Recommendation System

Introduction

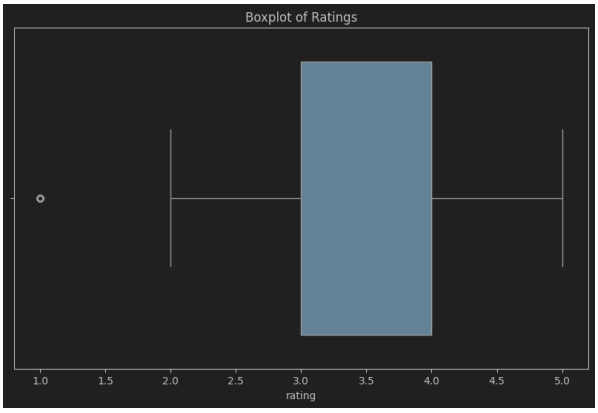
Recommendation systems are integral to enhancing user experience by offering personalized suggestions. There are two main types: collaborative filtering, which relies on user preferences, and content-based filtering, which considers item characteristics. The implemented approach here focuses on collaborative filtering, specifically using the K Nearest Neighbors (KNN) algorithm to recommend movies based on user ratings from the MovieLens dataset.

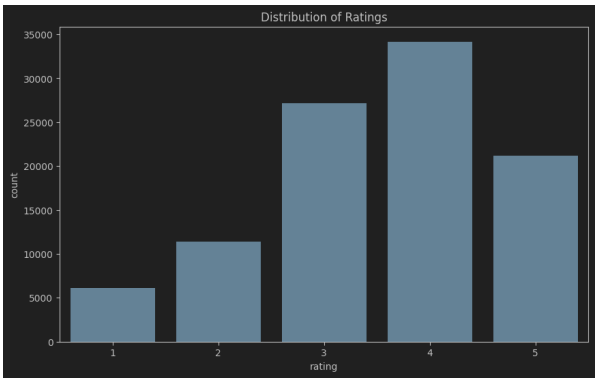
Approach:
The collaborative-based KNN model identifies similar users and suggests movies based on their preferences. Leveraging the simplicity and adaptability of KNN, the system evaluates its performance using metrics like precision, recall, and F1 score ..

Exploratory Data Analysis

Ratings Distribution

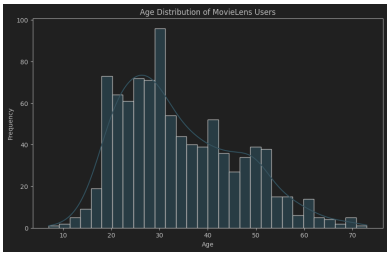
- Here we could notice almost normal distribution for movies ratings except of “1” which is an outlier

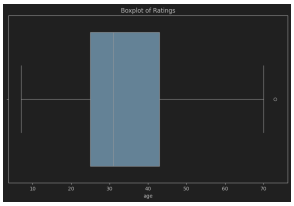




Users Distribution

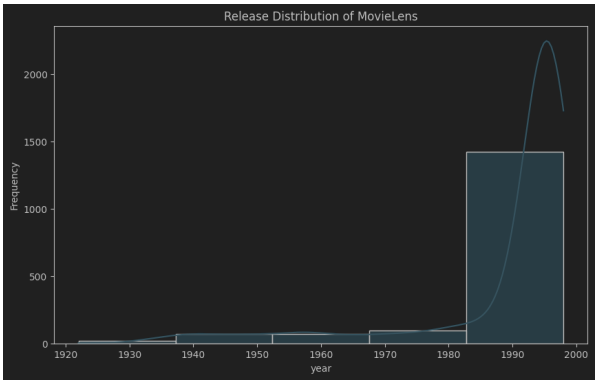
- Here we could look at the age distribution of the users and we could notice outliers for users younger than 10 and older than 70.

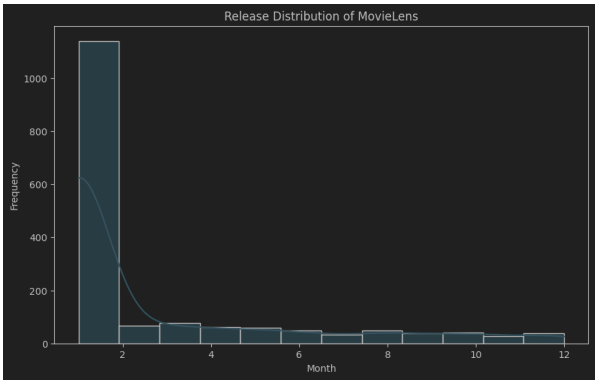




Release year Distribution

- Given the distinct movies associated with each era, it made sense to categorize release years into intervals, grouping them in spans of 10 years rather than examining continues dates.
- When encoding release months, sinusoidal and cosine transformations were applied to preserve their circular nature. This approach is particularly apt as movies released on same season might be related (Christmas movies for example).





Model Implementation, Evaluation and Results

1. Initialization:

- The KNN model is initialized using the `NearestNeighbors` class from the scikit-learn library.
- `metric='cosine'` specifies the cosine similarity metric. Cosine similarity measures the cosine of the angle between two non-zero vectors and is well-suited for collaborative filtering.
- `algorithm='brute'` indicates the use of the brute-force algorithm, which calculates distances between all pairs of users. While computationally expensive for large datasets, it is feasible for the used dataset.

2. User-Item Matrix:

- The MovieLens dataset is organized into a user-item matrix where users are in rows, movies in columns, and ratings as values.
- This matrix captures the interactions between users and movies, forming the basis for similarity calculations.

3. Nearest Neighbors Function:

- The `get_k_nearest_neighbors` function retrieves the k-nearest neighbors for a given user based on their ratings.
- It utilizes the `kneighbors` method of the k-NN model to find the indices and distances of the nearest neighbors.
- The function excludes the user itself from the list of neighbors.

4. k-Fold Cross-Validation:

- To evaluate the model's performance robustly, k-fold cross-validation is employed.
- The dataset is split into training and testing sets for each fold.
- The KNN model is trained on the training set, and evaluation metrics (precision, recall, F1 score) are calculated on the testing set.

5. Evaluation Metrics:

- Precision at k (P@k) measures the accuracy of the top-k recommendations by calculating the proportion of correctly recommended items.
- Recall at k (R@k) assesses the completeness of the recommendations by calculating the proportion of relevant items found in the top-k recommendations.
- F1 Score at k provides a balanced metric by considering both precision and recall.

6. Achieved Results:

K Recommendation Per User	P@k	R@k	F1@k
K = 5	0.0456	0.0063	0.0111
k = 10	0.0453	0.0126	0.0197
K = 20	0.0453	0.0197	0.0311

Model Advantages and Disadvantages

Advantages

1. Conceptually simple and easy to understand
2. No need to build a model

Disadvantages

1. Computationally and memory expensive especially with large datasets as it requires calculating distances between all pairs of users or items, thus will be faced with scalability issues
2. Challenging to find similarities in sparse data
3. Hard to finetune its hyperparamter, as in hard to pick a good value for K
4. Not suitable to add more features to the data