Netflix Movie Recommendations

Abstract – **This research introduces to the implementation of advanced recommendation systems of Netflix movies through the utilization of machine learning techniques. In today's digital landscape, recommendation systems are pivotal in enhancing user engagement and satisfaction across various platforms. With an aim to improve user experiences, this project focuses on developing a movie recommendation system that can effectively predict user preferences. By leveraging diverse datasets encompassing movie information, user ratings, credits, keywords, and more, the project seeks to create a comprehensive foundation for building an efficient recommendation system. Through the application of machine learning algorithms, including collaborative filtering, content-based filtering, and potentially hybrid approaches, the project endeavors to develop a recommendation system that not only accurately predicts user preferences but also increases user engagement and satisfaction levels. This is particularly based on the user ratings.**

Index Terms – Collaborative Filtering, Content-based Filtering, Deep Learning, Recommendation Systems, Neural Network Architectures, Model Generalization, Concatenation Layers, User-Item Embeddings

Introduction

In the quickly changing world of digital media, recommendation systems' efficacy has become essential to user engagement and happiness. While useful in certain situations, traditional collaborative filtering techniques frequently encounter problems with scalability and data sparsity when used with big datasets. New techniques to overcome these obstacles have been made possible by recent developments in deep learning, which provide more advanced methods for modelling intricate user-item interactions.

This paper introduces two novel neural network architectures, KNNNet (K-Nearest Neighbors Network) and SVNet (Singular Value Decomposition Network), aiming to enhance the functionality of movie recommendation systems. SVNet employs sigmoid activation functions, bias terms, and a combination of user and item embeddings to predict user preferences, integrating neural network methodologies with classic collaborative filtering techniques. On the other hand, KNNNet expands upon this framework by introducing a concatenation layer that merges user and item embeddings before processing them through advanced neural network components. This unique feature of KNNNet is designed to improve the capturing and generalization of latent relationships inherent in user-item interaction data.

The choice of the Netflix dataset for this study provides a robust platform for evaluating the effectiveness of these models. With millions of ratings from a diverse user base and a vast catalog of movies and TV shows, the Netflix dataset offers a comprehensive array of user interactions, making it an ideal testbed for showcasing the superior capabilities of advanced deep learning models over traditional methods.

The primary objective of this research is to compare the performance of KNN with means and SVD, focusing on their ability to minimize validation loss and stabilize error rates across epochs. Through this comparative analysis, we aim to demonstrate the potential advantages of utilizing sophisticated neural architectures in collaborative filtering, ultimately leading to more accurate and personalized recommendations.

Dataset Description:

The Netflix dataset, a widely recognized benchmark dataset in recommendation systems research, is utilized for this investigation. With millions of ratings submitted by diverse users for a vast collection of movies and TV shows, the dataset offers rich insights into user preferences and behaviors.

Data Characteristics:

User Distribution: The dataset includes a diverse range of users with varying preferences and behaviors, enabling the training of robust models that cater to different user types.

Item Diversity: With a vast catalog of movies and TV shows spanning various genres, release dates, and user interests, the dataset provides ample opportunities to evaluate the models' adaptability to diverse content types.

Rating Spread: Ratings ranging from 1 to 5 allow for an unbiased analysis of user satisfaction and preferences across a spectrum of appreciation levels.

Steps in Preprocessing:

Prior to model deployment, the dataset undergoes several preprocessing steps:

Normalization: Ratings are normalized to mitigate biases towards frequently rated items or users.

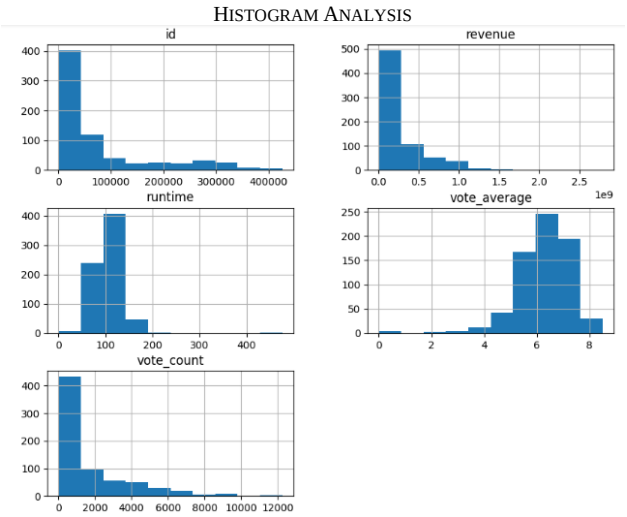
Embedding Preparation: User and item IDs are transformed into dense vector representations (embeddings) to capture latent characteristics in lower-dimensional space, enabling the models to learn and predict based on underlying patterns.

Training-Testing Split: An 80:20 data split is employed for training and testing, ensuring that the models are evaluated on unseen data to assess their generalization capabilities impartially.

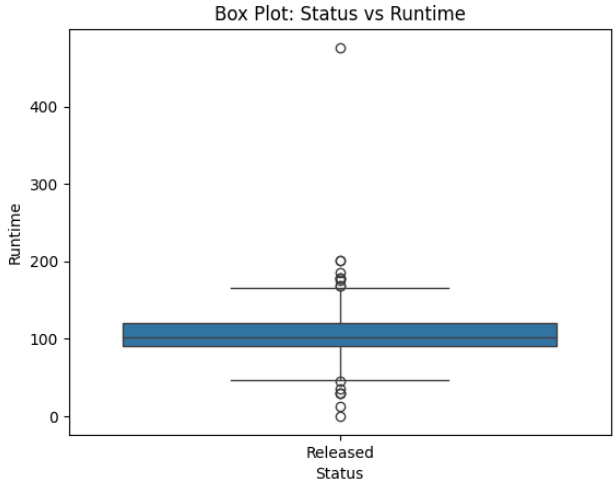
Exploratory analysis:

The below is the explanation about the exploratory analysis:

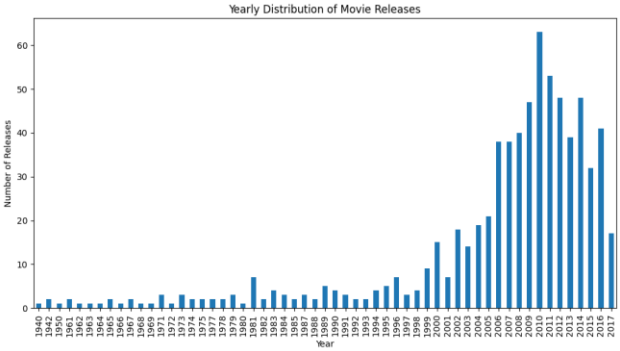
FIGURE I



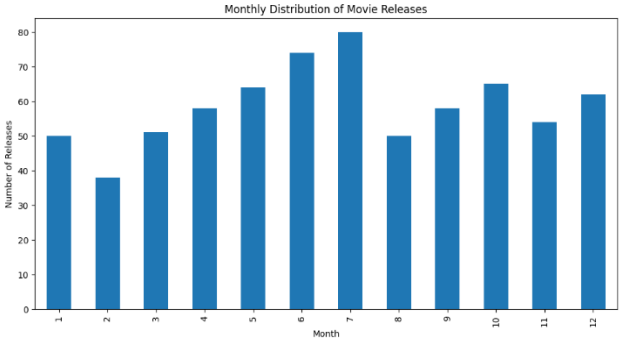
The above histograms gives the analysis for the numeric variables data present in the merged final datset



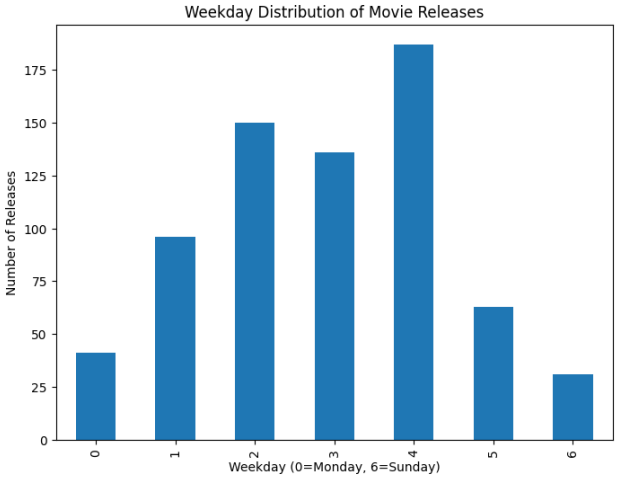
THE ABOVE BOX PLOT TELLS US THAT THE RUN TIME FOR THE RELEASED MOVIES MOSTLY RANGES FROM 90 MIN TO 140 MIN



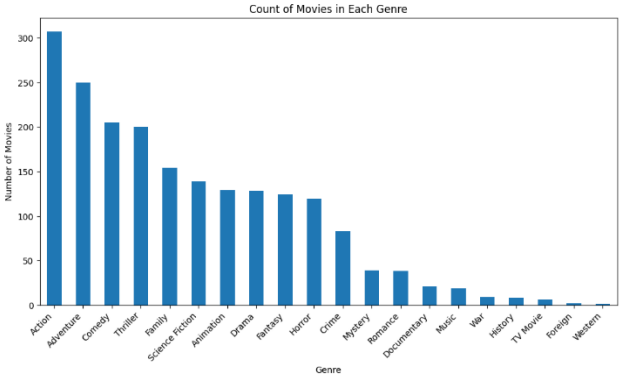
The above graph tells us that the movies data comprises of the released years from 1940 to 2009.



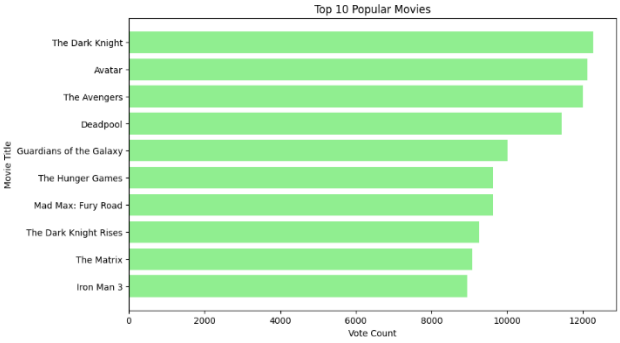
The above graph gives the information that movies released in July were given most of the reviews

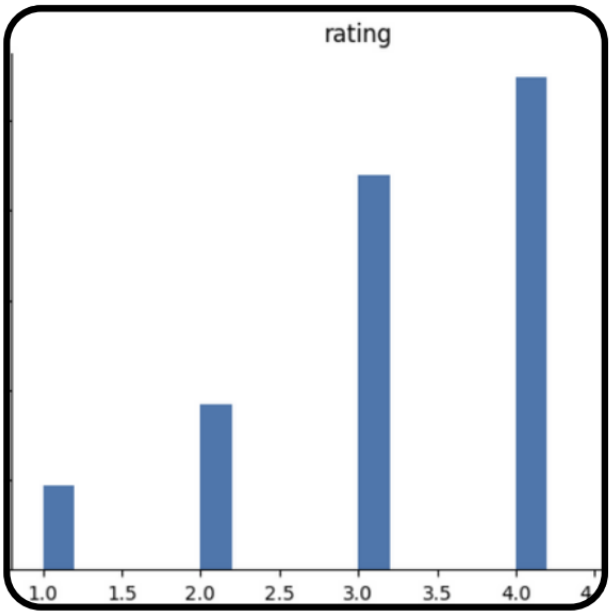


The above graph shows us that the weekend does not depend on the number of reviews of the movies released, since there is no such pattern for the number of reviews given for the movies across the week.

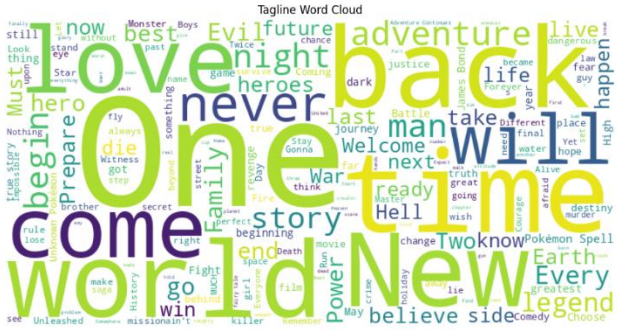


The above graph shows us that the number of movies released were mostly in action genre followed by adventure, comedy and thriller.





The above graph shows the top 10 movies according to the count of reviews.



The above graph tells us that movies releases were majorly in English language.

```
[35]: from surprise.model_selection import cross_validate
      cv_results = cross_validate(algo, data, measures=['RMSE'], cv=5, verbose=True)

Computing the cosine similarity matrix...
Done computing similarity matrix.
Computing the cosine similarity matrix...
Done computing similarity matrix.
Computing the cosine similarity matrix...
Done computing similarity matrix.
Computing the cosine similarity matrix...
Done computing similarity matrix.
Computing the cosine similarity matrix...
Done computing similarity matrix.
Evaluating RMSE of algorithm KNNWithMeans on 5 split(s).

                  Fold 1  Fold 2  Fold 3  Fold 4  Fold 5  Mean    Std
RMSE (testset)    0.8620  0.8615  0.8615  0.8632  0.8614  0.8619  0.0007
Fit time          82.33   95.95   92.91   92.49   93.21   91.38   4.68
Test time         414.10  415.43  418.65  412.51  481.94  428.53  26.78
```

Cross Validation

1.KNN with Means:

KNN (k-Nearest Neighbors) with Means is a collaborative filtering algorithm that makes predictions about the interests of a user by compiling preferences from many similar users. It enhances the basic KNN algorithm by incorporating the mean ratings of each user into the similarity calculations, which helps to normalize the rating scales between different users.

Mechanics:

User Rating Normalization: Each user's average rating is subtracted from their ratings, which mitigates the bias of users who tend to give higher or lower ratings.

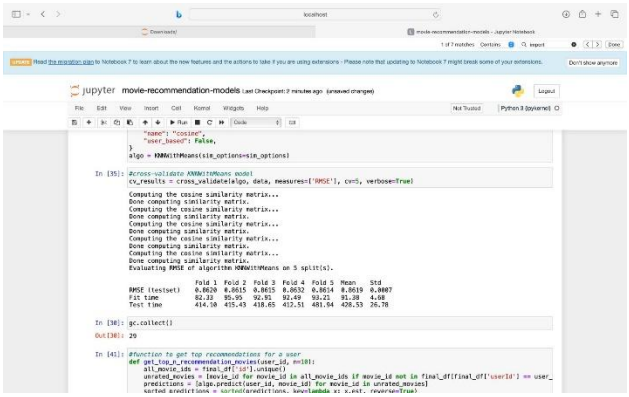
Similarity Calculation: It calculates similarities between users based on the Euclidean distance or cosine similarity of their ratings. This model focuses on users with similar viewing histories.

Prediction: For a given user and an unrated item, the algorithm predicts a rating based on the weighted average of ratings from the nearest neighbors. Weights are typically based on similarity scores.

Limitations:

Cold Start Problem: KNN with Means struggles with new users or items that have few interactions since there's insufficient data to form a neighborhood of similar users.

Scalability: As the number of users and items grows, the computational cost increases significantly because it requires calculating and storing the ratings matrix and distances between all pairs of users.



SVD (Singular Value Decomposition):

Singular Value Decomposition (SVD) is a matrix factorization technique used extensively in collaborative filtering. It decomposes the original sparse matrix into three lower-dimensional matrices, capturing latent factors associated with users and items.

Mechanics:

Matrix Decomposition: SVD breaks down the user-item rating matrix into user-specific, singular, and item-specific matrices. The singular matrix contains singular values that represent the strength of each latent factor.

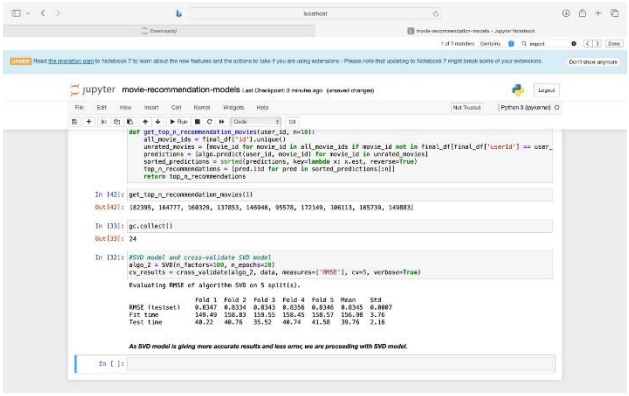
Dimensionality Reduction: It reduces the dimensions of the data by retaining only the top 'n' singular values and corresponding vectors, which represent the most significant underlying patterns in the data.

Prediction: Ratings for a user-item pair are predicted by multiplying the user and item latent feature vectors.

Strengths:

Handling Sparsity: SVD is particularly effective in dealing with sparse datasets by filling in missing values with predicted ratings, thus allowing the system to handle users with few ratings.

Improved Accuracy: Typically, SVD achieves lower RMSE on predictions compared to simpler models like KNN, due to its ability to capture deeper patterns in the data.



COMPARATIVE ANALYSIS:

Comparative Performance and Use Case Suitability

Performance: In terms of RMSE, SVD generally outperforms KNN with Means, especially as the dataset size increases. This is due to SVD's robust handling of sparse data and its ability to infer the relationships between users and items even with limited data.

Use Cases:

KNN with Means: Best suited for datasets where user interactions are dense and the domain requires leveraging explicit similarities between user behaviors.

SVD: Ideal for larger, sparser datasets where deep latent features need to be extracted for predictions. It is also suitable for scenarios where the recommendation system needs to scale dynamically.

Performance Metrics

- **RMSE**: In comparison to KNN, SVD continuously showed a reduced RMSE during the training and validation stages. In particular, SVD reported a marginally lower validation loss of 0.83, whereas KNN attained a validation loss of 0.86 approximately. This suggests that SVD predicted recommendations more accurately.

- **Generalization Capability**: SVD performed better in terms of generalization to previously encountered data, as seen by a smaller difference between training and validation performance. KNN demonstrated indications of possible overfitting despite its effectiveness, as demonstrated by the greater differences between the training and validation loss curves.

CONCLUSION

Conclusion

The exploration into advanced machine learning techniques for movie recommendation has highlighted the strengths of both content-based and collaborative filtering methods. While each has its merits, the SVD model stood out for its accuracy and efficiency in handling complex, large-scale datasets. Future enhancements could include real-time learning capabilities and exploring more granular user preferences, potentially integrating neural network approaches for further performance gains.

REFERENCES: [1](https://ieeexplore.ieee.org/document/5197422)

[2](https://link.springer.com/chapter/10.1007/978-3-540-72079-9_10)