variable was used solely to merge the portfolio data file by person id. This left the profile data looking like this after being preprocessed:



```
profile.head()
```

	gender	id	income
0	Unknown	68be06ca386d4c31939f3a4f0e3dd783	
1	F	0610b486422d4921ae7d2bf64640c50b	
2	Unknown	38fe809add3b4fcf9315a9694bb96ff5	
3	F	78afa995795e4d85b5d9ceeca43f5fef	
4	Unknown	a03223e636434f42ac4c3df47e8bac43	

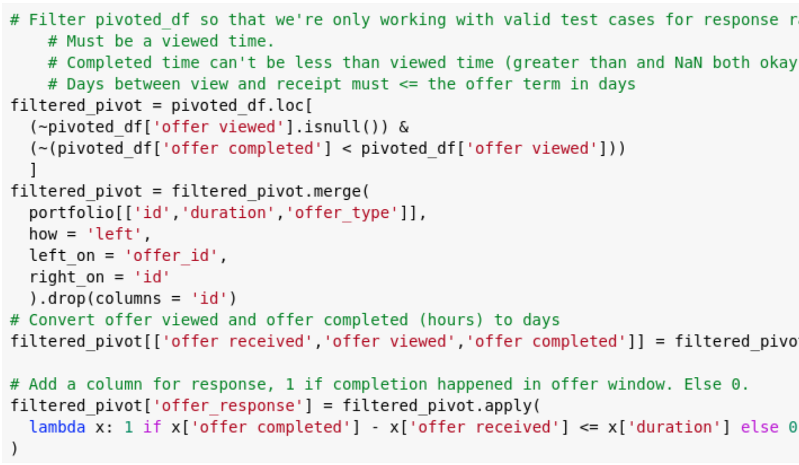
Preprocessing the transcript data required pulling out the offer_id and the amount from the value variable:



```
#closer look at 'value' for each event type
for e in transcript.event.unique():
    print(transcript.loc[transcript.event == e, ['event',

[['offer received' {'offer id': '9b98b8c7a33c4b65b9aebfe6a7
[['offer viewed' {'offer id': 'f19421c1d4aa40978ebb69ca19b0
[['transaction' {'amount': 0.8300000000000001}]]
[['offer completed'
  {'offer_id': '2906b810c7d4411798c6938adc9daaa5', 'reward'
```

The data then needed to be cleaned as offers could be completed without ever being seen; which means they were being bought regardless of the offer in the rewards program and not relevant to the problem I am solving. Transactions that didn't follow the "offer received -> offer_viewed -> offer completed" model are deemed unsuccessful:



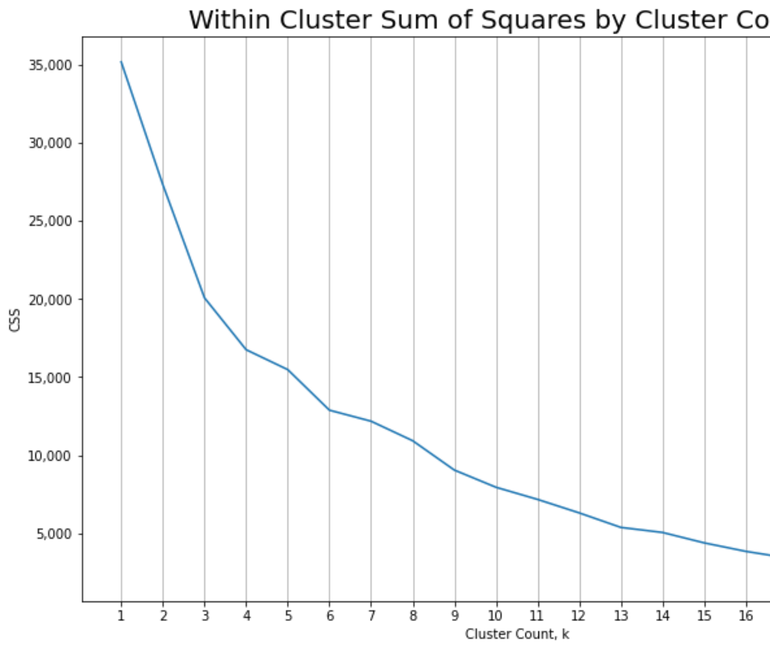
```
# Filter pivoted_df so that we're only working with valid test cases for response r
    # Must be a viewed time.
    # Completed time can't be less than viewed time (greater than and NaN both okay
    # Days between view and receipt must <= the offer term in days
filtered_pivot = pivoted_df.loc[
  (~pivoted_df['offer viewed'].isnull()) &
  (~(pivoted_df['offer completed'] < pivoted_df['offer viewed']))
  ]
filtered_pivot = filtered_pivot.merge(
  portfolio[['id','duration','offer_type']],
  how = 'left',
  left_on = 'offer_id',
  right_on = 'id'
  ).drop(columns = 'id')
# Convert offer viewed and offer completed (hours) to days
filtered_pivot[['offer received','offer viewed','offer completed']] = filtered_pivo

# Add a column for response, 1 if completion happened in offer window. Else 0.
filtered_pivot['offer_response'] = filtered_pivot.apply(
  lambda x: 1 if x['offer completed'] - x['offer received'] <= x['duration'] else 0
)
```

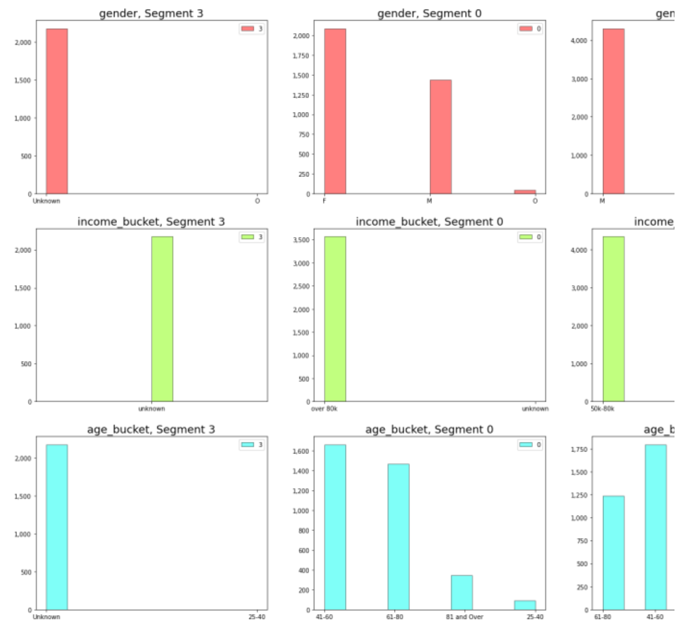
Preprocessing offers data required the least amount of preprocessing. Since I was not concerned specifically about how the offers went out to the customer, this column was not used in analysis. This data file was merged to the portfolio data by offer_id. Future questions could be answered using this data such as: Are there specific channels in the rewards program that are more effective?

III. Solution

I dummy encoded the profile data rather than one-hot encoding in order to avoid correlation among variables and improve clustering. Using the dummy encoded profile data, I determined the optimal centroids using within-cluster sum of squares. Using the elbow method, It was determined that somewhere between 4 and 6 clusters was optimal.



I initially attempted to use 6 clusters, but the cluster came out very ambiguous and would be difficult to interpret to a marketing team. 5 clusters was the final settled upon cluster number. The cluster label was added to the original profile dataset and the clusters were able to be easily viewed as below:



The clusters were identified as having the following properties:

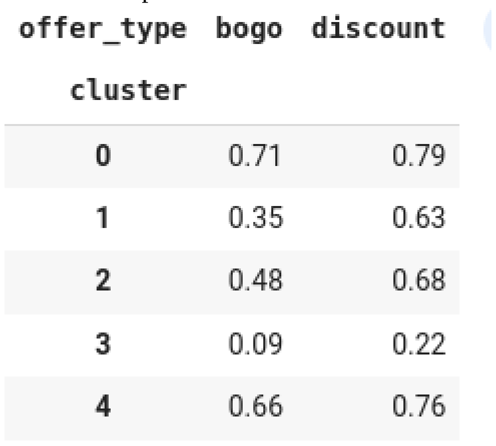
In general:

- Income was a prominent difference (1 unknown, 1 high, 1 low, 2 middle segments)
- Gender was the secondary difference
- Segments are mostly income/gender-based

Cluster-specific:

- Cluster 0: High-Earners (more than 80k income). All ages.
- Cluster 1: Low-Earners (under 50k income). All ages.
- Cluster 2: Middle-Earners Male.
- Cluster 3: The unknowns. Users who do not provide demo data.
- Cluster 4: Middle-Earners Female.

Next, I merged the responses calculated from the transcript data to the profile cluster data. There are no responses for informational type offers, so this information was ignored. I then viewed response rates for each cluster as seen below:



offer_type	bogo	discount
cluster		
0	0.71	0.79
1	0.35	0.63
2	0.48	0.68
3	0.09	0.22
4	0.66	0.76

This gave me the necessary information to make recommendations regarding the clusters.

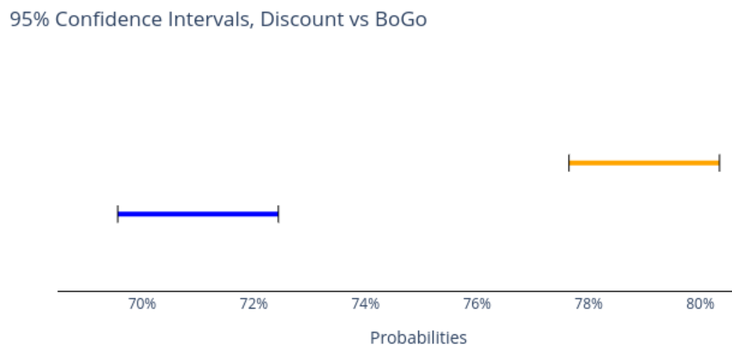
IV. Evaluation Metrics

The optimal centroids were determined by within-cluster sum of squares:

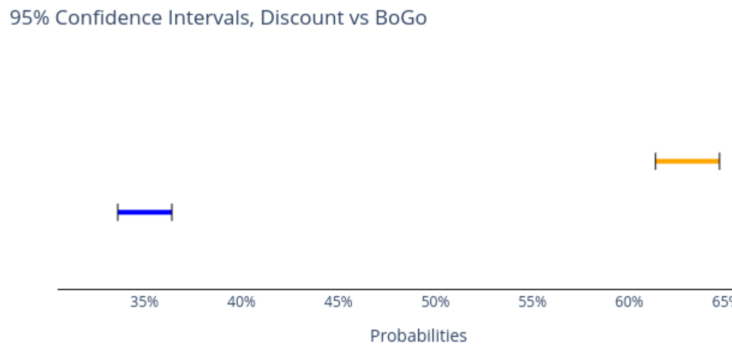
```
# determine optimal centroids using within cluster su
css = []
for i in range(20):
    kmeans = KMeans(i+1)
    kmeans.fit(profile_x)
    css.append(kmeans.inertia_)
```

The elbow method along with interpretability was used to select the optimal cluster number. I calculated the 95% confidence interval for each cluster's response rates for the two offers, they can be viewed below[2]:

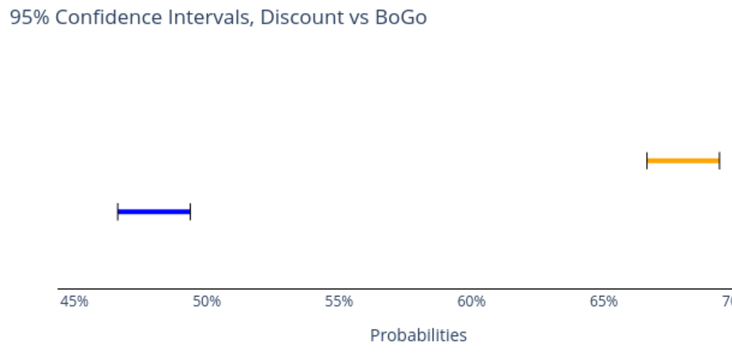
Cluster 0:



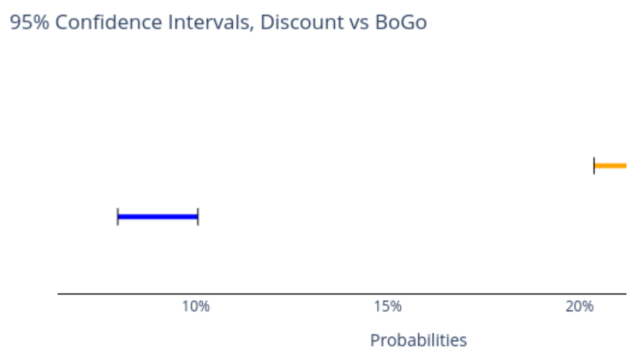
Cluster 1:



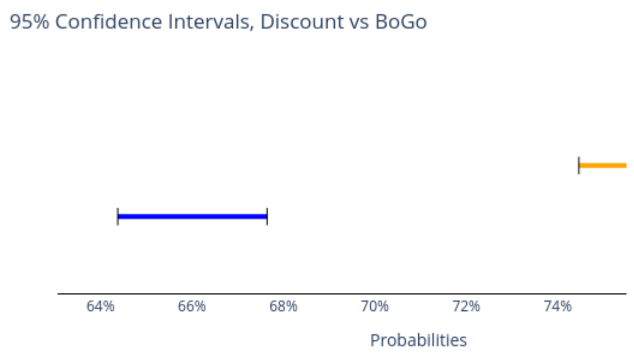
Cluster 2:



Cluster 3:



Cluster 4:



V. Conclusion

The recommendations I would make to Starbucks regarding their app marketing based on this data are as follows:

Cluster 0, the High-Earners - This app user group responded over 70% to both bogos (71%) and discounts (79%). It is by far the most successful group regarding bogos. With this group, I would recommend changing very little, they tend to respond well to both bogos and discounts.

Cluster 1, the Low-Earners - This app group responded to 35% of the bogo offers, but 63% of the discount offers. The inference here would suggest that low earners could also likely be single and bogo offers are generally less relevant as they don't often travel in pairs. My suggestion for this group would be to focus nearly entirely on discount offers rather than bogos.

Cluster 2, The Male Middle-Earners - The male middle earners responded to 48% of the bogo offers and 68% to the discount offers. I would suggest focusing on more discount offers to entice more visits, but I would not stop bogo offers, they are still effective, but could be released to this user group at a slower pace than the discount offers. I would also see if overall instore data was available to look at the types of orders men make, so that the discounts can be tailored to them.

Cluster 3, the Unknowns - This group was by far the least successful in responding as they responded to bogos at a 9% rate and discounts at a 22% rate. The response rate along with the fact they did not provide their demographics data suggests these users have very little involvement with Starbucks. My suggestion would be to entice them with a special discount to provide their demographics data first and foremost. Specialized campaigns to build more engagement with the app could also bring these fringe customers into the Starbucks fold.

Cluster 4, the Female Middle-Earners - This group is the 2nd most successful group responding to 66% of the bogos and 76% of the discounts. I would not change much with this group; although knowing it is female, you could tailor the offers to orders more females make, if that data is available.

Acknowledgment

I would like to acknowledge Kaggle for providing the dataset.

I would like to thank my wife, Sara Krasinski, for watching our 2 children and sacrificing time together, so that I could complete this project, I would not have been able to complete it without her.

I would like to acknowledge University of Illinois professor, Elham S. Buxton, for answering questions I had and providing guidance with this project.

Lastly, I'd like to acknowledge University of Illinois professor, Liang Kong, for his availability to answer questions and his one-on-one meetings to keep this project on track.

References

[1] Albon, Chris. Machine Learning with Python Cookbook: Practical Solutions from Preprocessing to Deep Learning. Japan: O'Reilly Media, 2018.

[2] S. K. Dash, "Understanding confidence intervals with python," *Analytics Vidhya*, 10-Feb-2022. [Online]. Available: https://www.analyticsvidhya.com/blog/2022/01/understanding-confidence-intervals-with-python/. [Accessed: 02-Nov-2022].

[3] F. K, "Starbucks app customer rewards program data," *Kaggle*, 03-Jun-2020. [Online]. Available: https://www.kaggle.com/datasets/blacktile/starbucks-app-customer-reward-program-data. [Accessed: 15-Sep-2022].