Analyzing Video Streaming Platforms

Team Stream

Group members: Huy Tran, Kelsey Kua, Kenneth Ven

Overview

Online video streaming is commonly used by people of all ages. One of the first big streaming platforms was Netflix, and after their success, other new streaming services popped up. Now there are several services to choose from, and it is not uncommon for an individual consumer to subscribe to multiple streaming services.

In this project, we will be exploring the availability of movies on the popular streaming platforms Netflix, Prime Video, and Hulu.

Our Approach

The question we want to answer is the following: **What kind of movie content does each major streaming platform tend to feature (e.g. genre, language, age group, year)?** To go about answering this question, we will be using public datasets found online. The current datasets that we will use can be found here from Kaggle (last updated: 5 months ago). This report explores the accuracy and intent of the current data.

From this dataset, we will examine the following variables:

1. Streaming platform
    a. Netflix
    b. Prime Video
    c. Hulu
2. Movie year
3. Age rating
4. IMDb score
5. Rotten Tomatoes score
6. Runtime
7. Genre
8. Primary country
9. Secondary country
10. Primary language
11. Secondary language

With these variables, we hope to identify content patterns within each platform to provide a high-level summary of movie content. The features we can control include the streaming platform, movie year, age rating, genre, countries, and languages. For example, we can see how the variety of movie year and age ratings differ by platform. The features we cannot control are ratings (IMDb and Rotten Tomatoes), which have been determined by users of these websites.

Data Pre-Processing

To process that data, we took the database on Kaggle and separated the database by its streaming platform to create three excel files (Netflix, Hulu, and Prime). Below are the steps we took to filter the data:

1. Delete the first column, ID, and Type
2. Filter the streaming platform that you want by making it be 0. This would create a sheet that contains movies that are not on that platform.
3. Delete the current sheet
4. Delete all the streaming platform type
5. Filter genres and delete all blanks
6. Filter age, IMDb, and runtime together to delete all blanks.
7. Freeze top row
8. Change age and IMDb to be recognized as a number

If you follow these steps, then it should recreate our current three excel files for each platform. Our goal for data preprocessing was to filter the original data by streaming platform and delete any information that was not needed.

Data Plots

Of the plots we have learned about, we think a scatterplot matrix and histograms are the most useful. We also plotted normal distributions of each feature for each class to determine if we could assume normality. These plots are described in the following sections.

Scatterplot Matrix

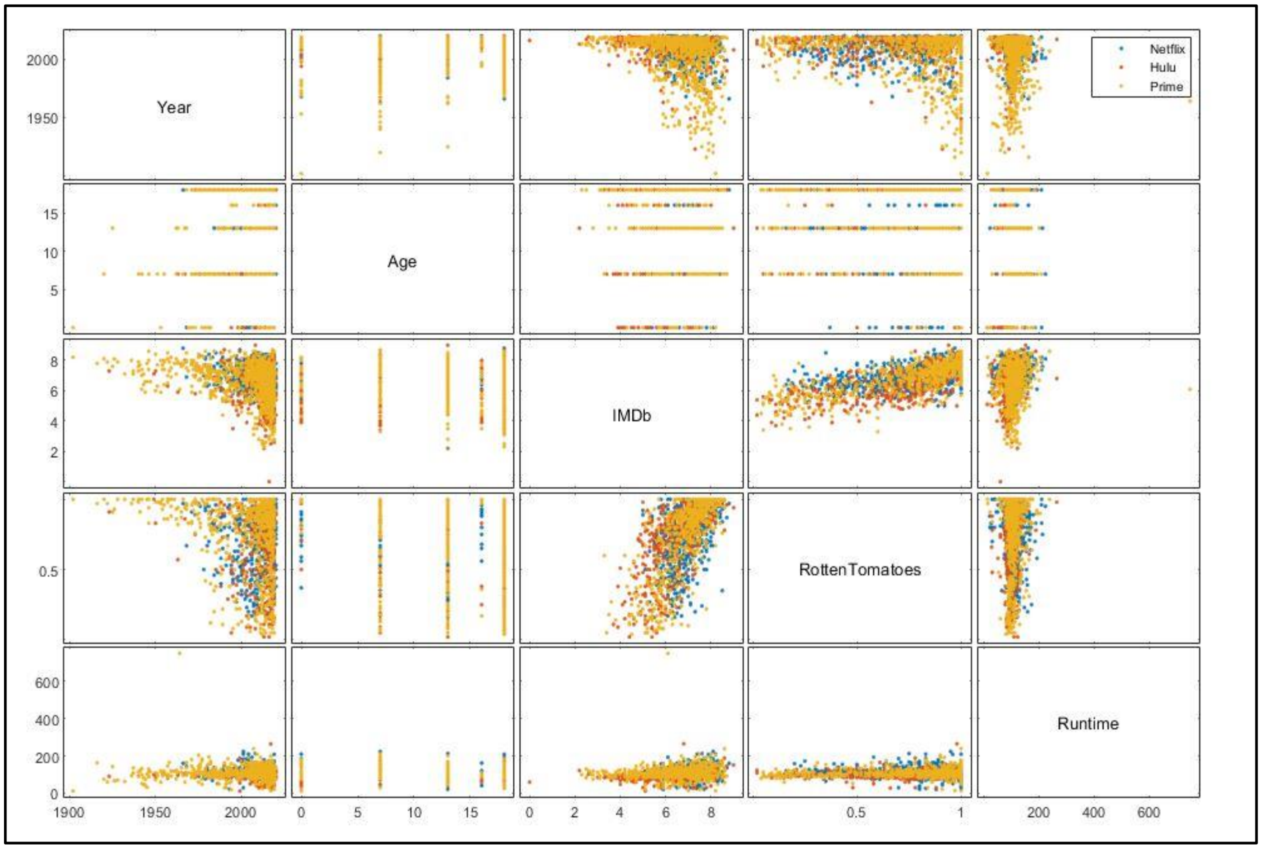


Figure 1: Scatterplot matrix of relevant features

The scatterplot matrix in Figure 1 shows that Netflix, Hulu, and Prime have a heavy correlation with each other. However, the reason that there is more data for Prime is that there is less missing data compared to the two other streaming platforms Which resulted in more Prime data appearing on the matrix. It seems that primes stretch further compared to the two other platforms as it covers more of the space, while Hulu and Netflix seem to be grouped at. This lets us assume that Prime video seems to cover a variety of movies, while Hulu and Netflix like to cover a specific type as shown when observing the relationship between Year. Also, it seems that all three platforms do not care about the Rotten Tomatoes and IMDb ratings as the plot stretches far.

Histograms

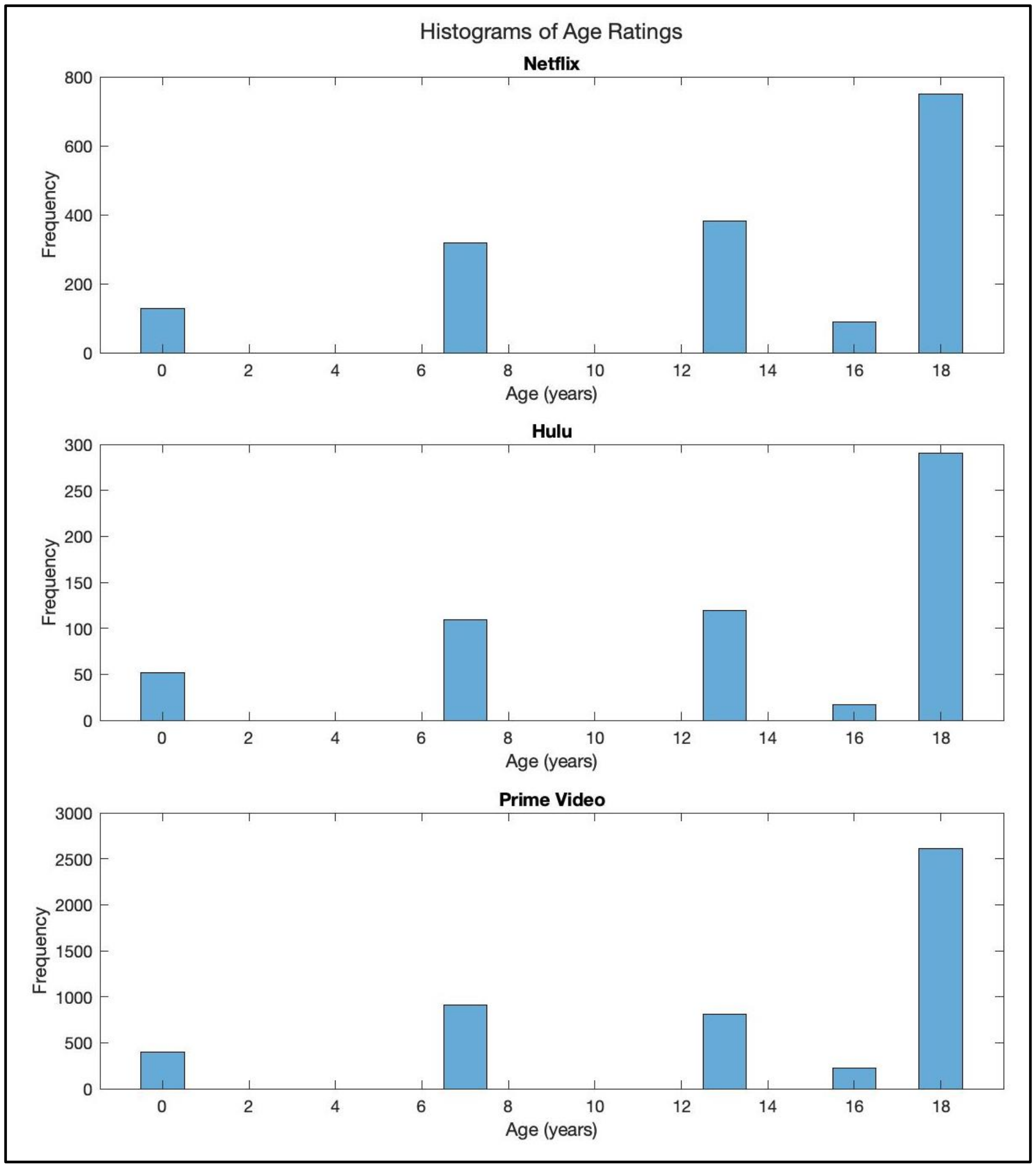


Figure 2: Histograms of age ratings by platform

The five age rating categories of films are as follows: all ages (which we have corrected to be 0+ years), 7+ years, 13+ years, 16+ years, and 18+ years. The histograms for each streaming platform look similar: The older the rating, the higher the frequency. Each platform has the greatest frequency of films in the 18+ category, meaning they tend to feature films for more mature audiences.

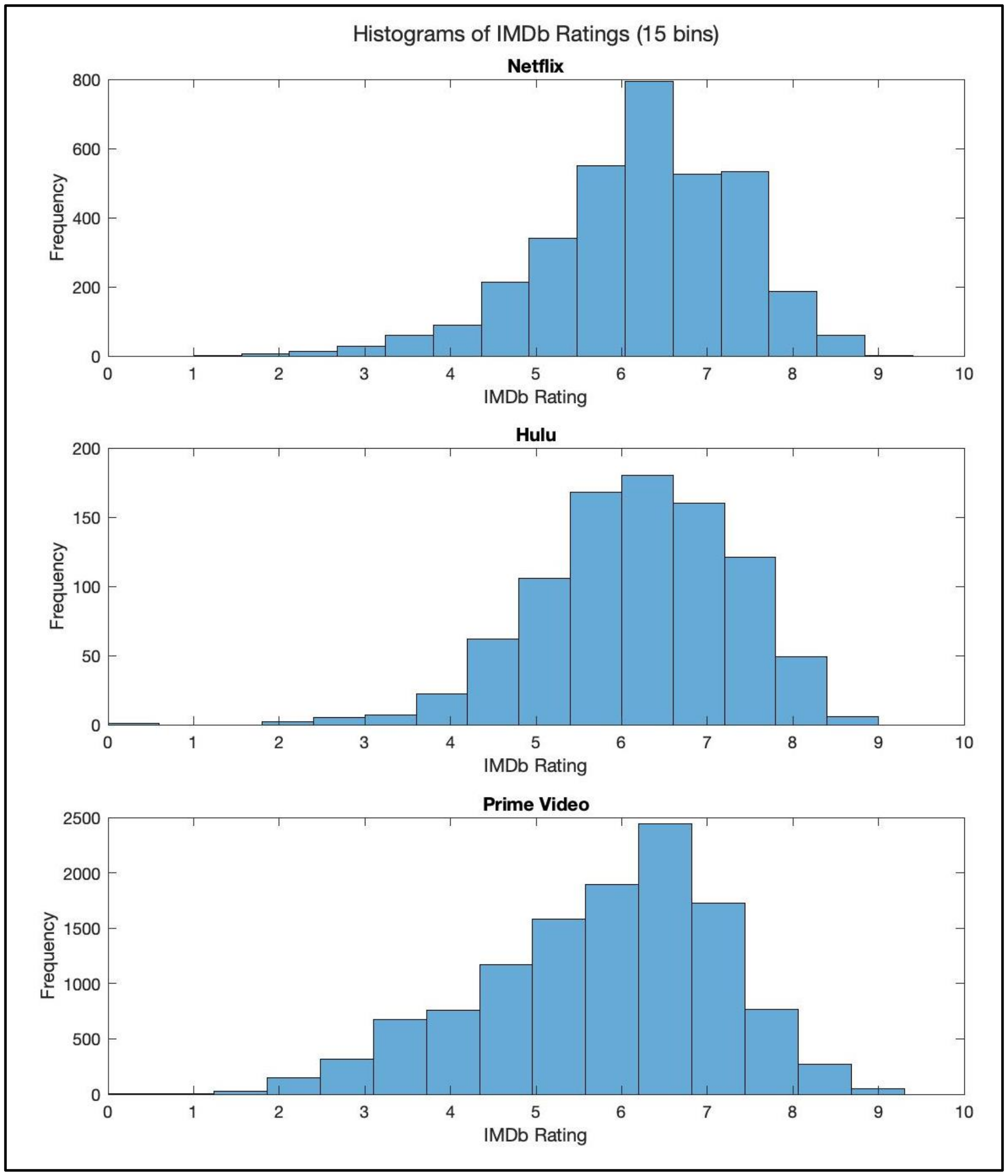


Figure 3: Histograms of IMDb ratings by platform

These histograms show the spread of films' IMDb ratings. For each platform, the majority of films have ratings between 5.5 and 7.5, which seems like a decently high rating. If we knew how films tend to be rated on IMDb (e.g. well-received movies tend to be rated between X and Y, mediocre movies tend to be rated between X and Y), we would have a better feel for the quality of films on each platform. To provide more context to the rating systems themselves, IMDb ratings can be contrasted with Rotten Tomatoes scores.

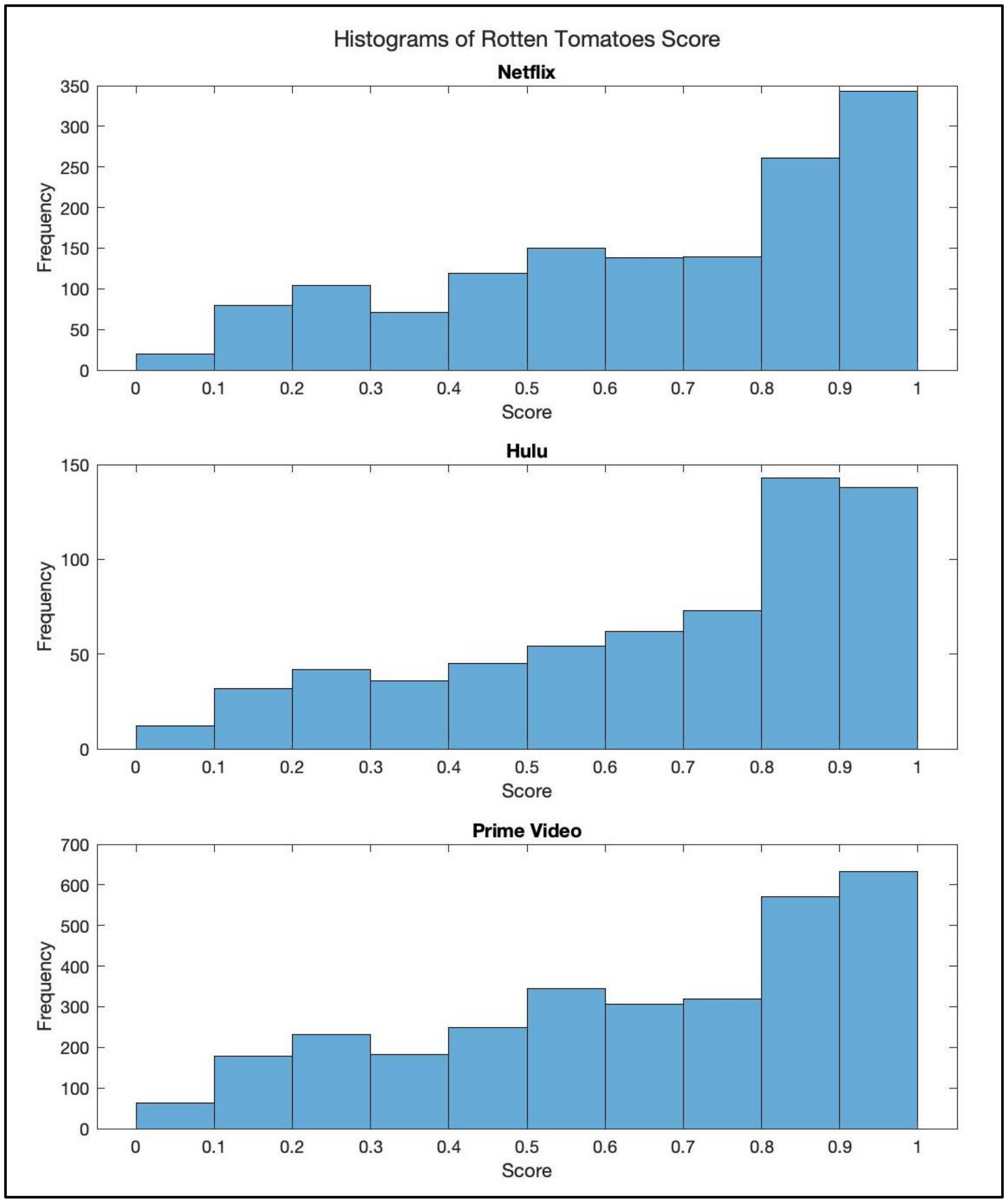


Figure 4: Histograms of Rotten Tomatoes scores by platform

These histograms show the distribution of films by Rotten Tomatoes score. An interesting part of this plot is that, for each streaming platform, the frequency of films stays somewhat consistent below 80%. A high frequency of films has scored between 80% and 100%, which indicates that they were very well-received. In contrast with the IMDb ratings, Rotten Tomatoes scores tend to be higher. Perhaps this means that highly regarded films have 80%+ approval ratings on Rotten Tomatoes but have lower IMDb ratings (between 6.0 and 8.0).

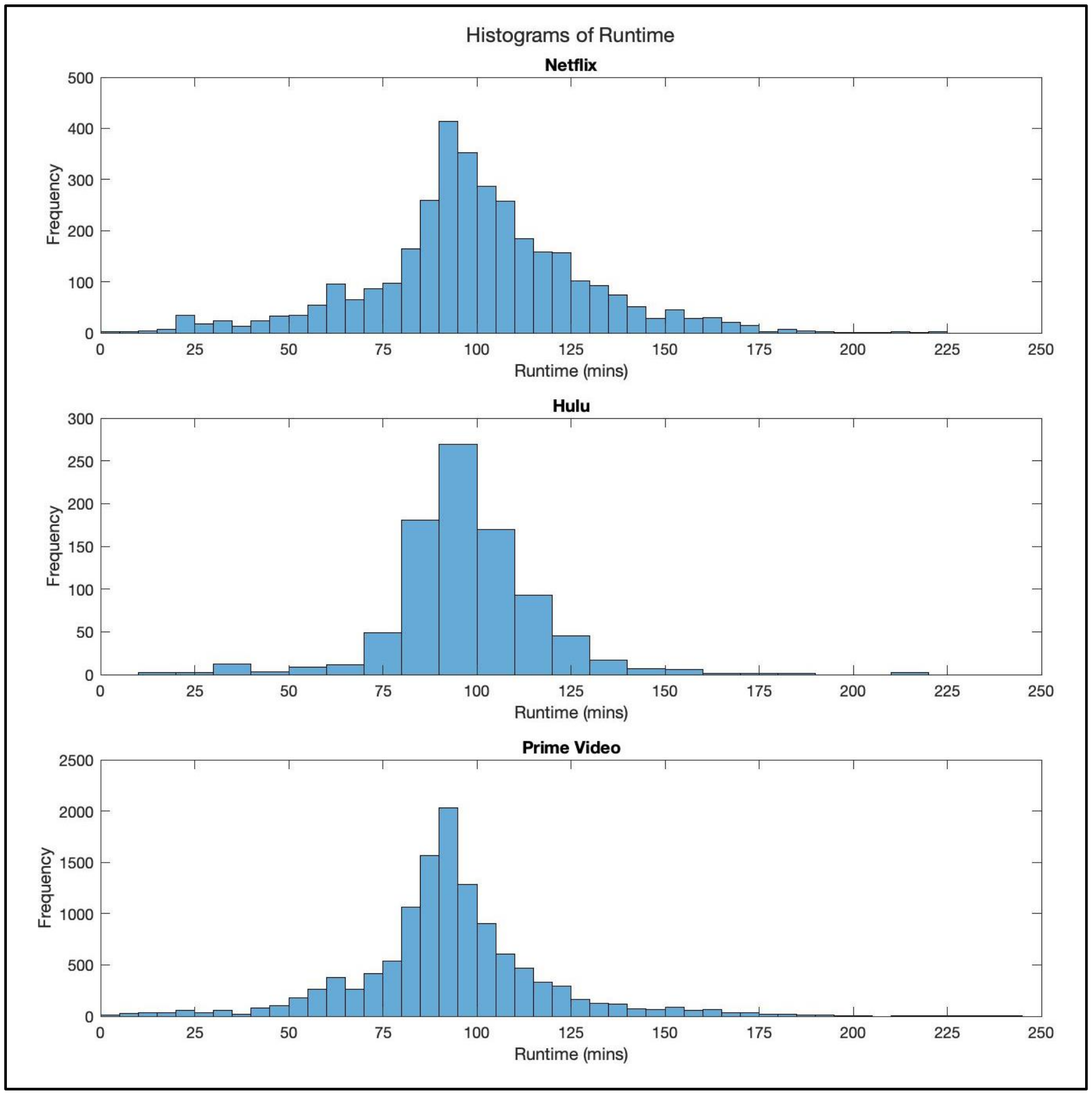


Figure 5: Histograms of runtimes by platform

These histograms show the distribution of films' runtimes in minutes. Prime Video had films beyond the 250-minute mark, but we set the x-axis limit to be 250 minutes to provide fair comparisons among the streaming platforms. The center of the distribution seems to be just under 100 minutes, which is between 1.5 and 2 hours. From personal experience watching films, this is not surprising. For Netflix and Prime Video, however, there are quite a few films past the 2-hour (120 minutes) mark, which is longer than most casually viewed films today.

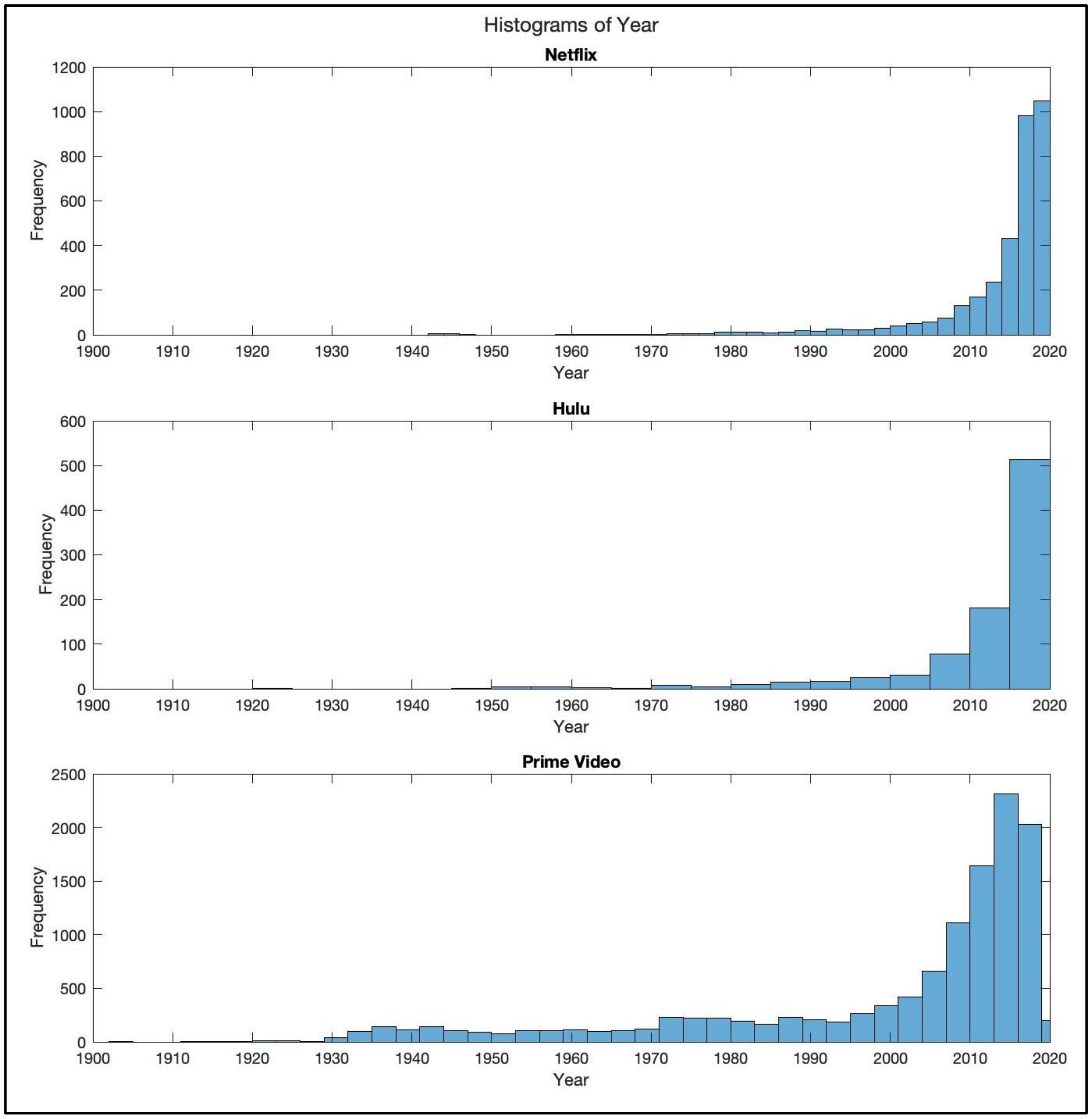


Figure 6: Histograms of film release year by platform

These histograms show the distribution of films by year released. On each platform, a heavy concentration of films was released between 2015 and the present. Compared to the other two streaming platforms, Prime Video features significantly more films between 1900 and 1980. This occurrence may be attributed to the fact that Prime Video has more observations in the dataset than the other two streaming platforms.

Normal Distributions

To determine if we can assume normality, dot plots and normal distributions were made for each class and each feature. Because the values of the dot plots often overlapped, the values of only one class were visible. To make all values visible, the dot heights of the classes were adjusted. Dot plots, however, are one-dimensional and have no value on the y-axis.

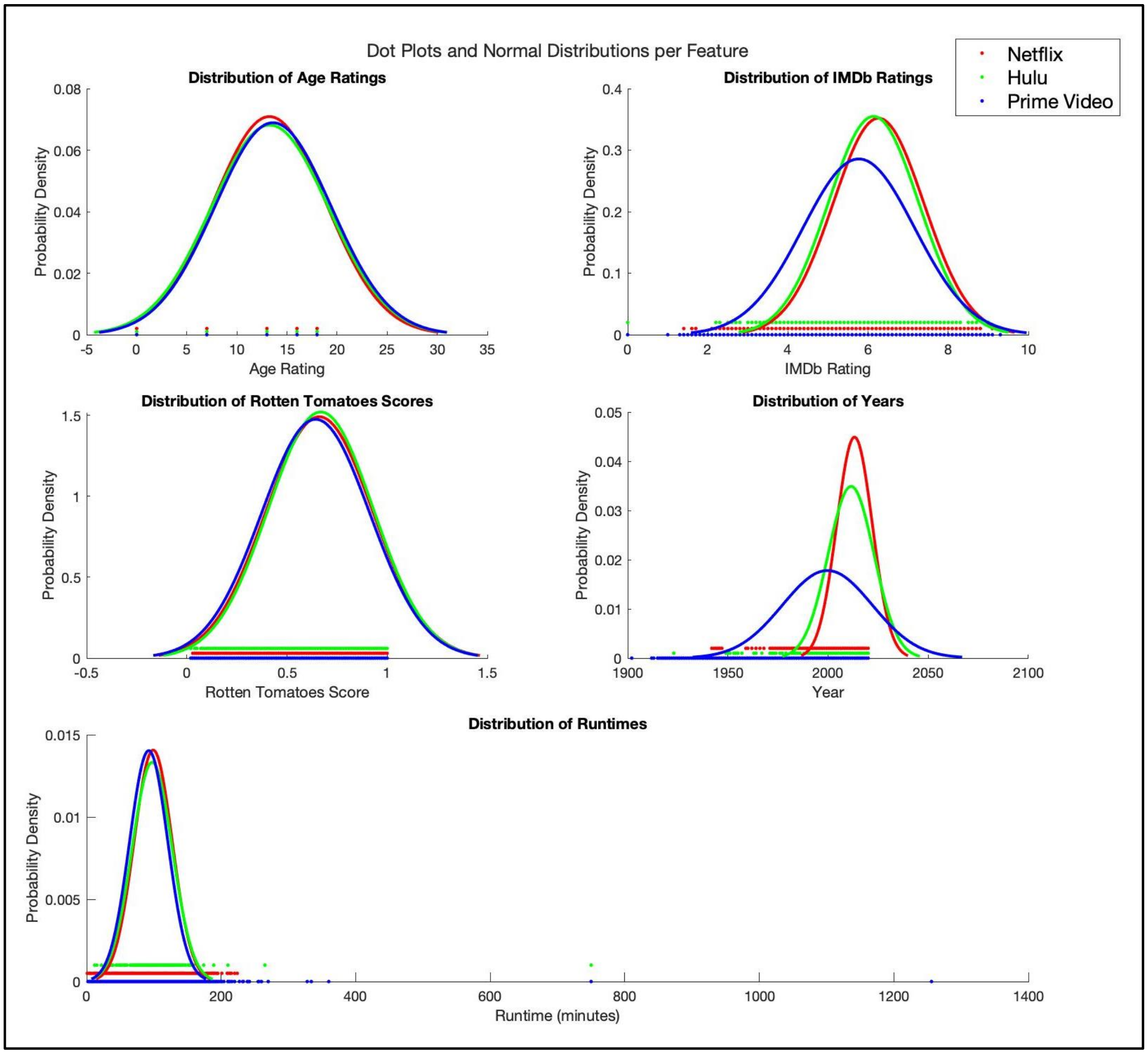


Figure 7: Dot plots and normal distributions of each feature

For almost every feature, the normal distributions of each streaming platform line up almost perfectly. This makes sense for the distribution of age ratings because there are only five possible values (0+, 7+, 13+, 16+, 18+), which can be seen in the dot plot values. This also makes sense for IMDb ratings and Rotten Tomatoes scores because the histograms for these features were very similar across all three platforms. The normal distributions for release year, however, have the most variation: Netflix and Hulu have similar distributions, but Prime Video's distribution is wider and centered about 15 years earlier. According to the histogram of the release year, Prime Video has a sizable number of older movies (between 1900 and 1980) compared to the other two platforms. Another interesting feature is runtime. According to the dot plot, Prime Video supposedly has films that run at 750 minutes and 1250 minutes. Including these points forces a zoomed-out view of the distribution, so we limited the x-axis to values between 0 and 300 minutes to get a clearer view as shown by Figure 8.

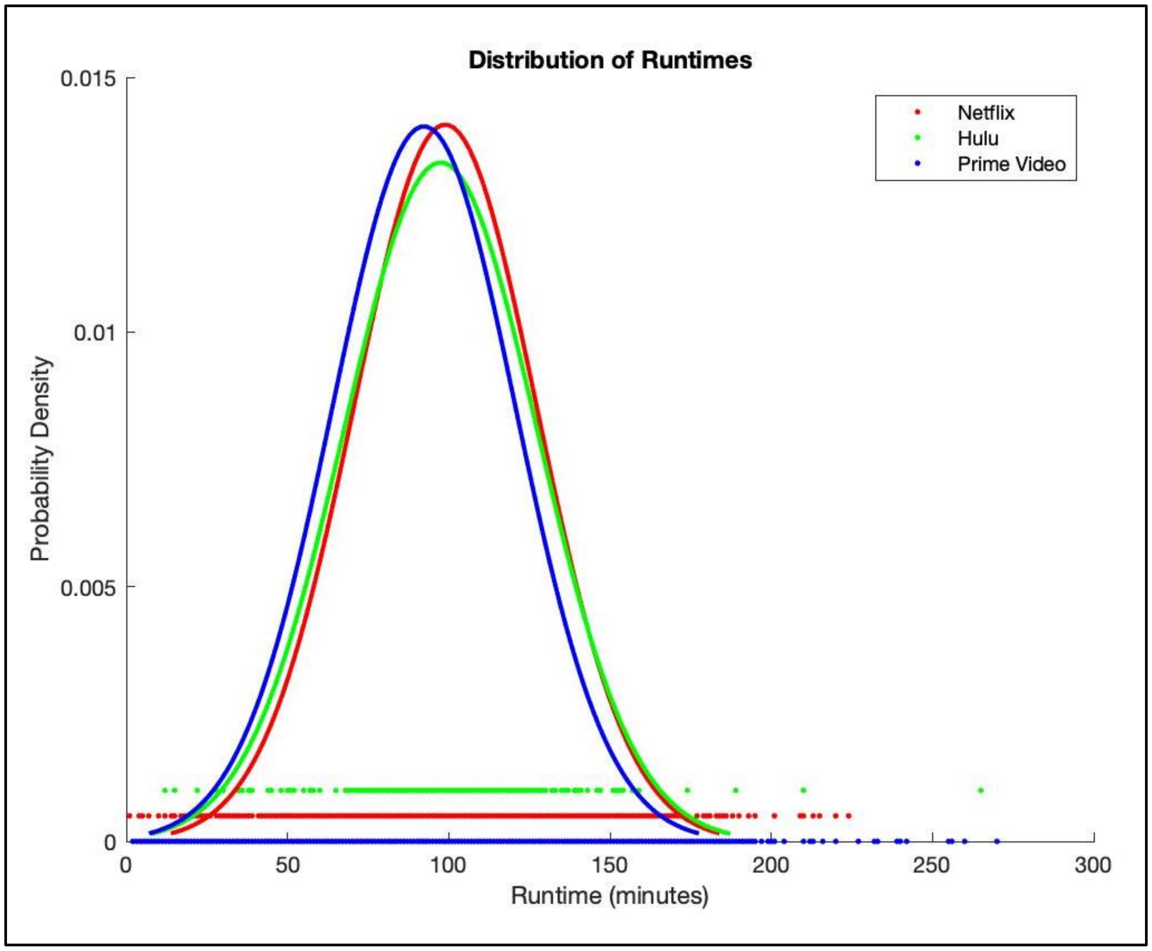


Figure 8: Dot plots and normal distributions of runtimes with x-axis limit

With this clearer view of the runtime distributions, we see that the distributions for each class line up almost perfectly.

From these dot plots and normal distributions, it is reasonable to assume normality for all features except release year. Prime Video's distribution for the release year may differ from that of the other two platforms because Prime Video had ~12,000 observations compared to Netflix's ~3,500 observations and Hulu's ~900 observations.

Independence of Data Observations

We are assuming that each class(Netflix, Hulu, Prime) has multiple related movies that they have in common. For example, Netflix and Hulu will have IP Man 3.

For this reason, there isn't independence between the observations. This is because the comparisons of the movies depend on whether or not the other platform streams the movie. If the streaming platforms were to have all unique movies compared to each other then we could assume that our observations are independent of one another.

However there will be occasions where some platforms have unique movies, these observations could be assumed as independent because there are no other observations to compare it with and are not dependent on other factors.

Other factors to be considered when figuring out if we can assume dependency of observations:

- If the movie is on the same platform as the other, then the specific movie is dependant on the other one(s) from the different platforms
- The rating of the movie could be dependent on which genre the movie is categorized as. This is assuming that some people like horror movies more than comedy movies and will rate them higher because of it.
- The runtime could be dependent on the year the movie was made. We are assuming that most movies made in the 1900s have shorter runtimes than movies made in the 2000s.
- The runtime of the movies can be dependent on the age range of the movie. We are assuming that the duration of kid movies tend to be shorter than adult movies.

However, some factors can be independent of others. For example, the ratings of the movies are independent of the runtime of the movie. We are assuming that people don't base their ratings on how short/long the movie was. Another factor that we can observe is that the ratings of a movie are independent of the year the movie was made. We are assuming that people don't base their ratings on how old the movie is.

In conclusion, even though there may be a few factors that could make our observations independent of one another, we assume that most of the observations are dependent on each other and their factors.

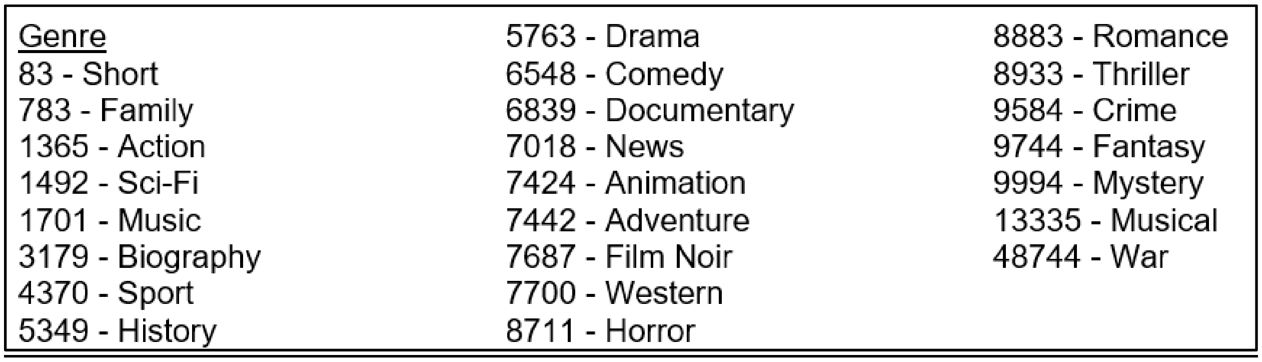
PCA

Additional Data Preprocessing

When preprocessing the data, we realized that we needed to convert some categorical words into numbers. The categories that we needed to change were countries, languages, and genres. There were some problems with the process as the three categories listed would have multiple categories in a cell. To fix this problem, we created a Matlab code-named “Preprocessing. mlx” that would take the genres and create additional rows with only one genre. For example, the movie “Strange Wilderness” has the genres adventure and comedy. The Matlab code would use that information to create two “Strange Wilderness” that would contain either adventure or comedy.

After creating a database with each genre listed, we need to figure out a way to change the other categories. To fix this problem, we use the Excel tools to create additional columns that would separate the other categories in the cells to a new column similar to the genre example. However, we would only take the first two countries and languages and bring over the first language into the second column if it only has one language to create a clean dataset. This approach would help create two features from a category that represents the primary and secondary type of the movie.

After cleaning the excel sheet, we created the Matlab code that would preprocess genre, countries, and languages to be converted into numerical values. When researching for values to represent a category, we find some databases that use numerical values to describe our categories. For genres, we use the Netflix categories database as Netflix has a numerical system to represent their genres. The list in Figure 9 shows which numerical values we choose to describe our categories from the Netflix database. For countries, we use country codes from the United Nations. For languages, we use the language ID numbers that are currently supported by the Windows database. With these three codepages, we converted all word categories to become numerical values.



Genre	5763 - Drama	8883 - Romance
83 - Short	6548 - Comedy	8933 - Thriller
783 - Family	6839 - Documentary	9584 - Crime
1365 - Action	7018 - News	9744 - Fantasy
1492 - Sci-Fi	7424 - Animation	9994 - Mystery
1701 - Music	7442 - Adventure	13335 - Musical
3179 - Biography	7687 - Film Noir	48744 - War
4370 - Sport	7700 - Western	
5349 - History	8711 - Horror	

Figure 9: Genre ID legend

Finally, we then clean our excel file by turning any percentage into numerical values. We also needed to change age as Matlab would not see their values as numbers. To solve this problem, we removed the “+” for each number. After everything is done, we would then combine all the streaming platform indicators as movies are recognized in a platform by having a “1” in their particular platform (ex. Netflix, Hulu, Prime, Disney Plus). After following the preprocessing, we would have a dataset that contains movies for their streaming platform with the information, except the Title, being a numerical value.

After encoding the additional features, we created a new scatter plot matrix containing all 10 features as shown in Figure 10. In this scatter plot, we were able to see the relationship between similar features. For example, the scatter plots for primary and secondary language all have similar shapes. The same occurs with primary and secondary countries. This indicates that these feature pairs may be redundant. Additionally, some scatter plots have distinct lines because they only contain a handful of unique values. For example, the age rating has five distinct categories. Outside of these linear patterns, there is no clear clustering.

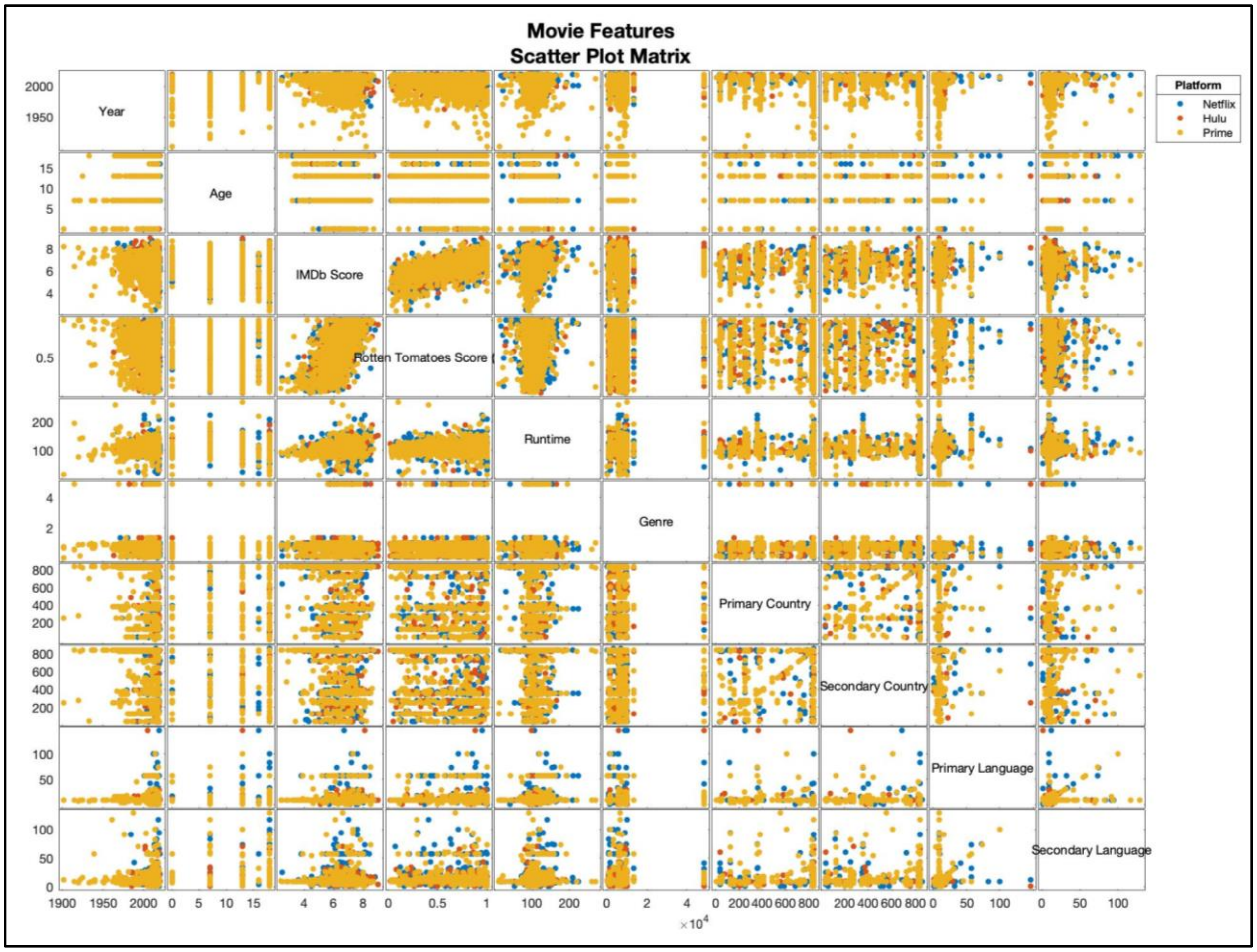
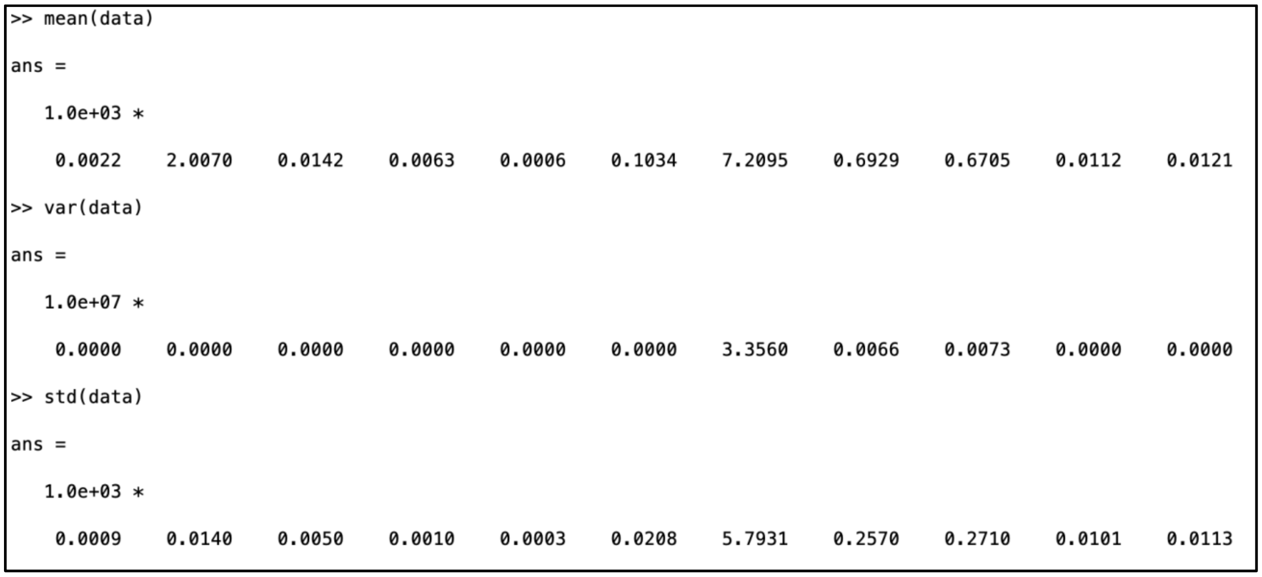


Figure 10: Scatter plot matrix with new encoded features

Basic Statistics

To begin our analysis, we looked at the basic statistics of our data (mean, variance, and standard deviation). The results shown in Figure 11 indicate a variety of means, variances, and standard deviations. The feature with the greatest variance and standard deviation is runtime; in the previous section, we observed the wide variability of runtime across all films, so a wide variance and standard deviation are to be expected.

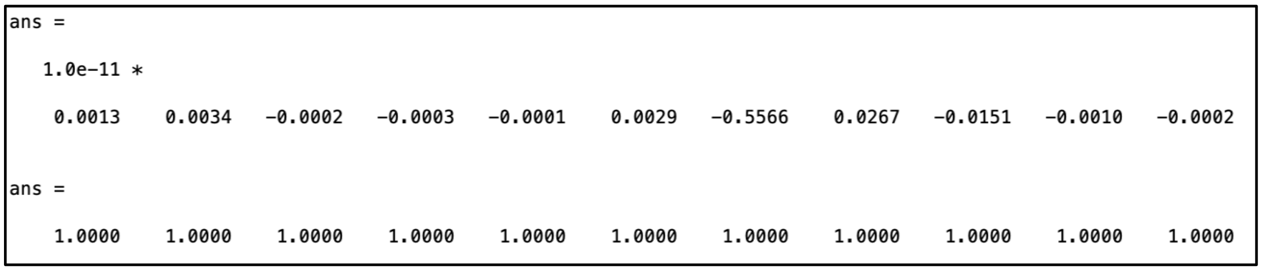
Based on the normal distributions from the previous section, we would also expect the release year to have high variability, however, this is not apparent in these statistics



```
>> mean(data)

ans =

   1.0e+03 *

    0.0022    2.0070    0.0142    0.0063    0.0006    0.1034    7.2095    0.6929    0.6705    0.0112    0.0121

>> var(data)

ans =

   1.0e+07 *

    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    3.3560    0.0066    0.0073    0.0000    0.0000

>> std(data)

ans =

   1.0e+03 *

    0.0009    0.0140    0.0050    0.0010    0.0003    0.0208    5.7931    0.2570    0.2710    0.0101    0.0113
```

Figure 11: Basic statistics before centering and scaling

After calculating these statistics, we centered all features at 0 and normalized them so that they are more equally comparable. Our results in Figure 12 confirm that the data was centered and scaled properly since all the means are 0 and all the variances are 1. Looking at these results, we can see that there are redundant features. It seems that from features 1 to 6 and 9 to 11, they were redundant values in our data. I would consider features 1 and 4 as statistically useless features as the standard deviation is close to 0. Some assumptions we can make of our data is that our data is more likely to cluster together as they would have similar values from each other as some columns have redundant values.



```
ans =

   1.0e-11 *

    0.0013    0.0034   -0.0002   -0.0003   -0.0001    0.0029   -0.5566    0.0267   -0.0151   -0.0010   -0.0002


ans =

    1.0000    1.0000    1.0000    1.0000    1.0000    1.0000    1.0000    1.0000    1.0000    1.0000    1.0000
```

Figure 12: Mean and variance after centering and scaling

Scree Plot

To determine how much information each feature contributes, we created a scree plot/cumulative scree plot as shown in Figure 13. The cumulative scree plot has a roughly linear shape, which means that the features contribute a uniform and a somewhat equal amount of information.

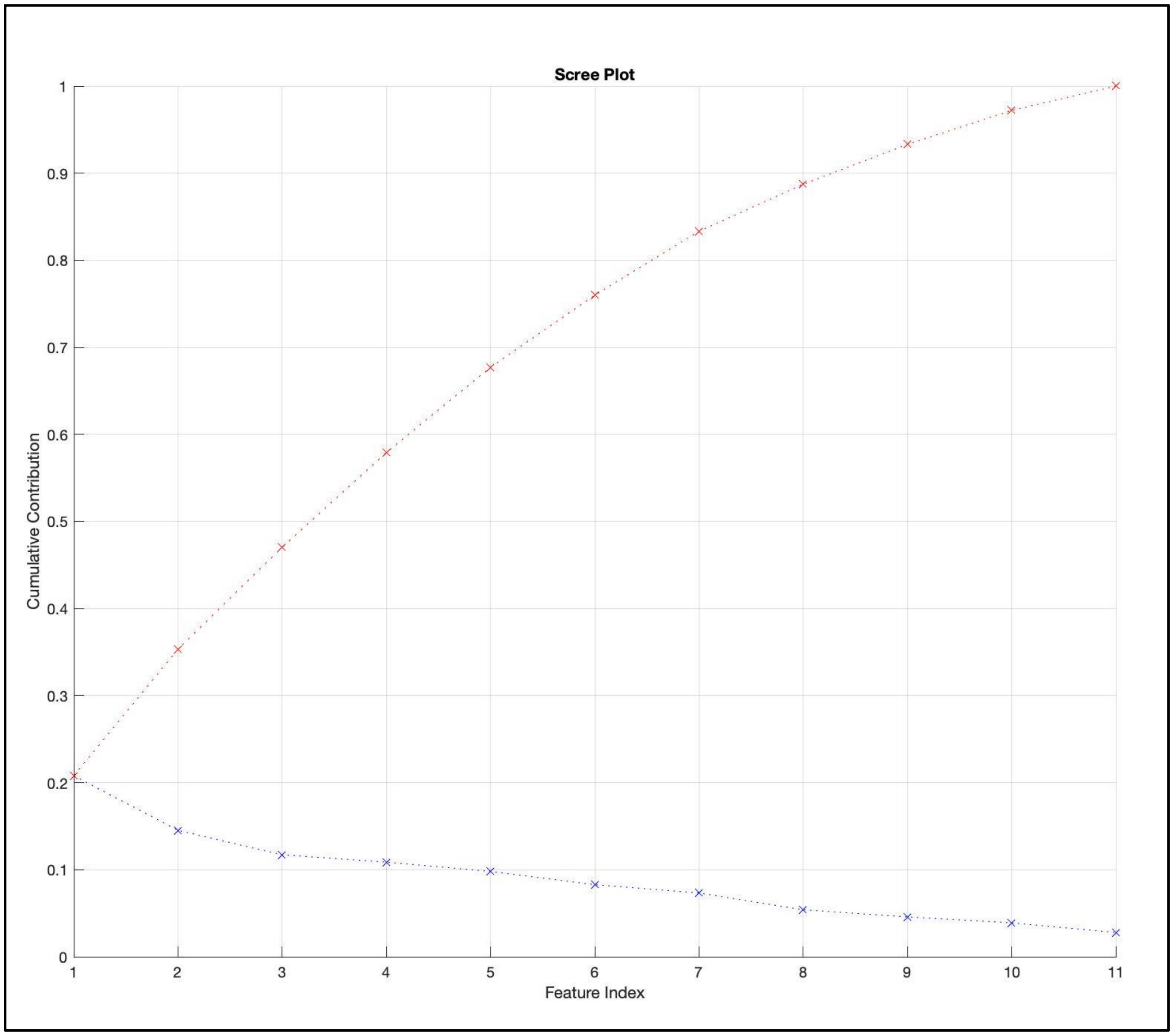


Figure 13: Scree plot of all features

Loading Vectors

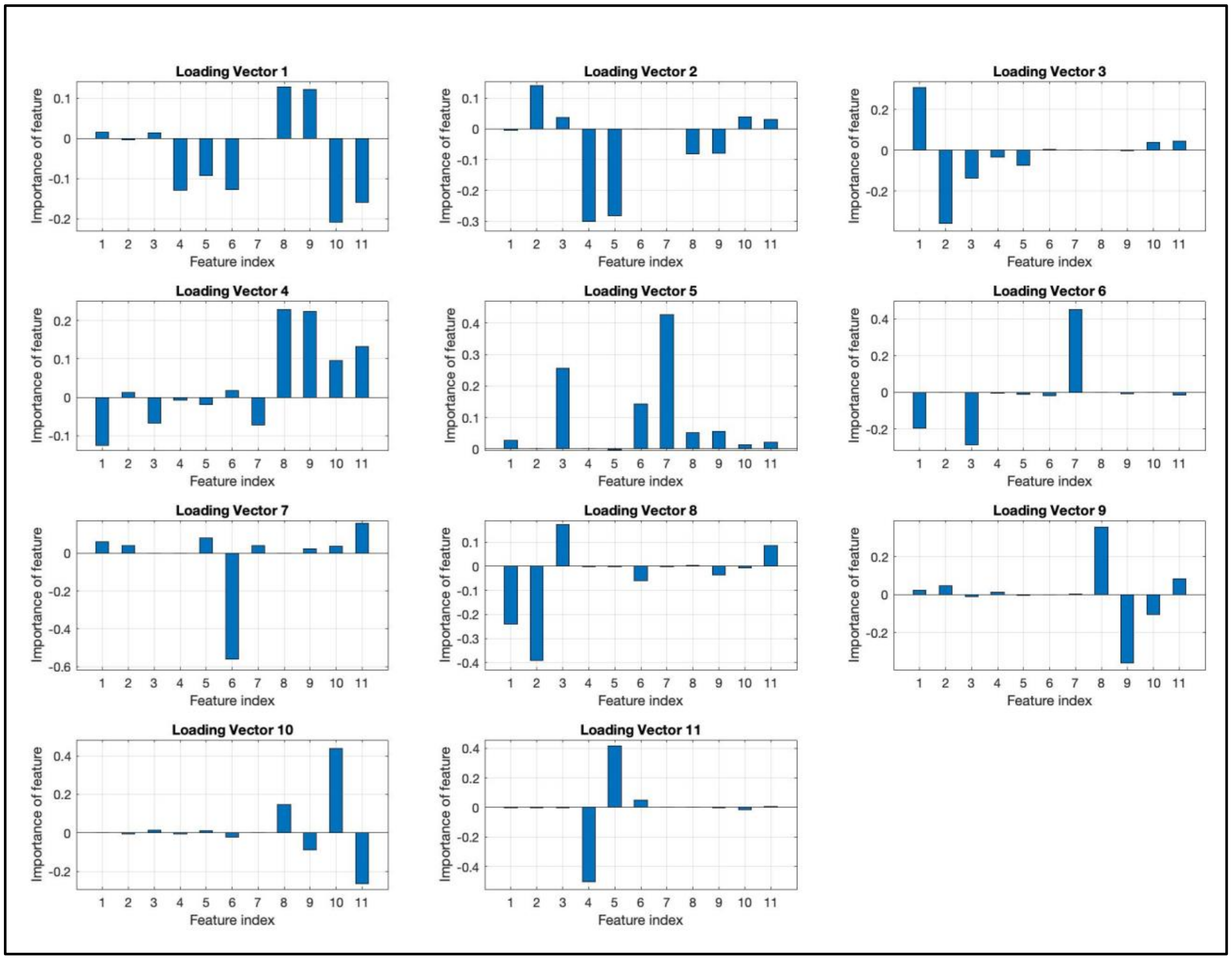


Figure 14: Loading Vectors 1-11

Loading Vector 1: Features 10 and 11 are the biggest contributors
We noticed in the first loading vector that features 10 and 11 have the largest magnitude out of the features. What we interpret from this is that they are the biggest contributors of the vector row to forming the new dimension. However, we don't interpret those features as the main contributors because some other features in the loading vector have a noticeable size magnitude. Features 4, 5, 6, 8, and 9 also have a larger magnitude compared to the other features. Our interpretation of this is that these features also contribute some information to forming the principal component. We also noticed that features 4, 5, and 6 might have some correlation with one another because they are grouped and have close magnitude sizes, at least 4 and 6. Another thing that we noticed is that features 8 and 9 also have about exact values for their magnitude, our interpretation of this is that they share some sort of correlation. Looking back at the scree plot, this loading vector contributes around 20% of the information to the principal components and provides the most information compared to the other loading vectors.

Loading Vector 2: Features 4 and 5 are the biggest contributors

The main thing we noticed in this loading vector is that features 4 and 5 have a large magnitude. Also, their magnitude is about the same as each other. Our interpretation of this is that they are the main contributors to the principal component and they may be correlated to each other as well. Even though features 8, 9, 10, and 11 have small magnitudes, each pair has around the same magnitude size. Meaning features 8 and 9 have about the same magnitude size and features 10 and 11 have about the same magnitude size. Again what we interpret from this is that 8 and 9 may be correlated to each other as well as 10 and 11. As this loading vector is near the beginning of the scree plot, this also provides some important information for the principal components.

Loading Vector 3: Features 1 and 2 are the biggest contributors
The main thing we notice from loading vector 3 is that features 1 and 2 have a large magnitude. What we interpret from this is that they contribute the most to forming the new dimension and provide the most information to the principal component. Another thing we noticed is that those two features have about the same size magnitude however they are in the opposite direction. So, our interpretation of this is that they have an inverse correlation. We also notice that again, features 10 and 11 have about the same size magnitude but it is quite small so there might be some correlation but doesn't provide too much information.

Loading Vector 4: Features 8 and 9 are the biggest contributors
What we noticed in this loading vector is that features 8 and 9 have a large magnitude and because of this we interpret those two features as being the main contributors to the principal component. Again, we notice that features 8 and 9 have about the same magnitude size and because of this we interpret that they are correlated with each other. From looking at the first few loading vectors, we are noticing a pattern that some pairs of features have a large magnitude and about the same size magnitude which we interpret as them being correlated to each other. Also, the information that has been provided from the prior loading vectors is quite important because up until now they have provided and contributed about 60% of the information to the principal components.

Loading Vector 5: Feature 7 is the biggest contributor
The main thing we noticed in this loading vector is that feature 7 has a large magnitude compared to the other features. Our interpretation of this is that this feature is the biggest and main contributor to the principal component. Feature 3 also has a noticeable size magnitude so we also interpret that feature to contribute some information, however it doesn't share the same importance as feature 7 in this loading vector. Also, we see another occurrence of features 8 and 9 sharing the same size magnitude and again we interpret these two features as having some correlation, however, their magnitude is quite small so it doesn’t seem to provide much important information to the principal component.

Loading Vector 6: Feature 7 is the biggest contributor
We noticed that in loading vector 6 most features have little to no magnitude and feature 7 has a large magnitude. Our interpretation of this is that feature 7 is the main contributor to the principal component as well as the other features don’t contribute much information and aren’t as

important. However, features 1 and 3 have larger magnitudes compared to the 0 magnitude features but we interpret them as not as important in this vector as well as contributing some but not much information compared to feature 7. Those two features also aren't correlated because their magnitudes aren't quite the same and we haven't seen a pattern where we've seen them being correlated. Looking back at the scree plot, this is where the loading vectors are starting to only provide less than 10% of the information to the principal components.

Loading Vector 7: Feature 6 is the biggest contributor
The main thing we noticed in this loading vector is that all the features except for feature 6 have a very small magnitude. With feature 6 having a large magnitude and no other feature having much of a magnitude, we interpret that feature 6 is the main contributor to the principal component and provides the most information to the forming of the new dimension.

Loading Vector 8: Features 1 and 2 are the biggest contributors
In this loading vector, we noticed that the features with the largest magnitude are 1 and 2. What we interpret from this is that these two features are the main contributors to the principal competence. This is also because most of the other features in this loading vector don't have many magnitudes. If there were any other important features for this loading vector, it would most likely be feature 3 because it has a medium-sized magnitude. Our interpretation of this is that it contributes to some information but is not as important compared to the other two features.

Loading Vector 9: Features 8 and 9 are the biggest contributors
The main thing we noticed in this loading vector is that features 8 and 9 have a large magnitude and the other features have little to no magnitude. Our interpretation of this is that features 8 and 9 are the main contributors to the principal component and provide the most information. However, we did notice that those two features are inversely correlated because they have the same size magnitude but are opposite directions of one another. Even though the magnitudes are small we noticed that features 10 and 11 are also inversely correlated because they have the same situation as the other two features. However, their magnitudes aren't as large compared to features 8 and 9 and so they don't contribute much to the principal component. Also, when we look back at the scree plot, at this point forward the loading vectors are not providing much information to the principal components and the transforming of the new dimensions because loading vector 9 only provides around 5% of the information.

Loading Vector 10: Features 10 and 11 are the contributors
The main thing we noticed in this loading vector is that it is very similar to loading vector 9 but the magnitudes for features 8&9 and 10&11 are swapped. Meaning that our interpretation for this loading vector features 10 and 11 are the main contributors to the principal component as well as being inversely correlated. Same as those two features, features 8 and 9 are again inversely correlated but not having as large of a magnitude this time.

Loading Vector 11: Features 4 and 5 largest contributors

The main thing we noticed in the last loading vector is that features 4 and 5 have a large magnitude and the rest of the other features have little to no magnitude. Our interpretation of this is that features 4 and 5 are the main contributors to the principal component and provide the most information. However, the information that is provided isn't very important or relevant because on the scree plot loading vector 11 only provides around 3% of the information. The last thing we noticed is that features 4 and 5 are also inversely correlated because they have the same size magnitude but are in different directions.

General analysis:
One main pattern that we noticed from looking at the loading vectors is that there were a few occasions where pairs of features were correlated with one another. More specifically, there were a few times where features 4&5 were correlated, features 8&9 were correlated and features 10&11 were correlated. Our interpretation of this is that since there are multiple occasions where the same features are correlated to one another, the features in the data itself are somehow correlated. Looking at the data itself features 4 and 5 are the IMDB and Rotten Tomato ratings for the movies, so it would make sense that we are seeing a pattern of features 4 and 5 and the loading vectors being correlated.

However, we also noticed that sometimes the pairs had an inverse correlation. These occurrences of the features being inversely correlated were in the later loading vectors. Our interpretation of this is that the features are providing redundant data for that principal component. But since this is in the later loading vectors, we assume that the information isn't that relevant or important because it is listed in the scree plot that they are only providing around 5% of the information to the principal components and the forming of the new dimensions.

So we interpret this as generally the features that were correlated most of the time are the ones that are the main contributors. This is also because they tended to have large magnitudes as well. Also, there was a pattern where feature 7 had a large magnitude in some of the loading vectors. Our interpretation of this is that feature 7 is also a big contributor to the information for the principal components and for forming the new dimensions.

Score Plots

To gauge the effectiveness of performing SVD, we created a 2D scatter plot of the features in the transformed space, Ur. The scatter plots are shown in Figure 15, and there is much more distinguishable clustering compared to the scatter plots in the original space. We no longer see a series of lines like in the original scatter plots; instead, we see blobs separated by class (streaming platform). PC7-PC10 has points that consistently fall outside the clusters, and this is due to how the features were encoded. These values are not in sequential numerical order, so consistent occurrences of particular data values lead to consistent appearances in the transformed space.

Though the data does appear to be better clustered in the transformed space, there are no immediately clear trends in the data. The scatter plots that stand out the most are those belonging to PC9: Every scatter plot is clustered in a tight line with some distinct data points outside of the line. This first round of PCA yielded promising results, and additional rounds of PCA can better cluster this data.

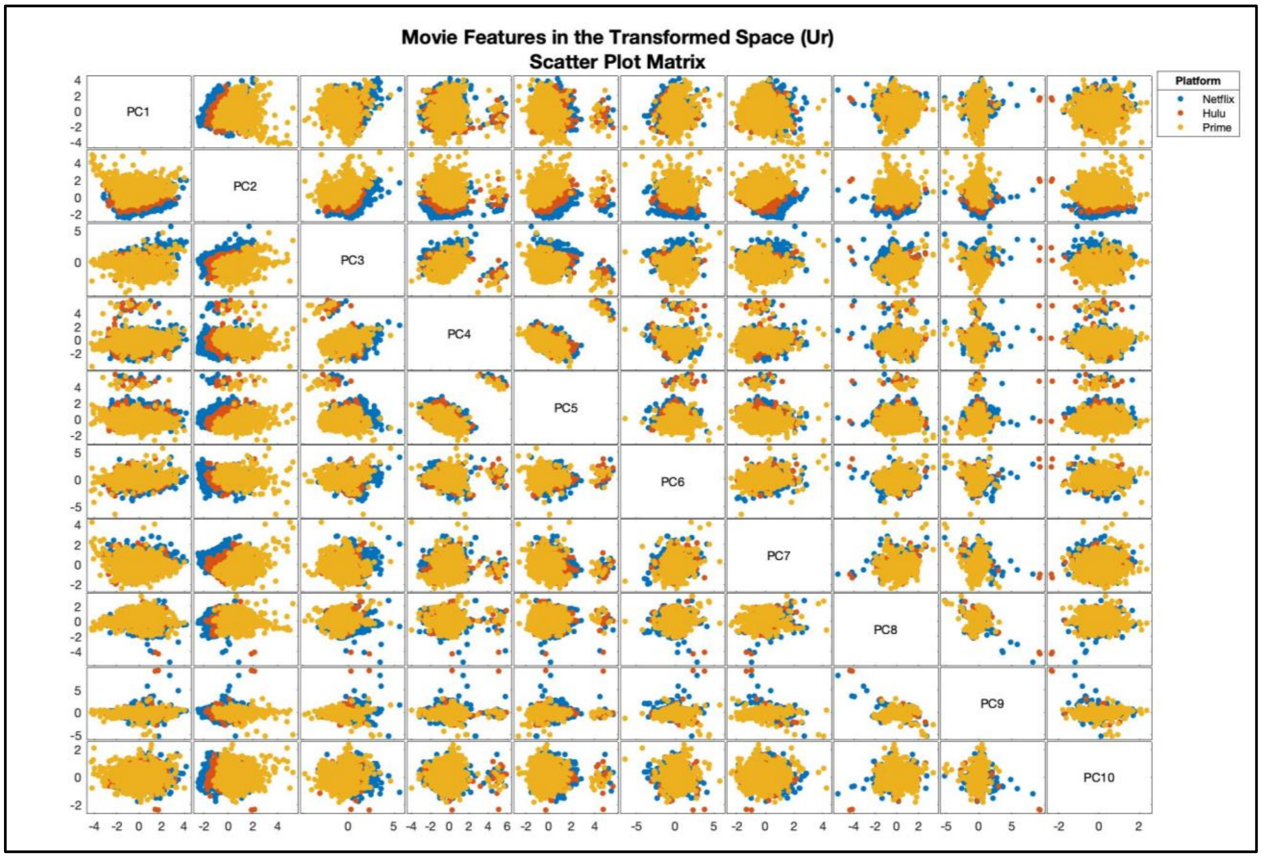


Figure 15: Scatter plot matrix in the transformed space (Ur)

Finally, we made a series of 3D scatter plots to better visualize clustering in the transformed space. These scatter plots are shown in Figures 16-19.

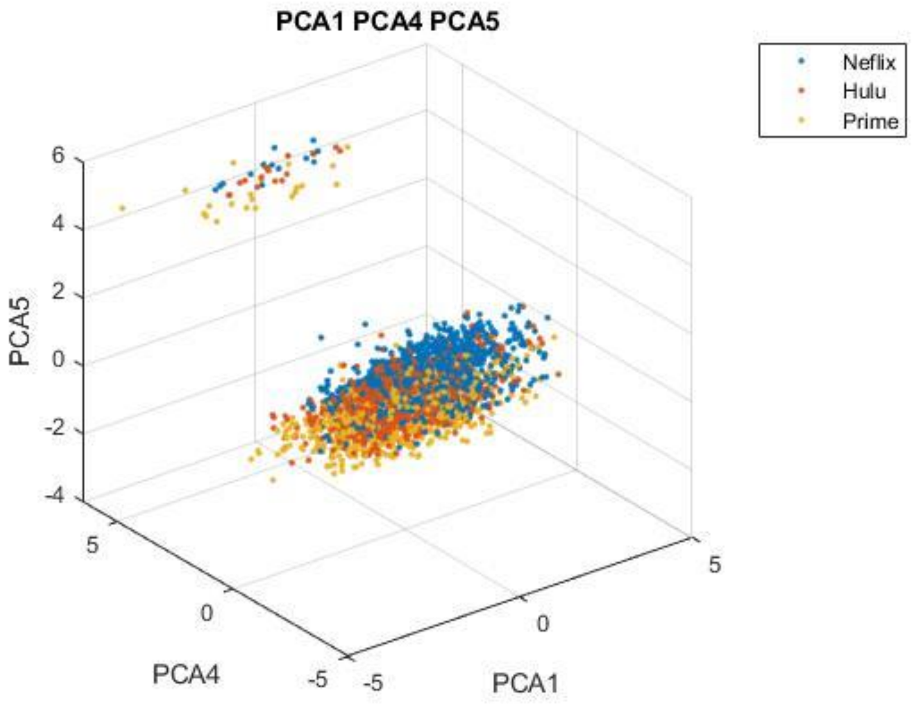


Figure 16: 3D scatter plot of PC1, PC4, and PC5

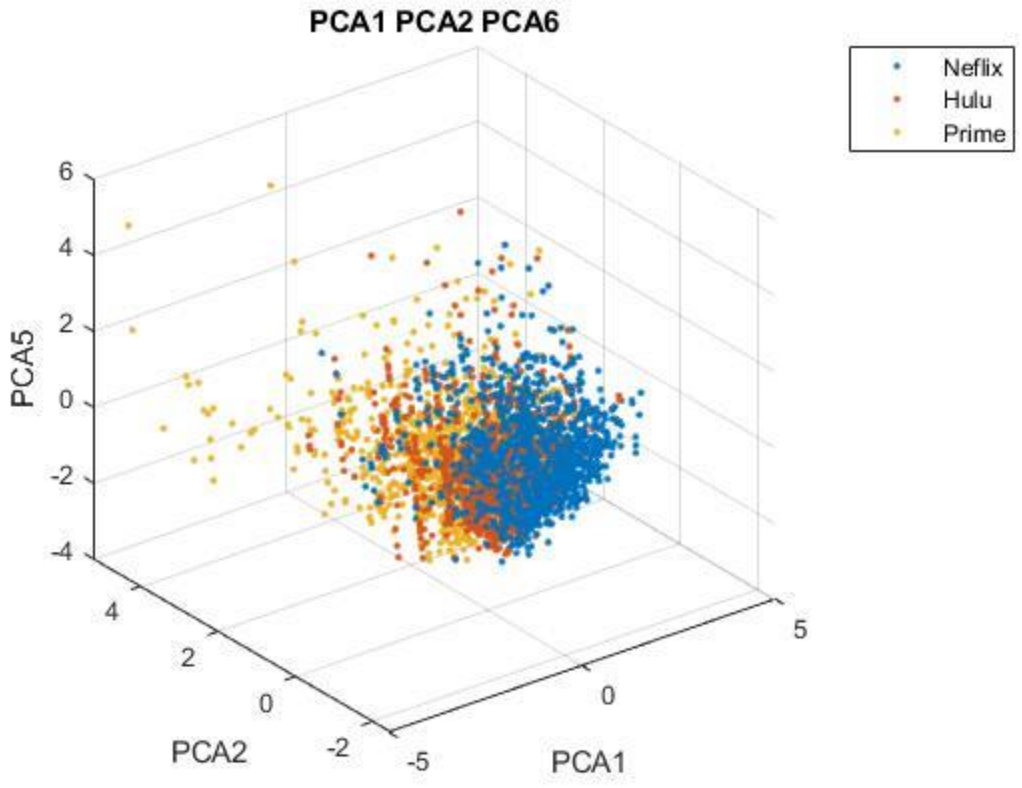


Figure 17: 3D scatter plot of PC1, PC2, and PC6

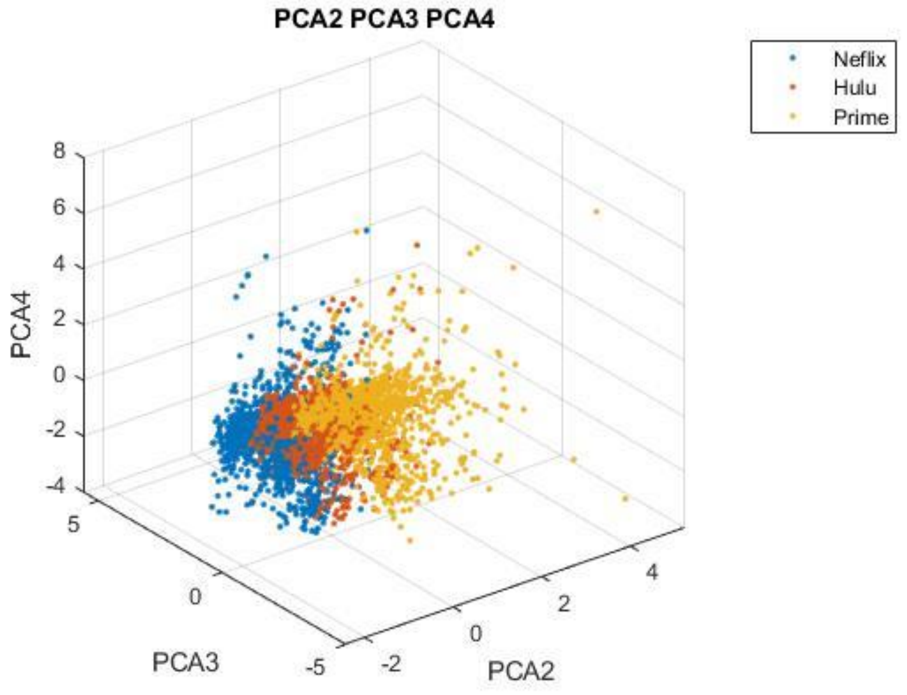


Figure 18: 3D scatter plot of PC2, PC3, and PC4

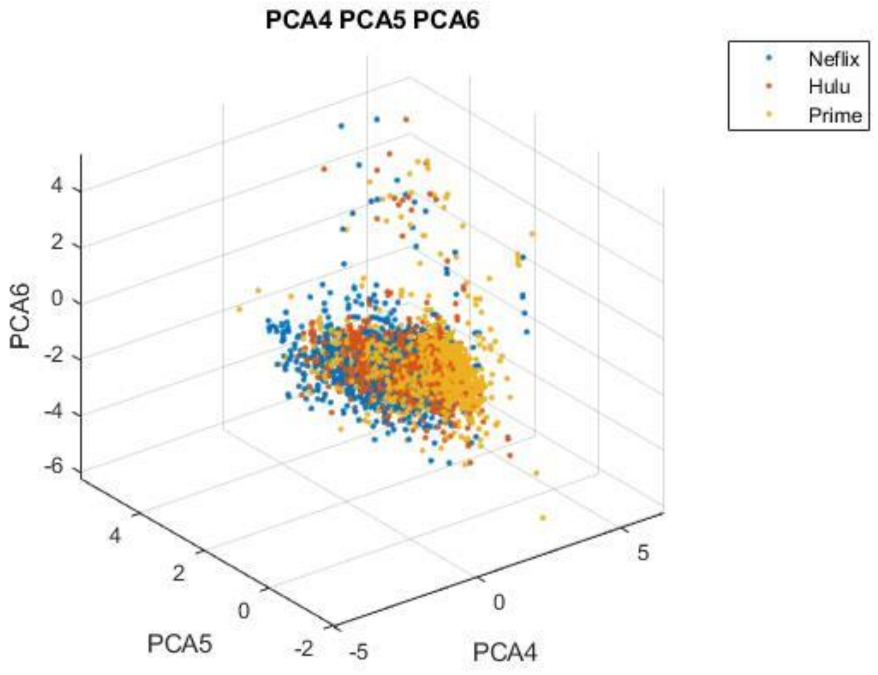


Figure 19: 3D scatter plot of PC4, PC5, and PC6

General Analysis for 3D plots:

In the first graph, the main thing we noticed is that there's a clear clump of the data around the middle area of the graph. However, there are some outlier data points from each platform, mostly from Prime, that are not near the clump of data. Our interpretation of this is that in general, all three platforms contributed around the same data for its loading vectors 1, 4, and 5, but had a few differences in the importance of the data they provided to the forming of the dimensions.

For the rest of the graphs, we noticed about the same pattern as we did in the first 3D plot where there is a clear clump in the data and then some data points from each platform out of the clump as outliers. Again, we interpret this as the three different platforms providing about the same importance of information but then have some occurrences where it differs from each other, separating them from being the same as one another.

Final Analysis:

Overall, we saw a general pattern of there being more clumping of the data compared to the 2D scatter plots we have been analyzing in our prior data. Even though we can generally see an overall clumping of the data in the 2D scatter plots, the analysis of looking for clumping of the data in the 3D plots is much easier to visualize. By also looking at the data in the 3D plots, we can better pinpoint the number of outliers in the PCA data.

From the 3D plots, it is easier to see that Netflix and Prime Video tend to have data points outside of the clusters compared to Hulu. Prime Video tends to have more scattered data points compared to Netflix, which is evident in Figures 17 and 18. This occurrence is supported by what we saw in the normal distributions of each feature: Prime Video had greater variability in release year and runtime compared to the other two platforms.

Using the Bagged Tree Classifier

Our group decided to use supervised machine learning techniques as we had information that we could input into our data to have it learn and predict future data. The technique we chose to use was the ensemble method, bagged trees. Ensemble methods combine several decision trees to produce better predictive performance than utilizing a single decision tree. The main idea of an ensemble model is a group of weak learners come together to form a stronger learner. Bagged tree's goal is to reduce the variance of a decision tree. The idea is to create several subsets of data from random training samples and repeat the process several times to train the many decision trees. This results in an ensemble of different models that averages together all the predictions from the different trees. We believe the bagged tree method can work well on our data because it may better capture data in less pronounced clusters.

Classification of Original Data (10 features)

From running the bagged tree classifier with our original data, we received an 81.5% prediction accuracy. We think that this classifier is decently accurate, however, it isn't as good as it could be because 20% of the data was misclassified.

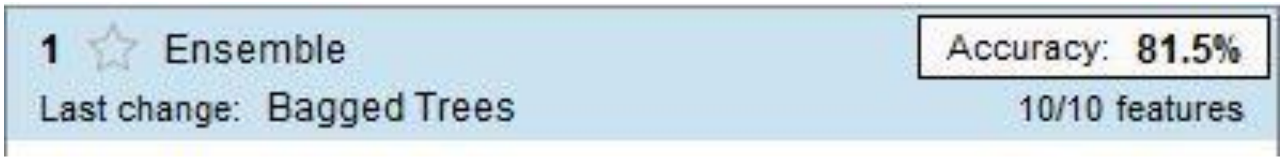

1 Ensemble
Last change: Bagged Trees
Accuracy: 81.5%
10/10 features

Figure 20: Accuracy of Bagged Trees classifier on original data

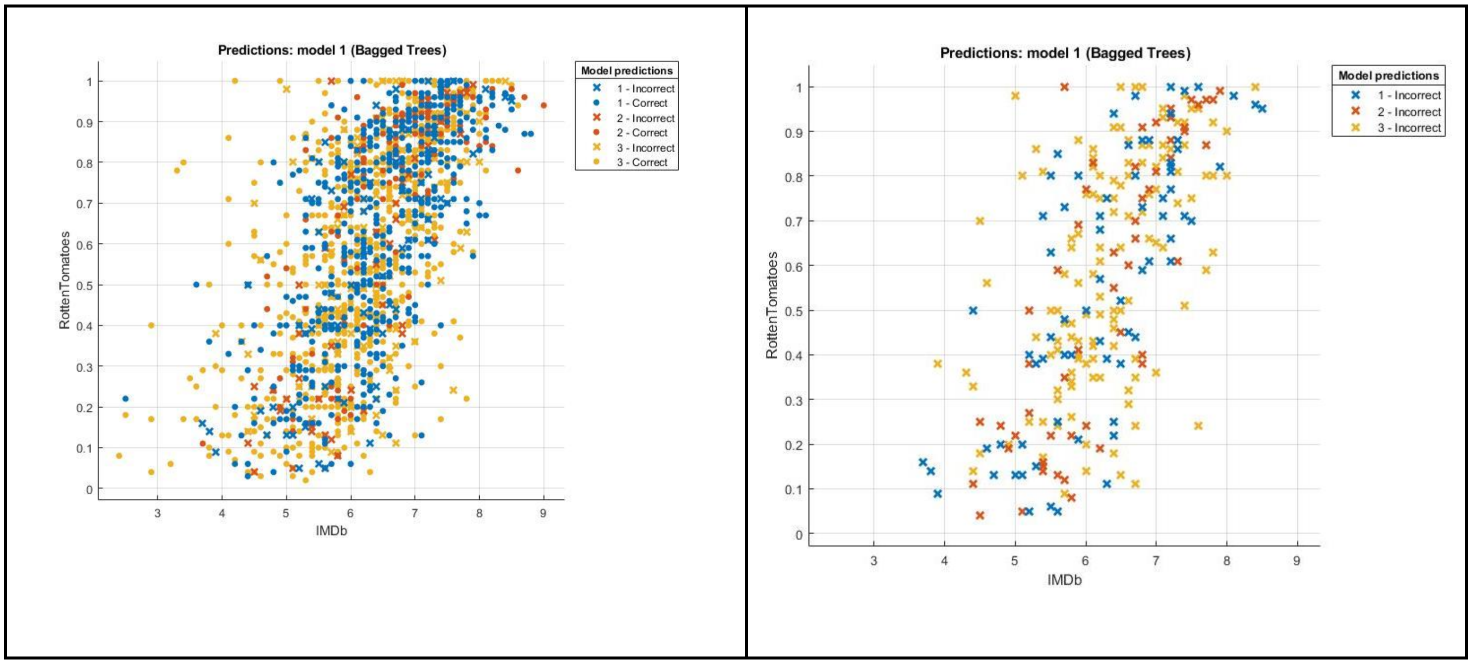


Figure 21: Bagged Trees classification on original data

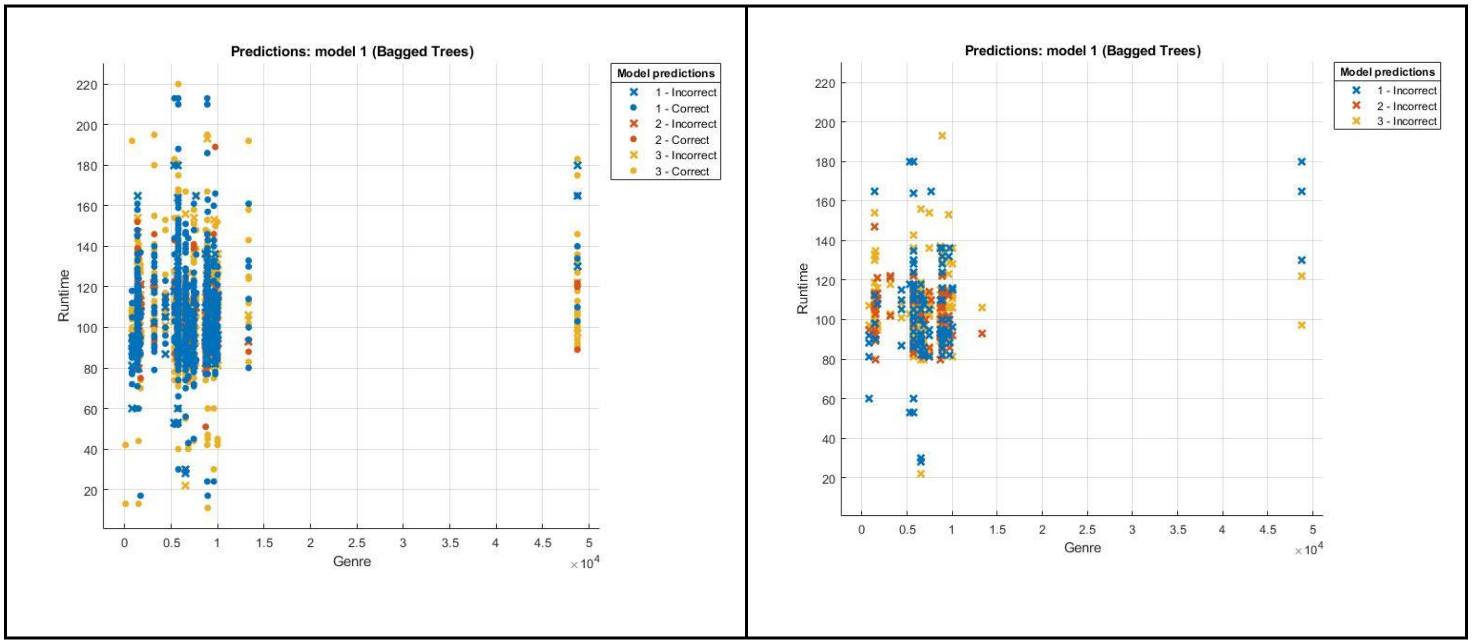


Figure 22: Bagged Trees classification on original data

From looking at these two paired graphs, we noticed a pattern where the more the data is clumped up the more the observations seem to be misclassified. Our interpretation of this is that the clumping of the data is because there are observations from the different classes that are similar to one another. This makes it harder for the classifier to accurately predict which class the observation is.

The first paired graph (Figure 21) is much more scattered than the second paired graph (Figure 22). Even though the results are more scattered, there is still a lot of camping within the graph. Also, when there are more outliers and scattered results, there is less misclassification. We noticed in the second paired graphs that there is much more clumping of the results. From this pattern, we assume that the classifier has trouble classifying the right class for the observations because there are occurrences with observations having similarities.

So again, our interpretation of this is that where there is clumping of the data there will be more occurrences of misclassification.

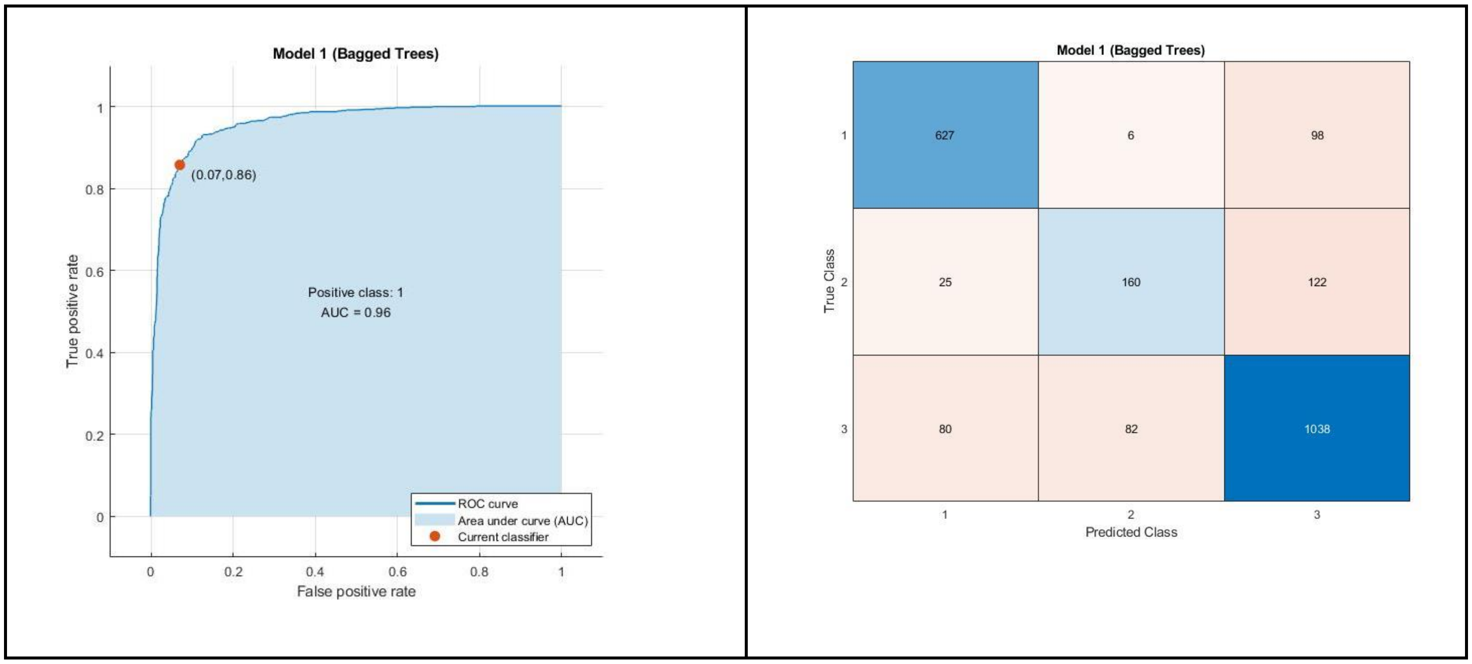
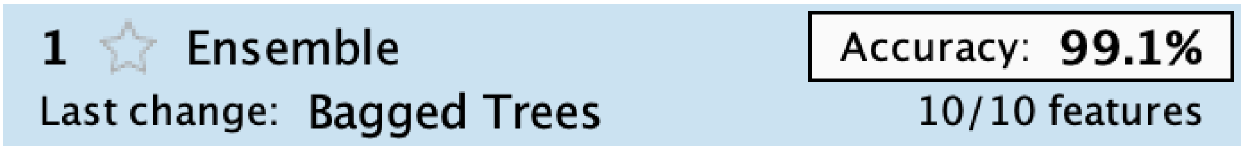


Figure 23: ROC curve and confusion matrix of original data

From looking at the ROC curve of the predictions from the classifier, the area under the curve value was 0.96 which is quite close to what a perfect value would be, at 1. However, we noticed that there is a current classifier plotted on the ROC curve itself. What we interpret from this is that even though the AUC value of the curve is very high, there is a threshold of the actual accuracy of the classifier.

Classification of Transformed Space (Ur)

The next step was to perform classification on the transformed data, Ur. Running the Bagged Trees classifier on Ur yielded an accuracy of 99.1% as seen in Figure 24. This was a marked improvement from the 81.5% accuracy rate from classifying the original dataset.



1 Ensemble — Accuracy: **99.1%**

Last change: Bagged Trees — 10/10 features

Figure 24: Accuracy of Bagged Trees classifier on transformed data

Figures 25 and 26 show the results of a classification in the transformed space. In both plots, there are roughly two sections: one with the bulk of the data points and another with a small collection of scattered points of the varying class. In both sets, most of the missed classifications were in the bigger cluster of data points. This makes sense because more densely clustered points will lead to more classification errors. In contrast, the other section of less dense data points has fewer missed classifications.

Within the densely clustered areas, there appears to be distinct “layering” of the classes. Though the data points of all classes have a decent amount of overlap, each class covers a visible range that is distinct from the other two classes. The missed classifications in this area are to be expected because there is so much overlap across the classes, and the classifier produced fewer missed classifications than we expected. The most surprising result was in the other area, the less dense scattering of data points. This area does not have the same obvious clustering as the denser area, so we expected several misclassifications in this area. The results show that there were few misclassifications in this area, perhaps because of the nature of bagged trees. In dividing a section into smaller sections, the classifier can identify finer patterns within the small sections that it would not have identified if examining a larger area.

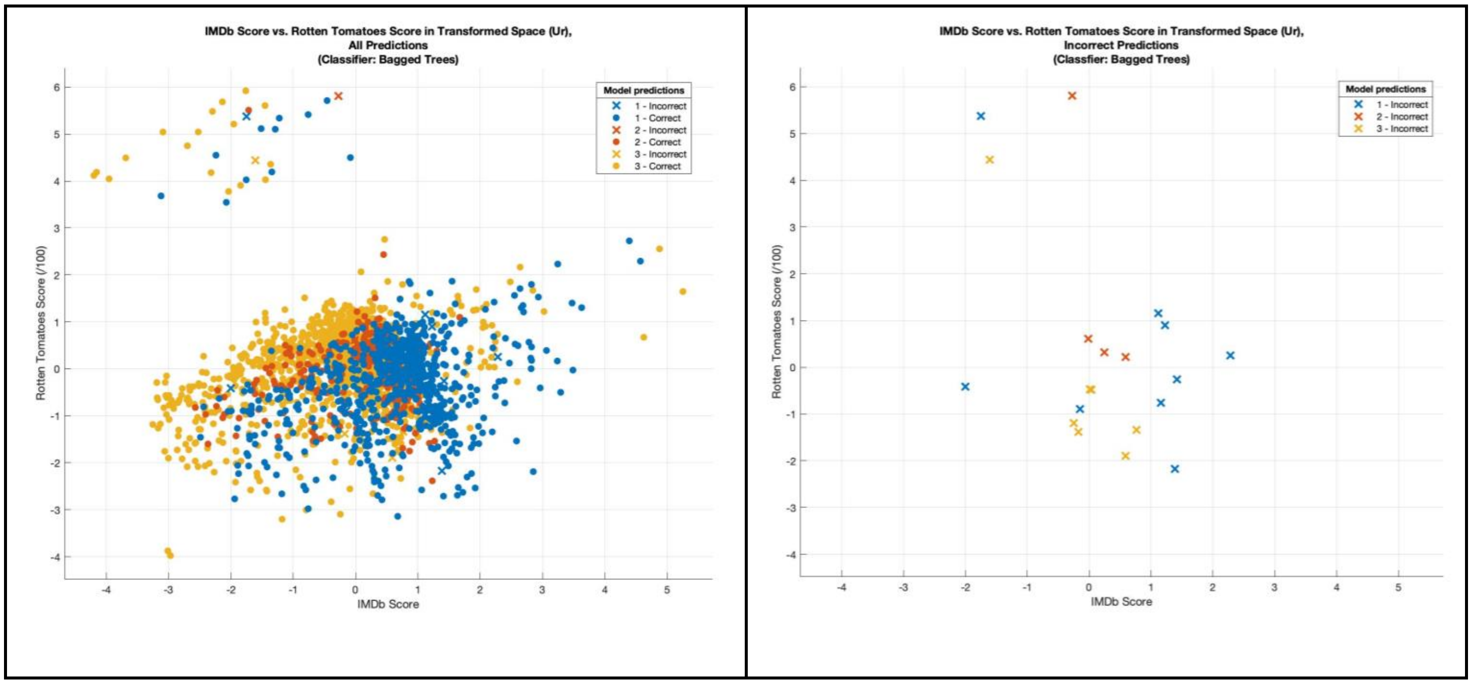
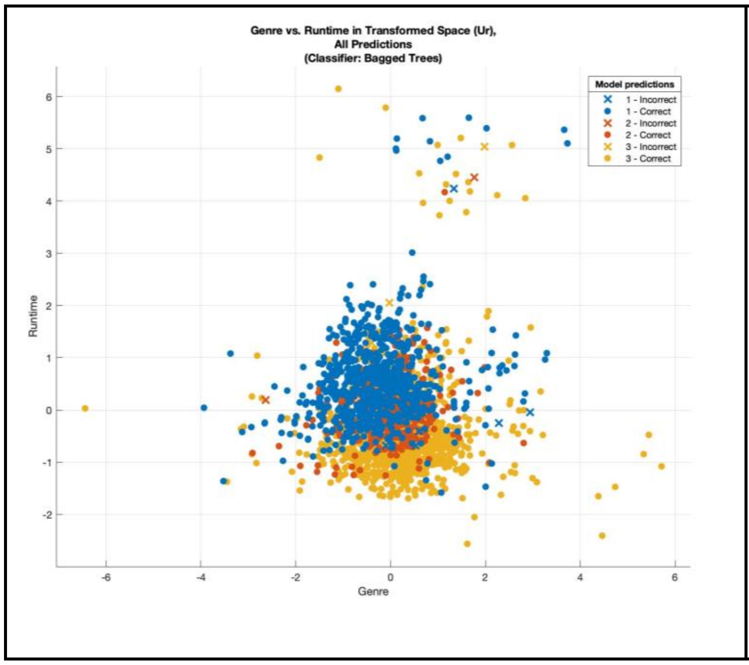


Figure 25: Bagged Trees classification on transformed data (Ur)



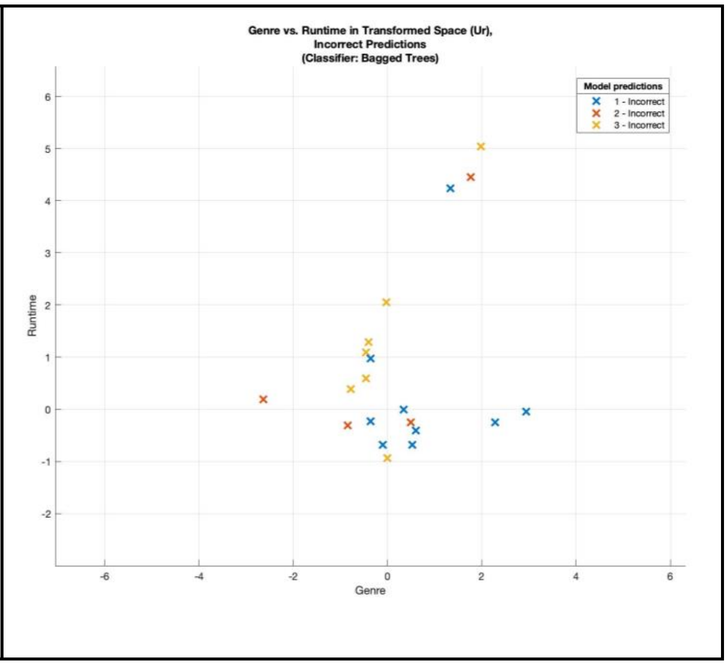


Figure 26: Bagged Trees classification on transformed data (Ur)

Last, we created a ROC curve and confusion matrix of the classification results on the transformed data (Figure 27). A ROC curve shows the performance of a classifier by comparing the false positive rate to the true positive rate. The area under the ROC curve for the transformed space turned out to be 1.00, which means our classifier is “perfect.” Though the larger the higher the better the classifier, one must be wary of a perfect score: An area of 1.00 may indicate data overfitting wherein the classification is fit exactly to the particular training set. The problem with overfitting is that the classifier may not be generalizable to another set of data because it was so finely tuned to a particular set of data. It is possible that using the bagged trees classifier caused overfitting because it is performed on fine subsets of data.

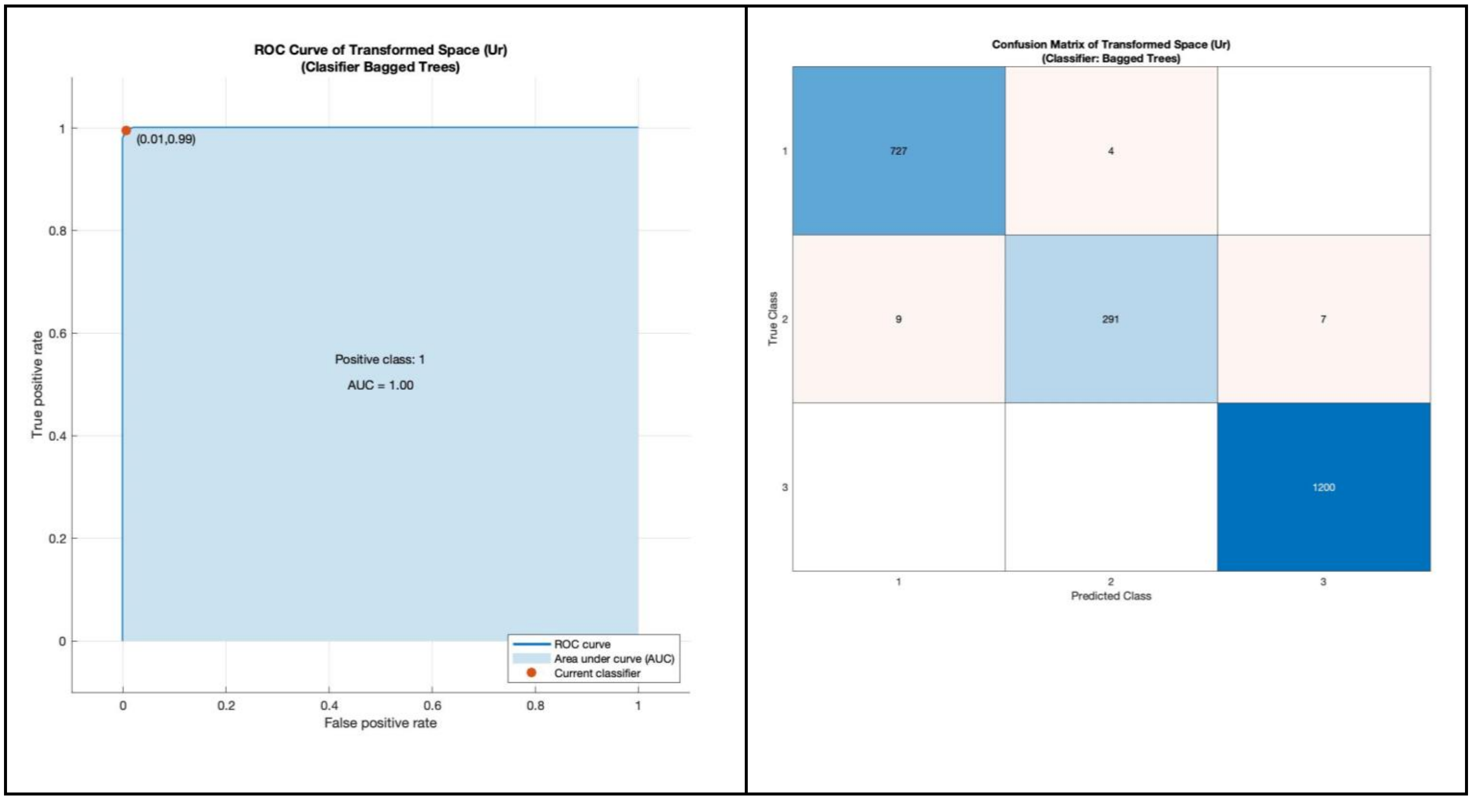
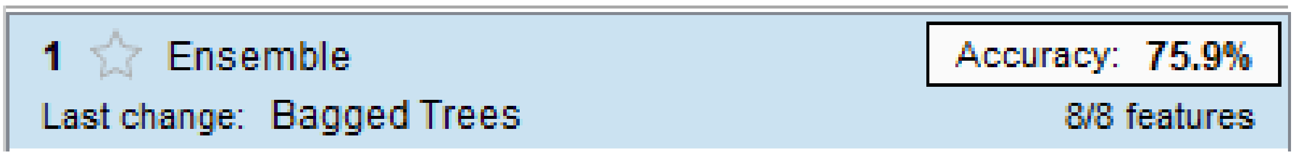


Figure 27: ROC curve and confusion matrix of transformed data (Ur)

Classification of Updated Original Dataset

Updated dataset (Took out features: Age and Rotten Tomatoes)

The reason why we chose to remove these two features from our dataset is that when we look back at the loading vectors, feature 3 (Age) had little to no magnitude in most of the loading vectors. Our interpretation of this is that feature 3 isn't providing important information for the forming of the newly transformed data. The reason for removing feature 5, Rotten Tomatoes, is that features 4 and 5 had many occurrences where they were correlated. So our interpretation of this is that the information it provides is redundant, and that is why we want to test our dataset without it.



1 Ensemble Accuracy: 75.9%

Last change: Bagged Trees 8/8 features

Figure 28: Accuracy of Bagged Trees classifier on updated original data

Looking at Figures 29 and 30, we can see that when we remove features from our bagged tree, it results in a less accurate prediction rate. It seems that every feature is required to give a more accurate reading of movies. This statement fully supports the idea that a bagged tree is a combination of decision trees so losing a feature would mean losing a stem on that decision tree. Losing a stem seems to impact accuracy if even more features are removed. Age and rotten tomatoes were perceived as repetitive and decided to be removed, however, it seems that this further detail in information gave a 5% increase in accuracy.

When looking at these graphs, it seems that more centralized data was harder to predict than outliers. For example, if you compare the two images in each figure, you can see that most of the incorrect predictions came from the center of the data. It seems that the classes are more clustered together which results in an inaccurate reading of predictions.

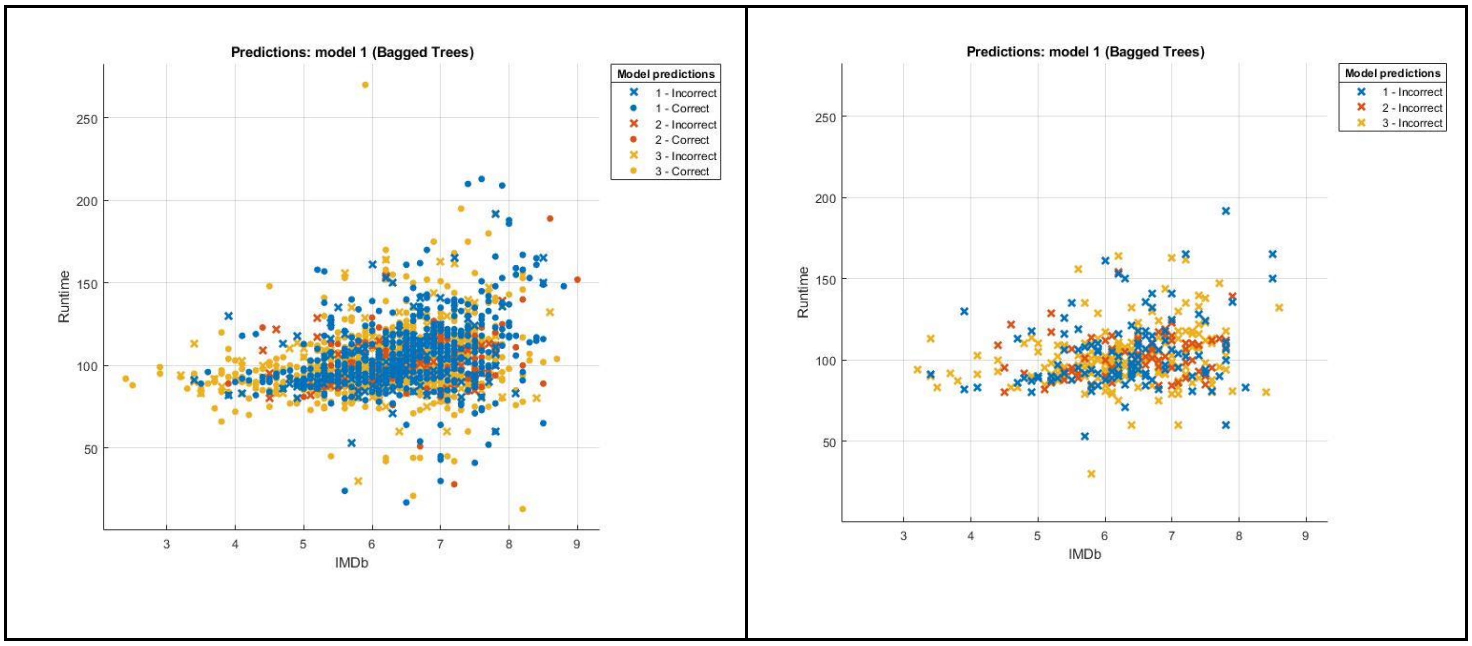


Figure 29: Bagged Trees classification on updated original data

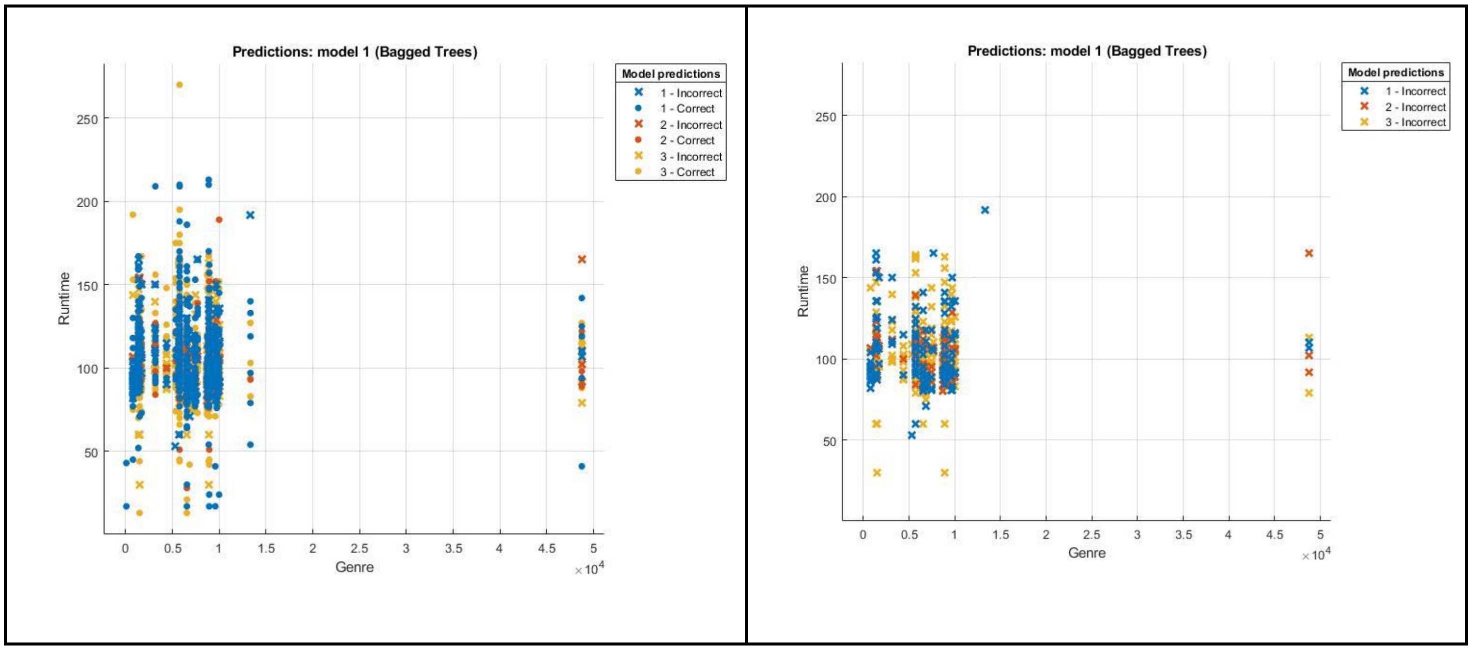


Figure 30: Bagged Trees classification on updated original data

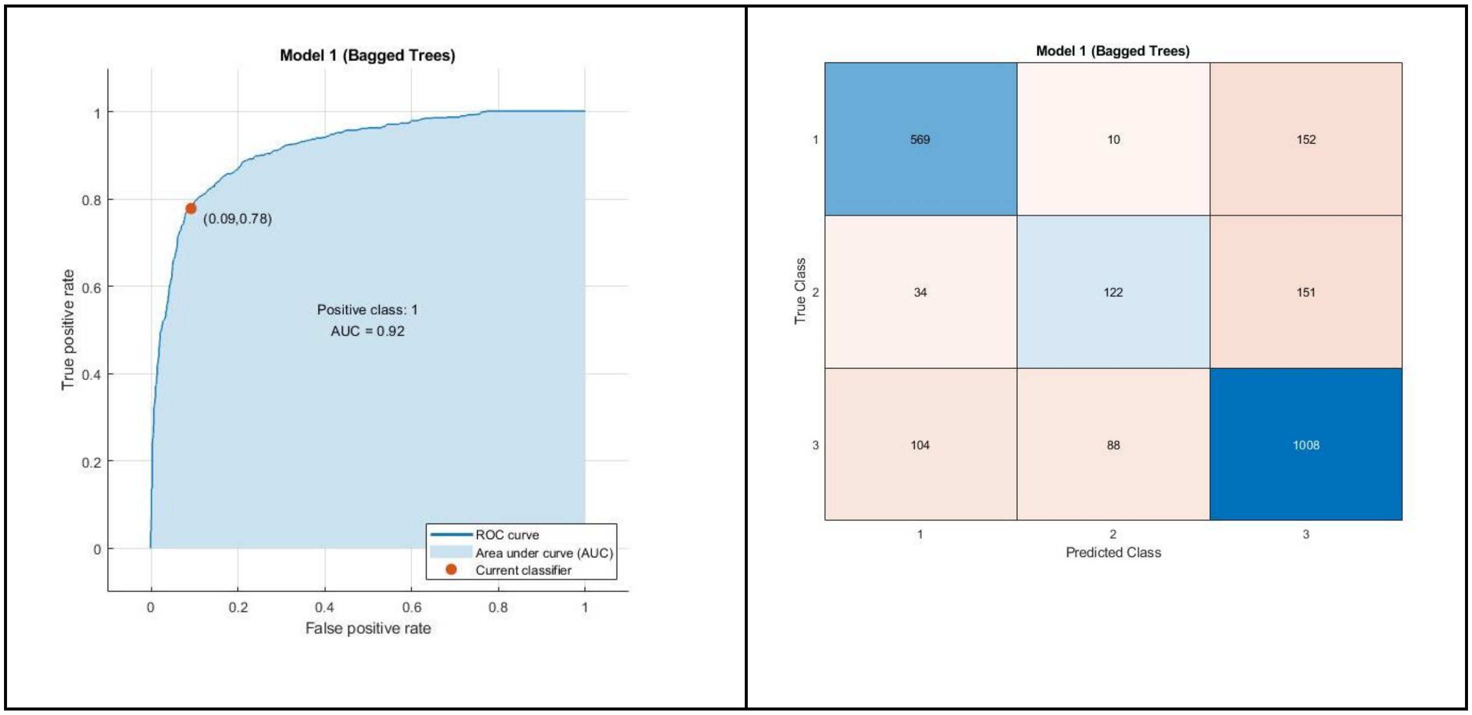


Figure 31: ROC curve and confusion matrix of updated original data

Looking at the ROC curve and confusion matrix, we can see that the area under the curve is about 0.92 which is close to 1 with the current classifier being located at 0.78 with a threshold of 0.09. It seems that 92% of the data was able to be read and learned through bagged trees which helped produce accuracy percentages of around 78%. However, the accuracy percentage was lower than our original data without removing any features.

Conclusion

Though other classifiers may be used, the bagged tree ensemble method worked well for our data. It may have worked too well, however; the “perfect” accuracy from the ROC curve makes us suspect that the classifier overfit the data. It may also be the case that there is so much similarity across the classes that the classifier is, in fact, highly accurate in predicting which class an observation belongs to. No matter the reason, decision trees may be the most effective classifiers for predicting which platform is most likely to have a given movie.

One classifier that we did not have time to test was a support-vector machine (SVM). Though it is possible to use them on more than two classes, SVMs usually work on only two classes. Seeing as our data had three classes (Netflix, Hulu, Prime Video) and given our lack of time, we were unable to explore the results of using an SVM on the data. If we had more time, we would try a linear or quadratic SVM because it may mitigate our issue with overfitting.

We realized that even though some data or features might seem noisy in the beginning, they could be very useful in the end for helping with classifying/predicting the observations.

It was very interesting to see the different amounts of data acquired with the various methods and functions. Also, we were very happy to have moments where we could look at the data/graphs from what we acquired from Matlab and be able to analyze and understand what we were looking at.

In the end, our team had an enjoyable time taking what we learned in class and being able to apply it to this project.

Contributions

Kelsey - I worked on the MATLAB code, especially for the plots. I also formed the written analysis for the plots. For classification, I ran the classifier app on the transformed dataset (Ur) and wrote its analysis. For the final presentation, I created the slides.

Kenneth - I focused on the analysis of the loading vectors of the data. I helped the team incorporate the previous iterations into this one and also helped edit the whole document. Also worked on some analysis of the 3D plots. I also worked on analyzing the classifier with the original dataset. In addition, I helped with adding a few comments in the conclusions.

Huy - I focused on creating functional MATLAB codes throughout the project and preprocess all the data. I also brought analysis for the classifier plots. In addition, I looked over the document to make sure we included everything. I helped do research on the data and how to use MATLAB. I helped decide which classifier was best suited for our data.

References

Genre feature encoding: https://www.whats-on-netflix.com/library/categories/

Language encoding:
http://producthelp.sdl.com/sdl_passolo_2016/en/html/Appendix/LanguageIDs.htm

Country encoding: https://www.nationsonline.org/oneworld/country_code_list.htm