Sprints

Capstone Project – Recommendation System

"Movie Recommendation System Development"

#team_8#

☒Objective :

This project aims to develop a robust movie recommendation system using the MovieLens dataset. The system will provide personalized movie suggestions for users based on their preferences and movie features. The system will incorporate collaborative filtering, content-based filtering, and a hybrid approach to enhance the accuracy and relevance of recommendations.

☒Technologies Used:

- **Python**: The main programming language for data loading, processing, and model implementation
- **Pandas**: For data manipulation, cleaning, and preprocessing.
- **NumPy**: For numerical operations.
- **Scikit-learn**: For machine learning algorithms and model evaluation.
- **Surprise Library**: For implementing collaborative filtering using Singular Value Decomposition (SVD).
- **TF-IDF: For text-based content filtering using movie genres.**
- **Cosine Similarity: For measuring similarities between users, items, and movie features.**
- **Matplotlib & Seaborn: For data visualization, including rating distributions and movie popularity trends.**

Sprints

☒Key Features

1. User Preferences:
   - User Ratings: Ratings provided by users for various movies (e.g., 1 to 5 stars).
2. Movie Features:
   - Movie Metadata: Information such as movie title, genres.
   - Genres: The categorization of movies into various genres (e.g., Action, Comedy, Drama, etc.), used for content-based filtering.
3. Collaborative Filtering:
   - User-based Collaborative Filtering: Recommending movies based on similarities between users' rating patterns.
   - Item-based Collaborative Filtering: Recommending movies based on the similarity between movies' ratings.
4. Content-Based Filtering:
   - TF-IDF of Genres: Analyzing movie genres using Term Frequency-Inverse Document Frequency (TF-IDF) to represent the importance of genres for movie recommendations.
   - Cosine Similarity: Used for measuring the similarity between movies based on genre features. (we used query_genre_input here).
5. Hybrid Recommendation System:
   - Combining Collaborative and Content-Based Models: Merging both approaches to leverage their strengths for more accurate and personalized recommendations.
6. Evaluation Metrics:
   - Mean Absolute Error (MAE): A metric for evaluating the difference between predicted and actual ratings.
   - Root Mean Squared Error (RMSE): A metric to assess the accuracy of the recommendation system by penalizing large errors.

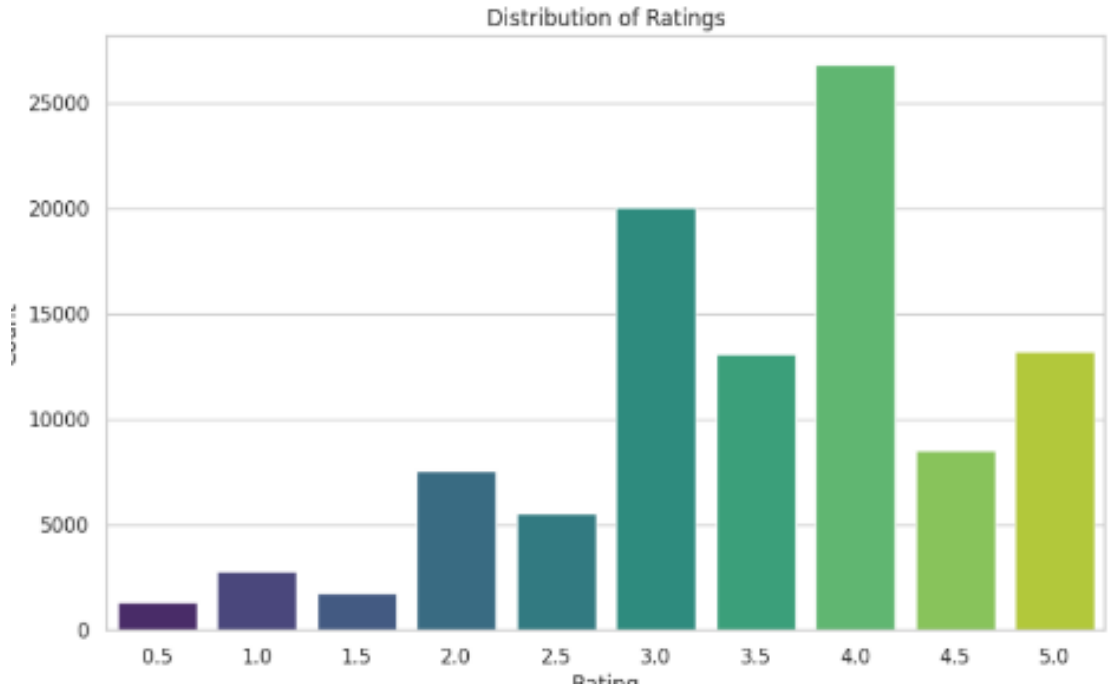
7. Optional Collaborative Filtering Using SVD:
   - Singular Value Decomposition (SVD): A matrix factorization technique for collaborative filtering to predict ratings and generate recommendations.

☒EDA

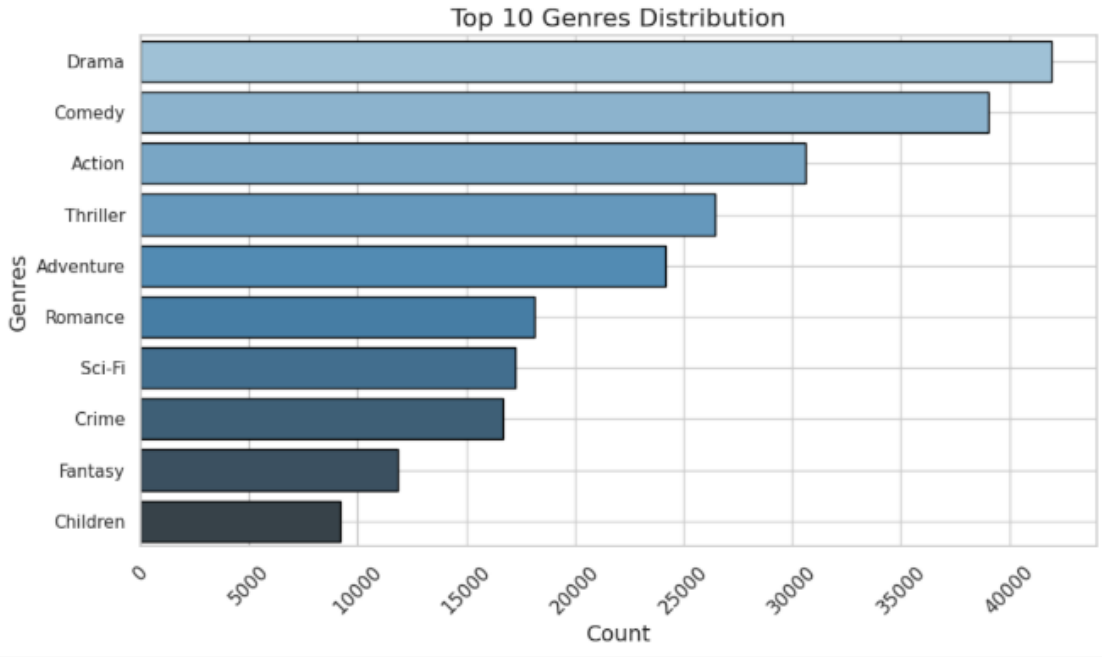
❖ Data Loading

```
# Load the ratings and movies data
ratings = pd.read_csv('ratings.csv')
movies = pd.read_csv('movies.csv')
```

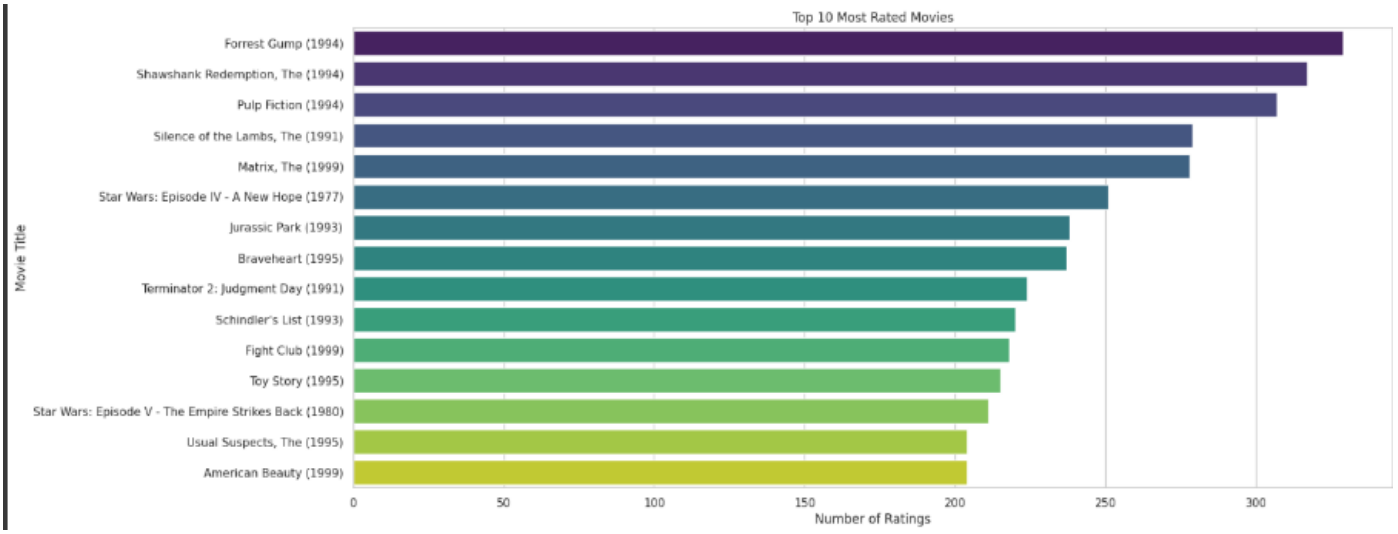
❖ Distribution of Rating



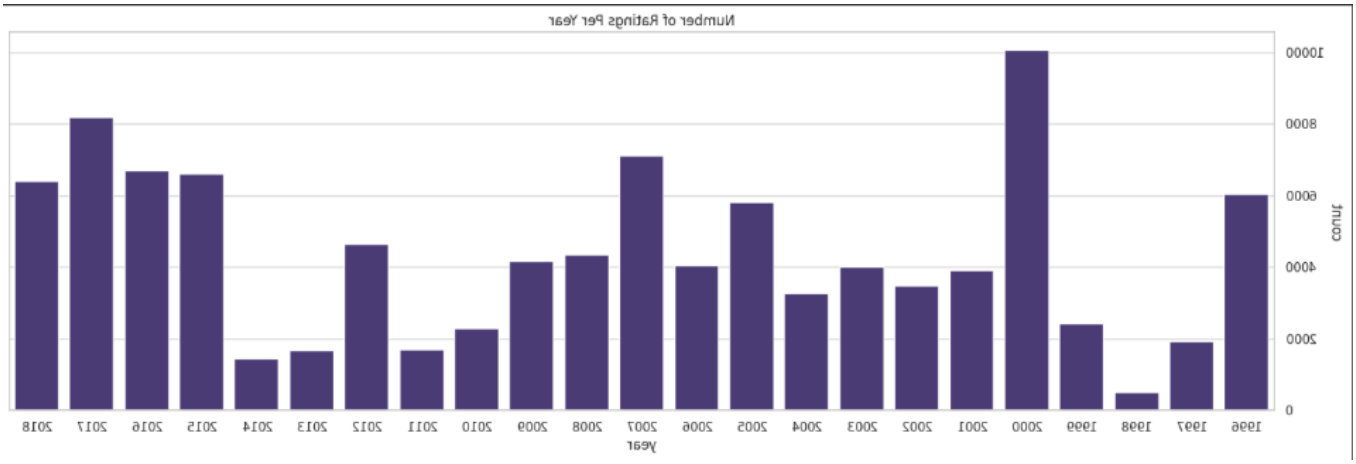
❖ Top 10 Genres Distribution



❖ Most Rated Movies



❖ Number Of Rating per Year



🔍 Insights :

- Popular Genres:
  - Drama, comedy then action are the mose frequent genres in the dataset
- Movie Popularity:
  - Forrest Gump then the shawshank redemtion have the most ratings (numbers)
- Rating Patterns:
  - in last 4 years starts to be kinda fixed about 7500 in year
- ratings :
  - most people ratings is between 3—-4

☒User-Based Collaborative Filtering

- This approach recommends movies to a user based on the preferences of similar users. It assumes that if users have similar ratings for certain movies, they will have similar preferences for other movies as well.
- Sample for the recommendation system

```
[ ]  1 recommended_movies = user_based_cf(user_id=28, top_n=5,number_of_movies=10)

[ ]  1 recommended_movies

[('Star Wars: Episode IV - A New Hope (1977)', 4.688723222127897),
 ('Seven (a.k.a. Se7en) (1995)', 4.434809088996766),
 ('Lord of the Rings: The Return of the King, The (2003)', 4.313940168889598),
 ("Ocean's Eleven (2001)", 4.097521567428778),
 ('Dead Poets Society (1989)', 4.08833963754269),
 ('Finding Nemo (2003)', 3.941673380336543),
 ('Shawshank Redemption, The (1994)', 3.864982900900911),
 ('Fight Club (1999)', 3.862099506203855),
 ('Shrek (2001)', 3.8126527855304277),
 ('28 Days Later (2002)', 3.667448019590248)]
```

☒Item-Based Collaborative Filtering:

- This method recommends movies that are similar to those the user has already rated highly. Similarity is calculated based on the overlap in ratings across users, and recommendations are made by finding items (movies) similar to the ones the user has interacted with.
- Sample for the recommendation system

```
[ ]  1 recommended_movies = user_based_cf(user_id=15, top_n=5,number_of_movies=10)

     1 recommended_movies

[('Braveheart (1995)', 4.098692625563315),
 ('Star Wars: Episode V - The Empire Strikes Back (1980)', 3.570708205060367),
 ('Lord of the Rings: The Return of the King, The (2003)', 3.4555463701458167),
 ('V for Vendetta (2006)', 3.445019614938157),
 ('Jurassic Park (1993)', 3.3731848896484657),
 ('Indiana Jones and the Last Crusade (1989)', 3.1641374530198525),
 ('Monsters, Inc. (2001)', 3.156532877823064),
 ('WALL·E (2008)', 2.987256675862717),
 ('Fight Club (1999)', 2.754520778178784),
 ('Dark Knight, The (2008)', 2.6330981996513514)]
```

☒Content-Based Collaborative Filtering:

- Content-based filtering recommends movies by analyzing the features of movies, such as genres, and matching them to the user's preferences. The system suggests movies with similar characteristics to those the user has liked or rated highly in the past.
- Sample for the recommendation system

```
[ ]  1 content_based_cf(query_genre="Sci-Fi  Mystery  Psychological Cyberpunk  Thriller  Dystopian",user_id=5,number_of_movies=10)

##USER...Found....
[('Twelve Monkeys (a.k.a. 12 Monkeys) (1995)', 0.9999999999999999),
 ('Stepford Wives, The (1975)', 0.9999999999999999),
 ('Narcopolis (2014)', 0.9999999999999999),
 ('District 9 (2009)', 0.9999999999999999),
 ('Seconds (1966)', 0.9999999999999999),
 ('Unforgettable (1996)', 0.9999999999999999),
 ('X-Files: I Want to Believe, The (2008)', 0.9644756218734531),
 ('Soylent Green (1973)', 0.9644756218734531),
 ('Moon (2009)', 0.9644756218734531),
 ('Forgotten, The (2004)', 0.9644756218734531)]
```

☒Hybrid Collaborative Filtering:

- A hybrid approach combines both collaborative and content-based filtering methods to provide more accurate and diverse recommendations. By leveraging the strengths of each method, it compensates for their individual weaknesses, leading to better overall recommendations.
- Approach for hybrid model
    1- Alpha: Weight for content-based filtering
    2- Beta: Weight for user-based filtering
    3- Gamma: Weight for item-based filtering
    4- The Sum of the Weights should be 1 (alpha + beta + gamma = 1)
- Sample for the recommendation system

```
[ ]  1 hybrid_recommend(user_id=15, query_genre="adventure sci-fi fantasy action romance",alpha=0.3, beta=0.3, gamma=0.3, number_of_movies=10)

##USER...Found....
[('Wolverine, The (2013)', 4.119616878210716),
 ('Braveheart (1995)', 3.926230364167818),
 ('Star Wars: Episode V - The Empire Strikes Back (1980)', 3.6414714792632803),
 ('Lord of the Rings: The Return of the King, The (2003)', 3.5070164977111893),
 ('V for Vendetta (2006)', 3.4966421391785225),
 ('Jurassic Park (1993)', 3.4941576030942993),
 ('Indiana Jones and the Last Crusade (1989)', 3.387450723950941),
 ('Monsters, Inc. (2001)', 3.360202899770482),
 ('WALL·E (2008)', 3.2614585655635118),
 ('Fight Club (1999)', 3.1715597016078143)]
```

- Consider the user-based , item-based , content based :

content-based recommendation :

```
[84] content_based_cf(query_genre="adventure sci-fi fantasy action romance",user_id=15,number_of_movies=10)

[('Shin Godzilla (2016)', 0.9300684306110806),
 ('Trip to the Moon, A (Voyage dans la lune, Le) (1902)', 0.9300684306110806),
 ('Krull (1983)', 0.9300684306110806),
 ('Marvel One-Shot: Agent Carter (2013)', 0.9300684306110806),
 ('Thunderbirds (2004)', 0.9300684306110806),
 ('Hellboy II: The Golden Army (2008)', 0.9300684306110806),
 ('Batman v Superman: Dawn of Justice (2016)', 0.9300684306110806),
 ('Starcrash (a.k.a. Star Crash) (1978)', 0.9300684306110806),
 ('Wolverine, The (2013)', 0.9300684306110806),
 ('Masters of the Universe (1987)', 0.9300684306110806)]
```

user-based recommendation :

```
user_based_cf(user_id=15)

[('Braveheart (1995)', 4.098692625563315),
 ('Star Wars: Episode V - The Empire Strikes Back (1980)', 3.570708205060367),
 ('Lord of the Rings: The Return of the King, The (2003)', 3.4555463701458167),
 ('V for Vendetta (2006)', 3.445019614938157),
 ('Jurassic Park (1993)', 3.3731848896484657),
 ('Indiana Jones and the Last Crusade (1989)', 3.1641374530198525),
 ('Monsters, Inc. (2001)', 3.156532877823064),
 ('WALL·E (2008)', 2.987256675862717),
 ('Fight Club (1999)', 2.754520778178784),
 ('Dark Knight, The (2008)', 2.6330981996513514)]
```

item-based recommendation :

using one of his most ratings

```
item_based_cf(movie_title="Star Wars: Episode IV - A New Hope (1977)",user_id=15,top_n=10,number_of_movies=10)
```

```
[('Star Wars: Episode V - The Empire Strikes Back (1980)', 3.7122347546800167),
 ('Star Wars: Episode I - The Phantom Menace (1999)', 3.6458308195817106),
 ('Indiana Jones and the Last Crusade (1989)', 3.6107639960111784),
 ('Jaws (1975)', 3.713250736538723),
 ('Mars Attacks! (1996)', 3.633944951279581),
 ('Total Recall (1990)', 3.681613763429202),
 ('Princess Bride, The (1987)', 3.7947601711002985),
 ('E.T. the Extra-Terrestrial (1982)', 3.8196640578766328),
 ('X-Men (2000)', 3.576235903233364),
 ('RoboCop (1987)', 3.594776194753969)]
```

❖ Insights for the output of hybrid method (consider other outputs)

- Hybrid Model Insights :
- Top Recommendation:
  - The Wolverine (2013) –-> Likely influenced by content-based recommendations, as it wasn't in user-based or item-based lists.
- Balanced Approach:
  - Retains key user-based picks like Braveheart (1995) and Lord of the Rings: The Return of the King, The (2003)
  - along with item-based favorites like Star Wars: Episode V - The Empire Strikes Back (1980).
- Diversity in Recommendations:
  - Includes animated films like Monsters, Inc. (2001) and WALL·E (2008), suggesting a broader genre coverage.
- Lower Predicted Scores: Hybrid model applies weighted average by alpha,beta and game leading to more moderate ratings compared to individual models.

```
Conclusion:
 A well-rounded mix of different recommendation strategies but slightly influenced by content-based picks.
 May need fine-tuning to ensure strong user preferences aren't diluted.
```

❖ Check for our functionality is right ---> (same output) we are good :D

>>> if beta = 1 and others = 0 –-> we will get recommendations from only user based ... lets check it

```
[86] user_based_cf(user_id=15)
```

```
[('Braveheart (1995)', 4.098692625563315),
 ('Star Wars: Episode V - The Empire Strikes Back (1980)', 3.570708205060367),
 ('Lord of the Rings: The Return of the King, The (2003)', 3.4555463701458167),
 ('V for Vendetta (2006)', 3.445019614938157),
 ('Jurassic Park (1993)', 3.3731848896484657),
 ('Indiana Jones and the Last Crusade (1989)', 3.1641374530198525),
 ('Monsters, Inc. (2001)', 3.156532877823064),
 ('WALL·E (2008)', 2.987256675862717),
 ('Fight Club (1999)', 2.754520778178784),
 ('Dark Knight, The (2008)', 2.6330981996513514)]
```

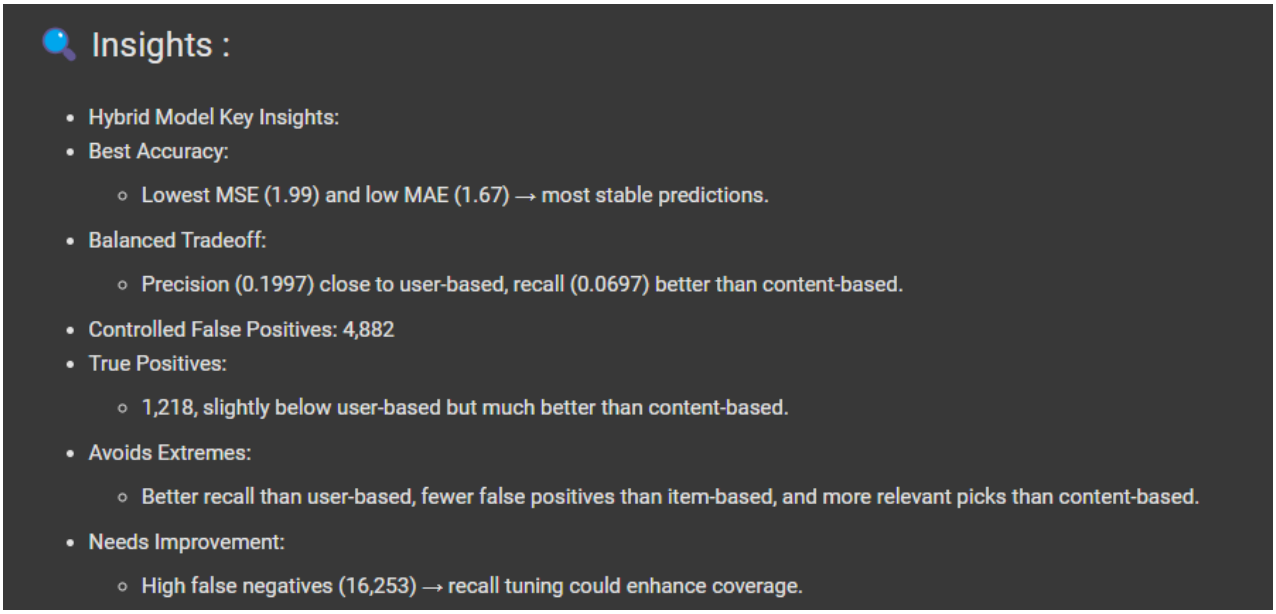
```
recommended_movies = hybrid_recommend(user_id=15, query_genre="adventure sci-fi fantasy action romance",alpha=0, beta=1, gamma=0, number_of_movies=10)

display(recommended_movies)
```

```
[('Braveheart (1995)', 4.098692625153445),
 ('Star Wars: Episode V - The Empire Strikes Back (1980)', 3.5707082047032963),
 ('Lord of the Rings: The Return of the King, The (2003)', 3.455546369800262),
 ('V for Vendetta (2006)', 3.445019614593655),
 ('Jurassic Park (1993)', 3.373184889311147),
 ('Indiana Jones and the Last Crusade (1989)', 3.164137452703439),
 ('Monsters, Inc. (2001)', 3.1565328775074106),
 ('WALL·E (2008)', 2.9872566755639913),
 ('Fight Club (1999)', 2.754520777903332),
 ('Dark Knight, The (2008)', 2.6330981993880416)]
```

☒Evaluation for models :

❖ How !!..

1- Extract User Data: Get user ratings from test_data for user_id.
2- Liked Movies: Identify movies rated above a threshold (e.g., 4).
3- Process Recommendations: Extract and average scores of recommended movies.
4- Metrics Calculation: Calculate true positives, false positives, and false negatives.
5- Precision & Recall: Compute precision (TP / (TP + FP)) and recall (TP / (TP + FN)).
6- RMSE & MAE: Compute sum of absolute errors (MAE) and squared errors (RMSE).
7- Return: Return metrics in a dictionary for the user.



☒Prediction for specific user-item pairs

1. Prediction Example

```
[ ]  1 item_based_cf(movie_title="Dark Knight, The (2008)",user_id=15,top_n=10)

[('WALL·E (2008)', 3.535660456351459),
 ('Batman Begins (2005)', 3.5231220823430514),
 ('Departed, The (2006)', 3.5854673945617077),
 ('Social Network, The (2010)', 3.458581445531892),
 ('Bourne Ultimatum, The (2007)', 3.494310449639571),
 ('Iron Man 2 (2010)', 3.374030198959302),
 ('Pirates of the Caribbean: The Curse of the Black Pearl (2003)',
  3.5122725878165646),
 ('Lord of the Rings: The Return of the King, The (2003)', 3.5584866264455677),
 ('300 (2007)', 3.5343565203165053),
 ('Taken (2008)', 3.5107362716802784)]
```

- For instance, lets recommend a group of movies based on a movie with title "Dark Knight" for a user (user_id=15)
    - User_15 predicts rating 3.53 on movie "WALL-E (2008)"
    - User_15 predicts rating 3.52 on movie" Batman Begins"
    - User_15 predicts rating 3.58 on movie "Departed, The (2006)"

2. Explanation

- The prediction is based on the similarity between User 15's rating pattern and other users who have rated Movie A highly. Additionally, the model considers Movie A's genre and compares it to User 15's preferred genres (e.g., Action, Thriller). The prediction of 3.5 is generated by factoring in the collaborative filtering approach (user similarity) and content-based filtering (movie features).

☒Strategies for Enhancing Model prediction

1. Enhanced Data Quality

- Handle missing values effectively by using imputation techniques or removing sparse data points.
- Normalize ratings to account for user biases (e.g., users who rate highly across all movies).
- Incorporate additional features like user demographics, movie descriptions, or context data (e.g., time of year) to improve recommendations.

2. Use Advanced Models:

- Matrix Factorization (SVD, NMF): Use matrix factorization techniques like Singular Value Decomposition (SVD) or Non-Negative Matrix Factorization (NMF) to capture latent factors that influence ratings.
- Deep Learning Models: Explore deep neural networks to model user-item interactions more accurately by learning complex patterns in large datasets.

3. Optimization and Tuning:

- Use hyperparameter tuning (e.g., grid search, random search) to optimize model parameters for better accuracy.
- Regularize the models to avoid overfitting and ensure generalization to unseen data.

4. Evaluation and Validation:

- Implement cross-validation techniques to validate the model on unseen data and avoid overfitting.

5. Incremental Learning:

- Update the recommendation system over time as new ratings and data come in, allowing the model to adapt and provide more up-to-date recommendations.