Movie recommendation system

Ahmed Kedić
Džanan Habibija
Anđela Jeftović

Table of contents

01

Introduction

Introduction

- What is movie recommendation system?
- Collaborative filtering VS Content-based filtering
- How did we implement a movie recommendation system?

02

Literature Review

Social and economic landscape

Content-based filtering

- Recommend movies based on attributes
- Solves cold-start problem
- Personalized reccomendations

MOVREC System

- Combines collaborative filtering + K-means clustering
- Uses IMDb dataset
- Reduces information overload

Hybrid approaches

- Collaborative, content-based, hybrid systems
- Cold-start, sparse ratings, large-scale data
- Suggests novel similarity metric

Hotel recommendation system

- Collaborative filtering, NLP, classification
- High precision and recall with large datasets
- ML applicability across domains

Systematic review

- Covers algorithms: filtering, clustering, metaheuristics
- Datasets: MovieLens, IMDb, Netflix
- Hybrid systems, future trends

03

Dataset

Dataset

For our movie recommendation system we are using 9000+ Movies Dataset.
This dataset contains following columns:

- Release_Date
- Title
- Overview
- Popularity
- Vote_Count
- Vote_Average
- Original_Language
- Genre
- Poster_Url

	Release_Date	Title	Overview	Popularity	Vote_Count	Vote_Average	Original_Language	Genre	Poster_Url
0	2021-12-15	Spider-Man: No Way Home	Peter Parker is unmasked and no longer able to...	5083.954	8940	8.3	en	Action, Adventure, Science Fiction	https://image.tmdb.org/t/p/original/1g0dhYtq4i...
1	2022-03-01	The Batman	In his second year of fighting crime, Batman u...	3827.658	1151	8.1	en	Crime, Mystery, Thriller	https://image.tmdb.org/t/p/original/74xTEgt7R3...
2	2022-02-25	No Exit	Stranded at a rest stop in the mountains durin...	2618.087	122	6.3	en	Thriller	https://image.tmdb.org/t/p/original/vDHsLnOWKl...
3	2021-11-24	Encanto	The tale of an extraordinary family, the Madri...	2402.201	5076	7.7	en	Animation, Comedy, Family, Fantasy	https://image.tmdb.org/t/p/original/4j0PNHkMr5...
4	2021-12-22	The King's Man	As a collection of history's worst tyrants and...	1895.511	1793	7.0	en	Action, Adventure, Thriller, War	https://image.tmdb.org/t/p/original/aq4Pwv5Xeu...
...	...	...	...	...	...	...	...	...	...
9832	1973-10-15	Badlands	A dramatization of the Starkweather-Fugate kil...	13.357	896	7.6	en	Drama, Crime	https://image.tmdb.org/t/p/original/z81rBzHNgi...
9833	2020-10-01	Violent Delights	A female vampire falls in love with a man she ...	13.356	8	3.5	es	Horror	https://image.tmdb.org/t/p/original/4b6HY7rud6...
9834	2016-05-06	The Offering	When young and successful reporter Jamie finds...	13.355	94	5.0	en	Mystery, Thriller, Horror	https://image.tmdb.org/t/p/original/h4uMM1wOhz...
9835	2021-03-31	The United States vs. Billie Holiday	Billie Holiday spent much of her career being ...	13.354	152	6.7	en	Music, Drama, History	https://image.tmdb.org/t/p/original/vEzkxuE2sJ...
9836	1984-09-23	Threads	Documentary style account of a nuclear holocau...	13.354	186	7.8	en	War, Drama, Science Fiction	https://image.tmdb.org/t/p/original/lBhU4U9Eeh...

9837 rows × 9 columns

04

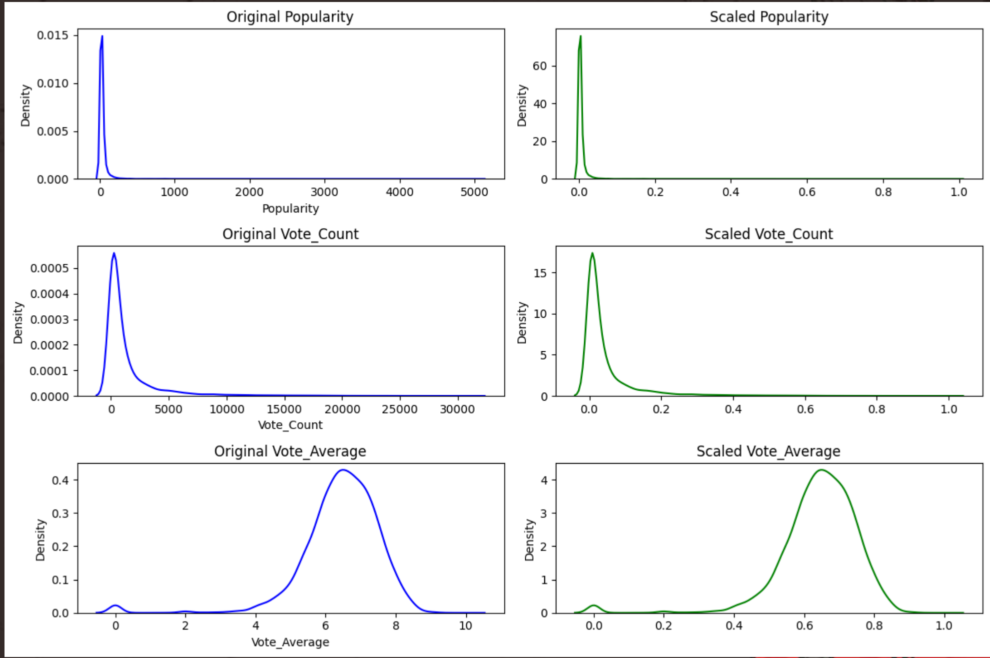
Data Cleaning and Preprocessing

Data cleaning and preprocessing

.info() .head() .describe()	Explored dataset
.isnull().sum() **.duplicated().any()**	Checked for missing values and duplicates
.fillna()	Replaced missing Vote_Count with column mean.
.median() .mean() .mode()	Analyzed distribution: mean, median, mode, quartiles
.min() .max()	Scaling: Min-Max scaling for numerical features

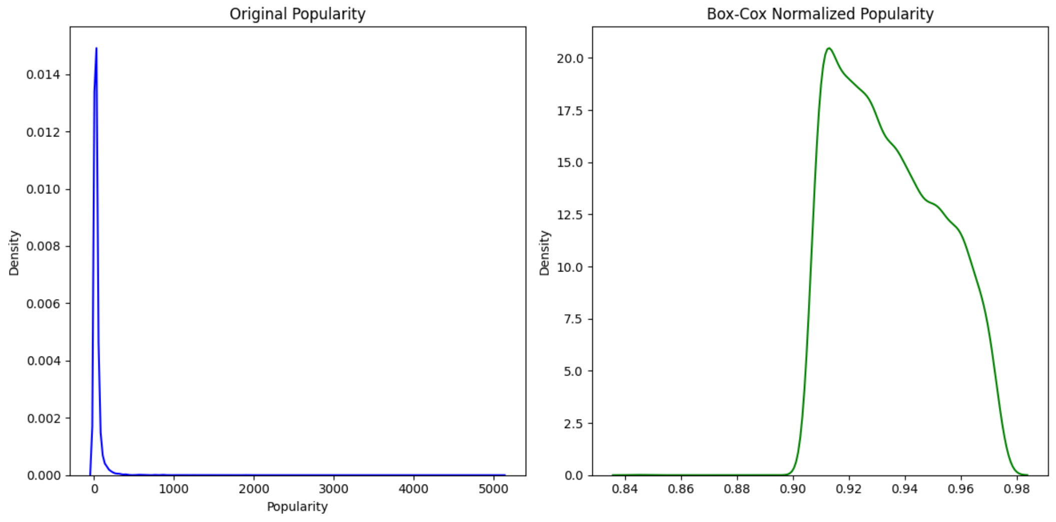
Data cleaning and preprocessing – Scaling

```python
import pandas as pd
from sklearn.preprocessing import minmax_scale
import seaborn as sns
import matplotlib.pyplot as plt
from scipy import stats

data['Vote_Count'] = pd.to_numeric(data['Vote_Count'], errors='coerce')
data['Vote_Average'] = pd.to_numeric(data['Vote_Average'], errors='coerce')

columns_to_scale = ['Popularity', 'Vote_Count', 'Vote_Average']

scaled_data = minmax_scale(data[columns_to_scale], axis=0)

fig, ax = plt.subplots(len(columns_to_scale), 2, figsize=(12, 8))

for i, col in enumerate(columns_to_scale):
    sns.kdeplot(data[col], ax=ax[i][0], color='blue')
    ax[i][0].set_title(f"Original {col}")
    sns.kdeplot(scaled_data[:, i], ax=ax[i][1], color='green')
    ax[i][1].set_title(f"Scaled {col}")

plt.tight_layout()
plt.show()
```



Data cleaning and preprocessing – Normalization

```python
from scipy import stats

positive_popularity = data['Popularity'][data['Popularity'] > 0]

normalized_popularity, _ = stats.boxcox(positive_popularity)

fig, ax = plt.subplots(1, 2, figsize=(12, 6))
sns.kdeplot(positive_popularity, ax=ax[0], color='blue')
ax[0].set_title("Original Popularity")
sns.kdeplot(normalized_popularity, ax=ax[1], color='green')
ax[1].set_title("Box-Cox Normalized Popularity")
plt.tight_layout()
plt.show()
```



05

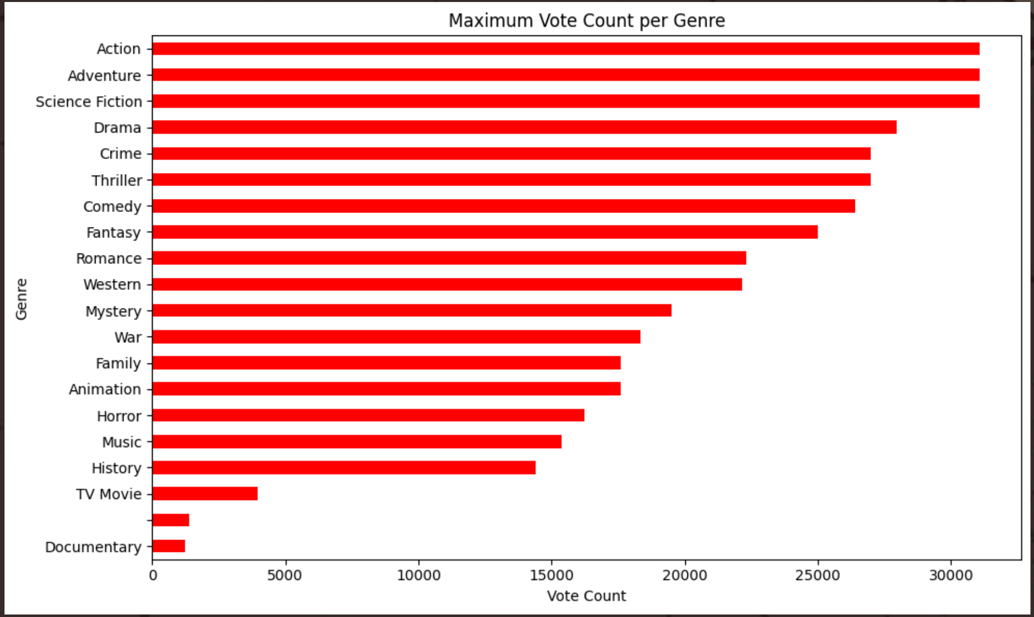
Data Analysis

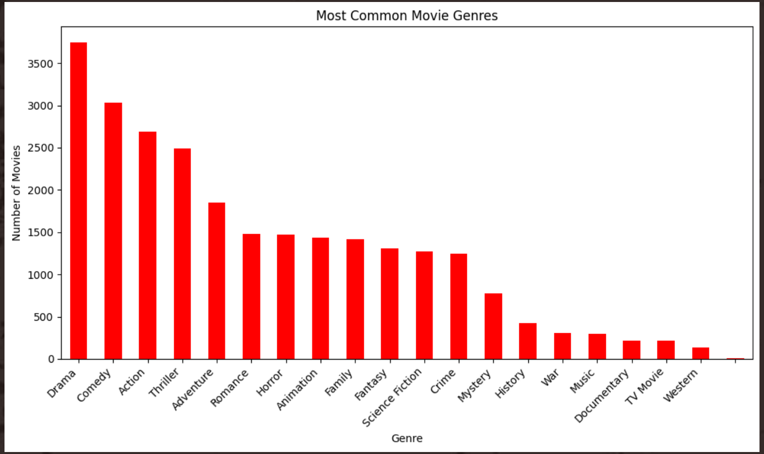
Data analysis overview

- Genre analysis
- Popularity trends
- Language insights
- Top movies and ratings
- Keywords analysis

Genre analysis

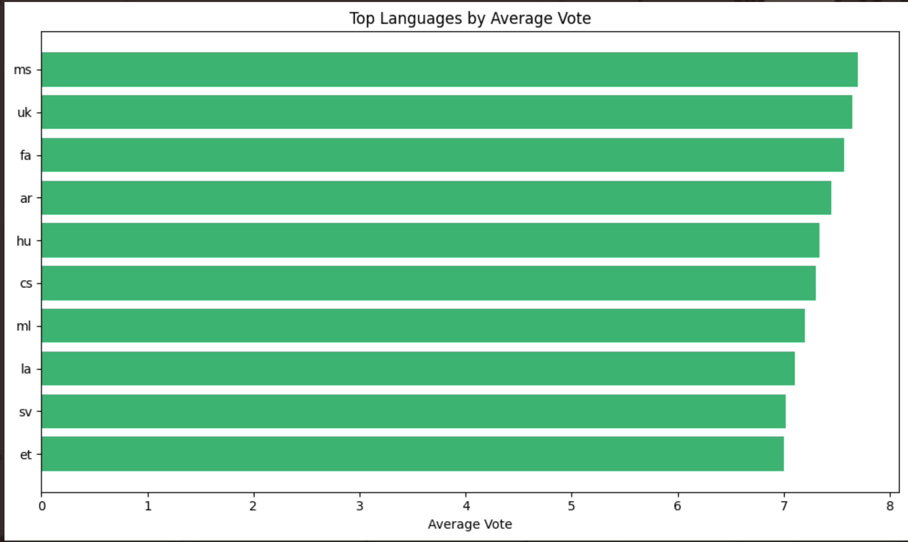
- Genre popularity analysis
- Maximum vote count per genre
- Biggest vote average per genre
- Most common genres
- Top rated movies per genre

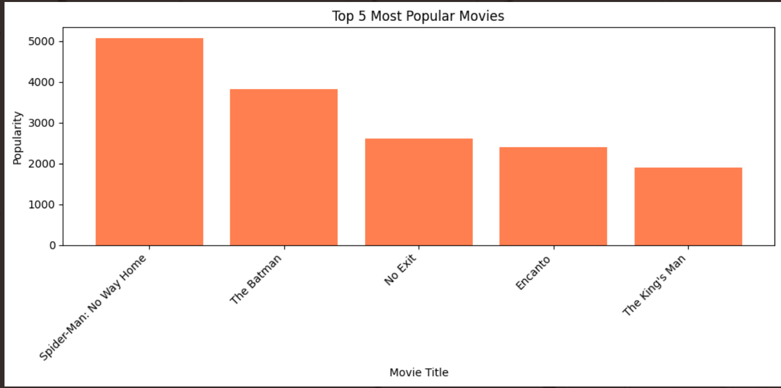




Popularity, top movies and language insight

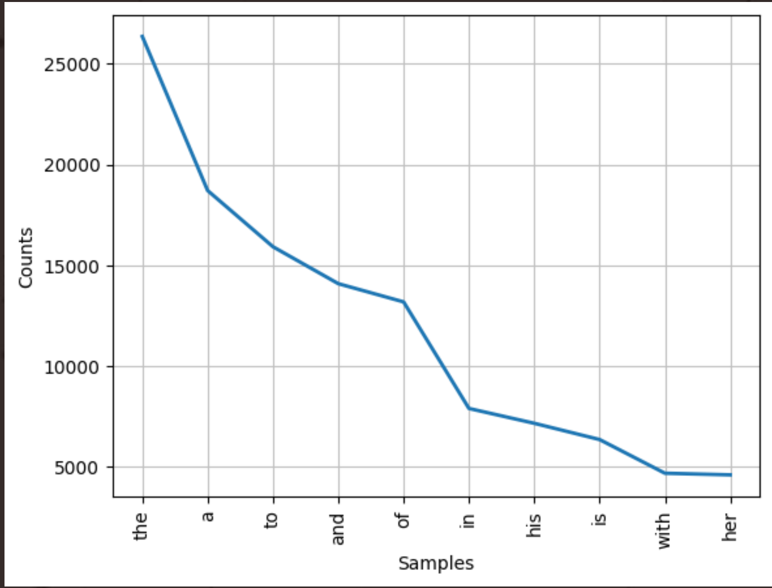
- Popularity per year
- Top 5 most popular movies per language
- The most popular movie for each language
- Top largest movies with largest popularity
- Top rated movies per genre
- Top languages by average vote





Keywords analysis

- Most Common Words in Movie Overviews



06

Machine Learning Algorithms

Machine learning algorithms

- K-Nearest Neighbors (KNN)
- KMeans Clustering
- Random Forest Classifier

K-Nearest Neighbors (KNN)

- Based on movie **overview text** using **TF-IDF + Cosine Similarity**
- Returns top 5 similar movies for any input movie
- Simple and effective for small/medium datasets

```python
def recommend_movies(title, num_recommendations=5):
    title = title.lower()
    if title not in indices:
        return f"Movie '{title}' not found in dataset."

    idx = indices[title]
    distances, indices_rec = nn_model.kneighbors(tfidf_matrix[idx], n_neighbors=num_recommendations + 1)
    recommended_indices = indices_rec[0][1:]  # Exclude the input movie itself

    return data[['Title', 'Overview']].iloc[recommended_indices]
```

```python
recommend_movies("Spider-man")
```

	Title	Overview
7949	Beyond the Ultimate Spin: The Making of 'Spide...	Documentary on the making of 'Spider-Man.'
201	Spider-Man 3	The seemingly invincible Spider-Man goes up ag...
1500	Spider-Man	When an extortionist threatens to force a mult...
168	Spider-Man: Homecoming	Following the events of Captain America: Civil...
191	Spider-Man: Into the Spider-Verse	Miles Morales is juggling his life between bei...

KNN Evaluation

```
def evaluate_knn_recommender(title, num_recommendations=5):
    title = title.lower()
    if title not in indices:
        return f"Movie '{title}' not found in dataset."

    idx = indices[title]
    distances, indices_rec = nn_model.kneighbors(tfidf_matrix[idx], n_neighbors=num_recommendations + 1)
    recommended_indices = indices_rec[0][1:]

    # Similarity scores (1 - distance)
    similarities = 1 - distances[0][1:]
    avg_similarity = np.mean(similarities)

    print(f"\nKNN Recommender Evaluation for '{title}':")
    print(f"Average Cosine Similarity to Recommended Movies: {avg_similarity:.3f}")
    return data[['Title', 'Overview']].iloc[recommended_indices]

evaluate_knn_recommender("Parasyte: Part 1")
```

```
KNN Recommender Evaluation for 'parasyte: part 1':
Average Cosine Similarity to Recommended Movies: 0.334
```

	Title	Overview
1746	Parasyte: Part 2	Alien pods come to Earth and, naturally, start...
9731	Crayon Shin-chan: Fierceness That Invites Stor...	The Noharas get abducted by aliens who claim t...
7158	Alien Nation	A few years from now, Earth will have the firs...
3649	Attraction	After an alien ship crash lands in a Russian c...
9627	Crayon Shin-chan: Action Mask vs. Leotard Devil	Everyone's favorite TV superhero Action Mask s...

KMeans Clustering

- Used MultiLabelBinarizer to convert genres to binary matrix
- Grouped movies into **5 clusters** based on genre similarity
- Recommends movies from the same cluster

```
score = silhouette_score(genre_matrix, data['Genre_Cluster'])
print(f"Silhouette Score: {score:.3f}")
```

```
Silhouette Score: 0.217
```

```
def recommend_by_genre_cluster(title, n=5):
    title_lower = title.lower()
    matches = data.index[data['Title'].str.lower() == title_lower]
    if len(matches) == 0:
        return f"Movie '{title}' not found."

    idx = matches[0]
    cluster_id = data.at[idx, 'Genre_Cluster']

    peers = data[data['Genre_Cluster'] == cluster_id].drop(idx)
    if peers.empty:
        return f"No other movies in cluster {cluster_id}."

    sample = peers.sample(min(n, len(peers)), random_state=42)
    return sample[['Title', 'Genre_list']]
```

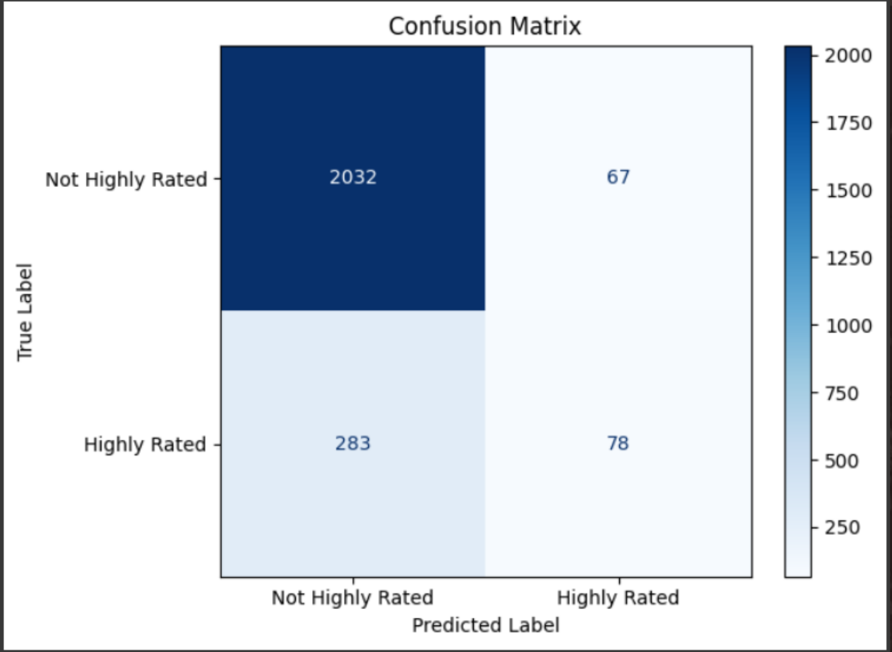
```
recommend_by_genre_cluster("The Batman", n=5)
```

	Title	Genre_list
1885	Friday the 13th Part 2	[Horror, Thriller]
7559	Sexy Beast	[Crime, Drama, Thriller]
2514	Run and Gun	[Action, Thriller]
4421	The Silence	[Horror, Drama, Thriller, Fantasy]
6791	Psycho II	[Horror, Mystery, Thriller]

Random Forest Classifier

- Supervised learning to predict if a movie is **Highly Rated (>= 7.5)**
- Used features: Popularity, Vote Count, Genre
- Trained on 75% of data, tested on 25%

	precision	recall	f1-score	support
0	0.88	0.97	0.92	2099
1	0.54	0.22	0.31	361
accuracy			0.86	2460
macro avg	0.71	0.59	0.61	2460
weighted avg	0.83	0.86	0.83	2460



Recommendations for 'Batman':

	Title	Overview
260	Batman Returns	While Batman deals with a deformed man calling...
1639	The Batman vs. Dracula	Gotham City is terrorized not only by recent e...
2162	Batman Beyond: Return of the Joker	The Joker is back with a vengeance, and Gotham...
221	The Dark Knight	Batman raises the stakes in his war on crime. ...
2487	Batman Unmasked: The Psychology of the Dark Kn...	Delve into the world of Batman and the vigilan...

ML Algorithms Comparison

	KNN	KMeans	Random Foresr
Goal	Recommended similar movies based on Overviews.	Clear understanding with well-developed ideas	Predict whether a movie will be highly rated
Type	Unsupervised	Unsupervised	Supervised
Personalization	Yes	Yes	No
Output	Top N movies based on similarities	Top N movies in same cluster	Wheather movie is highly rated or not
Performance	Good	Clusters are overlapping	Good prediction
Strengths	Fine-grained, story-based similarity. Great for personalized recommendations	Captures genre-based groupings. Simple, fast	Predictive power Uses multiple features -Interpretable outpu
Weaknesses	Ignores genre or popularity. Doesn't explain why results are similar	Clusters may be ambiguous due to genre overlap. Less personalized	Needs labeled data. Not used for content recommendation

07

Conclusion

Conclusion

Each model had its own purpose and strengths:

- **KNN** gave accurate and personalized results based on movie overviews.
- **KMeans** grouped movies effectively by genre, though with some overlap.
- **Random Forest** achieved good predictive performance using multiple features.

Our system demonstrates that combining multiple ML models creates a more accurate and informative movie recommendation and analysis tool.

Thank you!