Capstone Project

Machine Learning Engineer Nanodegree

Mofetoluwa Adeyemi
August 31st, 2019

Starbucks Project (Customer Segmentation)

Definition

Project Overview

Customer Segmentation involves dividing a customer base into similar groups based on certain attributes the customers have. Companies and organisations perform customer segmentation to analyse, draw conclusions and make business decisions concerning their customer base.

Algorithms such as K-means clustering algorithm is very popular with customer segmentation and tend to fare well with clustering customers according to their RFM (Recency, Frequency and Monetary) attributes. Another well used algorithm is the Agglomerative Hierarchical clustering algorithm which can be applied to transactional data of customers.

This project focuses on performing customer segmentation on the customer base of the American beverage company, Starbucks. Customers are grouped based on demographic traits such as age, income and gender and their behaviour towards types of offers sent from the company. To perform the segmentation, 3 datasets containing the demographic information of customers, offer types and customer transactions were used with the Gaussian Mixture Models (GMM) clustering algorithm. Each cluster was analysed to view similarities.

Problem Statement

Starbucks would like to how certain customers react to offers sent to them. This project aims at identifying the groups of individuals that are responsive to these offers by performing the following tasks:

1. Download and Explore the data
2. Data Cleaning and Exploration
3. Feature Engineering and Dimensionality Reduction
4. Clustering the data using GMM
5. Selecting the best clustering algorithm
6. Extracting the trained model attributes and visualizing clusters
7. Stating the observations of each cluster

Every customer is assigned to a cluster. These clusters have attributes that is used to analyse and make decisions on the customer-offer relationship.

Metrics

The average Silhouette score is used as a measure of accuracy of clustering. The silhouette score indicates how close a sample is to its own cluster compared to other clusters. The best value for a silhouette score is 1 and the worst is -1.

$$silhouette\ score\ = \frac{(b - a)}{max(a,\, b)}$$

Where a is the mean intra-cluster distance and b is the mean inter-cluster distance.

The silhouette score is used to measure the performance of each of the clustering algorithm by calculating the average silhouette score of the samples in clusters formed by the algorithm. In other words, in this project it is used to compare the performance of the model used in the project and the benchmark model.

Analysis

Data Exploration

Three (3) datasets were used in this project.

1. profile.json: This contains demographic data about the rewards program users. There are 17000 users and 5 fields which include:
    - gender: (categorical) M, F, O, or null
    - age: (numeric) missing value encoded as 118
    - id: (string/hash) The customer id. Same values as those in the ‘person’ feature of the transcript.json dataset.
    - became_member_on: (date) format YYYYMMDD
    - income: (numeric)

    This dataset contained 2175 missing values in the gender and income features. This was taken care of using imputation.
    Also since customer segmentation is the task at hand, profile.json dataset formed a base for the final dataset to be used.
2. portfolio.json: This contains the offers sent during the 30-days simulation period. There are 10 offers and each has 6 fields which are:
    - reward: (numeric) money awarded for the amount spent
    - channels: (list) web, email, mobile, social
    - difficulty: (numeric) money required to be spent to receive reward
    - duration: (numeric) time for offer to be open, in days
    - offer_type: (string) bogo, discount, informational
    - id: (string/hash)

    There were no missing values in this dataset.
3. transcript.json: This is the transactional data that shows the events such as when a user views, receives or completes an offer. It contains 306648 events and 4 fields which are:
    - person: (string/hash)

- event: (string) offer received, offer viewed, transaction, offer completed
- value: (dictionary) different values depending on event type
- offer id: (string/hash) not associated with any "transaction"
- amount: (numeric) money spent in "transaction"
- reward: (numeric) money gained from "offer completed"
- time: (numeric) hours after the start of test

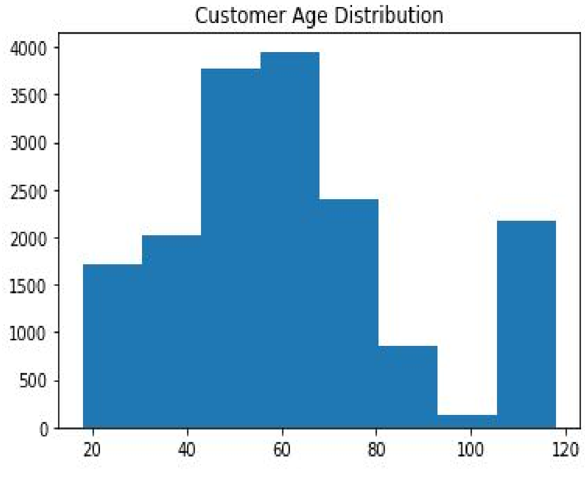
There were also no missing values in this data.

To form a more detailed dataset, the transcript.json dataset is merged with the profile.json dataset on the ‘id’ and ‘person’ features. Then more relevant features are created and a groupby operation is performed on the dataset. This is explained in more detail in the **Data preprocessing** section.
These datasets were provided by Udacity.

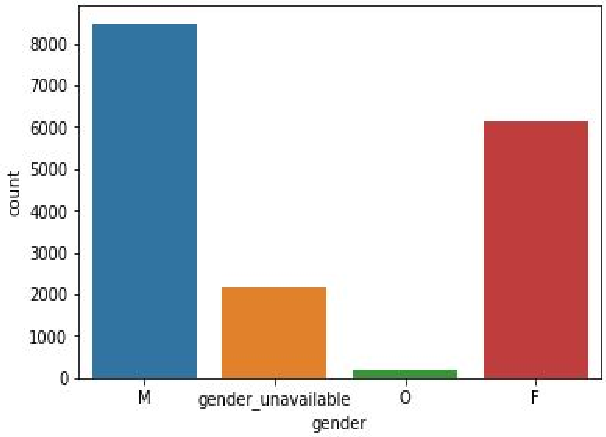
Exploratory Visualization

Data preprocessing is done to create new features for customers such as total amount spent, total number of purchases, total number of offers received, amongst others. This features were created for better visualization and analysis. The steps taken to create them are explained in the Data Preprocessing section.
On getting the final merged dataset we examine the following:

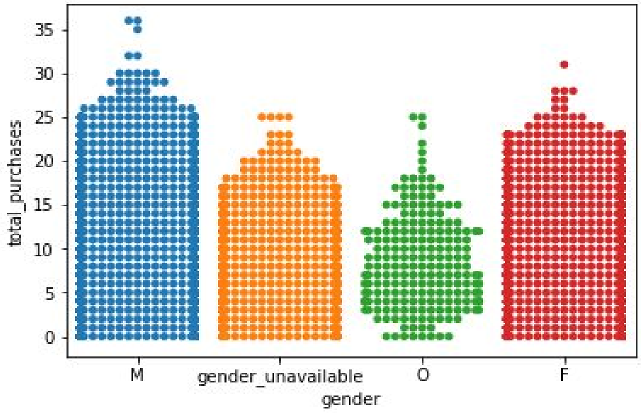
1. The age distribution and gender distribution of the customers. From the histogram below, we see that most customers are within the ages 40 to 60.



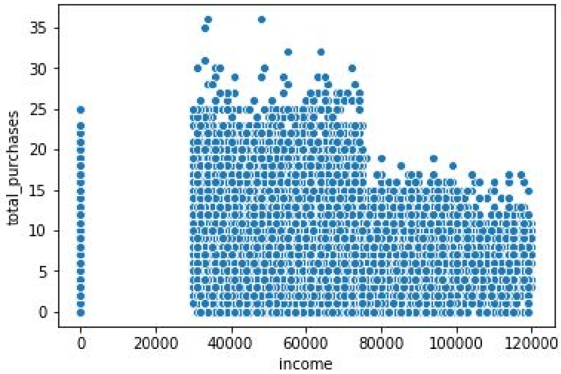
The count plot below shows that there is a huge percentage of customers who are male.

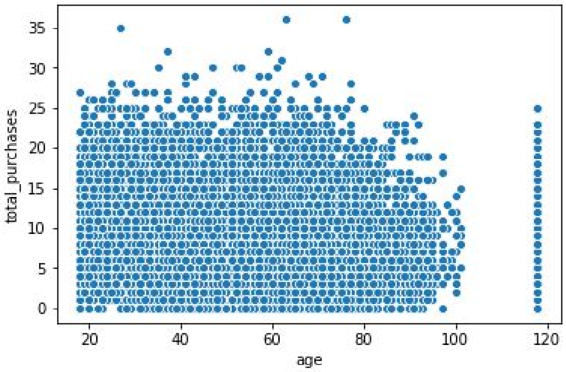


2. Total purchases based on gender, income and age. From the swarm plot of the total purchases and gender features, we observe that the highest number of purchases were made by the male gender. A lot of purchases were also made by those with missing gender values.

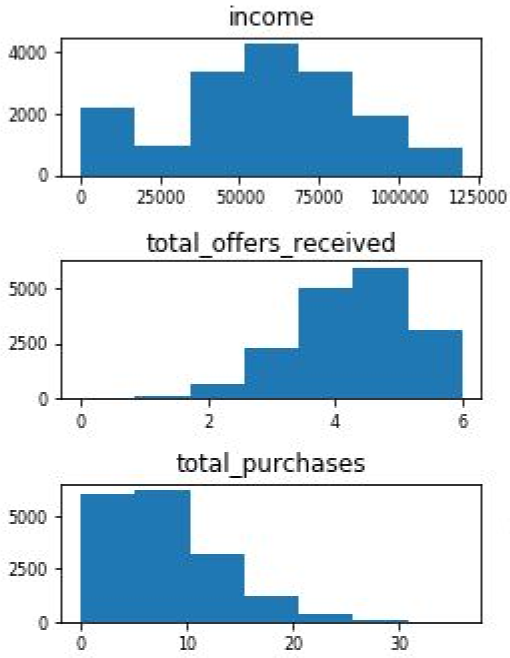


The scatter plots below shows the highest number of purchases were made by those between 25-80 years of age and those earning between 30,000 to 60,000.





3. Distribution of customers income, total purchases, total offers received, total offers viewed, total offers completed and total spend.



The following observations are made from the histograms:

- Most customers earn between 50000 and 75000
- Most customers received between 4 and 5 offers
- Most viewed between 3 and 4 offers
- Most didn't complete offers.
- Most customers made between 5 to 10 purchases

Algorithms and Techniques

1. Gaussian Mixture Models (GMM) for clustering the data.
   The clustering algorithm used in this project is the Gaussian Mixture Models (GMM): The Gaussian Mixture Model , an unsupervised learning algorithm, gives more clustering flexibility than the K-Means algorithm. GMM models each cluster with a different Gaussian distribution which results in the soft assignment of data points to clusters i.e.

each data point is assigned to a cluster based on probability, unlike k-means which does hard-assignment i.e. the point is either in the cluster or not. This is because K-Means defines each cluster by the mean, but clusters in GMM are defined by the 2 Gaussian parameters, mean and covariance. Clusters formed by the GMM algorithm come from different Gaussian Distributions. Therefore it tries to model the dataset as a mixture of several Gaussian Distributions.

For multivariate data (n-dimensional), the Probability Density Function of a Gaussian distribution is:

$$G(X|\mu,\ \Sigma)\ =\ \frac{1}{\sqrt{2\pi\ |\Sigma|}}\ exp\ (-\tfrac{1}{2}(X\ -\ \mu)^T\Sigma^{-1}(X\ -\ \mu))$$

Where μ is an n-dimensional vector denoting the mean of the distribution and Σ is the $n\ \times n$ covariance matrix.

Since there would be K clusters, the probability density is defined as a linear function of densities of all these K distributions, i.e.

$$p(X)\ =\ \sum_{k=1}^{K}\pi_k\ G(X|\mu_k,\ \Sigma_k)$$

Where π_k is the mixing coefficient for the k-th distribution.

Given a dataset and the number of required components or clusters, the GMM algorithm makes use of an optimization algorithm known as Expectation-Maximization (EM) to find the parameters (i.e the mean and covariance) of the Gaussian of each cluster. Using the parameters for each cluster, the probability that a data point belongs to that cluster is then computed iteratively until there is convergence.

The parameters used to tune the GMM model in this project are:

- Number of mixture components: This value represents number of clusters.
- Type of covariance: Can be full, spherical, tied or diagonal
- Maximum Iterations
- The number of initializations to perform

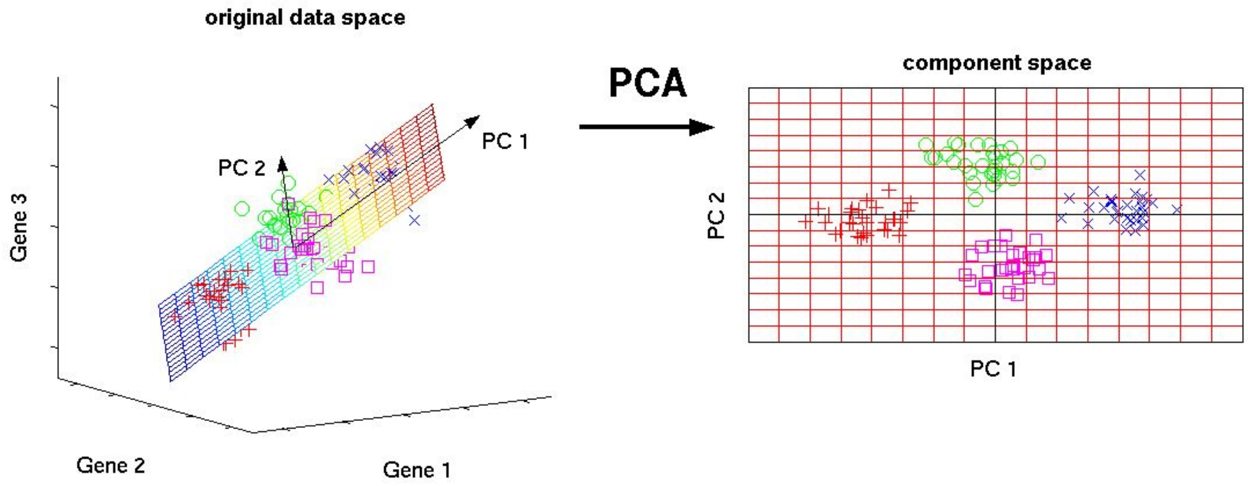
Relevant attributes produced by the model include:

- Weights of each mixture components
- The mean of each mixture component. This is also likened to the centroid of a cluster.
- The covariance of each mixture component

The algorithm is fit to and predicts the processed data to give clusters of similar data points.

2. Principal Components Analysis (PCA) for Dimensionality Reduction.

The PCA algorithm is used for dimensionality reduction in order to easily deal with and cluster high-dimensional data. Principal components are created which are linear combinations of the original features. An illustration of this is seen below where 2 principal components are obtained from 3-dimensional data.



From the image above, the data points in the original data space seem related; they fall close to a 2D plane, and just by looking at the spread of the data, we can see that the original, three dimensions have some correlation. So, we can instead choose to create two new dimensions, made up of linear combinations of the original, three dimensions. These dimensions are the Principal Components and are represented by the two axes/lines, centered in the data.

The components which account for the highest data variance are used for the clustering. Parameters taken into consideration include:

- Number of components
- Seed used for random number generation.

The model attributes include:

- Principal axes in feature space i.e. the Principal components and their respective linear combinations of features.
- Explained variance of each component
- Singular values of each component

The components which account for at least 80% variance of the data was chosen.

Benchmark

The Benchmark model used is the K-means clustering algorithm. Silhouette scores for the same number of clusters in the K-means and Gaussian Mixture Models are compared and the Gaussian Mixture Model is expected to have a higher silhouette score.
Both silhouette scores obtained should be above 0.5 as a threshold.

Methodology

Data Preprocessing

Preprocessing of the dataset was done using the following approach:

1. Merging the Datasets.
   Since the task at hand is Customer Segmentation, a dataset containing customer demographic information as well as transactional information is needed to analyze how demographic traits relate to offer type responses. Therefore the transcript dataset was merged with the profile dataset on the ‘person’ and ‘id’ features of the respective datasets. This produced a ‘profile_transcript’ dataset where each event had associated customer attributes. The ‘portfolio’ dataset wasn’t merged as not much of its information was needed at that point.

   A little bit of Preprocessing is done on the ‘value’ feature of the new profile_transcript dataset. The values for the keys ‘offer_id’ and ‘amount’ in the dictionaries were extracted and saved in a new feature called “unzipped _values”.

2. Grouping the Dataset by Customers.
   The dataset created above only gives information on the customers attached to each event. To make customer segmentation easier, there is a need to create a dataset containing features per customer, hence the need to group the dataset by customers.
   To do this, the following steps are taken:
   a. Creating new features.
      In order to perform customer segmentation, features that tell more about each customer is created. These features give transactional and offer information about each customer. The features include:
      - ❖ Number of purchases made
      - ❖ Number of offers received
      - ❖ Number of offers viewed
      - ❖ Number of offers completed
      - ❖ Total amount spent
      - ❖ How each customer relates to each offer i.e was the offer received? was the offer viewed? This in particular would generated 30 new features (3 for each type offer which represents if the offer was received, viewed and completed)

      In total, 35 new features were created.

   b. Dealing with missing values using Imputation
      The missing values in the ‘gender’ and ‘income’ features were imputed to avoid losing customer information when performing the groupby operation. Missing values in the ‘income’ feature were imputed as 0.0 and those in the ‘gender’

feature were imputed as ‘gender_unavailable’. The imputation was done using SimpleImputer from Scikit learn’s .impute class.

c. Grouping the Dataset by Customer Id and Demographic Information
Next, the dataset is grouped by the customer id and demographic information and the sum of each of the new columns is calculated for each customer. This would give a new dataset having only the features needed.

3. Dealing with Categorical Features

The only categorical feature left was the 'gender' feature. I dealt with this using One Hot encoding and Column Transformer of the Scikit Learn library.

4. Normalizing Numerical Features using Min-Max Scaler
Numerical features are also normalized in order to consistently compare the values of different features. This leaves us with a scaled dataset.

5. Dimensionality Reduction using PCA.

Data preprocessing left the dataset with 40 features. Since the dimension of the data is very high, PCA is used to perform dimensionality reduction to find smaller sets of features that help separate the data.
The number of principal components instantiated for the model was 30.
The scaled dataset (obtained from the last data preprocessing step) is fit to the PCA model and the model attributes are examined to determine the top components that account for at least 80% variance. To determine the number of top principal components that account for 80% variance, the explained variance of the components with the top singular values is calculated.

$$Explained\ Variance = \frac{The\ sum\ of\ squared\ s\ values\ for\ all\ top\ n\ components}{sum\ of\ squared\ s\ values\ for\ all\ components}$$

$where\ s = singular\ values\ of\ components$

From the explained variance attribute, it is observed that the top 11 to 12 components account for up to 80% of the data variance. Therefore the 12 top components are kept.

Then the dataset is transformed using the PCA model and the top 12 components are kept. The transformed data is stored in a new dataset.

Implementation

- Step 1: Creating a Gaussian Mixed Model

Using Scikit Learn’s GaussianMixture class, the Gaussian Mixed Model is defined using the number of components and covariance type hyperparameters. From the k-means model, the optimal number of clusters defined by the silhouette score is 3. Therefore, 3 is set as the number of components for the GMM and the covariance type is varied. Possible covariance types include Spherical, full, diagonal and tied. The covariance type is set to default which is ‘full’.

- Step 2: Call the fit_predict method on the model created using the transformed dataset. The fit_predict method serves as a two-in-one process. It fits the data to create clusters and predicts the cluster each data point belongs. In lieu of this, the transformed dataset obtained from the PCA model is then fit into the model and predicted. The result is an array of the cluster for each data point.

- Step 3: Then average Silhouette score is calculated using the clusters generated from GMM model and the transformed dataset.
  On obtaining the silhouette scores for the k-means and the GMM algorithm, there is not much of a difference in both scores which is not the expected results. The values are stated below:

Algorithm	**Silhouette score**
K-means	0.40328860761643825
GMM	0.40278647052464367

From the results above, the challenge faced is that the GMM is not performing as it is expected to. Infact, it is performing lower than the benchmark model, K-means.

Refinement

To improve the performance of the GMM model, we take the following into consideration the following hyperparameters;

- The covariance type
- Maximum iterations
- Number of initializations
- Number of params

Examining the silhouette score corresponding to each covariance type:

Covariance Type	**Silhouette score**
Diagonal	0.40278647052464367
Tied	0.40278647052464367

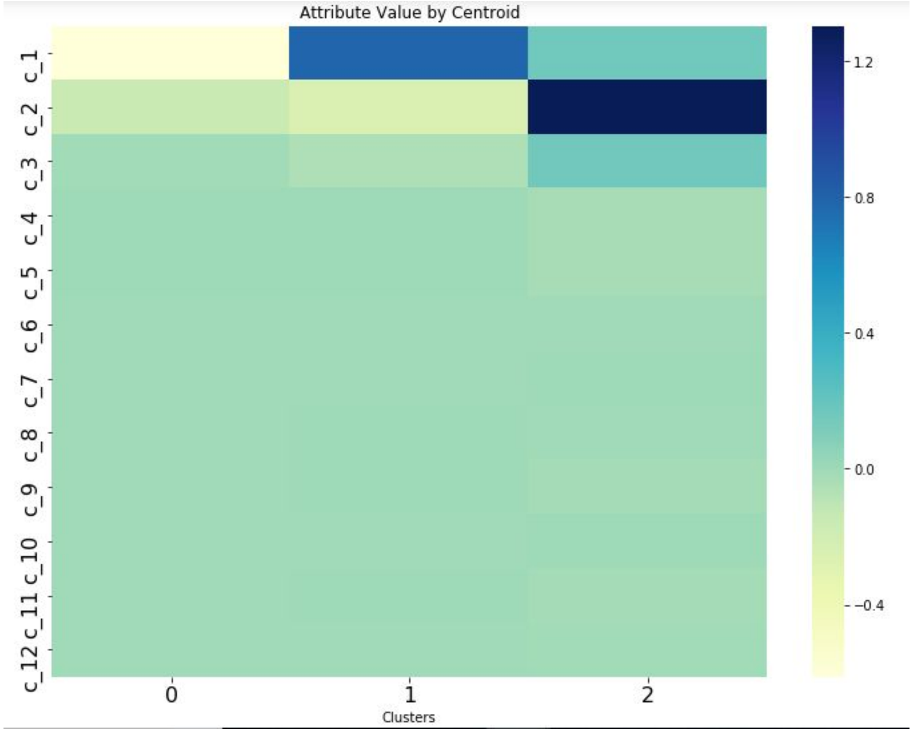
Spherical	0.40334750093084665
Full	0.40278647052464367

The spherical covariance type is the only covariance which gives a score slightly higher than that of the K-means clustering.
On using the rest of the hyperparameters, there was no increase in the value of the silhouette score.

Results

Model Evaluation and Validation

Extracting some of the model attributes can assist with understanding the different clusters. The natural groupings of the data points can be done by mapping the GMM cluster results to the main dataset to know the cluster a customer belongs to. A great advantage GMM has is to give the probability a point has in belonging to a cluster. The predict_proba_ method of the GMM model is used to determine these probabilities and these probabilities can be mapped to the major dataset.
First we take a look at the heatmap showing the relationship between the components of each cluster.

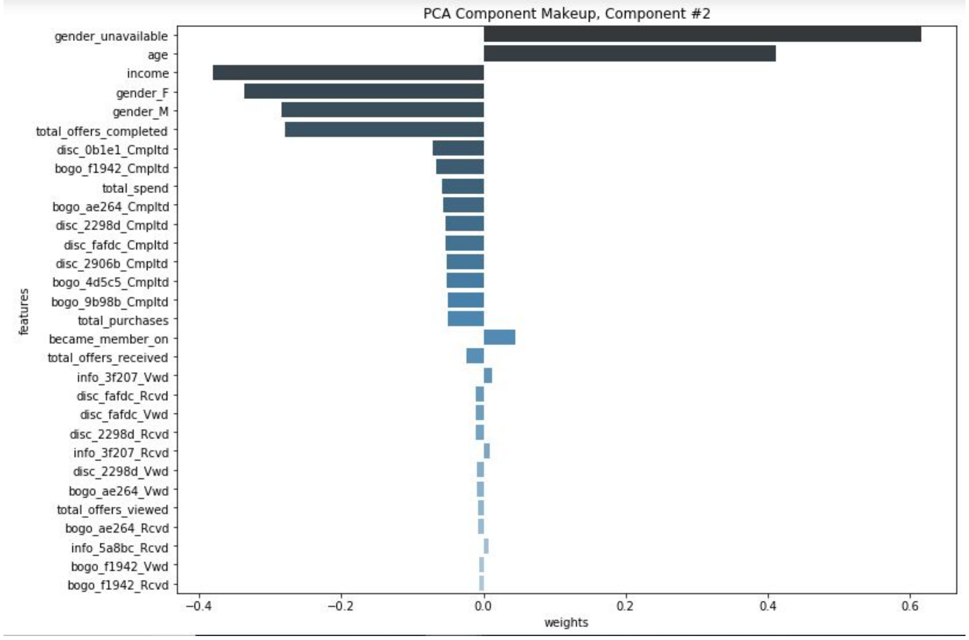


From the heatmap above, we see that:

- Cluster 0 is very low in component 1 and 2.

- Cluster 1 has a very high value for component 1, and low for component 2.
- Cluster 2 has a very high value for component 2, followed by components 1 and 3
- The variance between components 4 down to 12 isn't much at all.

Recall that each component is made up of the initial 40 features. We can examine the weights of features of each component using the components_ attribute of the PCA model created earlier. Taking a look at components 2;



From the bar plot above, it can be observed that the 'gender_unavailable' feature has the largest weight. Therefore mapping back to the heatmap, we would see that most data points in cluster 2 have the feature 'gender_unavailable'. From our observation, the same goes for clusters 0 and 1; customers in cluster 0 are only 'male' and 'O' while those in cluster 1 are only 'female' and O. It seems the clusters were formed based on gender. This is not the exact nature of clustering that is been searched for.

Justification

The GMM algorithm performed slightly better than the K-Means algorithm, but not well enough to map out the meaningful clusters needed for the type of data processing done on the data. Therefore, not much specific offer-customer information or observation can be drawn from the clusters, except more general information on how a person's gender affects response to an offer.

Conclusion

A lot of improvements can be done to boost the clustering of this data. This could be taking another approach to preparing data for customer segmentation and/or using a clustering algorithm that works well with data of this nature.