Recommender System

Load the data

In [22]:
```
import pandas as pd
import numpy as np
# Load the data
links_df = pd.read_csv('/Users/nickblackford/Downloads/ml-latest-small/links
movies_df = pd.read_csv('/Users/nickblackford/Downloads/ml-latest-small/movi
ratings_df = pd.read_csv('/Users/nickblackford/Downloads/ml-latest-small/rat
tags_df = pd.read_csv('/Users/nickblackford/Downloads/ml-latest-small/tags.c
```

Data Preprocessing

In [27]:
```
# Merge movies and ratings data
data = pd.merge(ratings_df, movies_df, on='movieId')

# Create a utility matrix
utility_matrix = data.pivot_table(index='userId', columns='title', values='r
```

Pearson Correlation

In [28]:
```
# Calculate the Pearson correlation between movies
movie_similarity = utility_matrix.corr(method='pearson')
movie_similarity.head()
```

Out[28]:

title	'71 (2014)	'Hellboy': The Seeds of Creation (2004)	'Round Midnight (1986)	'Salem's Lot (2004)	'Til There Was You (1997)	'Tis the Season for Love (2015)	'burbs, The (1989)	'night Mother (1986)	[S
title									
'71 (2014)	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
'Hellboy': The Seeds of Creation (2004)	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
'Round Midnight (1986)	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
'Salem's Lot (2004)	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
'Til There Was You (1997)	NaN	NaN	NaN	NaN	1.0	NaN	NaN	NaN	

5 rows × 9719 columns

Generate Recommendations

In [25]:
```
# Create function that takes a movie title and returns the top 10 most simil
def recommend_movies(movie_title, num_recommendations=10):
    similar_scores = movie_similarity[movie_title].dropna().sort_values(asce
    recommendations = similar_scores.iloc[1:num_recommendations + 1].index
    return recommendations

# Example
recommendations = recommend_movies("Toy Story (1995)")
recommendations
```

Out[25]:
```
Index(['Claim, The (2000)', 'Stalker (1979)',
       'Halloween III: Season of the Witch (1982)',
       'Eddie Murphy Delirious (1983)', 'Guy Thing, A (2003)',
       'Brigadoon (1954)',
       'Hearts of Darkness: A Filmmakers Apocalypse (1991)',
       'Hall Pass (2011)', 'Perfect Candidate, A (1996)',
       'Perfect Blue (1997)'],
      dtype='object', name='title')
```

Summary

Data Loading and Preprocessing:

We loaded the MovieLens dataset, merged ratings with movie metadata, and created a utility matrix where rows represent users and columns represent movie titles.

Similarity Calculation:

We calculated the Pearson correlation between movies to create a similarity matrix. This matrix captures the linear relationships between movie ratings.

Generating Recommendations:

We implemented a function to recommend movies based on the highest Pearson correlation values. Example recommendations were generated for a specific movie to demonstrate the system.

Interpretation and Analysis:

Pearson correlation values indicate the strength and direction of the linear relationship between movie ratings. High correlation values suggest similar user ratings and form the basis for recommendations. The model's effectiveness can be evaluated using coverage, diversity, novelty, accuracy, and user feedback.

Sources

- https://analyticsindiamag.com/ai-mysteries/how-to-build-your-first-recommender-system-using-python-movielens-dataset/
- https://www.geeksforgeeks.org/recommendation-system-in-python/#