IMDB Score Predictor
Part I

Milan Bidare
Juan Antonio Martinez Castellanos

CME 250 - Introduction to Machine Learning
Stanford University

1 Motivation and Definition

Before choosing to become an engineer, one of the authors of this project dreamed of becoming a professional filmmaker. Eventually, he decided that a career in STEM would probably be significantly more lucrative. However, his passion for cinema remained.

Meanwhile, the other author was sitting at home scrolling through Netflix movies on a Tuesday morning, trying to decide what to watch. At that moment, she realized what she really needed in life. A way to predict movie scores to optimize her movie watching time, to ensure maximum happiness right before the drudgery of class started up again. Thus, this project was born.

For this project, we will be building an algorithm to predict movie ratings based on input features such as the duration, budget, gross sales, and number of critical reviews received by the movie.

2 Data Sources

Our data comes form a kaggle dataset titled "IMDB 5000 Movie Dataset" [4]. As the title implies, this dataset contains the information of 5000 different movies provided in IMDB. The information provided is organized in the following categories:

1. Color
2. Director name
3. Number of critic reviews
4. Duration
5. Director Facebook Likes
6. Facebook Likes of 3 principle actors.
7. Names of 3 principle actors.
8. Gross revenue
9. Genres
10. Title
11. Number of users who voted
12. Total number of Facebook likes of the cast
13. Number of faces in the poster
14. Plot keywords
15. Link to the IMDB page
16. Number of users who wrote a review
17. Language
18. Country
19. Content rating
20. Budget
21. Year
22. IMDB score
23. Aspect ratio
24. Number of Facebook likes of the movie

From inspection, it is clear that some of the data features provided do not

influence the ratings of the movie. Some examples of such features include the link to the IMDB page and the aspect ratio of the movie. In addition, non-numerical data such as title and keywords are difficult to process and visualize in large quantities, so for our project we will only use the following features:

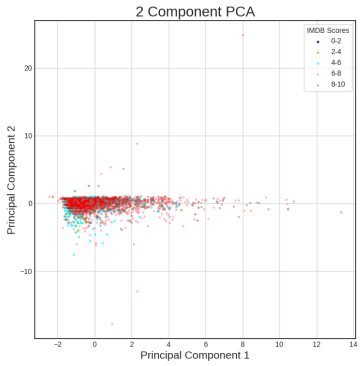
1. Number of critic reviews
2. Duration
3. Gross revenue
4. Number of faces in the poster
5. Budget
6. Number of Facebook likes
7. IMDB score.

3 Analysis

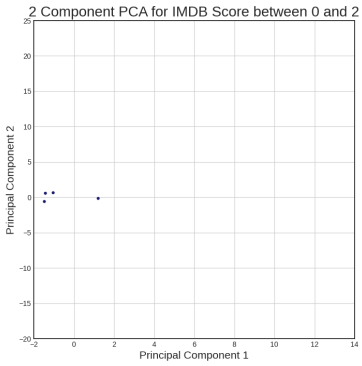
3.1 Dimensionality Reduction

In order to reduce the dimensionality of the data and make it easier to visualize, we performed Principal Component Analysis (PCA) on the dataset [3]. First, the data was standardized to remove the effect of having different scales and ranges for each numerical feature of the input. Then, PCA was performed on the standardized data with 2 components. After performing PCA, the principal components were plotted for different IMDB scores in Figure 1. To more clearly see each range of scores from high to low, the scores were also plotted separately.

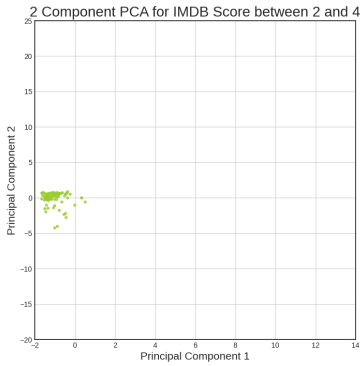
Additionally, the principal components were plotted against the IMDB scores in search of a strong correlation. This can be seen in Figure 2



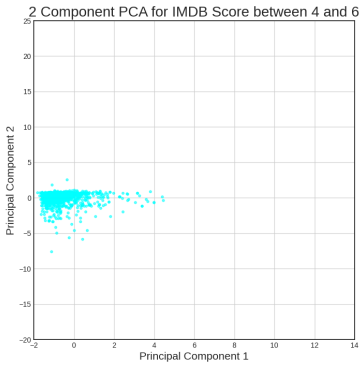
(a)



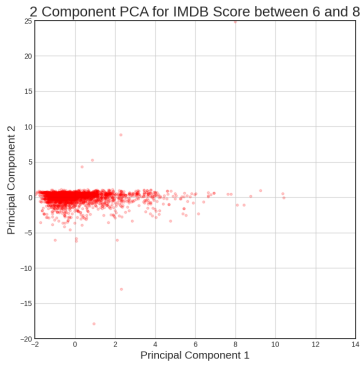
(b)



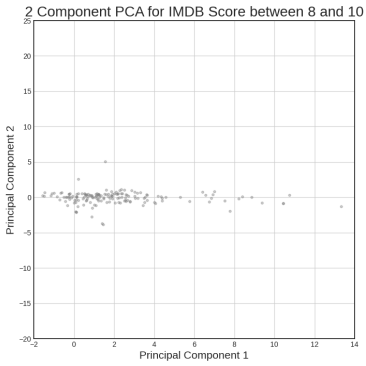
(c)



(d)

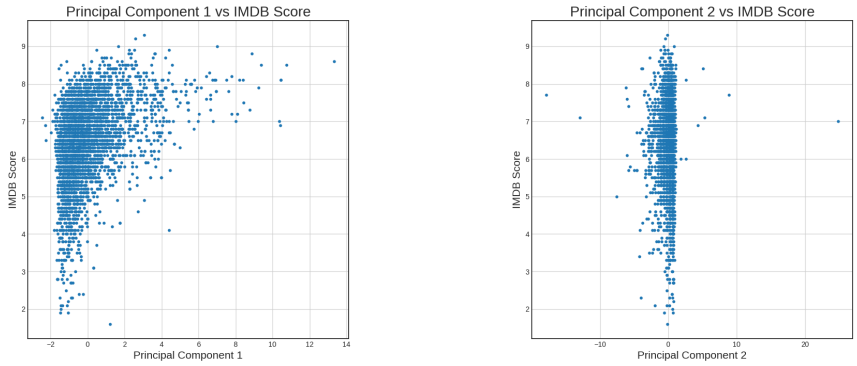


(e)



(f)

Figure 1: Plot of principal components for varying score ranges

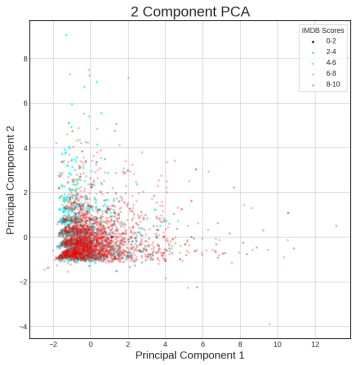


(a) (b)

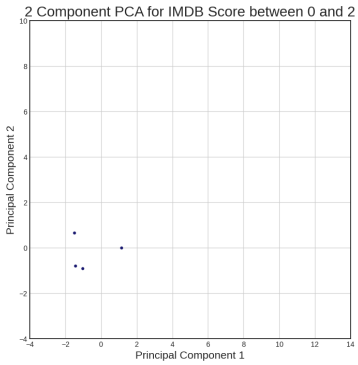
Figure 2: Plot of individual principal components against IMDB score

It is difficult to see the distribution of points in these plots because of outliers in the data that skew the axis to be much larger than required for most of the data clustered in the middle. In addition, looking at the variance values of PCA, it was found that only 54% of the original data was preserved when performing PCA. To improve this, we could perform PCA with 3 or more components. In fact, we found that using 3 components increases the variance value, showing that 75% of the data is now captured by the components. However, having 3 components makes it difficult to visualize the data.

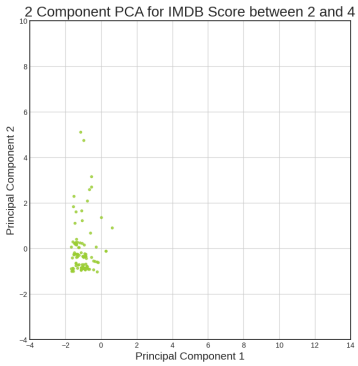
Instead, we decided to remove the outliers from the PCA. Specifically, the datapoints that had an absolute value greater than 10 for the principal component 2 were removed from the original dataset. Then, PCA was run on this new dataset. Once again, the axis was similarly much larger than required for most of the data because of a single datapoint that had a large value. This point was also removed and the PCA was applied again. In total, only 4 datapoints were removed from the original set to achieve the more informative plots that can be seen in Figure 3



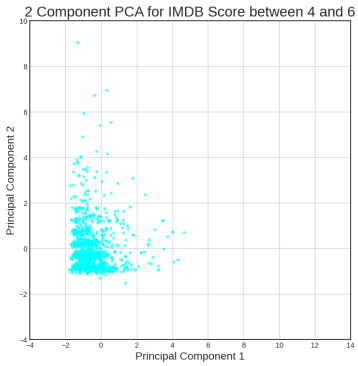
(a)



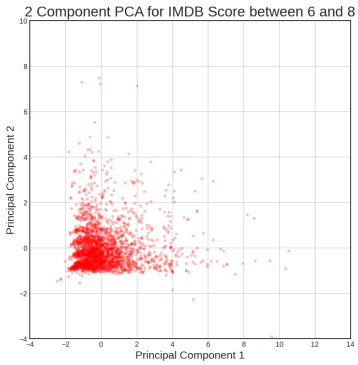
(b)



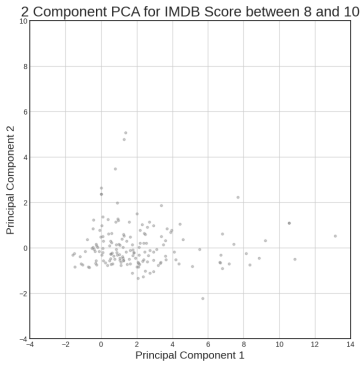
(c)



(d)



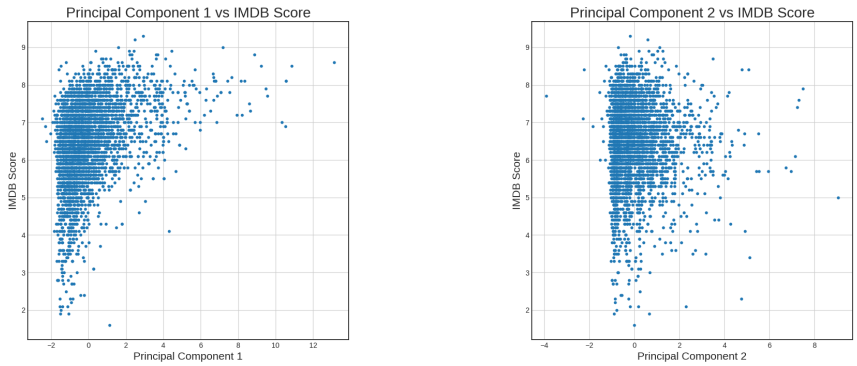
(e)



(f)

Figure 3: Plot of principal components for varying score ranges

Once again, we also plotted the principal components against the IMDB scores separately, as seen in Figure 4.



(a) (b)

Figure 4: Plot of individual principal components against IMDB score

Because there are so many data points overlapping, we decided to plot the different score ranges in different plots for easier comparison. We also plotted the individual principal components against the scores to see if there were any trends that could be used for prediction. Clearly, from figures 3, it can be seen there are more data points with high IMDB scores than there are with low IMDB scores. In addition, there seems to be a correlation with principal component 1 and the IMDB score as seen in figure 4. Therefore, this is our most likely candidate for the component we will use for IMDB score prediction. Perhaps, we could utilize some sort of regression method on this principal component 1.

3.2 Clustering

Once we performed PCA, we then looked at methods of clustering [2]. We applied k-means to cluster the data that we got from the PCA. In particular, we tried to find relevant clusters in the two-dimensional data containing the 2 principal components related to each movie. Since the goal was to find clusters that would easily map new movies to IMDB scores, we used a k value equal to the number of score ranges we used for figures 1 and 3 (i.e. $k = 5$). We ran the algorithm 10 times with different centroid seeds, using a relative tolerance of 10^{-4} with regards to inertia, and selected the best output in terms of inertia. All of this was performed using the Scikit-Learn KMeans function [1]. Following our calculation, we plotted the clusters using the principal components as axes.

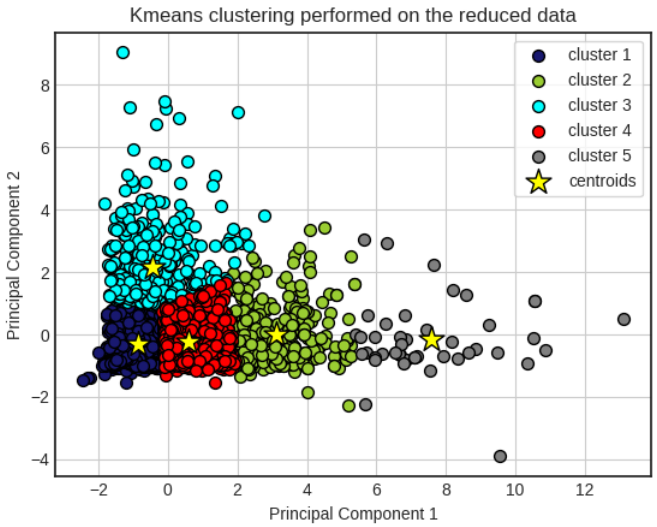


Figure 5: Clusters of the data obtained from PCA.

Comparing these clusters with the score ranges from Figure 3, we claim that there is no correlation between the clusters and the IMDB scores. Therefore, we need to search for alternative methods of predicting IMDB scores.

4 Conclusion

From our analysis, we concluded that k-means clustering was not very helpful and categorizing the data for prediction. On the other hand, PCA analysis could be useful for our prediction, if we use regression on principal component 1. Ideally, we would like to have a larger correlation for the principal components for a more robust prediction algorithm, but it seems like the features used as inputs do not have enough correlation with the output.

5 Source Code

The code we used for this analysis can be found here:
https://github.com/jam14j/BookScorePredictor

References

[1] Scikit-Learn Developers. *Scikit-Learn Documentation*. URL: `https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html#sklearn.cluster.KMeans.fit_predict`.

[2] Lorraine Li. *K-Means Clustering with scikit-learn*. June 2019. URL: https://towardsdatascience.com/k-means-clustering-with-scikit-learn-6b47a369a83c.

[3] mGalarnyk. *mGalarnyk/Python$_T$utorials*. URL: https://github.com/mGalarnyk/Python_Tutorials/blob/master/Sklearn/PCA/PCA_Data_Visualization_Iris_Dataset_Blog.ipynb.

[4] Yueming. *IMDB 5000 Movie Dataset*. Dec. 2017. URL: https://www.kaggle.com/carolzhangdc/imdb-5000-movie-dataset.