How do movie features influence ratings?

Ryan Stefanic

1 Introduction

Movies are one of the most consumed forms of entertainment in the world. As the movie industry continues to grow, filmmakers want to better understand what audiences want so they can make the most successful movies. Variables such as budget, revenue, and popularity can provide valuable insights into a movie's potential success. The goal of this project is to understand how these different movie variables are correlated and to develop a machine-learning model that can predict a movie’s rating. Machine learning is an appropriate tool for this task because it can analyze large datasets and find patterns that will help us understand how the variables are related to each other and a movie's rating. In addition, machine learning's predictive power can provide accurate ratings based on these variables.

2 Background & Related Work

One prior work relevant to my project is a project by San Jose State University students Shu Zhang and Yue Xu called "Movie Rating Prediction System." This work uses a dataset from MovieLens, which includes 100,000 ratings among 943 users and 1,682 movies. They used two models for predictions, K-means clustering and Stochastic Gradient Descent (SGD). By evaluating the performance of the models with the Root

Mean Squared Error (RMSE) metric, the study found that SGD outperformed K-means, providing more accurate predictions. The authors concluded that SGD, which incorporates both user and movie factors, yields better results than K-means, which only considered movie genres.

This project is relevant to my goal of predicting movie ratings based on numerical variables like budget and factors that measure user behavior. However, Zhang and Xu's work focuses on predicting ratings by using dimensionality reduction techniques. My project, on the other hand, will explore how different movie features, such as a movie's popularity, budget, and release date, are correlated using Principal Component Analysis (PCA). By applying PCA to reduce the dimensionality of the dataset, I aim to uncover the most influential features and then use them to assist in predicting movie ratings.

3 Data Description & Preprocessing

For my project, I used a dataset of 5000 movies sourced from Kaggle. The dataset was taken from TMDB, which is a movie database. The predictor variable is “vote_average,” which is on a scale of 0 to 10. The predictors are 6 numerical variables, including budget, revenue, and year. To clean the data, I started by dropping rows that contained null values or errors. I then converted all the data types to numerical. I also had to properly format the variables to make sure they were usable in my models. This includes the year variable, which had to be formatted to just be the year number, as it was originally in a JSON format.

Variable	Type	Explanation	Value
vote_average	Response	The average rating of a movie	0 - 10
Budget	Predictor	Total money used to produce a movie	0 - 380000000
Popularity	Predictor	represents how much online attention and how many people have watched the movie	0.023 - 876
Revenue	Predictor	How much the movie made	0 - 2800000000
runtime	Predictor	length of the movie in minutes	0 - 201
vote_count	Predictor	How many people rated the movie	0 - 13752
year	Predictor	The year the movie was released	1937 - 2017

Figure 1: Variables used in the analysis.

3.1 Data Visualizations

Variable <chr>	Mean <dbl>	Median <dbl>	SD <dbl>
budget	45895338.39	27000000.00	53692188.61
popularity	35.75	25.28	48.21
revenue	155008338.66	62548947.00	240871202.62
runtime	108.66	106.00	19.38
vote_count	1323.56	616.00	1792.46
year	2007.55	2009.00	8.33

Figure 2: mean, median, and standard deviation of each predictor

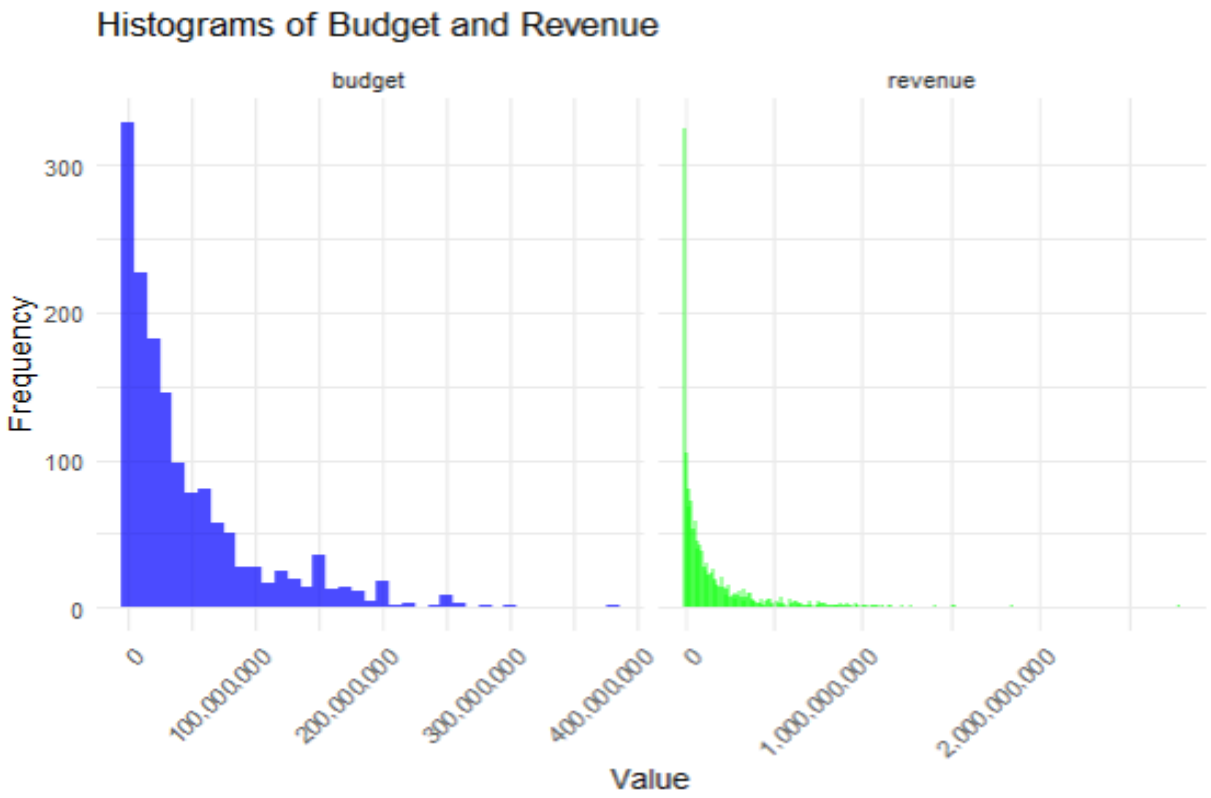

Figure 3: histogram of budget and revenue predictors

4 Modeling

Before performing the prediction models, I first applied Principal Component Analysis (PCA) to better understand the dimensionality of the dataset and explore the correlations between the various movie features. Following this, I will use XGBoost to predict movie ratings. I will compare the performance of XGBoost with and without PCA

to assess whether dimensionality reduction improves the model's accuracy and predictive power.

4.1 Model 1: Principal Component Analysis

Principal Component Analysis or PCA, is a dimensionality reduction technique that transforms variables into a set of principal components. To start PCA I created a Scree plot to visualize what percentage of variance can be explained by each principal component.

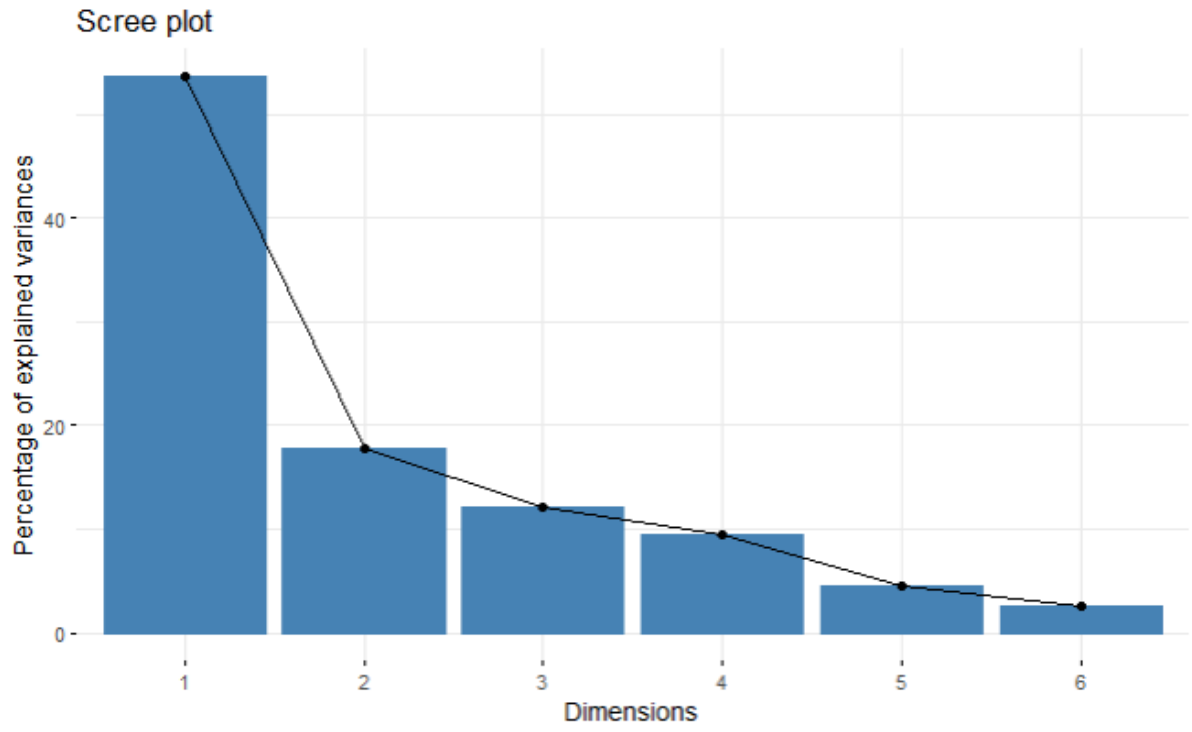

Figure 4: Scree plot showing the percentage of variance explained by each principal component.

In Figure 4, we see that around 95% of the variance in the data can be explained by the first 4 principal components. This indicates that these components capture the majority of the underlying variance in the dataset. This allows us to significantly reduce the

dimensionality while retaining most of the data by only using the first 4 principal components.

After determining that using 4 principal components is sufficient to explain approximately 95% of the variance in the data, I proceeded to run a PCA analysis on these components to examine how the different variables contributed to each principal component.

	PC1	PC2	PC3	PC4
budget	-0.4503892	0.17959604	0.1170693	-0.6138474
popularity	-0.4340008	0.02672052	-0.4036089	0.5951486
revenue	-0.5013226	0.01482871	-0.1341063	-0.3335714
runtime	-0.3189415	-0.34846827	0.8294453	0.2785216
vote_count	-0.5048898	-0.03024805	-0.1654350	0.1787329
year	-0.0288309	0.91894848	0.3001032	0.2195450

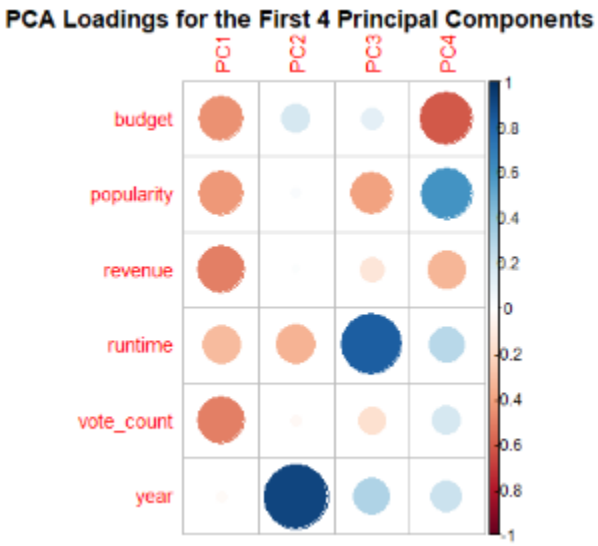

Figure 5: results of PCA analysis, and corrplot visualization

---

From this analysis, we can see the makeup of each principal component. The first principal component is primarily explained by the financial variables, with revenue, budget, and popularity contributing the most. This suggests that these financial factors play a significant role in the overall variance of the data and may have a strong influence on movie ratings. The second principal component is almost entirely driven by the year variable, indicating that the release year of a movie captures a large portion of the variance in the data. The third principal component is mainly explained by the runtime, suggesting that the length of a movie is another important factor but to a lesser extent than the financials.

4.2 XGBoost without utilizing PCA

In this model, I applied XGBoost directly to the full set of original variables without using principal component analysis. The purpose of this approach was to assess the model's performance using all available variables.

The performance of the model was evaluated using Root Mean Squared Error (RMSE). The RMSE value obtained for this model was 1.4359, which reflects the model's error when predicting movie ratings based on the full set of features.

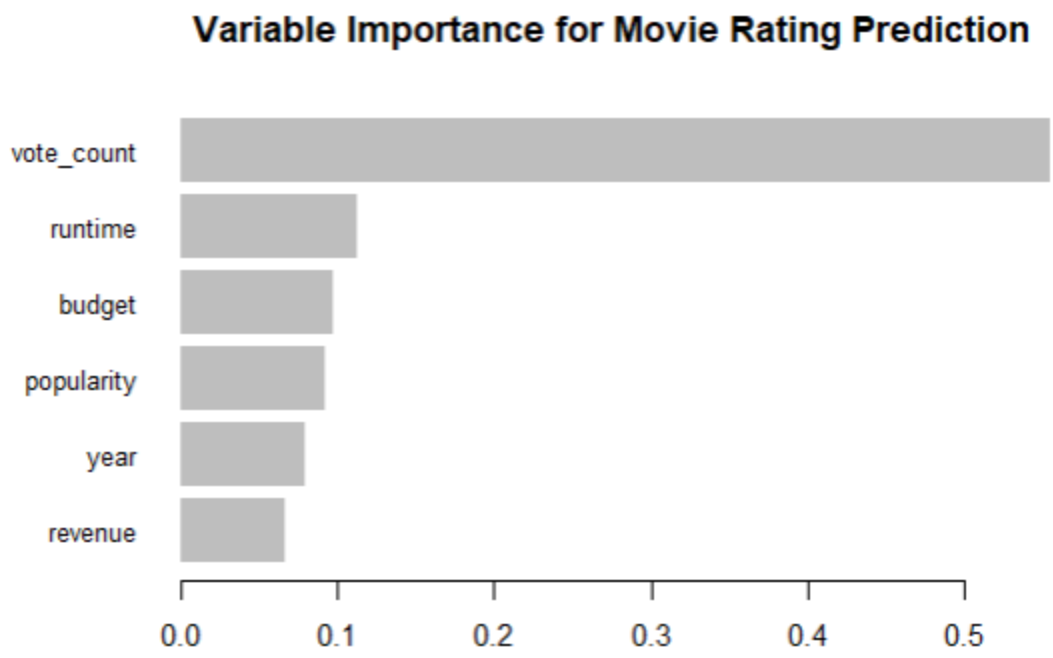

Figure 6: Variable importance for XGBoost model without pca

4.3 XGBoost utilizing PCA

In this model, I applied XGBoost after reducing the dimensionality of the dataset using Principal Component Analysis (PCA). As previously discussed, I used PCA to retain only the top 4 principal components, which accounted for approximately 95% of the data's variance. This step aimed to simplify the dataset by removing less informative features and reducing potential noise while still preserving the most significant data.

The performance of this model was also evaluated using RMSE, and the result was significantly improved compared to the model without PCA. The RMSE value for this model was 0.7353, indicating that utilizing PCA for dimensionality reduction led to an improvement in prediction accuracy.

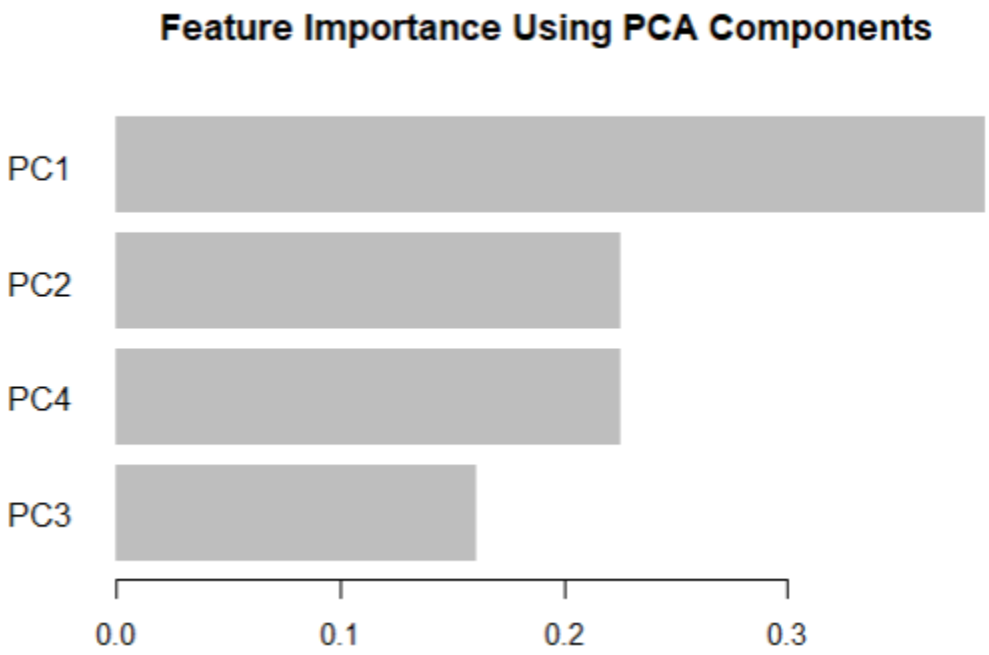

Figure 7: Variable importance for XGBoost model with pca

5 Conclusion

In conclusion, utilizing principal component analysis resulted in significant improvement to the RMSE value for XGBoost, reducing the error from 1.4359 to 0.7353. This suggests that dimensionality reduction through PCA simplified the dataset as well as improved the model's ability to make accurate predictions. PCA also allowed us to find the most important features influencing movie ratings. By focusing on the 4 most important principal components, the model was able to better capture the underlying patterns in the dataset. The results highlight the effectiveness of combining PCA with machine learning models like XGBoost to improve predictive performance.

6 Discussion

Principal Component Analysis (PCA) is a dimensionality reduction technique that works best when there are many predictors in a dataset. In this project, however, I was surprised to find that PCA still led to a significant improvement in model performance, even though the dataset contained only 6 predictors. One possible explanation for this improvement is that even with a relatively small set of features, there may still be considerable noise in the data, which may be due to a complex movie dataset.

References

Zhang, S., & Xu, Y. (n.d.). *Movie Rating Prediction System*. Retrieved December 17, 2024, from https://www.sjsu.edu/faculty/guangliang.chen/Math285F15/285ProjectPaper.pdf

https://www.kaggle.com/datasets/tmdb/tmdb-movie-metadata?select=tmdb_5000_movies.csv

Code Appendix

```
data1 <- df %>%
  select(budget, popularity, revenue, runtime, vote_count, year)
data2 <- df %>%
  select(budget, popularity, revenue, runtime, vote_count, year, vote_average)
```

```
data_pca <- prcomp(data1, scale = TRUE)
fviz_screeplot(data_pca)
```

```
data1_pca$rotation[, 1:4]
```

```
X <- data2 %>%
  select(budget, popularity, revenue, runtime, vote_count, year) %>%
  as.matrix()

y <- data2$vote_average

set.seed(123)
```

```
train_index <- sample(1:nrow(data2), size = 0.8 * nrow(data2))


X_train <- X[train_index, ]
y_train <- y[train_index]


X_test <- X[-train_index, ]
y_test <- y[-train_index]
```

```r
X_train_scaled <- scale(X_train)
X_test_scaled <- scale(X_test)

dtrain <- xgb.DMatrix(data = X_train_scaled, label = y_train)
dtest <- xgb.DMatrix(data = X_test_scaled, label = y_test)

dtrain <- xgb.DMatrix(data = X_train_scaled, label = y_train)
dtest <- xgb.DMatrix(data = X_test_scaled, label = y_test)


params <- list(
  objective = "reg:squarederror",
  max_depth = 6,
  eta = 0.1,
  nthread = 2,
  eval_metric = "rmse"
)


xgb_model <- xgboost(
  params = params,
  data = dtrain,
  nrounds = 100,
  verbose = 1
)
```

```r
predictions <- predict(xgb_model, newdata = X_test_scaled)


rmse <- sqrt(mean((predictions - y_test)^2))
print(paste("RMSE: ", rmse))
```

```
[1] "RMSE:  1.43587107439467"
```

```r
importance_matrix <- xgb.importance(feature_names = colnames(X_train_scaled),
                                    model = xgb_model)


xgb.plot.importance(importance_matrix,
                    main = 'Variable Importance for Movie Rating Prediction')
```

```r
pca_scores <- data1_pca$x[, 1:4]
y <- data2$vote_average

set.seed(123)
train_index <- sample(1:nrow(data2), size = 0.8 * nrow(data2))


X_train <- pca_scores[train_index, ]
y_train <- y[train_index]


X_test <- pca_scores[-train_index, ]
y_test <- y[-train_index]

dtrain <- xgb.DMatrix(data = X_train, label = y_train)
dtest <- xgb.DMatrix(data = X_test, label = y_test)


params <- list(
  objective = "reg:squarederror",
  max_depth = 6,
  eta = 0.1,
  nthread = 2,
  eval_metric = "rmse"
)


xgb_model <- xgboost(
  params = params,
  data = dtrain,
  nrounds = 100,
  verbose = 1
)
```

```r
predictions <- predict(xgb_model, newdata = X_test)


rmse <- sqrt(mean((predictions - y_test)^2))
print(paste("RMSE: ", rmse))
```

```
[1] "RMSE:  0.734527356303792"
```

```r
importance_matrix <- xgb.importance(feature_names = paste("PC", 1:ncol(X_train), sep = "")
                                    model = xgb_model)


xgb.plot.importance(importance_matrix,
                    main = 'Feature Importance Using PCA Components')
```