Christian Campbell

Recommender System

In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.metrics.pairwise import cosine_similarity
from sklearn.feature_extraction.text import CountVectorizer
import random
```

In [2]:
```
path = r"C:\Users\chris\Documents\Bellevue University\7 - Predictive Analytics\movies.csv"
movies = pd.read_csv(path)
movies.head()
```

Out[2]:

	movieId	title	genres
0	1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy
1	2	Jumanji (1995)	Adventure|Children|Fantasy
2	3	Grumpier Old Men (1995)	Comedy|Romance
3	4	Waiting to Exhale (1995)	Comedy|Drama|Romance
4	5	Father of the Bride Part II (1995)	Comedy

In [3]:

```
# Extracts the year and cleans the title
movies['year'] = movies['title'].str.extract(r'\((\d{4})\)', expand=False)
movies['title'] = movies['title'].str.replace(r'\(\d{4}\)', '', regex=True).str.strip()

# Converts genres to a string
movies['genres_str'] = movies['genres'].str.replace('|', ' ')

# Creates a count vectorizer and computes the cosine similarity matrix
count_matrix = CountVectorizer().fit_transform(movies['genres_str'])
cosine_sim = cosine_similarity(count_matrix, count_matrix)

# Functions to get recommendations
def get_index_from_title(title):
    return movies[movies['title'].str.contains(title, case=False)].index[0]

def get_recommendations(title, num_recommendations=10):
    movie_index = get_index_from_title(title)
    similarity_scores = list(enumerate(cosine_sim[movie_index]))
    similarity_scores = sorted(similarity_scores, key=lambda x: x[1], reverse=True)[1:]
    random.shuffle(similarity_scores)
    movie_indices = [i[0] for i in similarity_scores[:num_recommendations]]
    return movies['title'].iloc[movie_indices]

# Recommender system function
def movie_recommender_system(movie_title, num_recommendations=10):
    try:
        recommendations = get_recommendations(movie_title, num_recommendations)
        return recommendations
    except IndexError:
        return "Movie not found in the dataset. Please check the title and try again."
```

In [8]:
```
# Code for the recommender system
user_input = 'Casper'
print(f"Recommendations for '{user_input}':")
print(movie_recommender_system(user_input))
```

```
Recommendations for 'Casper':
455            Robin Hood: Men in Tights
7406                         Jackass 2.5
4641                                 Elf
1604                     St. Elmo's Fire
6287    Wind That Shakes the Barley, The
4786                      Spring Forward
5260                        Spider-Man 2
8128                         Mezzo Forte
2541                        Turtle Diary
1623                               Blade
Name: title, dtype: object
```

In [ ]:
```

```