Data Mining Final Project

Yiqing Chen, Jiusi Li, Xingyu Wang, Xueqing Wang

Abstract

This paper proposes a movie recommendation engine that employs two techniques: clustering and collaborative filtering, both of which are based on the user's past viewing history. The first approach uses dimensional reduction to cluster movies and recommends those in the same cluster as a user's viewed movies. The second approach employs collaborative filtering methods, leveraging the wisdom of the crowd and past rating information to predict the top five movies users are likely to enjoy. The system can be used by movie recommendation websites to improve user experience and engagement.

Introduction

In recent years, movie recommendation systems have become increasingly popular and essential for streaming platforms to enhance user experience and retain customers. It helps the entertainment industry to create personalized movie recommendations, content discovery, cross-selling, etc.

In this project, we aim to use clustering methods to explore similar movie attributes and build a collaborative filtering movie recommendation engine that incorporates rating scores to make personalized recommendations. We will use clustering techniques such as TSNE and Kmeans to better understand the movie dataset and identify similar movies. We will also integrate user ratings to build a collaborative filtering recommendation system that provides personalized movie recommendations based on past user behavior. This paper will explore the process of data preparation, visualization, clustering analysis, and collaborative filtering.

Paper Review and Model Approach

Clustering Methods:

K-Medians

K-Median clustering is a type of unsupervised machine learning algorithm used to group similar data points together in a given dataset. It is similar to K-Means clustering, with the difference being that K-Median clustering uses the median value of each cluster to calculate the centroid, whereas K-Means uses the mean value.

The paper suggests that K-Medians is a useful alternative to K-Means clustering in situations where robustness to outliers is important and the data is not well-suited for the assumptions of

K-Means. However, K-Medians may not be the best choice for large datasets or data with non-Euclidean distance metrics.

K-Means

One commonly used clustering algorithm is the K-Means algorithm, which iteratively partitions a dataset into K clusters by minimizing the sum of squared distances between the data points and their assigned cluster centers. The algorithm starts by randomly selecting K initial cluster centers from the dataset. Then, it assigns each data point to the closest cluster center based on the Euclidean distance between the data point and the center. After all data points are assigned, the algorithm updates the cluster centers by calculating the mean of all data points in each cluster. This process is repeated until convergence, which is when there is no further improvement in the sum of squared distances.

In terms of the challenges and limitations of the K-Means algorithm, one of the main challenges is choosing the optimal number of clusters. To overcome this, Jin & Han suggest several methods such as the elbow method, the silhouette method, and the gap statistic method. Overall, the K-Means algorithm is a widely used clustering technique that is simple and computationally efficient, but requires careful consideration of the initial cluster centers and the optimal number of clusters.

Dimensional Reduction

PCA

Hotelling proposed PCA methods in 1993. It is a dimensional reduction method that can linearly reduce the number of features while retaining the majority of the information. This method identifies the direction of maximum variance in the data and projects the data onto these axes to obtain a lower-dimensional representation of the data. It is an efficient method for doing dimensional reductions; however, it can only capture linear relationships and hard to interpret the resulting lower dimensional features (principle components)

T-SNE

T-SNE, short for T-distributed Stochastic Neighbor Embedding, is an unsupervised Machine Learning approach developed in 2008 by Laurens van der Maaten and Geoffery Hinton. It is one of the most popular algorithms for visualizing high-dimensional data and reducing dimensionality on non-linear separable datasets. The algorithm identifies neighbor points by creating a probability distribution that represents similarity between neighbors. For example, the similarity of the datapoint X_i to the datapoint X_j is the conditional probability $P_{(i|j)}$, that X_i would pick up X_j as its neighbor point.

T-SNE uses gradient descent with Kullback-Leibler divergence as the cost function and repeats a number of times chosen by the user to find the converged solution. Another important hyperparameter for users to choose is perplexity, which usually ranges from 5 to 50. It roughly determines the number of neighbors for each central point. So, higher perplexity indicates higher value variance.

Unlike PCA, T-SNE can deal with data that is non-linearly separated. It potentially has more application in the real world as many real-world datasets contain non-linearity. However, one of the drawbacks is that T-SNE does not generate deterministic solutions, which means that whenever new data comes in, it is necessary to recompute the solution of T-SNE.

Recommendation System

Recommendation Engine (User-based and item-based)

recommendations based on past data, which assumes that people with similar purchasing behaviors will have similar preferences. There are two primary types of collaborative filtering systems: user-based and item-based.

In a user-based collaborative filtering system, users with similar ratings for a product in the past are identified, and recommendations are made to a user based on the preferences of similar users. However, this method has some drawbacks due to sparsity and scalability issues.

To address these issues, Sarwar et al. (2001) proposed an item-based collaborative filtering recommendation algorithm. This method assumes that items that are frequently rated together by users are likely to be similar and makes recommendations based on those similarities.

Today, both algorithms are popular in different industries for creating collaborative filtering recommendation systems.

Data Analysis and Model Development

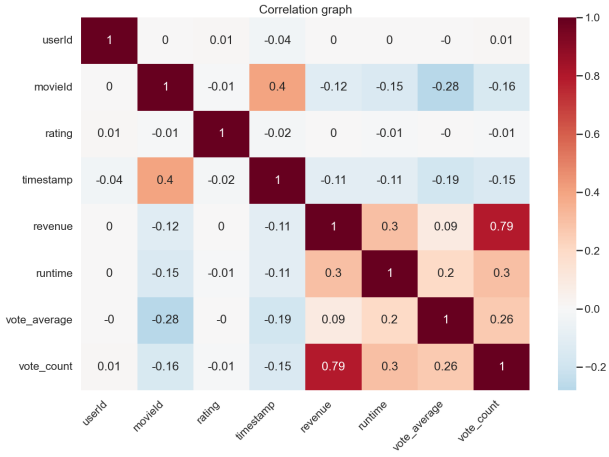
Data

The movie dataset contains a set of attributes for each movie, such as its title, genre, as well as user ratings for each movie. Using these attributes, clustering analysis is performed to group movies based on their similarities and differences.

Furthermore, the user ratings data is used to build a collaborative filtering system, which can generate personalized recommendations for users based on their past viewing history and preferences. By combining these analytical approaches, a deeper understanding of the movie industry can be gained and users can get movies meeting their interests through the recommendation engine.

Explorative Data Analytics

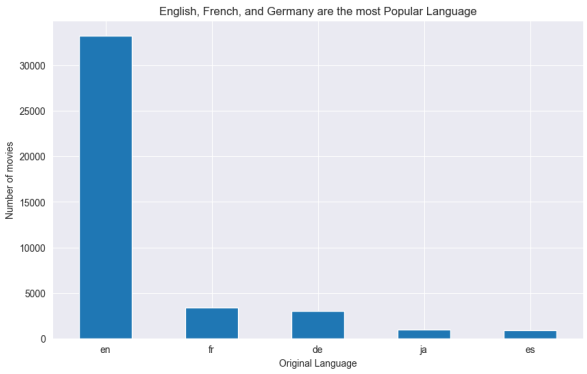
The basic explorative analysis is done to better the understanding of the dataset.

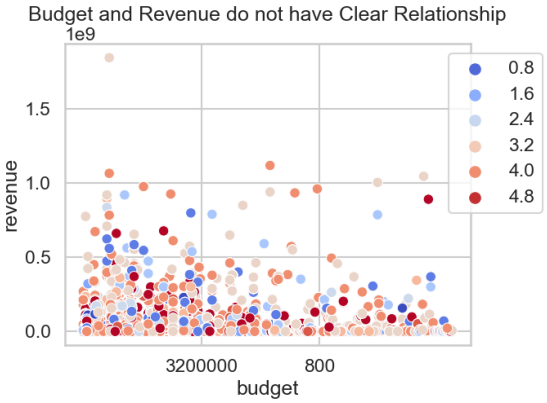


After examining the dataset's title, we discovered that words such as "Million", "Dollar," "Terminator", and "Rise" frequently occur, providing insights into the most popular genres among audiences.

While the features in the dataset are not generally highly correlated, we did find a positive correlation of 0.79 between revenue and vote count, indicating that films with higher revenues may have a wider audience but not necessarily higher ratings.

Our scatter plot confirmed this hypothesis, revealing that movie ratings are not strongly correlated with either budget or revenue. This finding is surprising and suggests that people's preferences are influenced by factors other than a movie's financial success.





Furthermore, we found that English, German, and French are the most commonly spoken languages in the dataset. We created dummy variables for the majority language classes and ignored the other language levels to avoid the curse of dimensionality.

Feature transformation

To ensure data cleanliness and expandability, we merged the movie information with user rating data, allowing for a more comprehensive analysis that considers users' perceptions of movies.

Furthermore, we created dummy variables for movie production country and genre on the majority levels to preserve categorical information for clustering tasks. This is particularly important since some clustering methods cannot handle categorical features.

To improve the accuracy and convergence speed of clustering algorithms, we standardized all the predictors. This ensures that differences in feature magnitude do not unduly influence the clustering results, particularly in distance-based algorithms like K-means.

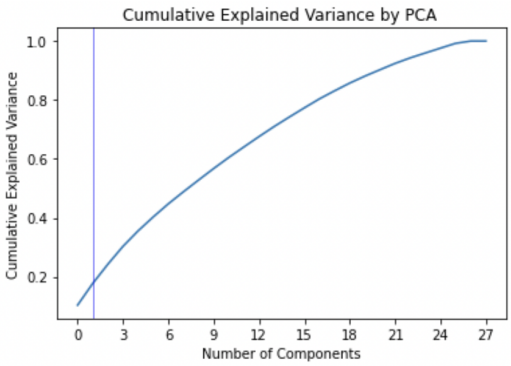
Overall, these steps represent important data preprocessing techniques that will facilitate more accurate and insightful clustering analyses in the future.

Dimensional Reduction

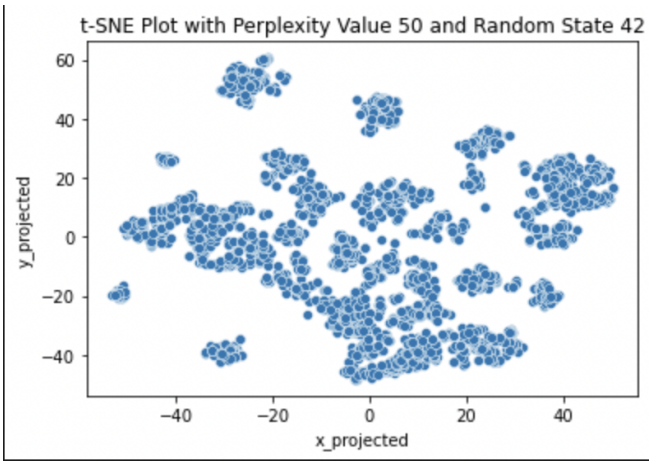
Before implementing clustering algorithms, we decided to first reduce dimensionality of the data as there were initially 30 columns in the dataset after applying EDA and feature transformation. We tried both linear and nonlinear dimension reduction techniques, PCA and T-SNE, at this step. T-SNE was also used for visualizing data after dimension reduction for later analysis.

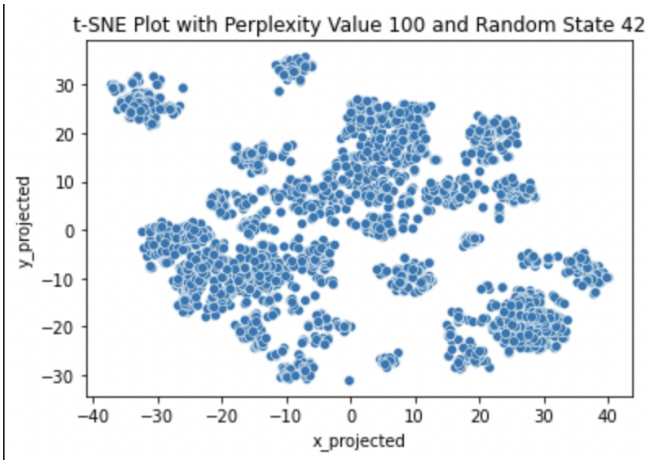
PCA

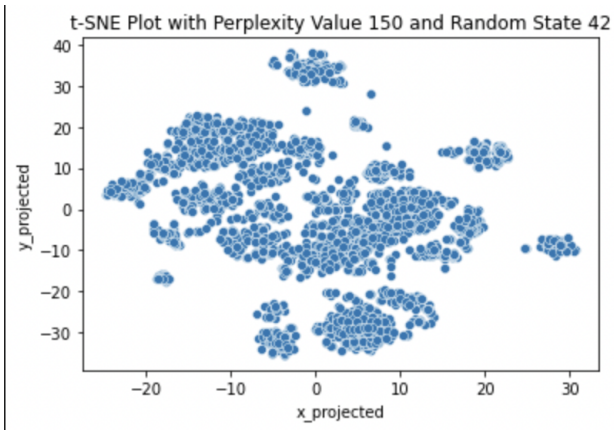
The cumulative explained variance was plotted when choosing the number of principal components for PCA. After analyzing the plot, we decided to use 18 principal components in PCA as dimensions could be reduced by almost half while still preserving 80% of the variance in data.



We then used T-SNE to visualize the 18-dimensional dataset to determine the proper value of perplexity and the number of potential clusters, and it turned out that choosing the perplexity of 100 for T-SNE visualization and the number of cluster 5-7 seemed to be reasonable based on the second graph shown below.





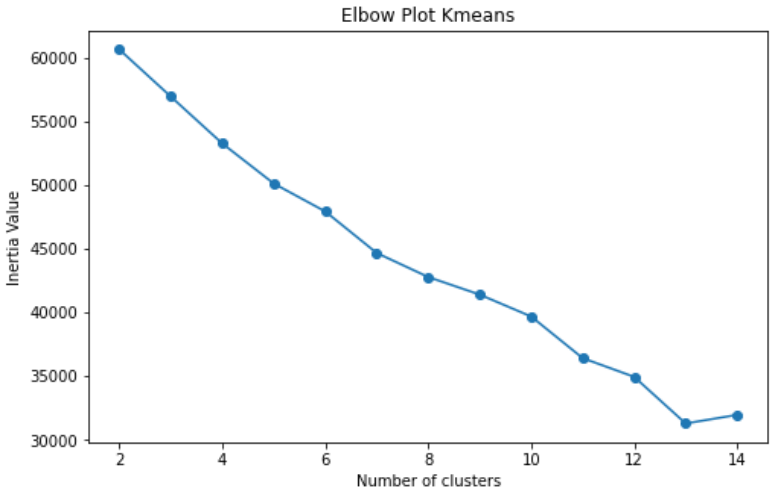


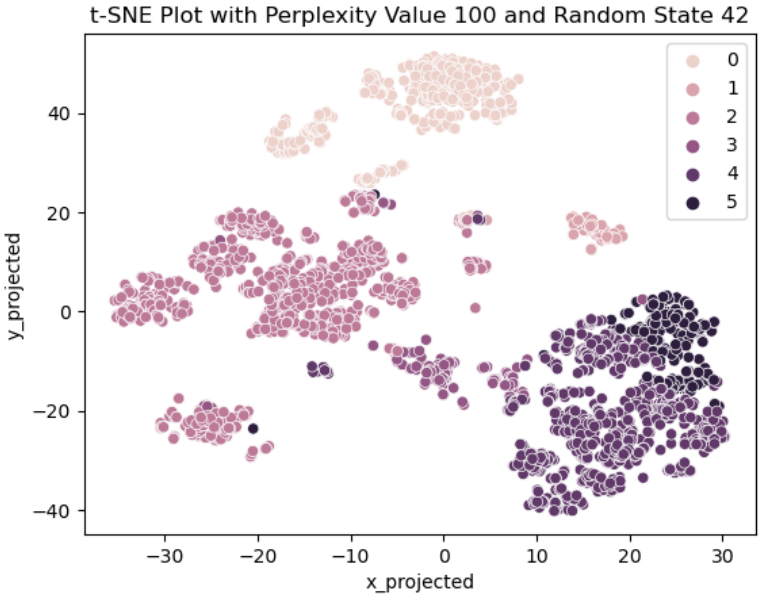
T-SNE

In our research, we attempted to utilize T-SNE as another dimensionality reduction technique; however, we found that it was not effective in separating the data during the clustering process. As a result, we opted to abandon this method and instead utilized PCA.
A Comparison between using T-SNE and PCA for dimension reduction will be shown in the **Dimensional Reduction findings section**.

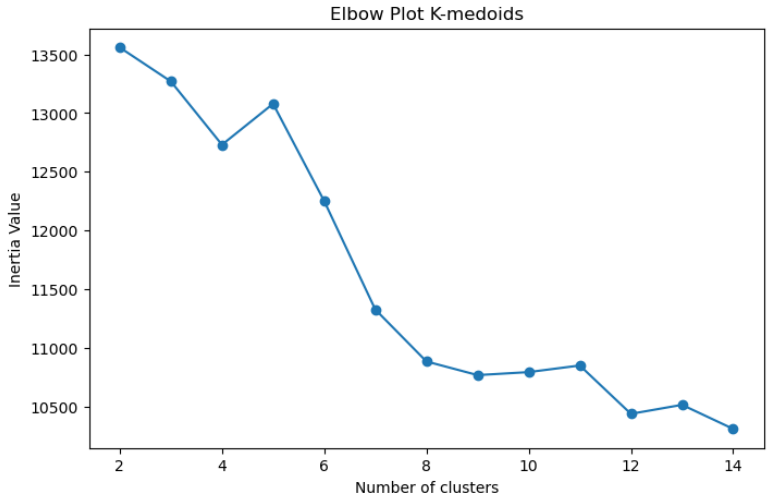
Clustering

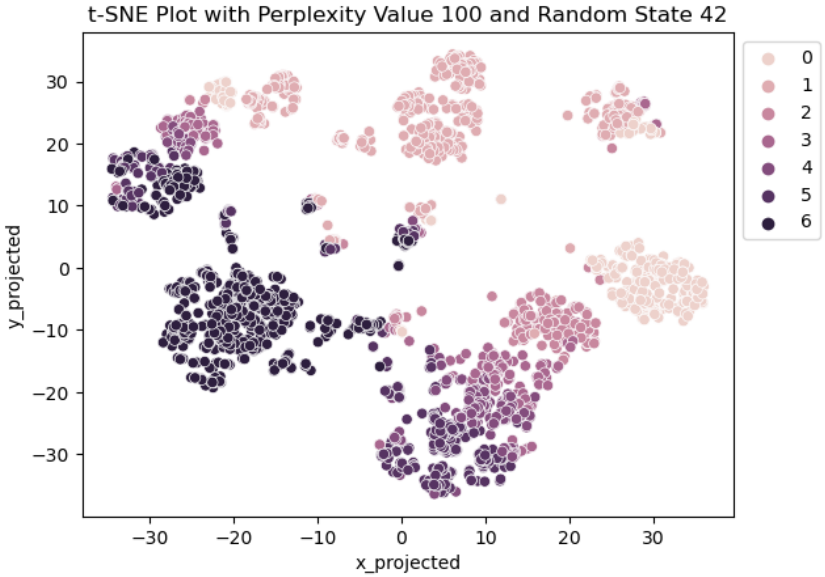
After conducting Principal Component Analysis (PCA) on our final standardized dataset, we utilized various clustering algorithms to evaluate the characteristics of each cluster based on initial research. Since the t-SNE plot revealed that the clusters were well-separated and roughly spherical in shape, we opted to first apply the K-Means algorithm. To determine the optimal number of clusters, we generated an elbow plot, which displayed a steep curve. However, after incorporating the t-SNE plot visualization, we identified that the optimal number of clusters (K) was 6. We then utilized the scikit-learn package in Python to implement the K-Means algorithm, assigning each data point a clustering label ranging from 0 to 5. A t-SNE plot was created again to visualize the clustering assignments.





To enhance the robustness of the clusters to noise or outliers, we subsequently employed the K-Medians algorithm, which follows a similar clustering process to K-Means. The elbow plot for K-Medians also indicated the presence of 7 clusters. We proceeded to implement the K-Medians method using the scikit-learn package and assigned clustering labels to each data point.





Recommendation Engine

In the development of a recommendation engine, the methods of user-based collaborative filtering and item-based collaborative filtering are employed.

The information of movie ID, user ID, and user rating for every movie that each user has rated was used to construct the rating matrix for the recommendation engine. This rating matrix serves as the foundation for the recommendation engine to identify users with similar preferences and to recommend movies that those similar users have rated highly. Or, identify movies with similar preferences and to recommend users that those similar movies have rated highly, depending on which recommender approach is applied.

movieId	2	3	5	6	11	12	13	14	15	16	17	18	19	20	21	22	24	25	26	27	28	30	35	38	55	58	59	62	63	64	65	66	68
userId																																	
1	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
2	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	5.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	3.0	NaN	NaN	NaN	NaN	NaN
3	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
4	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
5	NaN	4.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
667	NaN	NaN	NaN	4.0	3.0	NaN	NaN	NaN	NaN	2.0	2.0	NaN	NaN	NaN	3.0	NaN	NaN	2.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	4.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN
668	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
669	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
670	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	5.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
671	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN

671 rows × 2830 columns

Table: Rating Matrix for Recommendation Engine

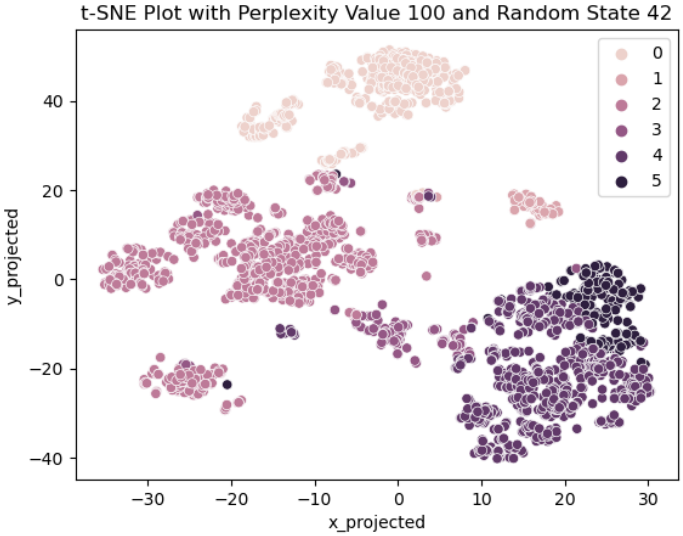
By utilizing the approaches of user-based collaborative filtering and item-based collaborative filtering, the recommendation engine can make predictions on users' ratings on the movie with no previous rating information, and offer personalized recommendations to users based on their predicted rating ranking.

Findings and Conclusion

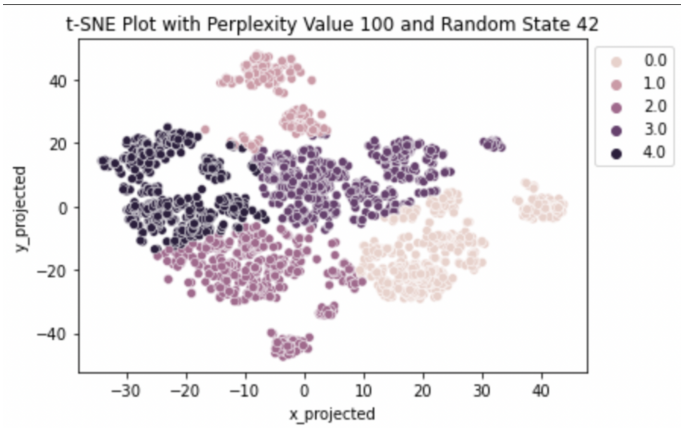
Dimensional Reduction findings

Based on the comparisons below, it can be concluded that PCA helps separate data points better than T-SNE does.

Visualizing K-means clustering using PCA for dimension reduction:



Visualizing K-means clustering using T-SNE for dimension reduction:



Clustering Findings

For both K-Means and K-Medians clustering algorithms, box plots (see appendix) of the features of interest were produced to visualize and compare the different clusters.

Our K-Means clustering analysis revealed six distinct clusters within the dataset. Cluster 0 was characterized by movies with the lowest revenue, an average runtime, the highest vote average, the lowest vote count, and genres of comedy and drama. Cluster 1 was associated with low revenue, an average runtime, high vote average, low vote count, and genres of thriller, science fiction, and drama. Movies in Cluster 2 were also low revenue, with an average runtime, an average vote average, a low vote count, and genres of comedy and drama. Cluster 3 included movies with relatively high revenue, short runtime, average voting, and genres of adventure, animation, fantasy, family, and comedy. Cluster 4 was composed of movies with low revenue, a relatively long runtime, average voting, and genres of comedy and drama. Finally, Cluster 5 encompassed movies with the highest revenue, longest runtime, high voting, and genres of adventure, thriller, action, and science fiction.

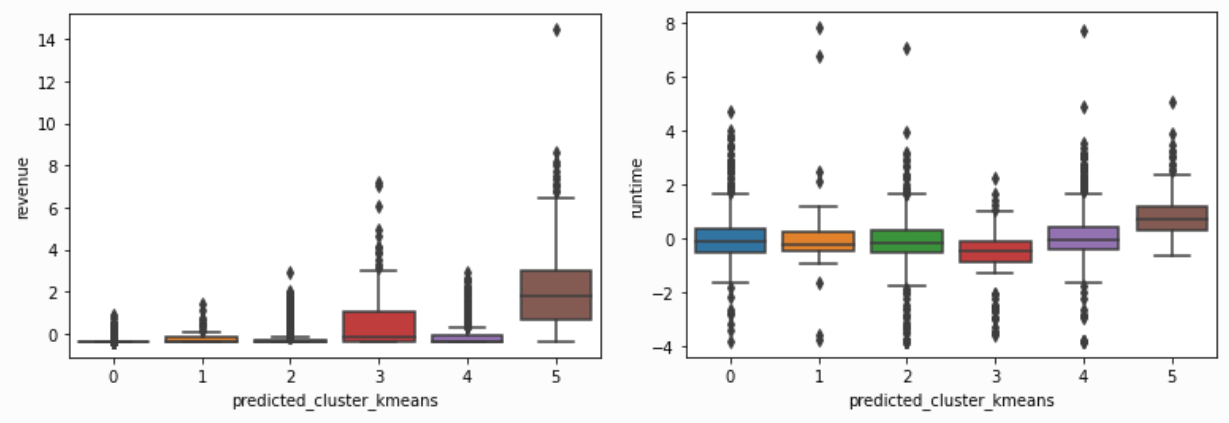
The K-Medians clustering algorithm yielded six clusters, each with distinct characteristics. Cluster 0 was comprised of movies with low revenue, low runtime, low voting, and mostly thriller, action, and comedy movies. Movies in Cluster 1 have low revenue, long runtime, high voting, and genre of drama. Cluster 2 included movies with the highest revenue, long runtime, the highest number of voting, and genres of adventure, thriller, action, fantasy, and science fiction. Movies in Cluster 3 had relatively low revenue, an average runtime, high voting, and genres of thriller, comedy, and drama. Cluster 4 consisted of movies with low revenue, long runtime, high voting, and genres of thriller and drama. Cluster 5 encompassed movies with the lowest revenue, short runtime, average voting, and genres of comedy and drama. Finally, cluster 6 consisted of movies with low revenue, short runtime, average voting, and genres of thriller, comedy, and drama. Movie recommendations would be made accordingly with cluster characteristics.

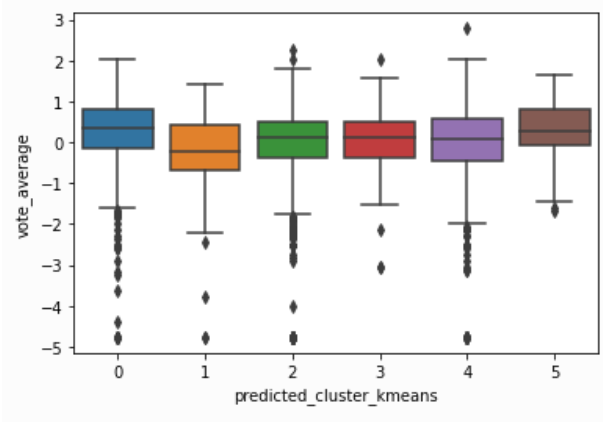
Reference:

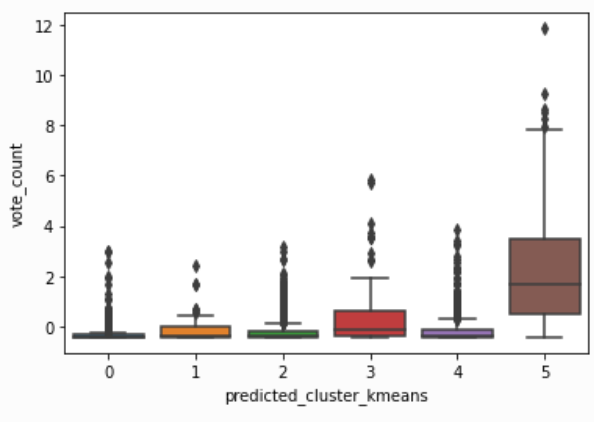
- BANIK, R. O. U. N. A. K. (2018). *Hands-on recommendation systems with Python: Start building powerful and personalized, ... recommendation engines with python*. PACKT Publishing Limited.
- Banik, R. (2017, November 10). *The movies dataset*. Kaggle. Retrieved March 7, 2023, from https://www.kaggle.com/datasets/rounakbanik/the-movies-dataset?select=ratings_small.csv
- Hotelling, H. (1933). Analysis of a complex of statistical variables into principal components. *Journal of Educational Psychology*, *24*(6), 417–441. https://doi.org/10.1037/h0071325
- Whelan, C., Harrell, G.K., & Wang, J. (2015). Understanding the K-Medians Problem.
- Jin, X., Han, J. (2011). *K*-Means Clustering. In: Sammut, C., Webb, G.I. (eds) Encyclopedia of Machine Learning. Springer, Boston, MA. https://doi.org/10.1007/978-0-387-30164-8_425
- Sarwar, B., Karypis, G., Konstan, J., & Riedl, J. (2001). Item-based collaborative filtering recommendation algorithms. *Proceedings of the 10th International Conference on World Wide Web*. https://doi.org/10.1145/371920.372071

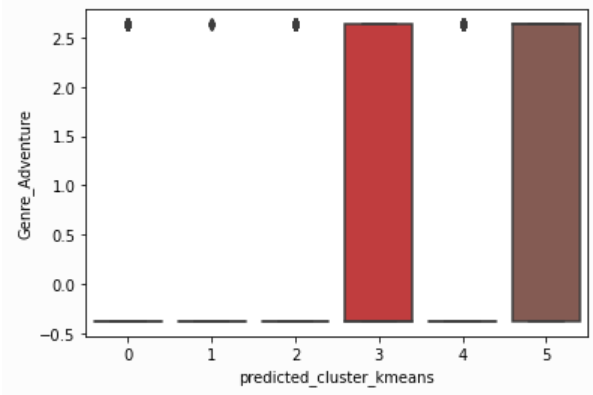
Appendix:

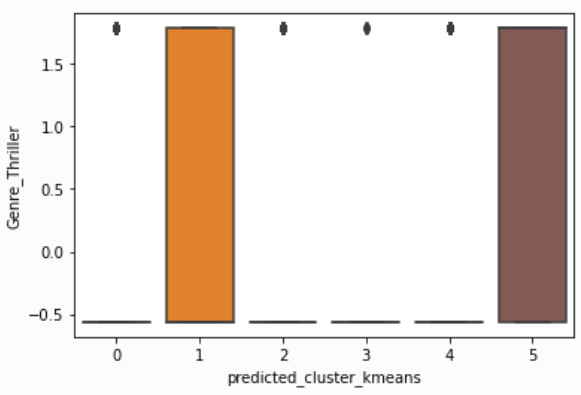
- Boxplots for K-means clustering

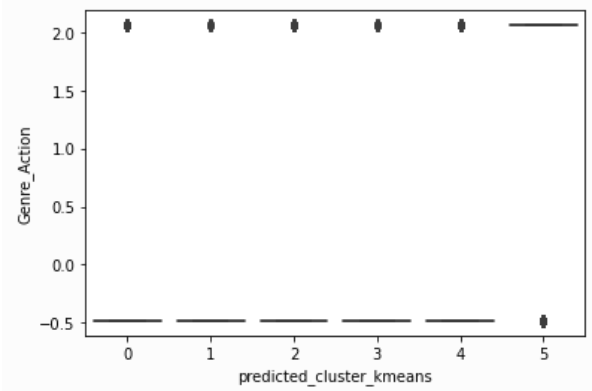


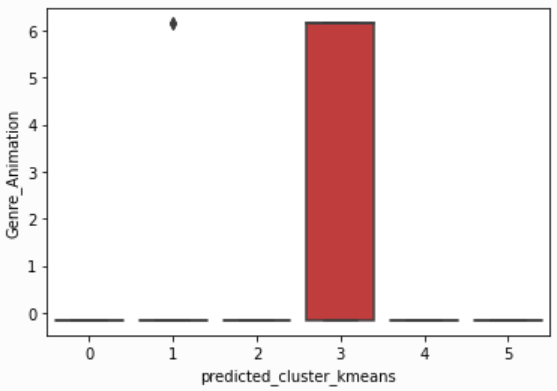

vote_average
predicted_cluster_kmeans

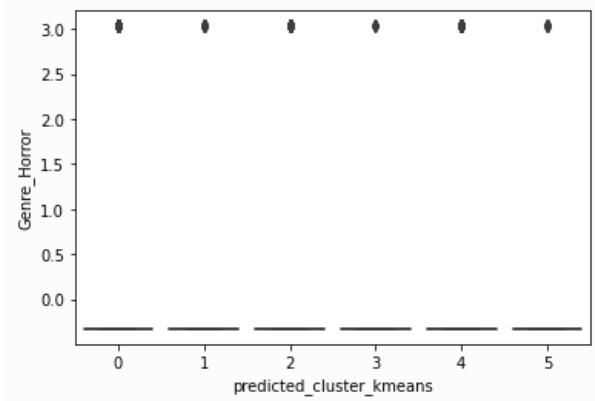

vote_count
predicted_cluster_kmeans

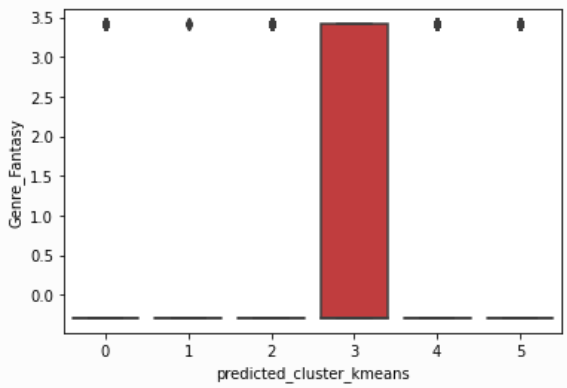

Genre_Adventure
predicted_cluster_kmeans

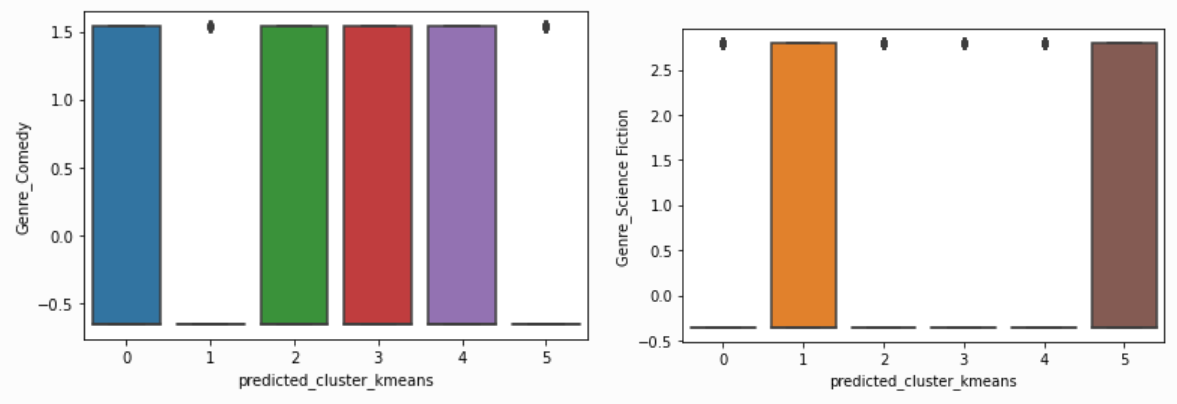

Genre_Thriller
predicted_cluster_kmeans

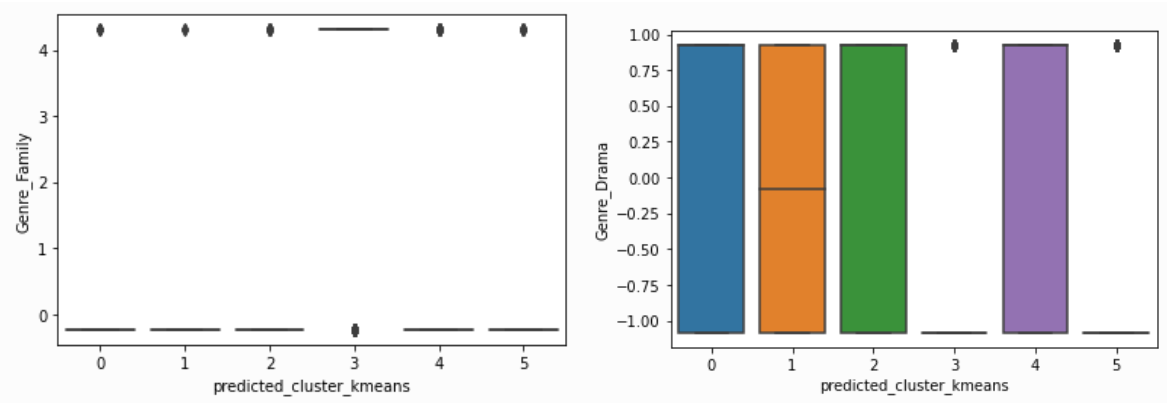

Genre_Action
predicted_cluster_kmeans

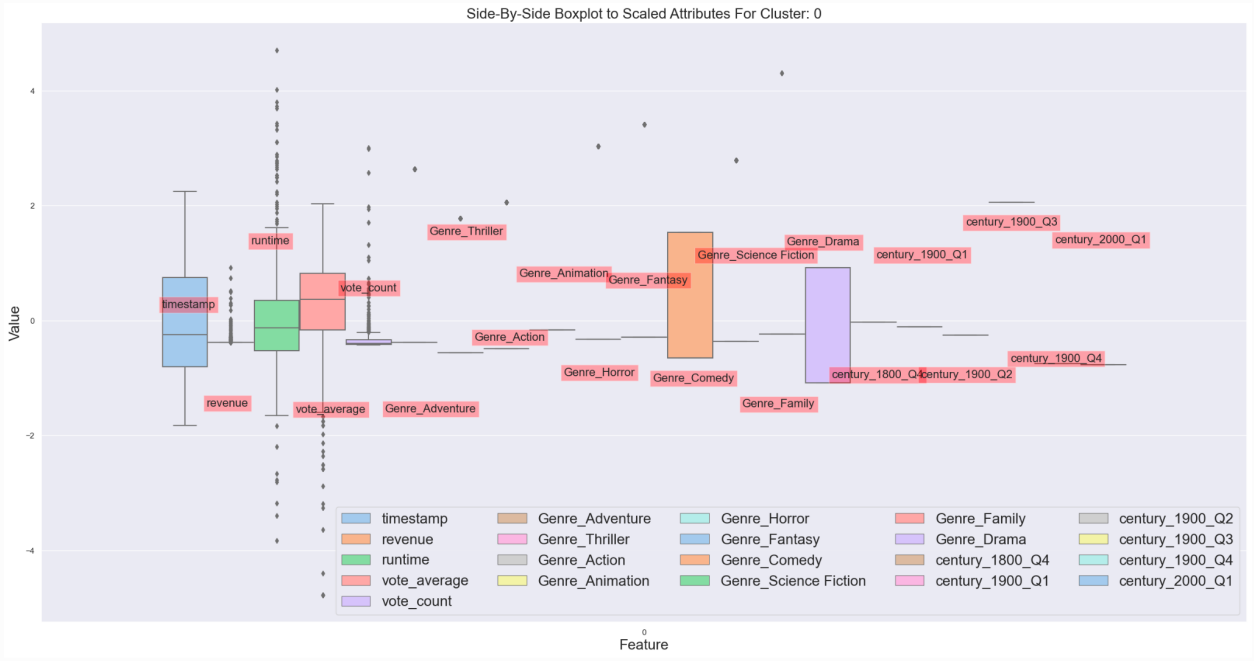

Genre_Animation
predicted_cluster_kmeans

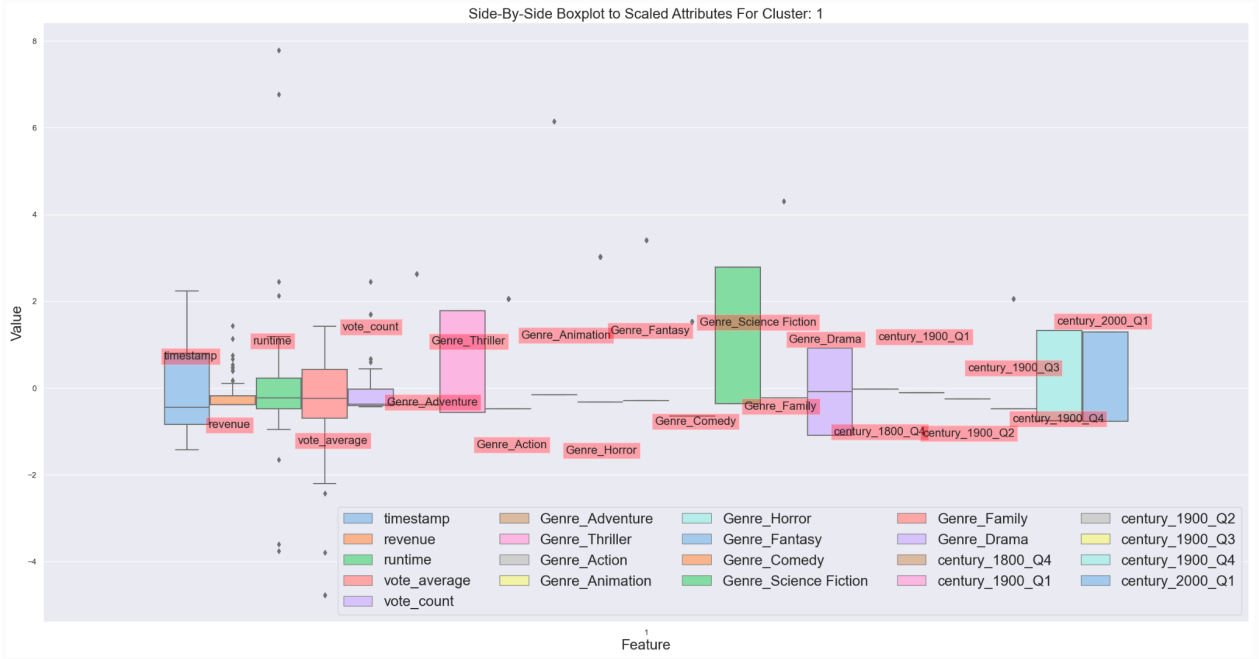

Genre_Horror
predicted_cluster_kmeans

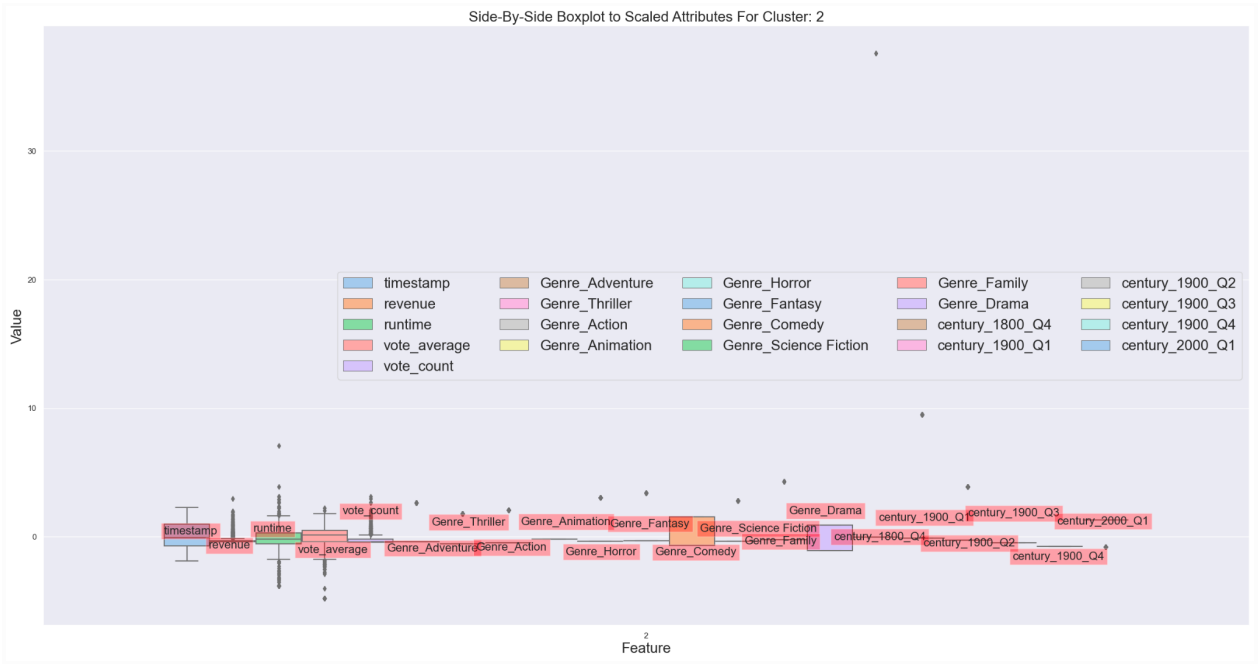

Genre_Fantasy
predicted_cluster_kmeans

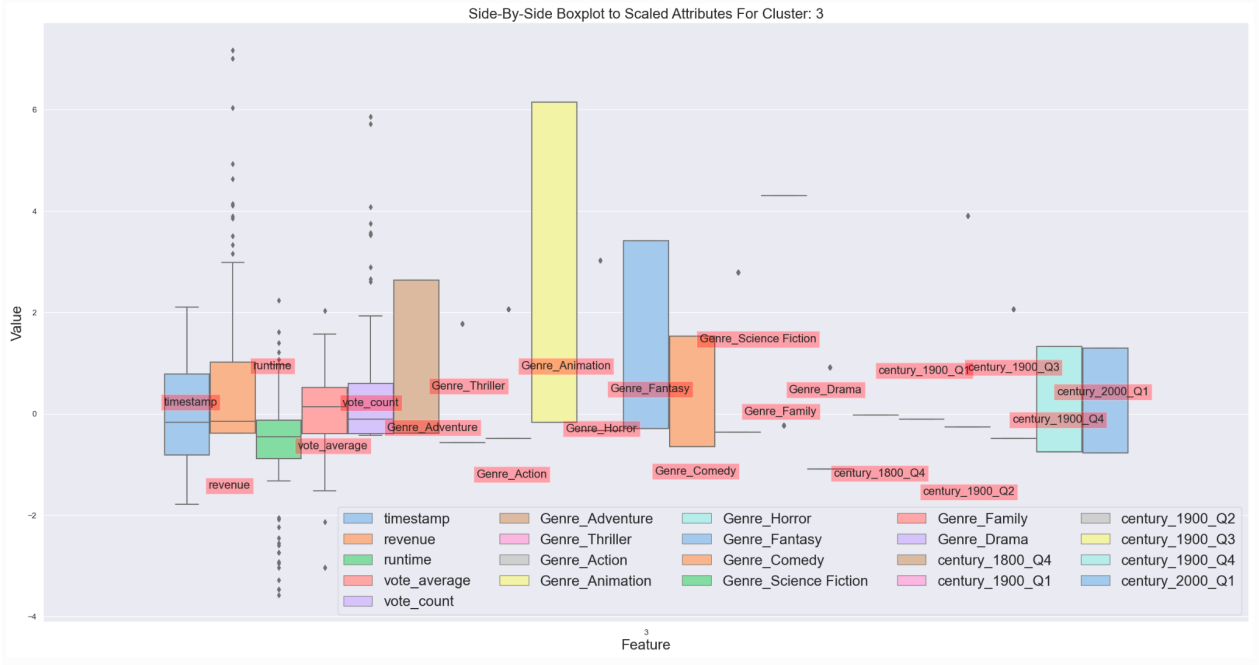

Genre_Comedy
predicted_cluster_kmeans
Genre_Science Fiction
predicted_cluster_kmeans

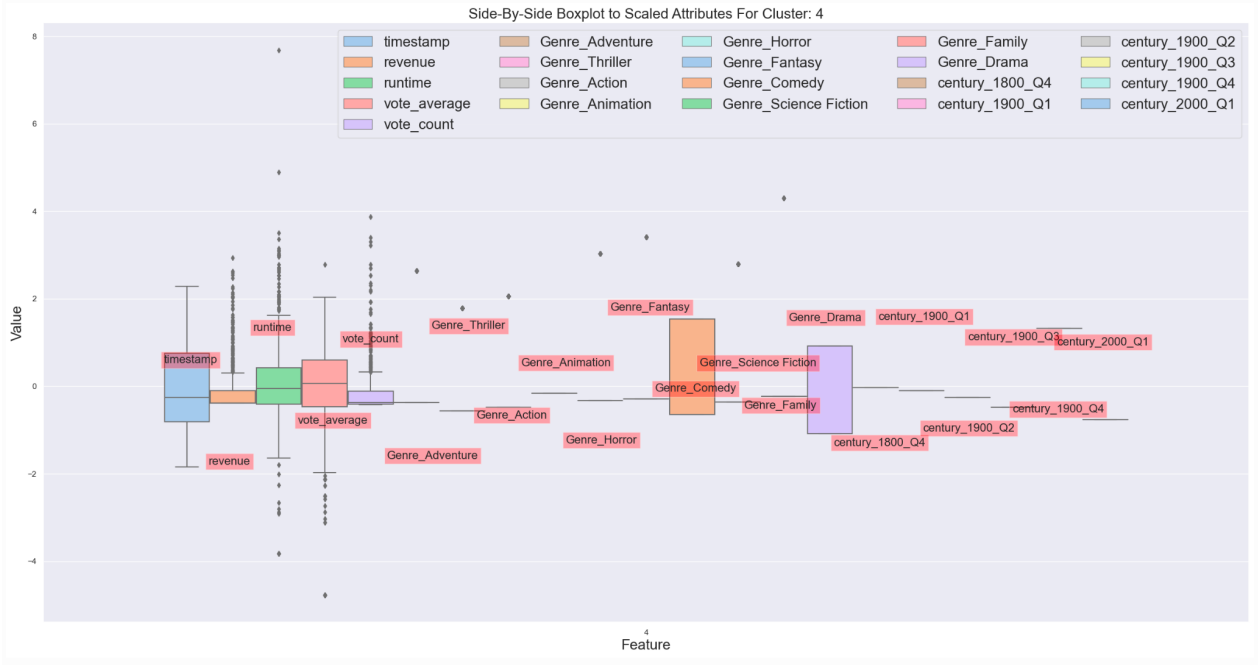

Genre_Family
predicted_cluster_kmeans
Genre_Drama
predicted_cluster_kmeans

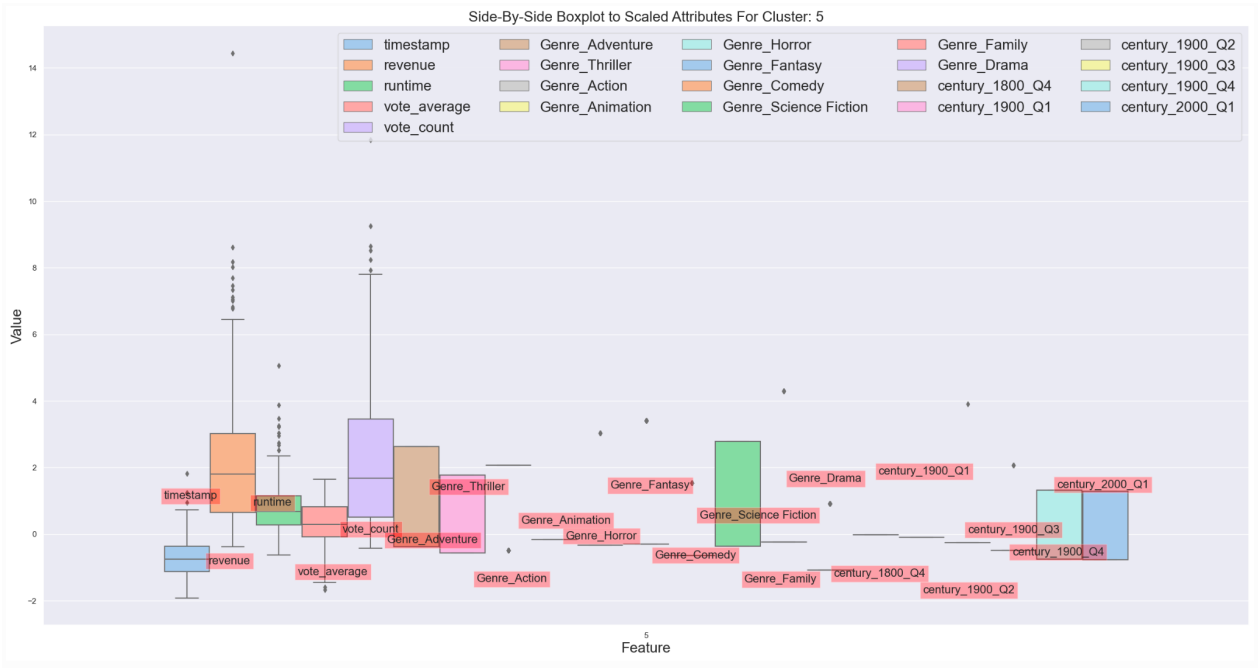

Side-By-Side Boxplot to Scaled Attributes For Cluster: 0
Value
Feature
timestamp
revenue
runtime
vote_average
vote_count
Genre_Adventure
Genre_Thriller
Genre_Action
Genre_Animation
Genre_Horror
Genre_Fantasy
Genre_Comedy
Genre_Science Fiction
Genre_Family
Genre_Drama
century_1800_Q4
century_1900_Q1
century_1900_Q2
century_1900_Q3
century_1900_Q4
century_2000_Q1


Side-By-Side Boxplot to Scaled Attributes For Cluster: 1
Value
Feature
1
timestamp
revenue
runtime
vote_average
vote_count
Genre_Adventure
Genre_Thriller
Genre_Action
Genre_Animation
Genre_Horror
Genre_Fantasy
Genre_Comedy
Genre_Science Fiction
Genre_Family
Genre_Drama
century_1800_Q4
century_1900_Q1
century_1900_Q2
century_1900_Q3
century_1900_Q4
century_2000_Q1

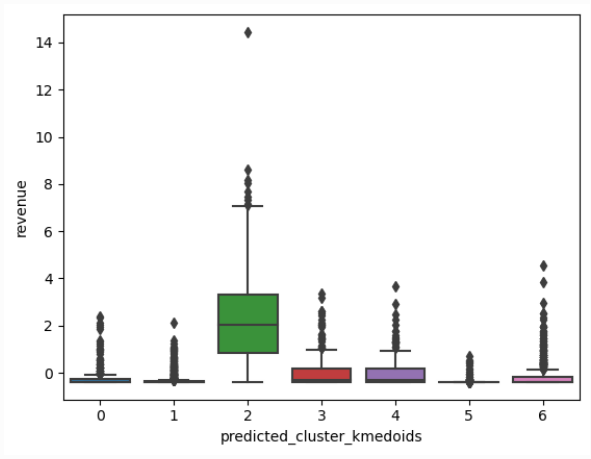

Side-By-Side Boxplot to Scaled Attributes For Cluster: 2
Value
Feature
2
timestamp
revenue
runtime
vote_average
vote_count
Genre_Adventure
Genre_Thriller
Genre_Action
Genre_Animation
Genre_Horror
Genre_Fantasy
Genre_Comedy
Genre_Science Fiction
Genre_Family
Genre_Drama
century_1800_Q4
century_1900_Q1
century_1900_Q2
century_1900_Q3
century_1900_Q4
century_2000_Q1

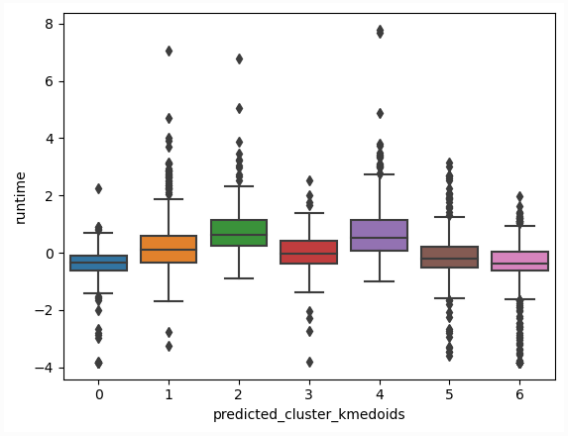

Side-By-Side Boxplot to Scaled Attributes For Cluster: 3
Value
Feature
timestamp
revenue
runtime
vote_average
vote_count
Genre_Adventure
Genre_Thriller
Genre_Action
Genre_Animation
Genre_Horror
Genre_Fantasy
Genre_Comedy
Genre_Science Fiction
Genre_Family
Genre_Drama
century_1800_Q4
century_1900_Q1
century_1900_Q2
century_1900_Q3
century_1900_Q4
century_2000_Q1

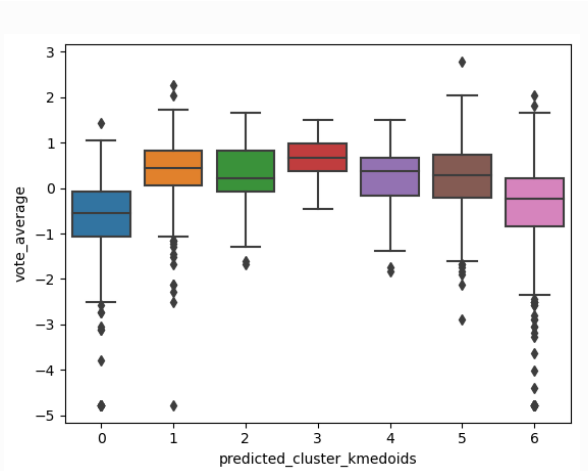

Side-By-Side Boxplot to Scaled Attributes For Cluster: 4
timestamp
revenue
runtime
vote_average
vote_count
Genre_Adventure
Genre_Thriller
Genre_Action
Genre_Animation
Genre_Horror
Genre_Fantasy
Genre_Comedy
Genre_Science Fiction
Genre_Family
Genre_Drama
century_1800_Q4
century_1900_Q1
century_1900_Q2
century_1900_Q3
century_1900_Q4
century_2000_Q1
Value
Feature

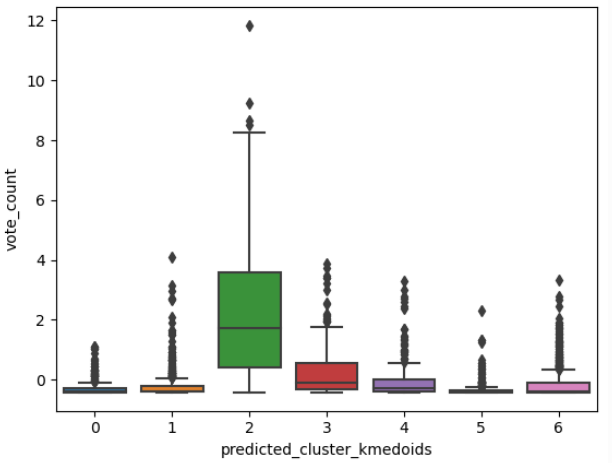

Side-By-Side Boxplot to Scaled Attributes For Cluster: 5
timestamp
revenue
runtime
vote_average
vote_count
Genre_Adventure
Genre_Thriller
Genre_Action
Genre_Animation
Genre_Horror
Genre_Fantasy
Genre_Comedy
Genre_Science Fiction
Genre_Family
Genre_Drama
century_1800_Q4
century_1900_Q1
century_1900_Q2
century_1900_Q3
century_1900_Q4
century_2000_Q1
Value
Feature

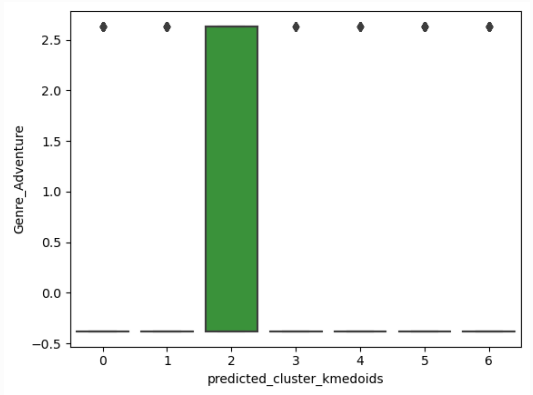
- Boxplots for K-medians clustering

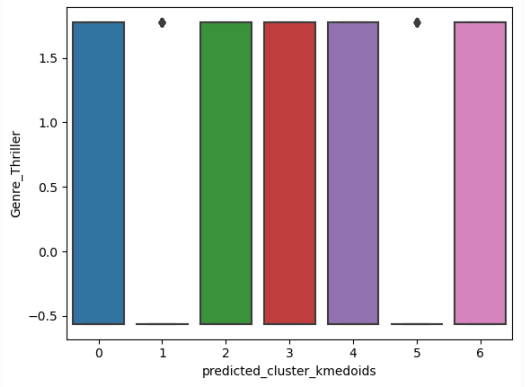


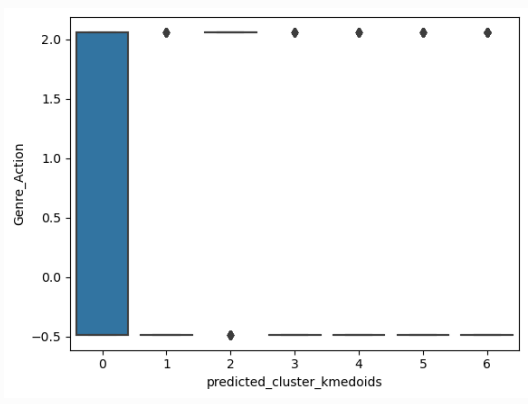


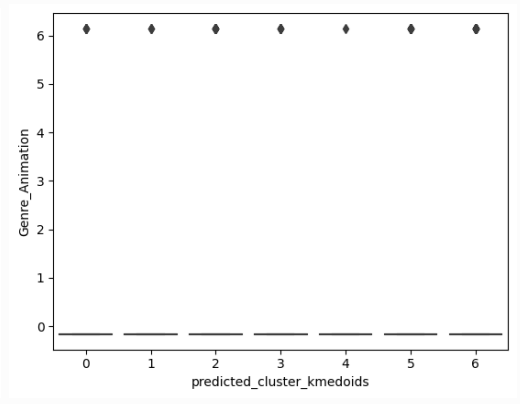


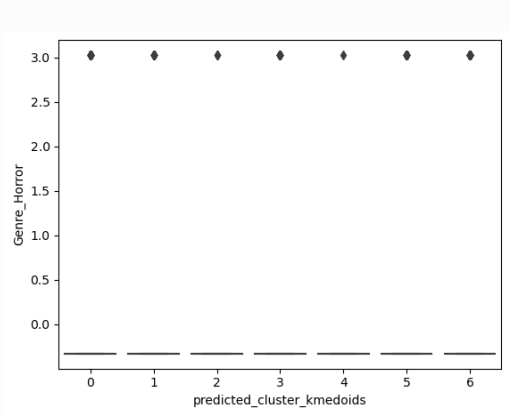


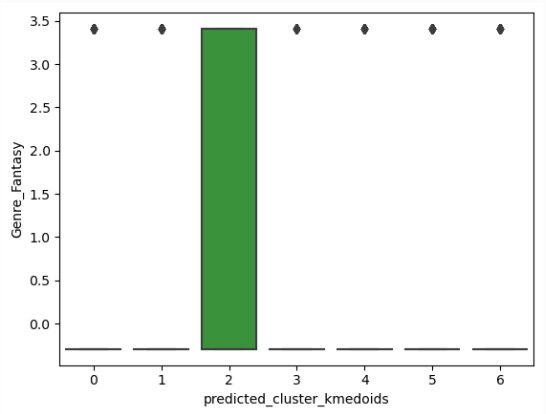


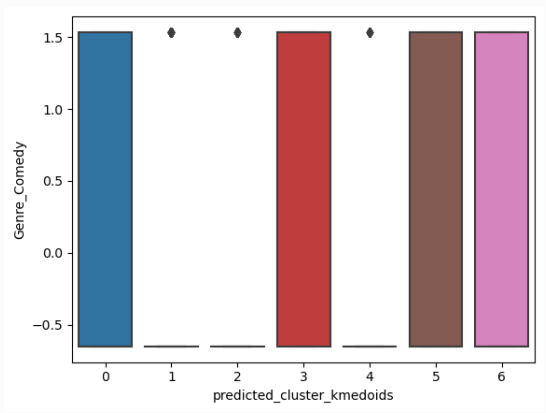


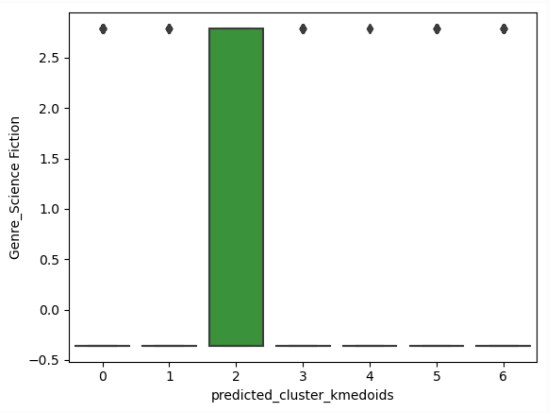

Genre_Action
predicted_cluster_kmedoids

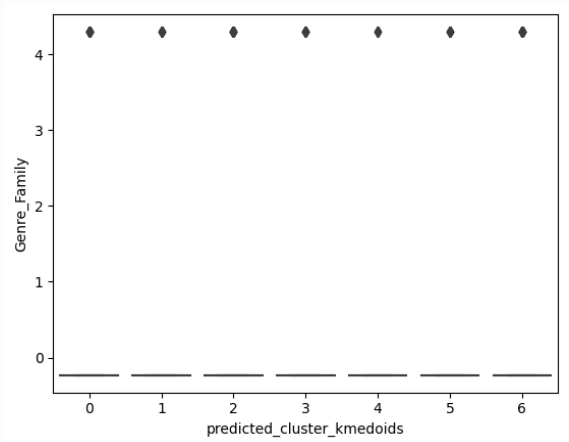

Genre_Animation
predicted_cluster_kmedoids

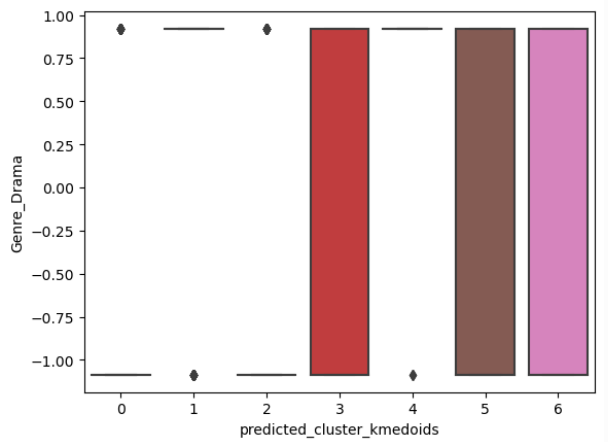

Genre_Horror
predicted_cluster_kmedoids

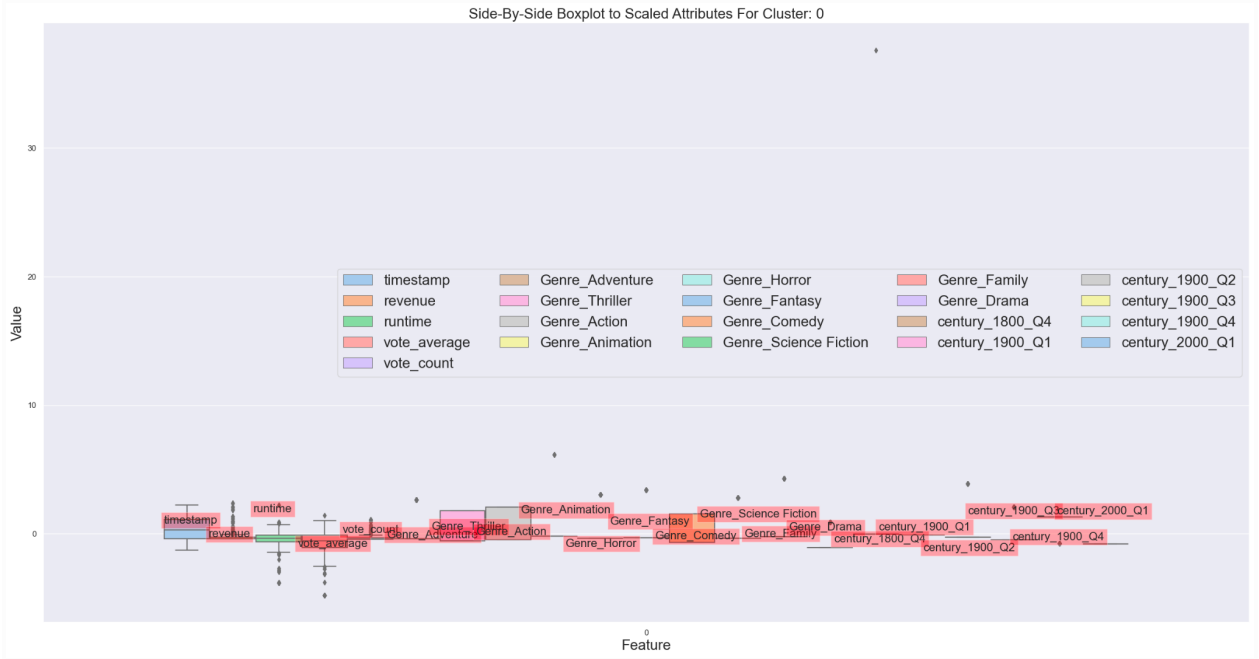

Genre_Fantasy
predicted_cluster_kmedoids

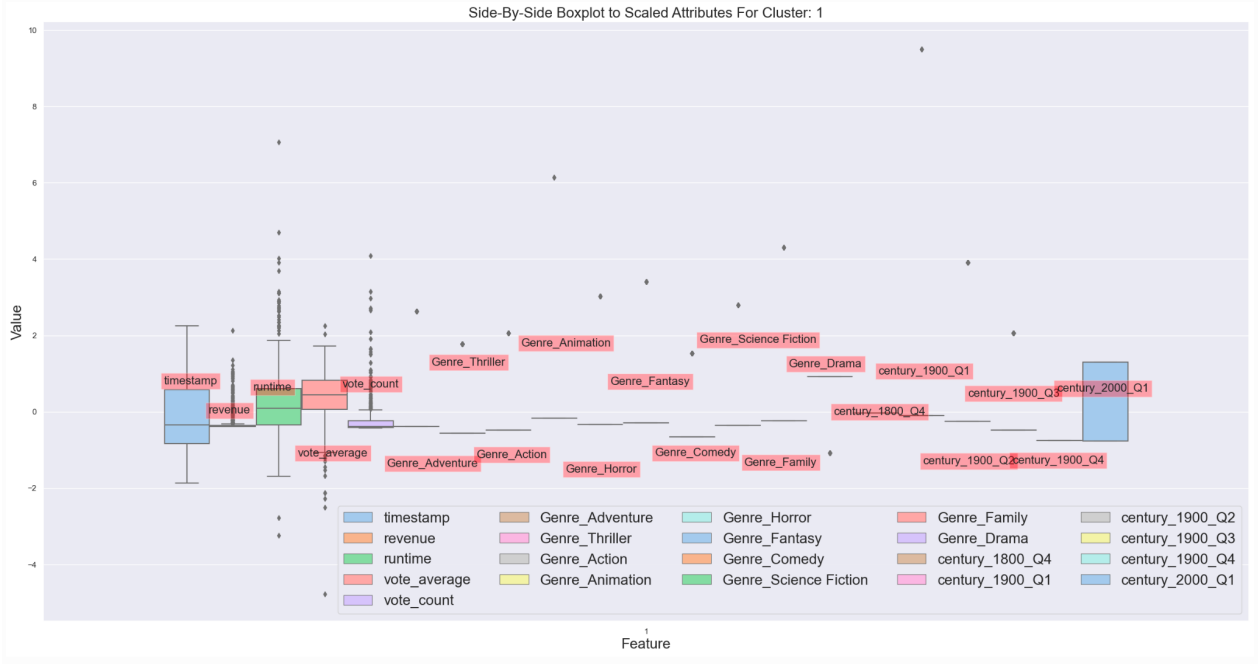

Genre_Comedy
predicted_cluster_kmedoids

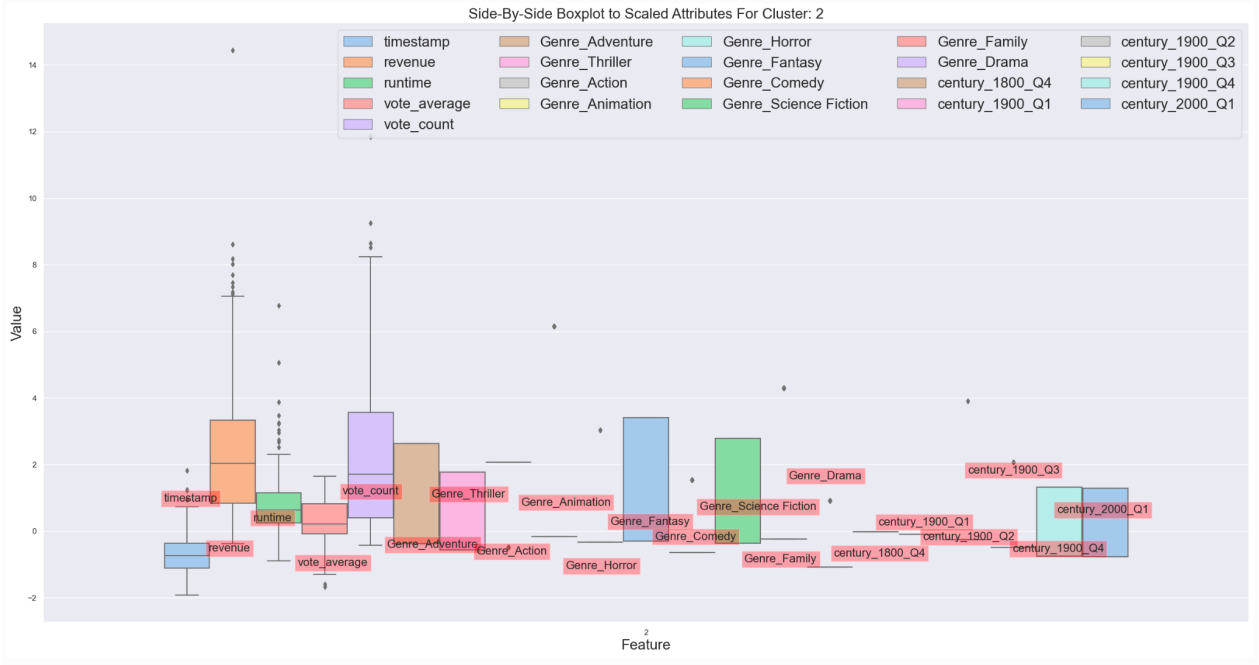

Genre_Science Fiction
predicted_cluster_kmedoids

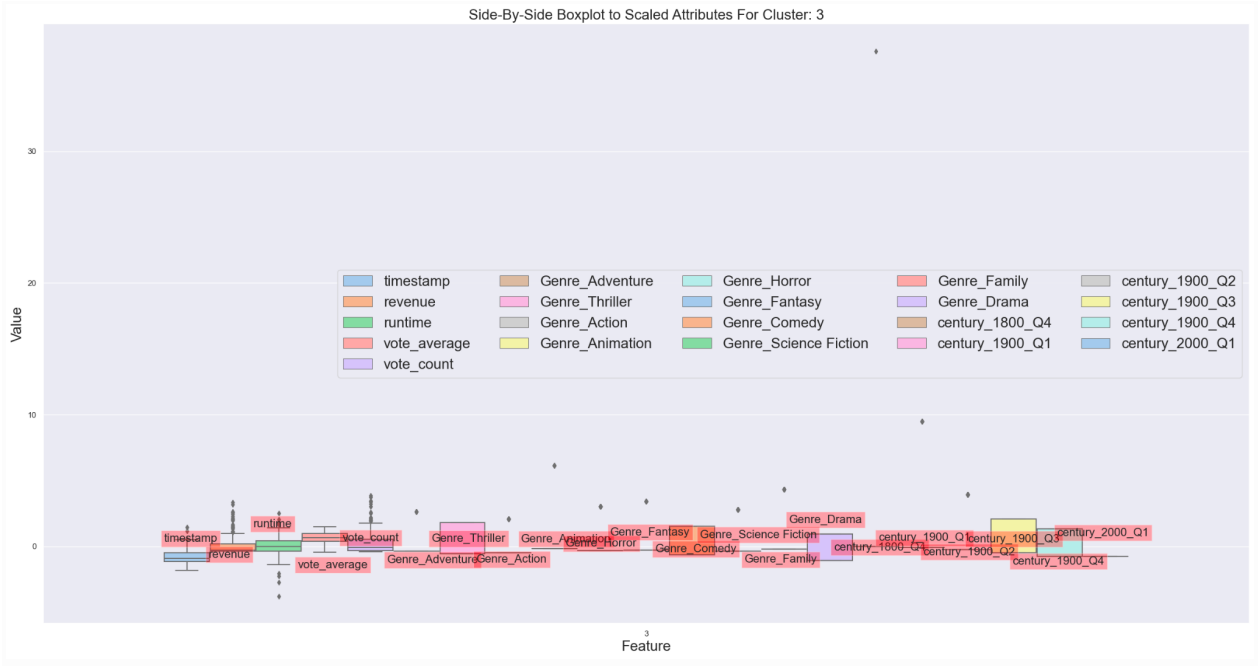

Genre_Family
predicted_cluster_kmedoids

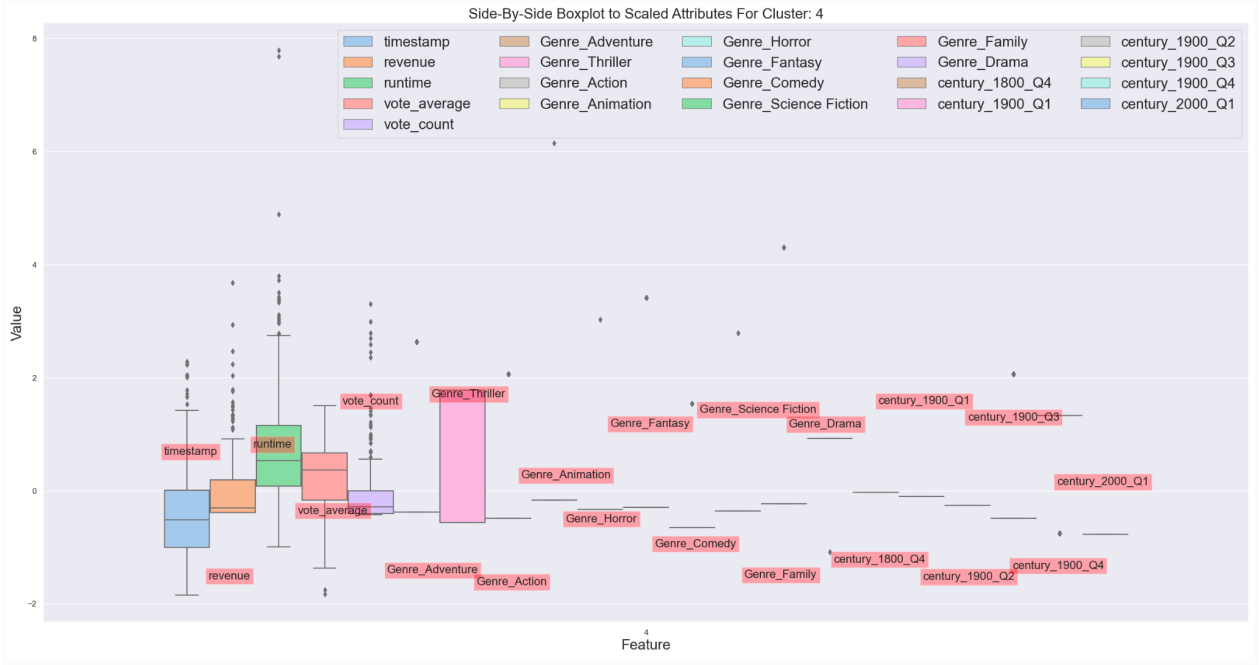

Genre_Drama
predicted_cluster_kmedoids

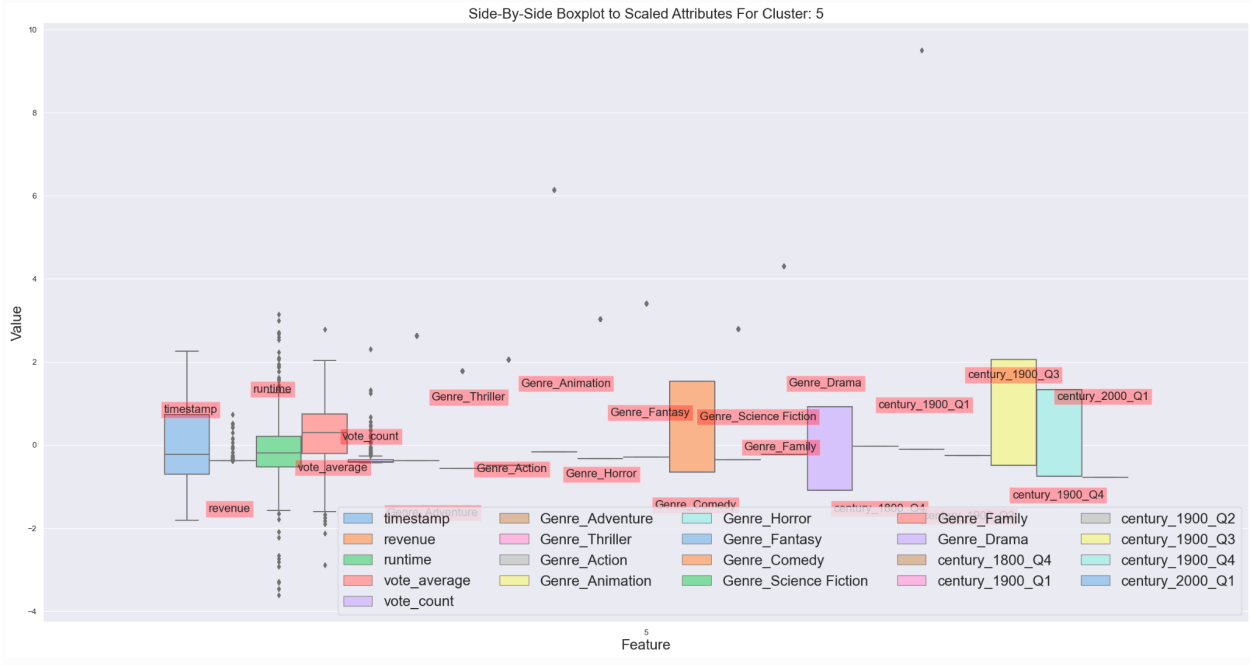

Side-By-Side Boxplot to Scaled Attributes For Cluster: 0
Value
Feature
0
timestamp
revenue
runtime
vote_average
vote_count
Genre_Adventure
Genre_Thriller
Genre_Action
Genre_Animation
Genre_Horror
Genre_Fantasy
Genre_Comedy
Genre_Science Fiction
Genre_Family
Genre_Drama
century_1800_Q4
century_1900_Q1
century_1900_Q2
century_1900_Q3
century_1900_Q4
century_2000_Q1

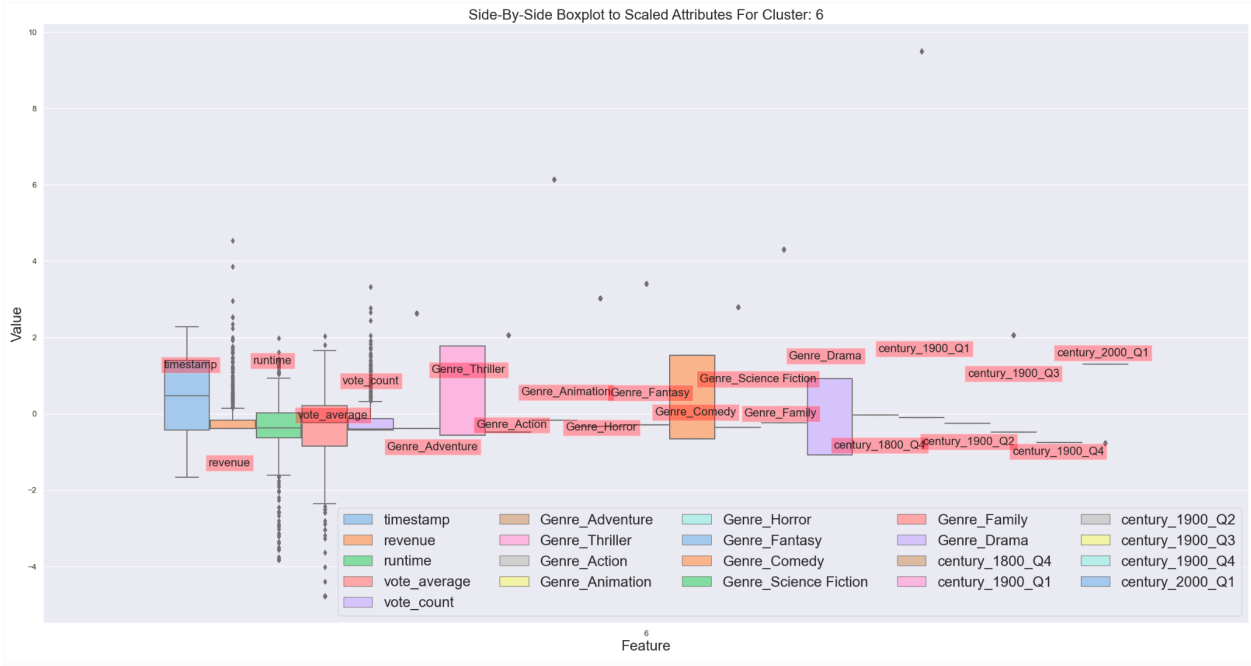

Side-By-Side Boxplot to Scaled Attributes For Cluster: 1
Value
Feature
1
timestamp
revenue
runtime
vote_average
vote_count
Genre_Adventure
Genre_Thriller
Genre_Action
Genre_Animation
Genre_Horror
Genre_Fantasy
Genre_Comedy
Genre_Science Fiction
Genre_Family
Genre_Drama
century_1800_Q4
century_1900_Q1
century_1900_Q2
century_1900_Q3
century_1900_Q4
century_2000_Q1


Side-By-Side Boxplot to Scaled Attributes For Cluster: 2
timestamp
revenue
runtime
vote_average
vote_count
Genre_Adventure
Genre_Thriller
Genre_Action
Genre_Animation
Genre_Horror
Genre_Fantasy
Genre_Comedy
Genre_Science Fiction
Genre_Family
Genre_Drama
century_1800_Q4
century_1900_Q1
century_1900_Q2
century_1900_Q3
century_1900_Q4
century_2000_Q1
Value
Feature
2


Side-By-Side Boxplot to Scaled Attributes For Cluster: 3
timestamp
revenue
runtime
vote_average
vote_count
Genre_Adventure
Genre_Thriller
Genre_Action
Genre_Animation
Genre_Horror
Genre_Fantasy
Genre_Comedy
Genre_Science Fiction
Genre_Family
Genre_Drama
century_1800_Q4
century_1900_Q1
century_1900_Q2
century_1900_Q3
century_1900_Q4
century_2000_Q1
Value
Feature
3


Side-By-Side Boxplot to Scaled Attributes For Cluster: 4
timestamp
revenue
runtime
vote_average
vote_count
Genre_Adventure
Genre_Thriller
Genre_Action
Genre_Animation
Genre_Horror
Genre_Fantasy
Genre_Comedy
Genre_Science Fiction
Genre_Family
Genre_Drama
century_1800_Q4
century_1900_Q1
century_1900_Q2
century_1900_Q3
century_1900_Q4
century_2000_Q1
Value
Feature


Side-By-Side Boxplot to Scaled Attributes For Cluster: 5
Value
Feature
5
timestamp
revenue
runtime
vote_average
vote_count
Genre_Adventure
Genre_Thriller
Genre_Action
Genre_Animation
Genre_Horror
Genre_Comedy
Genre_Fantasy
Genre_Science Fiction
Genre_Family
Genre_Drama
century_1800_Q4
century_1900_Q1
century_1900_Q2
century_1900_Q3
century_1900_Q4
century_2000_Q1


Side-By-Side Boxplot to Scaled Attributes For Cluster: 6
Value
Feature
6
timestamp
revenue
runtime
vote_average
vote_count
Genre_Adventure
Genre_Thriller
Genre_Action
Genre_Animation
Genre_Horror
Genre_Comedy
Genre_Fantasy
Genre_Science Fiction
Genre_Family
Genre_Drama
century_1800_Q4
century_1900_Q1
century_1900_Q2
century_1900_Q3
century_1900_Q4
century_2000_Q1