Chipo Jokonya

Section 1: Data Exploration

1. How many rows and columns are there in the movielens dataset?
2. How many zeros were given as ratings in the movielens dataset?
3. How many threes were given as ratings in the movielens dataset?
4. How many different movies are in the movielens dataset?
5. How many different users are in the movielens dataset?
6. Which movie has the greatest number of ratings?

For the above refer to juypter notebook

Section 2: Project report

Introduction/Overview/Executive Summary

This project analyzes a dataset based on the movielens dataset that shows ratings given for a list of movies. The dataset used contains information about the titles of the movies, the genres, userid of the users who rated the movies, and it shows different rating levels done for each movie. The project uses these different variables in the dataset to build a recommender systems that predicts movie ratings based on the available variables. Assessment of each variable was done to understand the relationships among the different variables. A multi linear regression model was used to create a model which can predict movie ratings based on the available variables. Variables used in the model were chosen based on the correlation score vs the target variable.

Project goals

I. Develop a movie recommender system that can predict movie ratings based on the movielens dataset.
II. To analyse the movieLens dataset and the relationships between variables to inform the model development.
III. To build and evaluate a predictive model using a multi-linear regression model that will predict movie ratings

Project steps

I. Data cleaning and preprocessing- The dataset is checked for null and duplicated values. The dataset structure is checked in terms of data types, columns and rows.
II. Data visualization & Interpretation- Analysis of the dataset through various visualizations.
III. Data Preprocessing- converting of variables through standard scaler and data aggregation.
IV. Feature Selection and Engineering- correlation analysis, preparation of validation, test and training data.
V. Model selection and Training- linear regression training and testing.
VI. Model Evaluation- the performance and accuracy of the model is assessed using matrices like RMSE.

Methods/Analysis

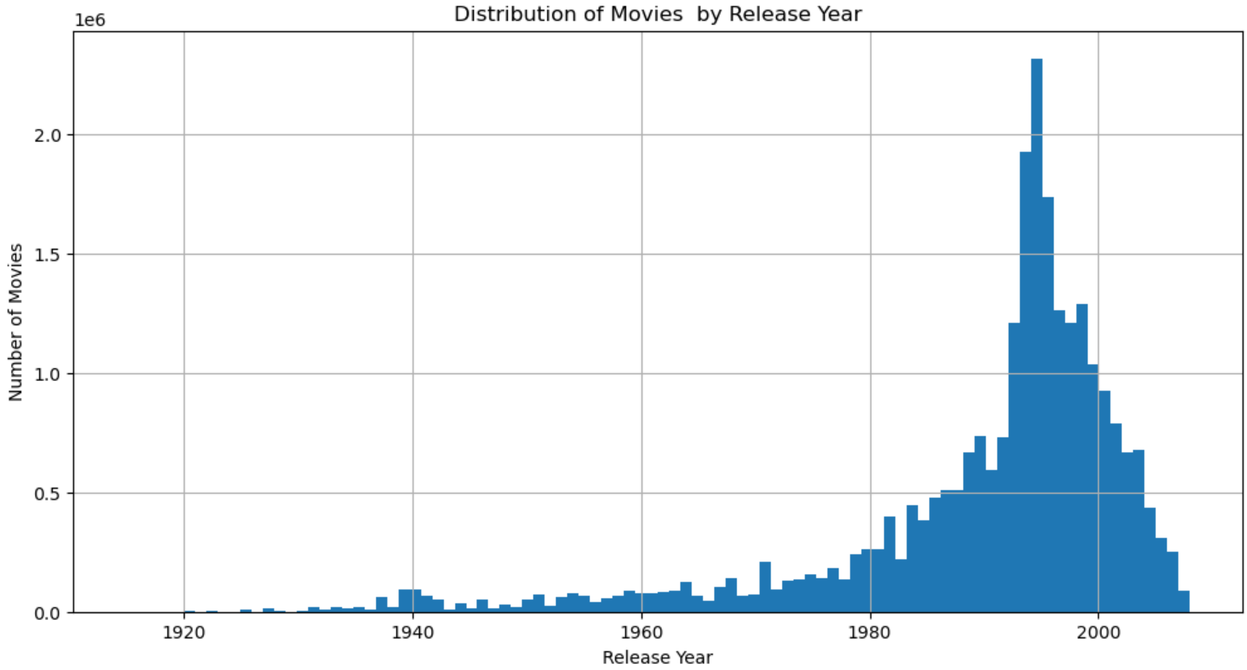
Step 1: Data cleaning and preprocessing

The movielens dataset did not have any null or missing values and it also did not have any duplicates. The dataset had objects, float and int datatypes. The dataset had no indexing errors, it had a total of 10000054 rows and 6 columns. In terms of summary statistics, the dataset had low standard deviation for all variables which showed that it had no outliers that needed to be processed. Highest rating that a movie got was 5 and the lowest being a 0.5.

Step 2: Data visualization & Interpretation

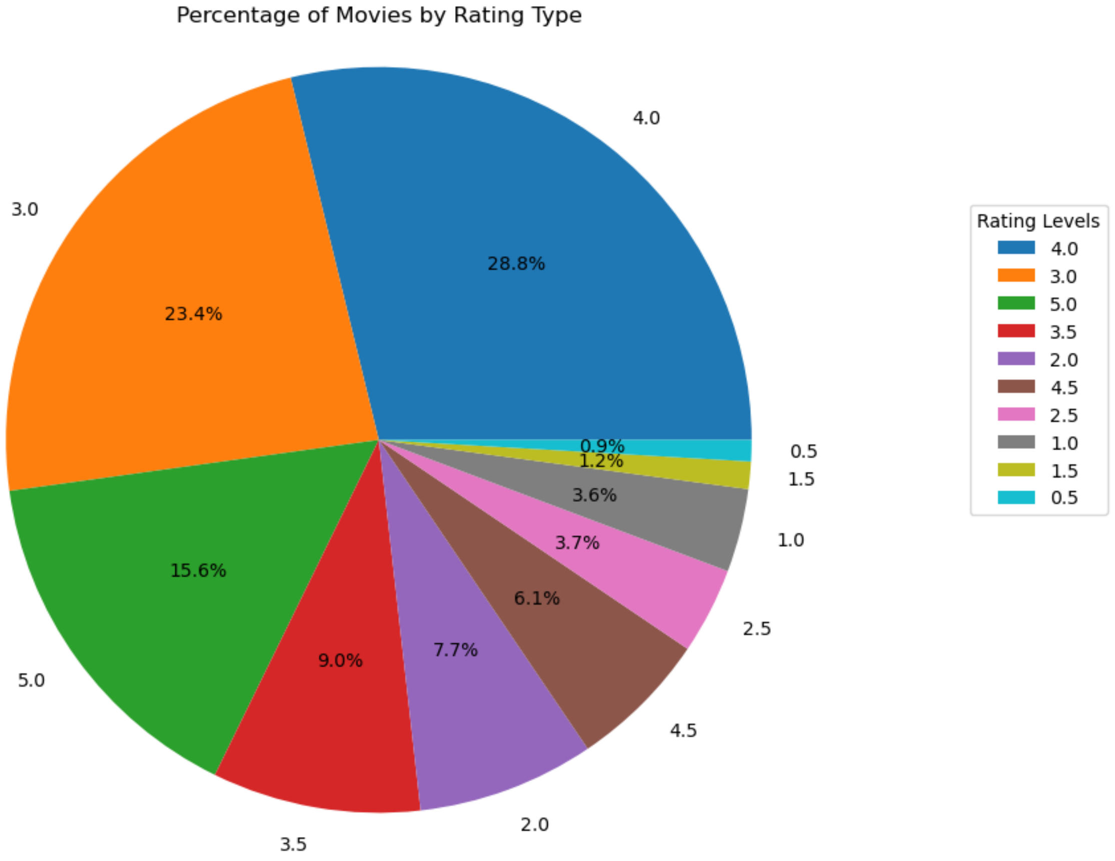
Looking at the given variables, analysis was done using different techniques to try and understand the dataset and to show patterns or trends that better explain the dataset.

I. Movie year release analysis



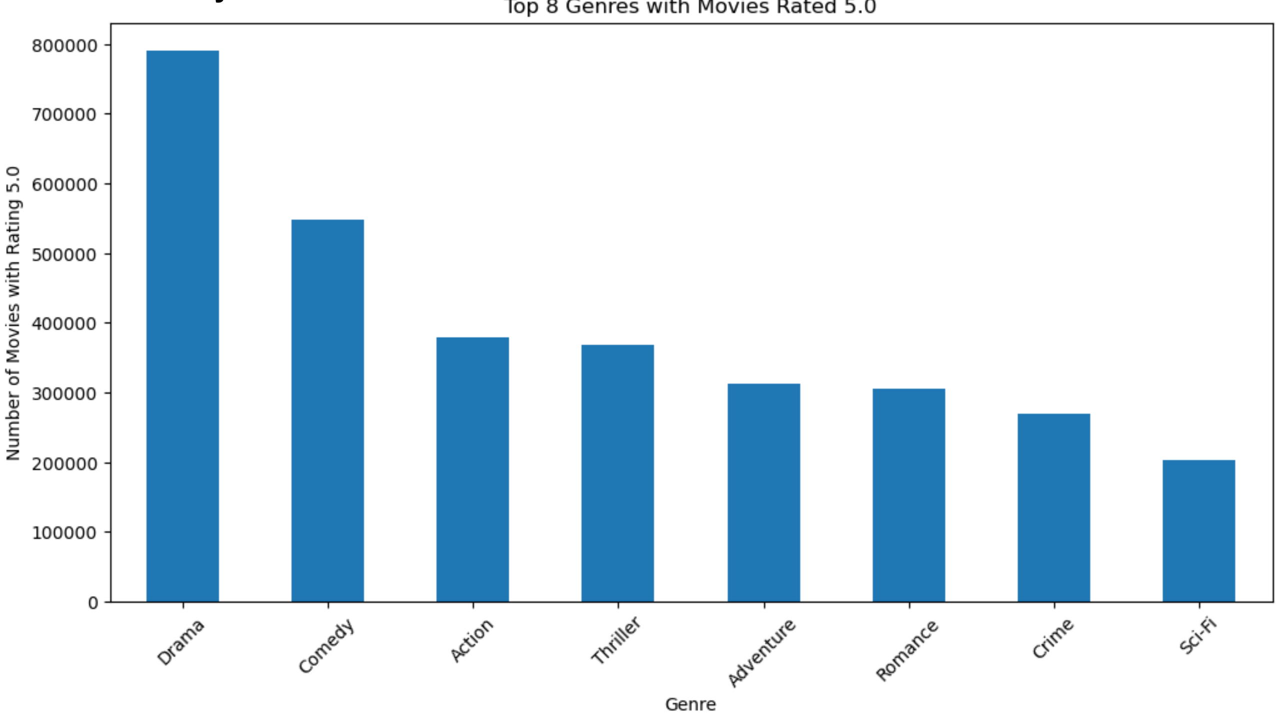
Analysis was done to determine which years saw the most movie releases. The title column was split to show movie release year. Splitting the data allowed analysis of the timelines in which movies were released. The above graph shows that majority of the movies were released in the late 1900s and 2000. It shows that there were limited movie activities in the early 1900s as per the dataset.

II. Movie ratings analysis



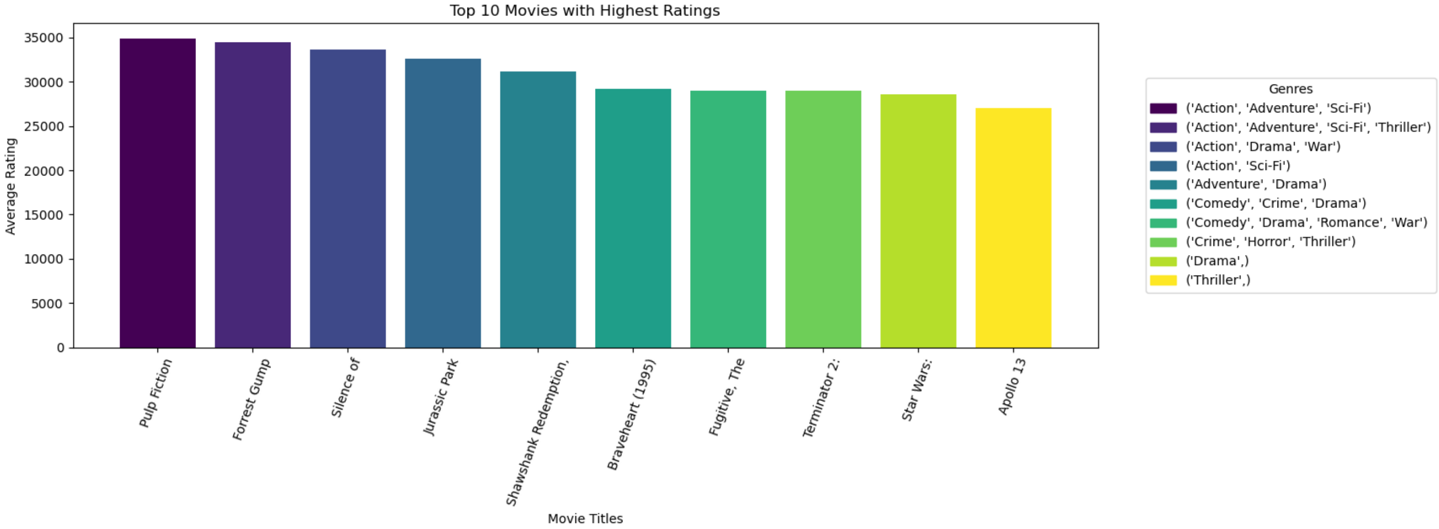
The above chart shows that most movies were rated a rate of 4 as the rate 4 occupies 28.8% of all the rating levels which is higher than any other rating level. Only 15.6% of the movies received a rating of 5, whilst 3.7% received a rating of 1. Those who received a rating less than 1 where only 0.9%. This shows that generally most movies had good ratings from 3 up to a rating of 5.

III. Genre’s analysis



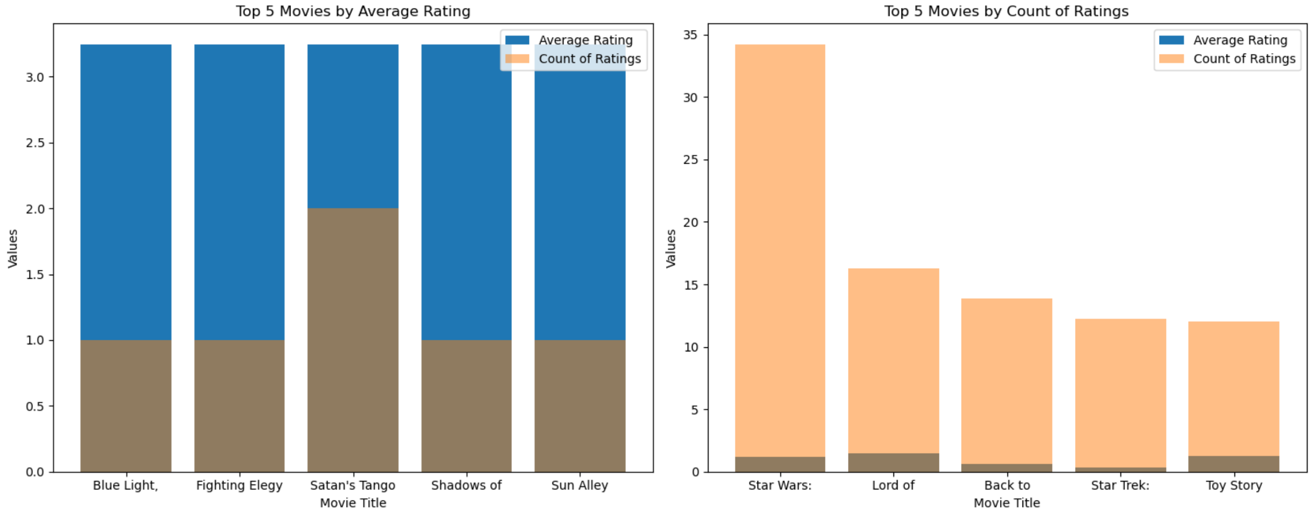
This analysis sought to understand which genres received top ratings. The above chart shows that movies in the Drama category mostly received a rating of 5 as compared to any other category. The lowest genres which received less ratings of 5 was the Sci-Fi. This analysis revealed that users mostly like Drama and comedy compared to other genres.

IV. Most popular Movies analysis



This analysis wanted to reveal the most popular movies based on the number of ratings. The count of ratings was done against each movie and the genres they belonged to. The above chart shows that Pulp Fiction had the highest ratings followed by Forrest Gump and they both belonged to the Action movie category. Looking at the chart above it shows most rated movies mainly belonged to the Action, Adventure and comedy movie categories.

V. Popular movies vs High rated movies analysis



The aim of the analysis was to answer the questions “are movies that are popular also highly rated that is they also had high ratings”. This analysis looked at both the count of movie ratings and the average rating per each movie. Movies were grouped based on count of ratings and average rating score. The aim of the analysis is to ascertain that a good movie should have a higher rating and should have been rated/watched by many. The analysis revealed that there was little correlation between the popularity of a movie and its ratings. The above chart on the left shows top 5 movies with high mean ratings whilst the chart on the right shows top 5 movies based on the count of ratings and average ratings. Both charts show different top 5s which shows the most popular movies didn’t necessarily have the highest ratings.

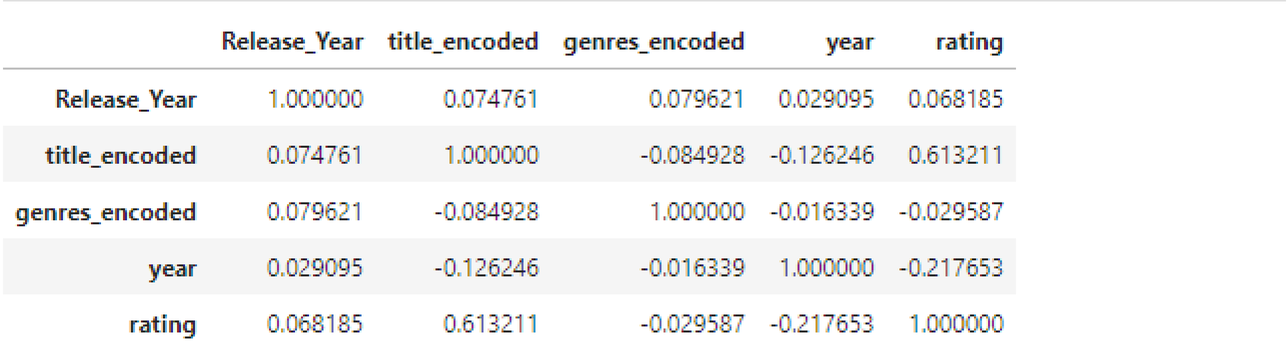
Step 3: Data Preprocessing

To make the data fit for processing through the regression model, non-numerical data was converted to numerical. The data was further processed by applying a standard scaler to improve model performance, enable easy feature selection, reduce feature dominance and enhance interpretability.

Data was also aggregated based on ratings, the features release year, title, year of rating and the genres were grouped according to the number of ratings. The aim of this process was to reduce noise and variability. The aggregation helped in revealing patterns and trends that can be hidden in individual data points this was proven by iteratively selecting different features to fit into the model and model evaluations.

Step 4: Feature Selection and Engineering

Correlation matrix table



	Release_Year	title_encoded	genres_encoded	year	rating
Release_Year	1.000000	0.074761	0.079621	0.029095	0.068185
title_encoded	0.074761	1.000000	-0.084928	-0.126246	0.613211
genres_encoded	0.079621	-0.084928	1.000000	-0.016339	-0.029587
year	0.029095	-0.126246	-0.016339	1.000000	-0.217653
rating	0.068185	0.613211	-0.029587	-0.217653	1.000000

The correlation matrix table was extensively used to understand the relationship between predictor variables and the target variable. As shown by the above table the correlation matrix table revealed that the title variable had the highest correlation with the rating variable with a correlation score of 0.613, which means that users rated movies based on how much they liked them. The year variable in which movies were rated had a negative correlation score of -0217 which meant that as the years moved ratings probably went down.

The dataset was split in to three that is validation data which was 10% of the original data, the remaining data was then split into testing and training data where testing data constituted only 10%. Variables were split into the following ways:

X variables/Predictors variables

- Title of movie
- Movie release date based on the title column
- Year based on timestamp column
- Genres

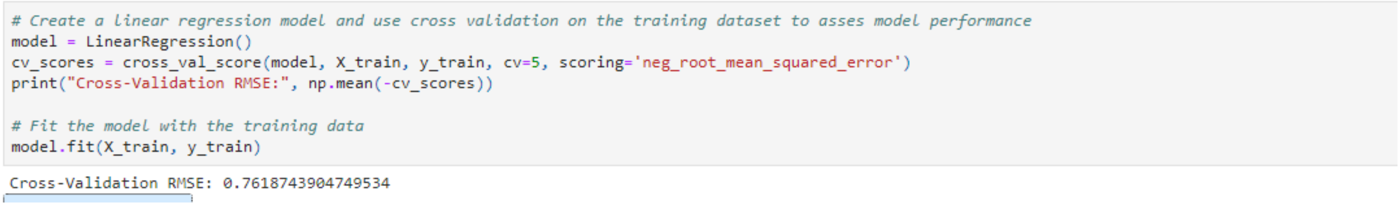
Y variable/ Target

- rating

Results

Model Fitting Process

I. **Model Initialization and Cross-Validation:**



```
# Create a linear regression model and use cross validation on the training dataset to asses model performance
model = LinearRegression()
cv_scores = cross_val_score(model, X_train, y_train, cv=5, scoring='neg_root_mean_squared_error')
print("Cross-Validation RMSE:", np.mean(-cv_scores))

# Fit the model with the training data
model.fit(X_train, y_train)
```

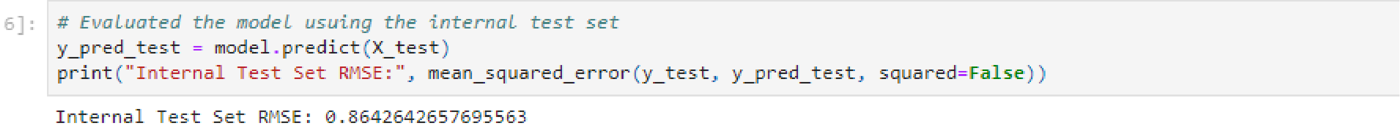
```
Cross-Validation RMSE: 0.7618743904749534
```

The model was initialized using LinearRegression(). Cross-validation (CV) was performed using cross_val_score with 5 folds (cv=5).

II. **Model Training:**

The model was trained on the training dataset using model.fit (X_train, y_train).

III. **Internal Test Set Evaluation:**

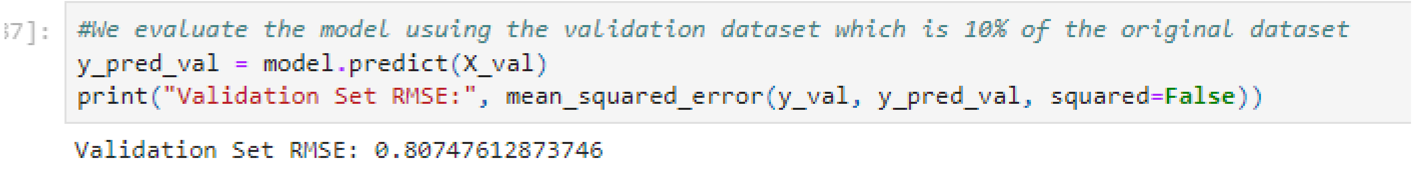


```
# Evaluated the model usuing the internal test set
y_pred_test = model.predict(X_test)
print("Internal Test Set RMSE:", mean_squared_error(y_test, y_pred_test, squared=False))
```

```
Internal Test Set RMSE: 0.8642642657695563
```

The model's performance was evaluated on the internal test set using the root mean squared error (RMSE).

IV. **Validation Set Evaluation:**



```
#We evaluate the model usuing the validation dataset which is 10% of the original dataset
y_pred_val = model.predict(X_val)
print("Validation Set RMSE:", mean_squared_error(y_val, y_pred_val, squared=False))
```

```
Validation Set RMSE: 0.80747612873746
```

An additional evaluation was performed on a separate validation set, which is 10% of the original dataset, also using RMSE.

Techniques Used

I. **Cross-Validation:**

Cross validation was used to evaluate the performance of the regression model on the training data using 5 folds meaning it was evaluated on each fold. This method was used to measure how well the model performs on unseen data. The use of 5-fold CV helps in providing a robust estimate of model performance.

II. **RMSE as Evaluation Metric:**

RMSE was used as the evaluation metric, both in cross-validation and for the internal test and validation sets. RMSE was used because it gives a sense of how much error there is between predicted and actual ratings.

III. **Model Validation:**

To make sure that the model performance was consistent across different subsets of data a separate validation process was done involving a separate validation set.

Results

I. Cross Validation RMSE- The cross-validation RMSE is approximately 0.762, indicating the average performance of the model across the training folds.

II. Internal Test Set RMSE- The internal test set RMSE is approximately 0.864. This is slightly higher than the cross-validation RMSE, suggesting that the model performs slightly worse on the unseen internal test data compared to the training data.

III. Validation Set RMSE- The validation set RMSE is approximately 0.807. This result is intermediate between the cross-validation RMSE, and the internal test set RMSE. According to the set metrics, the model performance exceeds performance target of <0.86490.

Interpretation

a. **Model Performance**

The model showed reasonable performance, with RMSE values indicating that the predictions were close to the actual ratings. The RMSE values were relatively low, suggesting that the model made accurate predictions.

b. **Consistency**

The consistency between the cross-validation RMSE (0.762), the internal test set RMSE (0.864), and the validation set RMSE (0.807) suggests that the model generalized well to unseen data. There is no significant overfitting or underfitting, as the RMSE values are close to each other.

c. **Feature Engineering**

The features used included the title of the movie, movie release date, year based on timestamp, and genres. These features are relevant in describing how movies are rated. The title of the movie was the biggest driver of the predictions because it had a higher correlation matrix score. The results show that to predict movie ratings it takes the title of the movie, the year the movie was released and when the ratings were done to make predictions that are close to accurate. Different combinations were tried to try and see the best predictor combinations that yields good performance results:

	Predictor variable	**Correlation score**	**RMSE**
Model 1	- userId	- 0.000223	3.511156

	- movieId - timestamp - Release_Year	- (-0.006501) - (-0.034750) - (-120770)	
Model 2 (aggregated data according to rating count)	- Realese_Year - Title - Genres	- 0.0609 - 0.934254 - (-0.0263)	0.320754
Model 3 (aggregated data according to rating count)	- Release_Year - Title - Genres - Year	- 0.068185 - 0.613211 - (-0.029587) - (-0.217653)	0.8074761

In the above table, model 1 did not perform well, it used the original variables as they were whilst the other two models performed well after transformations of the predictor variables. Model 3 was chosen because it is close to the set level of <0.86490. To get this desired model performance variables were transformed through splitting the timestamp and title, transformations also included aggregation based on the count of ratings per each year, title and genres.

Conclusion

The data analysis process revealed that most movies were released after 1980 and that most movies received a rating of 4. The most interesting pattern that the analysis revealed was that movies with high ratings were not necessary very popular and that movies with high count of ratings did not necessarily have high ratings.

The modelling process revealed that if there are at least two variables with higher correlation scores against the target variable then the model performs well. Initially when the variables were used as they are given in the dataset, the model did not perform well which meant that there was a need to create new variables that had more influence on movie ratings, the available variables did not have high correlation with the target variable enough to allow the model to perform well that is predict movie ratings. Consideration for additional features such as director, cast, or user-related data if available should be done. Splitting variables like timestamp and title created new variables that proved to be useful as the performance of the model improved because these variables had better correlation scores. Aggregating the variables also proved to be useful by revealing hidden patterns that improved model performance.

Overall, the model fitting process, techniques used, and results indicates that the model performed well with good generalization performance after using a combination of aggregated encoded title, encoded genres, year of rating and movie release year predictor variables. However, there's always room for improvement, especially in feature engineering and exploring other modelling techniques.

The current limitations of the model include low correlation of predictor variables with the target variable, this dataset requires further feature engineering to improve model performance.

Future work

Techniques like TF-IDF (Term Frequency-Inverse Document Frequency) or word embeddings (like Word2Vec or GloVe) can be used to encode the title text to get more meaning out of the data which can greatly help in creating a more robust recommender system.

There is room for deeper analysis for the timestamp variables, deep analysis can reveal things like seasonal effects or patterns.

The dataset could incorporate additional features like director, cast, budget, runtime and user demographics because when it come to movie ratings these additional features could offer significant model improvements.

With limited features, to help with model performance other models could also be incorporated such as decision trees, random forest, gradient boost machines or support vector. To combine these models, techniques like stacking or boosting can be used to improve the model performance.