```
import pandas as pd
from sklearn.metrics.pairwise import cosine_similarity

df = pd.read_csv("/Users/colinmichael/Desktop/Data Science/DSC
630/movies.csv")
df2 = pd.read_csv("/Users/colinmichael/Desktop/Data Science/DSC
630/ratings.csv")

left_join_df = pd.merge(df, df2, on='movieId', how='left')

left_join_df
df = left_join_df.drop(['timestamp', 'movieId'], axis=1)

#first, I want to find the average rating for each movie
movie_ratings = pd.DataFrame(df.groupby('title')['rating'].mean())
movie_ratings['num_ratings'] = pd.DataFrame(df.groupby('title')
['rating'].count())


#then, I want to find each user's movie ratings with a pivot table
user_ratings = pd.pivot_table(df, index='userId', columns='title',
values='rating')

# define function to recommend similar movies based on a given movie
def recommend_movies(movie_title, top_n=10, genre=None):

    # gets the row of the given movie
    movie_row = user_ratings[movie_title]

    # finds the correlation between given movie and all other movies
    similarity = user_ratings.corrwith(movie_row)
    similarity = similarity.dropna()

    # creates dataframe of similarity values and number of ratings for
each similar movie
    similar_movies = pd.DataFrame(similarity, columns=['similarity'])
    similar_movies = similar_movies.join(movie_ratings['num_ratings'])

    # filters out movies with too few ratings to make the engine run
faster
    similar_movies = similar_movies[similar_movies['num_ratings'] >
50]

    # if a genre is specified, filter out movies that do not match the
genre
    if genre is not None:
        genre_movies =
movies_df[movies_df['genres'].str.contains(genre)]
        genre_titles = set(genre_movies['title'].tolist())
        similar_movies =
similar_movies[similar_movies.index.isin(genre_titles)]
```

```
    # sort by similarity and select top 10 similar movies
    similar_movies = similar_movies.sort_values('similarity',
ascending=False)
    similar_movies = similar_movies.head(top_n)

    return similar_movies.index.tolist()

# ask the user to enter a movie title and year
movie_title = input("Please enter a movie title (with year):
").strip()

# take out the year of the movie
year_start = movie_title.rfind('(') + 1
year_end = movie_title.rfind(')')
year = movie_title[year_start:year_end]


# get recommended movies
recommendations = recommend_movies(movie_title)

# print the resulting recommended movies
print("Recommended movies based on", movie_title)
for movie in recommendations:
    print(movie)
```

```
Please enter a movie title (with year): Memento (2000)
Recommended movies based on Memento (2000)
Memento (2000)
Piano, The (1993)
Philadelphia (1993)
Dead Man Walking (1995)
Grumpier Old Men (1995)
Life Is Beautiful (La Vita è bella) (1997)
Children of Men (2006)
Sense and Sensibility (1995)
Crow, The (1994)
28 Days Later (2002)
```