IMDB clustering project

Sergio Navia, Erik Vela, Jason Wo

Background:

The dataset we explored in our project is “IMDB Top 1000 Movies”, which consists of movies, the observations, on the website IMDB which aggregates movie ratings and other attributes. The information gathered on IMDB gives viewers an idea of how much others are enjoying a particular movie, and informs their decision to watch the movie as well. The goal of our assignment was to cluster the movies into some number of clusters, so that the specific movies within each cluster were as similar to each other as possible, while still being able to interpret what the relationship among them was. In a real-world context, similar to the information given on IMDB, this would help inform viewers on their movie-watching decision if they chose to watch other movies in the same cluster as ones they have previously watched and enjoyed.

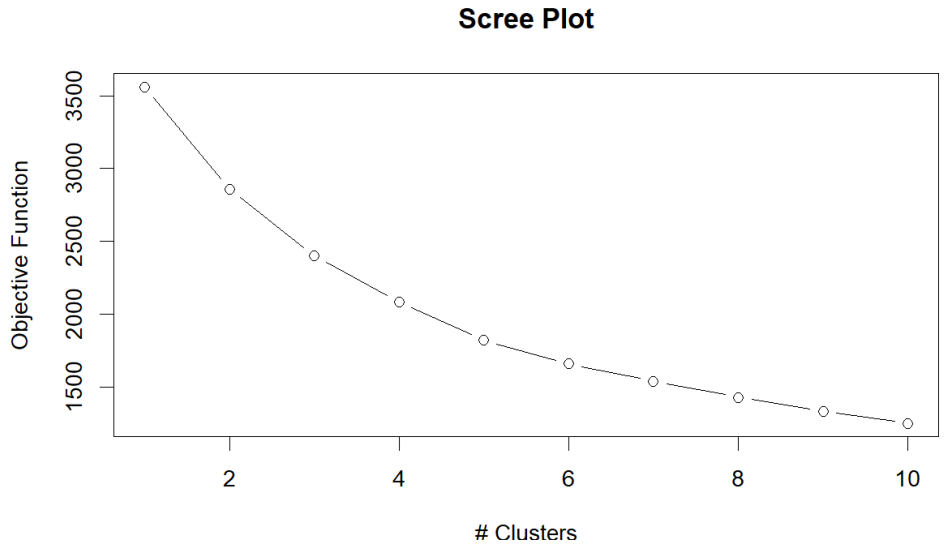
EDA and Feature Engineering:

The original dataset had 16 variables. The unusable variables are Poster Link, Overview, Director, Star 1, Star 2, Star 3, and Star 4. We removed Certificate because it was difficult to turn the ratings into an equal and measurable variable due to their being different rating systems in different countries. We also removed number of votes variable because it was skewed toward newer movies. Genre had 21 unique names and many movies had more than 1 genre. Thus we made each genre a binary variable and added it to our dataset.

During our preliminary cluster analysis, with all the different genre variables in the dataset, the scree plot and silhouette statistic gave 2 clusters as our optimal number of clusters. When we cluster with k=2, the data was only clustered into high-grossing movies in one cluster and low-grossing movies in the other. Since this clustering really didn't tell us anything new about the dataset, we decided to take off all the binary variables and only have our dataset consist of released year, runtime, IMDB rating, Meta score, and gross.

Determining the number of clusters(Scree Plot):

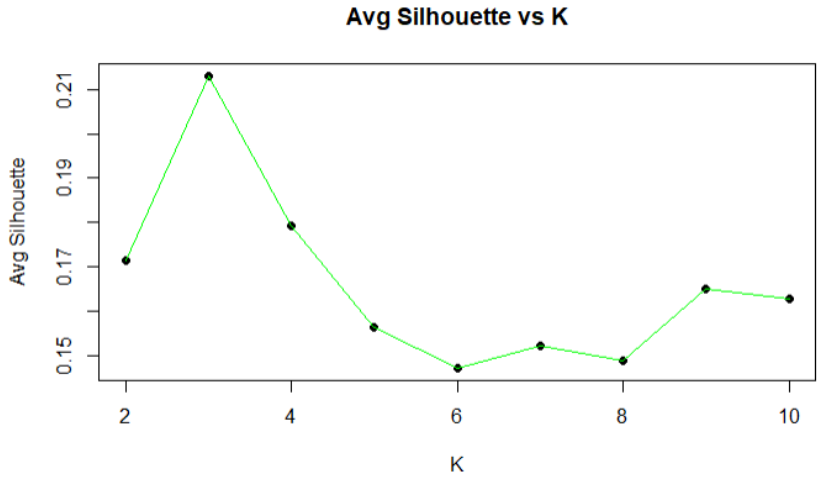
Before performing K-means or other clustering algorithms such as hierarchical clustering. We need to decide the number of clusters that we should use. First, we generated a scree plot for K-means:



From the plot we see that it is hard to decide which cluster to use at first, any cluster from k=2 to k=6 is reasonable based on the plot. After further examination of the plot, k=3 has the best elbow compared to the other clusters. However, we think the process of selecting the number of clusters from Scree Plot is too subjective, it is mostly based on our judgment, thus, we will use another statistical method to help us decide what number of clusters will be used.

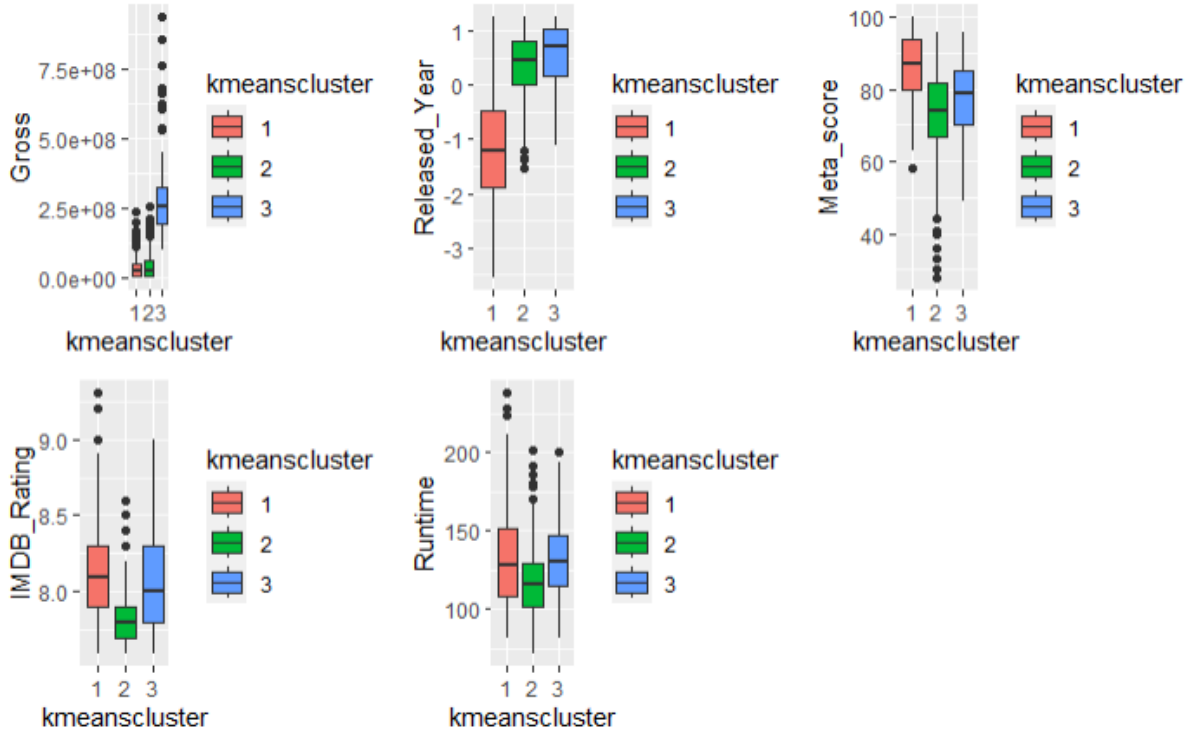
Determining the number of clusters(Silhouette Statistic):

We will be using the Silhouette method to decide the number of clusters. We obtained the average Silhouette score and then generated a plot to visualize:



From the plot, we can see that at k=3, the average Silhouette score reaches its maximum in the range of k=1 to 10. The higher average Silhouette score indicates the better the data is being clustered. Thus, from the Silhouette method, our optimal number of clusters will be 3. Hence we will perform k-means clustering with k-3.

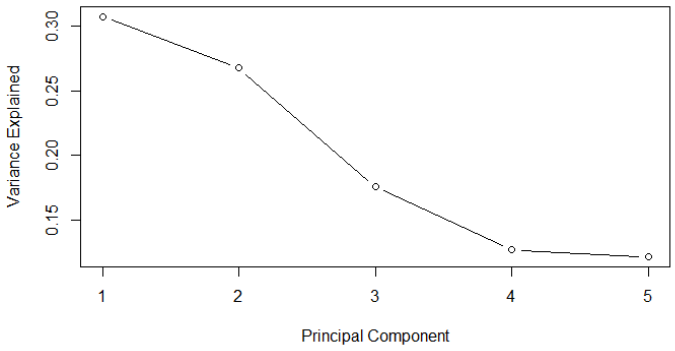
K-means clustering with k=3



The first cluster generated by k-means contains movies that have older release years, their IMDB rating and Meta_score are higher, but they have a lower gross income. We would conclude cluster 1 to be old classic movies where they tend to have high ratings but lower gross income due to differences in ticket value in the past vs. now. In cluster 2, the cluster contains movies that have relatively new release years, lower ratings in general compared to the other two clusters, and also a lower gross income. We conclude cluster 2 to be movies that are movies released in the past few decades, meaning newer release years, but they are not popular compared to the movies in cluster 3. In cluster 3, the cluster contains movies that have newer release years, and higher ratings on average lower than Cluster 1, but significantly higher gross income compared to the other two clusters. We conclude cluster 3 to be the blockbuster movie that is newly released with high ratings and very popular.

Principal Component Analysis:

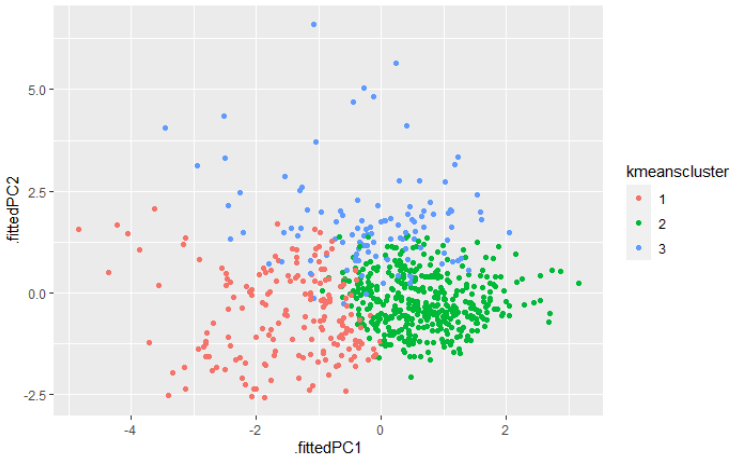
The principal component analysis we did helped us decide which portion of the data accounted for most of the variation in the set. Because principal components are linear transformations of the original variables in descending order of variation, we can account for most of the data variation by looking only at the first few principal components. The scree plot of the components shows us how much less variation is explained from one component to the next. We can see that our first “elbow” is at the third principal component, meaning that adding more principal components would add little explanation to the variation , while making the model less interpretable. Because we are only working with 5 numeric variables in our clustering algorithms, using more principal components seems unnecessary since we are trying to explain more with less coefficients. Since about 75% of the variation of the data can be explained by the first 3 components, this choice makes sense.

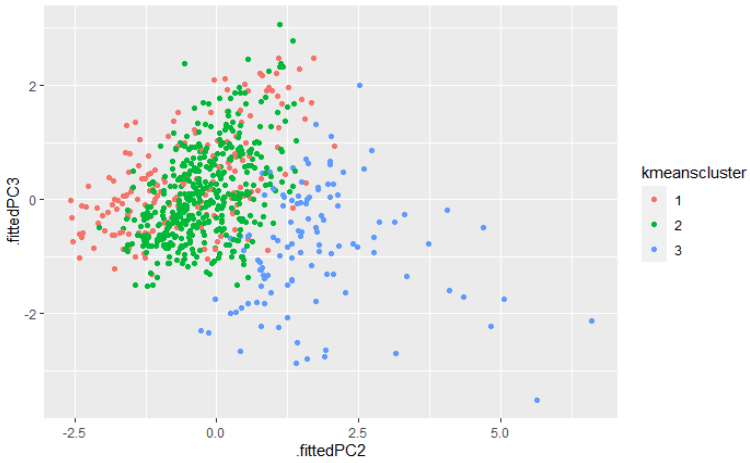


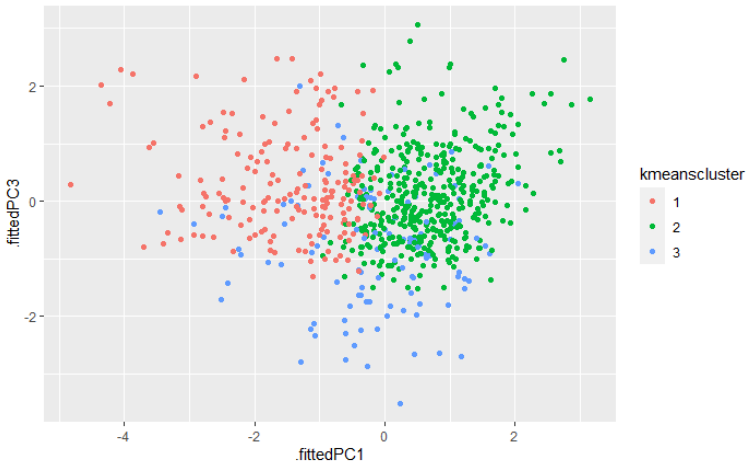

```
Standard deviations (1, .., p=5):
[1] 1.2395715 1.1568047 0.9382903 0.7972365 0.7805708

Rotation (n x k) = (5 x 5):
                     PC1        PC2         PC3         PC4         PC5
Released_Year  0.4880631  0.4417639 -0.25208172 -0.62759340  0.3304852
Runtime       -0.3010805  0.5112006  0.65381430  0.10421520  0.4579198
IMDB_Rating   -0.5993749  0.2606866 -0.02676844 -0.54696280 -0.5224054
Meta_score    -0.5584336 -0.1774663 -0.53837548 -0.02427944  0.6051611
Gross         -0.0068283  0.6663834 -0.46735101  0.54360641 -0.2048453
```

Looking at the coefficients that measure the PC value for each observation, we can see that movies with high PC1 would likely be newer , shorter runtime, and have lower ratings. These movies tended to be 90's to 00's dramsa that were moderately well received and generally. Movies with high PC2 were also fairly new, but had longer runtime and much higher gross. These movies tended to be crowd-pleasing , blockbusters that a wide range of audiences usually enjoy. Movies high in PC were usually differentiated by being older and low grossing niche films.

In order to determine how well our clusters separated the movies based on the variation involved in the set , we constructed plots between Principal Components colored by which K-Means cluster they are in.


5.0
2.5
0.0
-2.5
.fittedPC2
-4
-2
0
2
.fittedPC1
kmeanscluster
1
2
3


2
0
-2
.fittedPC3
-2.5
0.0
2.5
5.0
.fittedPC2
kmeanscluster
1
2
3


2
0
-2
.fittedPC3
-4
-2
0
2
.fittedPC1
kmeanscluster
1
2
3

By plotting the PCs against each other and visualizing how the K-Means algorithm separated the movies, it shows that this clustering method worked pretty well in terms of explaining the variance involved. Movies highest in PC1 were all in cluster 2, movies highest in PC 2 were all in cluster 3 , and movies high in PC 3 were a bit more mixed but tended to be in Cluster 1.

Conclusion:

Since Clustering is an unsupervised task we did not really know what to expect while running these algorithms. Our most important takeaways from this project were not just our results, but what we encountered in the process. Data standardization was very important given how different in scale the variables were . Before accounting for the different units, the clusters were almost totally determined based on the Gross variable. Additionally, our feature engineering of the Genres into several binary variables also dominated the clustering process. Because we did not want this to be a classification task in terms of Genre, we chose to omit Genre altogether. The results of the PCA showed us that the K-Means algorithm did a good job of clustering the variables based on how differently they explain the data variance. In the future we will use the pitfalls and successes we encountered here to create a model that helps improve movie lover's experience.