# Week6_Mounika_Lakureddy

February 25, 2024

[1]:
```python
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.metrics.pairwise import cosine_similarity, pairwise_distances
```

## 0.1 Load datasets

[2]:
```python
movies = pd.read_csv('movies.csv')
movies.sample(3)
```

[2]:
```
      movieId                                              title  \
7218    73160  Sorority Babes in the Slimeball Bowl-O-Rama (1…
9628   178615                                 Front Cover (2016)
2335     3093                           McCabe & Mrs. Miller (1971)

                   genres
7218         Comedy|Horror
9628  Comedy|Drama|Romance
2335         Drama|Western
```

[3]:
```python
ratings = pd.read_csv('ratings.csv')
ratings.sample(3)
```

[3]:
```
       userId  movieId  rating   timestamp
86897     561     7361     3.0  1491092287
22674     155     3286     4.0   963871903
64873     414    97306     4.0  1519594433
```

## 0.2 Subset for user 2

[4]:
```python
user_2_ratings = ratings[ratings['userId'] == 2]
user_2_ratings
```

[4]:
```
     userId  movieId  rating   timestamp
232       2      318     3.0  1445714835
233       2      333     4.0  1445715029
234       2     1704     4.5  1445715228
235       2     3578     4.0  1445714885
236       2     6874     4.0  1445714952
```

```
237        2     8798    3.5  1445714960
238        2    46970    4.0  1445715013
239        2    48516    4.0  1445715064
240        2    58559    4.5  1445715141
241        2    60756    5.0  1445714980
242        2    68157    4.5  1445715154
243        2    71535    3.0  1445714974
244        2    74458    4.0  1445714926
245        2    77455    3.0  1445714941
246        2    79132    4.0  1445714841
247        2    80489    4.5  1445715340
248        2    80906    5.0  1445715172
249        2    86345    4.0  1445715166
250        2    89774    5.0  1445715189
251        2    91529    3.5  1445714891
252        2    91658    2.5  1445714938
253        2    99114    3.5  1445714874
254        2   106782    5.0  1445714966
255        2   109487    3.0  1445715145
256        2   112552    4.0  1445714882
257        2   114060    2.0  1445715276
258        2   115713    3.5  1445714854
259        2   122882    5.0  1445715272
260        2   131724    5.0  1445714851
```

The output is showing the ratings given by User 2 for various movies. Each row corresponds to a movie rated by User 2, indicating the movieId, rating, and timestamp of each rating.

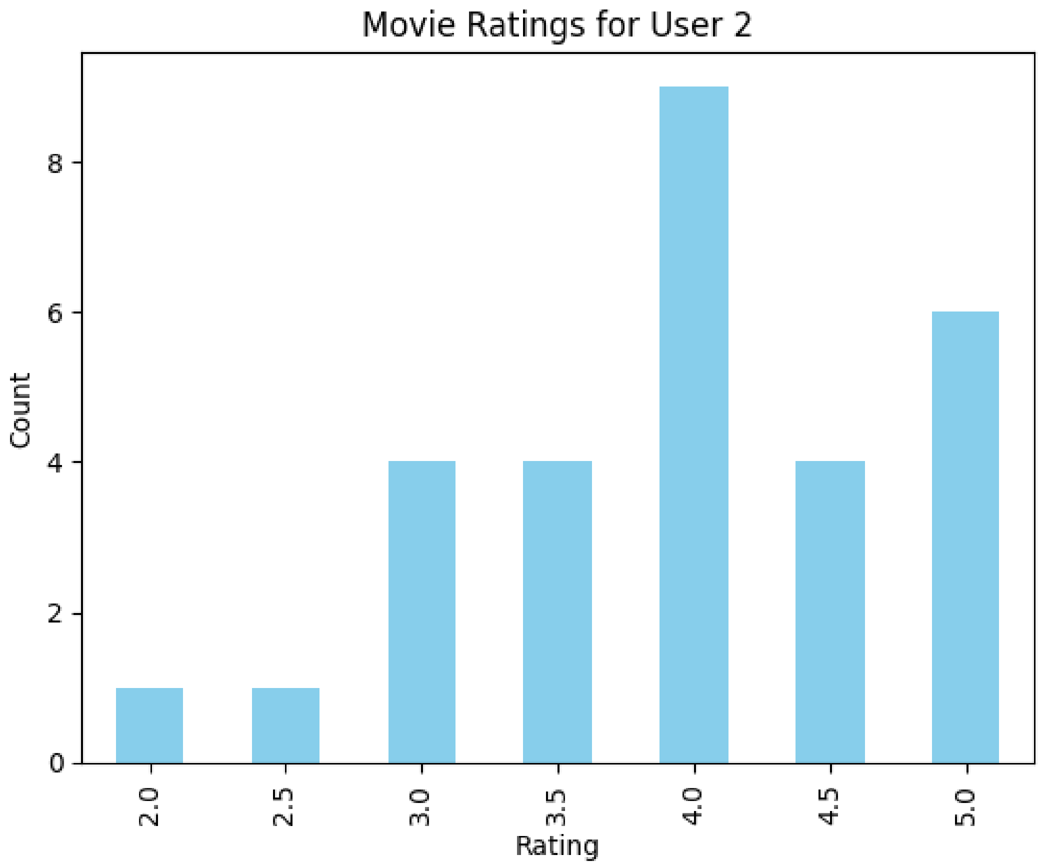
### 0.3 1. Movies user2 has watched

[5]:
```
num_movies_watched = len(user_2_ratings)
print(f"User 2 has watched {num_movies_watched} movies.")
```

```
User 2 has watched 29 movies.
```

### 0.4 2. Bar chart for user 2's movie ratings

[6]:
```
rating_counts = user_2_ratings['rating'].value_counts().sort_index()
rating_counts.plot(kind='bar', color='skyblue')
plt.xlabel('Rating')
plt.ylabel('Count')
plt.title('Movie Ratings for User 2')
plt.show()
```




`rating_counts = user_2_ratings['rating'].value_counts().sort_index()`

This line calculates the count of each unique rating given by User 2. It uses the value_counts() function to count occurrences of each unique rating in the 'rating' column of the user_2_ratings DataFrame. The sort_index() function is then used to sort the unique ratings in ascending order.

`rating_counts.plot(kind='bar', color='skyblue')`

We create a bar chart using the plot() function. The kind='bar' parameter specifies that a bar chart should be created. The color='skyblue' parameter sets the color of the bars to sky blue.

`plt.xlabel('Rating')`

Adds a label to the x-axis of the plot, indicating that it represents the different movie ratings.

`plt.ylabel('Count')`

Adds a label to the y-axis of the plot, indicating that it represents the count of movies for each rating.

`plt.title('Movie Ratings for User 2')`

Here we set the title of the plot to 'Movie Ratings for User 2'.

`plt.show()` - display the plot.

The bar chart visually represents the distribution of movie ratings given by User 2. Each bar on the chart corresponds to a unique movie rating, and the height of the bar represents the count of movies that received that rating from User 2. The x-axis shows the different ratings, and the y-axis shows the count of movies for each rating.

## 0.5 3. User 2's top movies

[7]:
```
user_2_top_movies = user_2_ratings.merge(movies, on='movieId')[['title',␣
↪'rating']].sort_values(by='rating', ascending=False)
print("User 2's top movies:")
print(user_2_top_movies.head())
```

```
User 2's top movies:
                                              title  rating
28  The Jinx: The Life and Deaths of Robert Durst …     5.0
27                        Mad Max: Fury Road (2015)     5.0
22                  Wolf of Wall Street, The (2013)     5.0
18                                   Warrior (2011)     5.0
9                              Step Brothers (2008)     5.0
```

## 0.6 4. Most similar user to user 2 using cosine and manhattan distances

[8]:
```
user_2_vector = user_2_ratings.pivot(index='userId', columns='movieId',␣
↪values='rating').fillna(0)
all_users_vector = ratings.pivot(index='userId', columns='movieId',␣
↪values='rating').fillna(0)
```

## 0.7 Ensuring both user vectors have the same columns

[9]:
```
common_columns = user_2_vector.columns.intersection(all_users_vector.columns)

user_2_vector = user_2_vector[common_columns]
all_users_vector = all_users_vector[common_columns]
```

## 0.8 Use the distances

[10]:
```
cosine_similarities = cosine_similarity(user_2_vector, all_users_vector)
manhattan_distances = pairwise_distances(user_2_vector, all_users_vector,␣
↪metric='manhattan')
```

## 0.9 Display results

[11]:
```
print(f"Most similar user to User 2 (cosine): {cosine_similarities}")
print(f"Most similar user to User 2 (manhattan): {manhattan_distances}")
```

```
Most similar user to User 2 (cosine): [[0.26086186 1.         0.
0.20751434 0.13834289 0.22825413
  0.24495082 0.13834289 0.         0.35144117 0.24455799 0.
```

| | | | | | |
|---|---|---|---|---|---|
| 0.18445719 | 0.13834289 | 0.49455168 | 0.39615764 | 0.43027895 | 0.77644315 |
| 0.2557961 | 0.18445719 | 0.60584711 | 0.46293651 | 0.18445719 | 0.48089623 |
| 0.46106757 | 0. | 0. | 0.47295934 | 0.38972905 | 0.37121293 |
| 0. | 0.13834289 | 0.24846359 | 0.23045899 | 0. | 0.23518292 |
| 0.13834289 | 0.13834289 | 0. | 0.13834289 | 0.58971767 | 0.35410056 |
| 0.13834289 | 0. | 0.33206224 | 0.13834289 | 0.33328977 | 0. |
| 0.26687054 | 0.61863671 | 0.26809823 | 0.55354535 | 0. | 0.13834289 |
| 0.18445719 | 0.22134863 | 0.27207436 | 0.22825413 | 0. | 0.24846359 |
| 0.22325714 | 0.77977403 | 0.57293207 | 0.38879219 | 0.46973355 | 0.39010257 |
| 0.3432886 | 0.83476077 | 0.22825413 | 0.24455799 | 0. | 0.13834289 |
| 0.60259073 | 0.18445719 | 0. | 0.49600239 | 0.34992614 | 0.13834289 |
| 0.18445719 | 0.53635901 | 0.13834289 | 0.18445719 | 0.38615679 | 0.13834289 |
| 0.20751434 | 0.35609211 | 0. | 0.44953772 | 0.13834289 | 0. |
| 0.38974874 | 0. | 0.24846359 | 0.13834289 | 0. | 0.18445719 |
| 0.18445719 | 0.35138059 | 0.22825413 | 0. | 0.18445719 | 0.13834289 |
| 0.67971357 | 0. | 0.79538375 | 0.13834289 | 0.13834289 | 0. |
| 0.13834289 | 0. | 0.76533592 | 0.43129063 | 0. | 0.31974512 |
| 0.18445719 | 0.27300317 | 0.21353049 | 0. | 0.67217531 | 0. |
| 0.13834289 | 0.65273801 | 0.41003535 | 0.13834289 | 0.65606915 | 0.13834289 |
| 0. | 0. | 0.26086186 | 0.13834289 | 0.39079356 | 0.48317866 |
| 0.13834289 | 0. | 0.19695383 | 0.18445719 | 0.22825413 | 0. |
| 0.4663538 | 0.42161507 | 0.5518402 | 0. | 0.22685235 | 0.18445719 |
| 0.13834289 | 0. | 0. | 0.28065806 | 0. | 0. |
| 0. | 0.45828757 | 0.35244558 | 0.30617717 | 0.26086186 | 0.30128404 |
| 0. | 0. | 0.37678124 | 0.18445719 | 0.13834289 | 0.13834289 |
| 0.20751434 | 0.18445719 | 0. | 0.42931207 | 0.13834289 | 0.20751434 |
| 0.28173982 | 0.13834289 | 0.13834289 | 0.18445719 | 0. | 0.18445719 |
| 0. | 0.13834289 | 0.65706475 | 0.13834289 | 0.13834289 | 0. |
| 0. | 0.39741418 | 0. | 0.45234816 | 0. | 0.18445719 |
| 0.28258593 | 0. | 0.43437052 | 0.35738581 | 0.13834289 | 0. |
| 0.13834289 | 0. | 0.29929141 | 0.30977174 | 0.13834289 | 0.29346959 |
| 0.37298329 | 0.44711516 | 0. | 0.13834289 | 0.25616126 | 0.35254487 |
| 0.31748392 | 0. | 0. | 0. | 0.46647338 | 0.20751434 |
| 0.48502631 | 0.63712365 | 0.20751434 | 0. | 0.24164924 | 0.18445719 |
| 0. | 0. | 0.48983055 | 0.36719262 | 0.13834289 | 0.62811663 |
| 0.40734296 | 0. | 0.18445719 | 0.45550032 | 0.62822688 | 0.13834289 |
| 0.13834289 | 0.35200082 | 0.30296944 | 0.59471542 | 0.63839333 | 0. |
| 0.13834289 | 0. | 0.33581513 | 0. | 0.47763126 | 0.22934757 |
| 0.41118582 | 0.13834289 | 0. | 0.18445719 | 0. | 0.36879624 |
| 0.6977416 | 0.20751434 | 0.82339478 | 0.13834289 | 0.24455799 | 0.18445719 |
| 0. | 0.34659106 | 0. | 0.47411066 | 0. | 0.13834289 |
| 0. | 0.24347427 | 0.39053846 | 0.13834289 | 0.24455799 | 0.18445719 |
| 0.13834289 | 0. | 0.18445719 | 0.20751434 | 0. | 0. |
| 0. | 0.25180416 | 0. | 0.72351453 | 0.30617717 | 0.18445719 |
| 0. | 0.13834289 | 0.70376786 | 0.33144012 | 0. | 0.63624856 |
| 0.24455799 | 0.22825413 | 0.20751434 | 0.25689247 | 0.18445719 | 0.39286742 |
| 0. | 0.23766083 | 0.20751434 | 0.47597877 | 0. | 0.18445719 |
| 0.35673856 | 0.30617717 | 0. | 0.83414426 | 0.18445719 | 0.40307602 |

| | | | | | |
|---|---|---|---|---|---|
| 0.31948923 | 0. | 0. | 0.24455799 | 0.75578313 | 0.19484292 |
| 0.50788978 | 0.34423393 | 0.13834289 | 0. | 0. | 0. |
| 0.22934757 | 0.13834289 | 0. | 0.18445719 | 0.69171446 | 0.67102847 |
| 0.38267884 | 0.22557955 | 0. | 0.35497272 | 0.19266935 | 0. |
| 0.13834289 | 0.58082836 | 0.3000034 | 0.47779536 | 0.13834289 | 0.50488132 |
| 0.61016953 | 0.45602509 | 0.18445719 | 0.34959159 | 0.22622482 | 0.23766083 |
| 0.13834289 | 0.22622482 | 0.59191141 | 0.13834289 | 0.24486267 | 0. |
| 0.28511644 | 0.13834289 | 0. | 0.27546733 | 0. | 0.34416734 |
| 0. | 0. | 0.53145804 | 0.7214811 | 0.22825413 | 0.39822899 |
| 0. | 0.4613716 | 0.55439421 | 0. | 0.20751434 | 0. |
| 0.13834289 | 0.46840006 | 0.24846359 | 0. | 0.62555243 | 0.54018706 |
| 0.24455799 | 0.18445719 | 0.18445719 | 0.22991761 | 0. | 0.13834289 |
| 0.13834289 | 0.13834289 | 0.22825413 | 0.30625026 | 0. | 0.56455746 |
| 0.13834289 | 0.58622459 | 0.35954901 | 0.52071514 | 0. | 0. |
| 0.13834289 | 0.22825413 | 0.43784064 | 0. | 0. | 0.20751434 |
| 0.25332902 | 0. | 0.45369696 | 0. | 0. | 0.13834289 |
| 0. | 0.25824007 | 0.18445719 | 0.39789582 | 0.18445719 | 0. |
| 0. | 0.13834289 | 0.31710049 | 0. | 0.20751434 | 0.57290041 |
| 0.13834289 | 0.13834289 | 0.13834289 | 0. | 0.38162752 | 0.81285723 |
| 0.42815419 | 0.2300827 | 0.63797738 | 0.35392682 | 0.50152921 | 0.28804078 |
| 0.13834289 | 0.13834289 | 0.13834289 | 0.65784851 | 0.42535965 | 0.13834289 |
| 0.18445719 | 0.2557961 | 0. | 0.13834289 | 0. | 0.51487694 |
| 0.39686301 | 0.54698899 | 0.51016418 | 0.22134863 | 0.13834289 | 0.42877776 |
| 0.13834289 | 0. | 0.42069787 | 0. | 0.26327685 | 0.13834289 |
| 0.42978741 | 0. | 0.13834289 | 0.83317362 | 0.13834289 | 0. |
| 0. | 0.18445719 | 0.29117315 | 0.18445719 | 0.13834289 | 0. |
| 0.18445719 | 0. | 0.36726586 | 0.44527091 | 0.43088789 | 0.41789961 |
| 0. | 0.4423443 | 0. | 0.45767134 | 0. | 0.13834289 |
| 0.2253933 | 0.13834289 | 0.27716573 | 0.34585723 | 0.13834289 | 0.42172618 |
| 0.44018307 | 0.13834289 | 0.57983571 | 0. | 0. | 0.46685142 |
| 0. | 0. | 0.59857647 | 0.51488916 | 0. | 0.13834289 |
| 0.37460742 | 0.23766083 | 0.4443302 | 0.39400306 | 0.23057149 | 0. |
| 0.18445719 | 0.18445719 | 0.81009624 | 0.20751434 | 0.41046935 | 0.22825413 |
| 0. | 0. | 0. | 0.13834289 | 0.35743337 | 0. |
| 0.13834289 | 0.21911417 | 0. | 0. | 0.5018984 | 0. |
| 0.43950173 | 0.13834289 | 0.13834289 | 0.61478442 | 0.43299991 | 0. |
| 0.29097047 | 0. | 0.24455799 | 0.30197878 | 0. | 0.6123471 |
| 0.49674264 | 0.13834289 | 0.55185926 | 0.26086186 | 0. | 0.56799016 |
| 0. | 0. | 0. | 0.23518292 | 0.18445719 | 0.52396154 |
| 0. | 0. | 0.50577508 | 0.20751434 | 0.19444977 | 0.18445719 |
| 0.13834289 | 0.27268144 | 0.26086186 | 0. | 0. | 0. |
| 0. | 0.33291319 | 0.35647306 | 0.47902654 | 0.42264638 | 0.20751434 |
| 0.27367005 | 0. | 0.22825413 | 0. | 0.13834289 | 0. |
| 0.19695383 | 0.75058407 | 0.71108708 | 0.48757338 | 0.18445719 | 0.41130484 |
| 0.13834289 | 0.13834289 | 0.51858499 | 0. | 0. | 0.35429365 |
| 0. | 0.24455799 | 0.68036964 | 0.13834289 | 0. | 0. |
| 0.13834289 | 0. | 0. | 0.43018733 | 0.37715714 | 0.51372095 |
| 0.26086186 | 0. | 0.43095879 | 0.48046792 | 0. | 0.22325714 |

```
  0.13834289 0.48140573 0.18445719 0.22134863 0.29426159 0.18445719
  0.         0.61198145 0.20751434 0.18445719 0.84138527 0.30503688
  0.59331773 0.13834289 0.27207436 0.         0.         0.40268434
  0.13834289 0.43193285 0.13834289 0.75986137]]
Most similar user to User 2 (manhattan): [[108.5   0.  114.5 113.5 111.5 110.5
109.5 113.5 114.5 102.5 108.5 114.5
  111.5 111.5  93.   99.   95.   56.  109.5 111.   80.5  93.  111.5  91.
   95.  114.5 114.5  91.5 100.  103.5 114.5 113.5 108.5 109.5 114.5 109.5
  112.5 113.5 114.5 113.5  83.5 102.5 112.5 114.5 103.5 112.5 103.5 114.5
  107.5  84.  108.5  85.  114.5 112.5 111.5 109.5 107.5 110.5 114.5 108.5
  109.5  57.   82.5  98.5  92.  100.  102.   48.5 110.5 108.5 114.5 113.5
   81.5 111.5 114.5  87.5 103.  113.  112.5  86.5 111.5 111.  101.  113.5
  111.5 100.5 114.5  94.  112.  114.5  99.5 114.5 108.5 113.5 114.5 110.5
  110.5 102.5 110.5 114.5 110.5 112.5  71.5 114.5  53.5 113.5 113.5 114.5
  113.5 114.5  56.   96.5 114.5 107.  111.5 107.5 110.5 114.5  69.  114.5
  113.5  74.5  97.  113.   72.  113.5 114.5 114.5 107.5 113.5  98.   93.5
  112.5 114.5 111.5 111.5 108.5 114.5  96.5  96.   85.  114.5 111.5 110.5
  111.5 114.5 114.5 109.  114.5 114.5 114.5  93.  106.5 106.5 106.5 105.5
  114.5 114.5 103.  111.5 113.5 113.5 110.5 111.5 114.5  96.5 113.5 110.5
  107.  113.5 113.5 110.5 114.5 110.5 114.5 113.5  72.  113.5 113.5 114.5
  114.5  98.  114.5  95.  114.5 110.5 105.5 114.5  93.5 100.5 112.5 114.5
  112.  114.5 105.5 105.  112.5 106.5 100.   94.5 114.5 112.5 108.5 103.
  104.5 114.5 114.5 114.5  92.5 110.5  92.5  75.  110.  114.5 109.  110.5
  114.5 114.5  91.5 101.5 112.5  75.5  97.5 114.5 111.5  93.   74.5 112.5
  113.5 103.5 105.5  81.   72.5 114.5 112.5 114.5 104.5 114.5  91.  108.5
   97.5 112.5 114.5 111.5 114.5 101.5  65.  112.   49.5 113.5 109.5 111.
  114.5 103.5 114.5  93.5 114.5 113.5 114.5 109.   99.  111.5 109.5 112.5
  112.5 114.5 111.5 111.5 114.5 114.5 114.5 106.5 114.5  62.  104.5 111.5
  114.5 113.5  64.5 103.  114.5  75.  109.5 108.5 110.5 108.  112.   99.5
  114.5 109.5 110.5  94.  114.5 113.5 103.  106.5 114.5  50.5 111.5  97.5
  104.  114.5 114.5 108.5  63.  109.   89.5 104.5 112.5 114.5 114.5 114.5
  108.5 113.5 114.5 110.5  66.5  72.  102.  109.  114.5 101.  111.  114.5
  113.5  80.  106.   94.5 112.5  90.5  79.5  93.  110.5 102.5 110.  109.5
  113.5 110.   81.5 113.5 108.5 114.5 106.5 113.5 114.5 107.  114.5 103.5
  114.5 114.5  88.   65.5 110.5  98.  114.5  93.5  81.5 114.5 110.  114.5
  111.5  90.5 108.5 114.5  79.   85.  109.5 111.5 111.  108.  114.5 113.5
  113.5 112.5 110.5 103.5 114.5  82.  111.5  78.5 101.   90.  114.5 114.5
  113.5 108.5  94.5 114.5 114.5 112.  109.5 114.5  94.5 114.5 114.5 113.
  114.5 107.5 111.5  98.  111.5 114.5 114.5 112.5 104.  114.5 110.5  82.5
  112.5 113.5 112.5 114.5 102.   52.   96.  108.   75.5 101.5  91.5 105.5
  113.5 113.5 113.5  73.5  96.  112.  110.5 108.  114.5 113.5 114.5  87.5
   98.5  84.   89.  109.5 113.5  96.  113.5 114.5  98.  114.5 108.5 112.5
   96.  114.5 113.5  44.  113.5 114.5 114.5 110.5 107.5 111.  113.5 114.5
  111.5 114.5 101.   95.   96.5 100.  114.5  96.  114.5  91.5 114.5 111.5
  110.5 112.5 106.5 104.5 113.5  95.5  96.  113.5  82.5 114.5 114.5  92.5
  114.5 114.5  79.5  89.5 114.5 113.5 100.  109.5  96.   97.5 110.  114.5
  111.5 111.5  53.5 110.   98.5 110.5 114.5 114.5 114.5 113.5 100.5 114.5
  112.5 110.  114.5 114.5  89.  114.5  97.5 113.5 113.5  77.   98.5 114.5
```

```
 108.   114.5 109.5 105.   114.5  78.5  92.5 113.5  82.5 108.5 114.5  84.5
 114.5 114.5 114.5 109.5 110.5  88.5 114.5 114.5  90.   110.   111.5 111.5
 112.5 107.5 107.5 114.5 114.5 114.5 114.5 103.   102.5  94.5  96.5 110.5
 107.5 114.5 110.5 114.5 113.   114.5 111.5  56.5  64.    91.5 111.   97.5
 112.5 113.5  91.5 114.5 114.5 101.5 114.5 109.5  70.5 113.5 114.5 114.5
 113.5 114.5 114.5  95.5 101.    87.5 108.5 114.5  96.    92.5 114.5 109.5
 113.5  90.   111.5 109.5 107.   111.   114.5  76.   110.5 111.5  46.5 105.
  81.   113.5 107.5 114.5 114.5  97.   113.5  93.5 112.5  55. ]]
```

This output shows the similarity scores between User 2 and all other users in the dataset using cosine similarity and Manhattan distance. Each row corresponds to a user in the dataset.

#### Cosine Similarity

The values range between 0 and 1, where 1 indicates perfect similarity. For each user, the cosine similarity score indicates how similar their movie preferences are to User 2. The most similar user to User 2 has the highest cosine similarity score (closest to 1).

#### Manhattan Distance

The values represent the Manhattan distance between the feature vectors of User 2 and other users. Smaller values suggest greater similarity. The most similar user to User 2 has the lowest Manhattan distance.

### 0.10 5. Recommend movies for user 2 using cosine similarity

[19]:
```python
import numpy as np

# Exclude User 2 from the calculation
cosine_similarities[:, 1] = -1
manhattan_distances[:, 1] = np.inf

# index of the most similar user using cosine similarity
most_similar_cosine_index = np.argmax(cosine_similarities)
most_similar_cosine_score = cosine_similarities[0, most_similar_cosine_index]

print(f"Most similar user to User 2 (cosine): User {most_similar_cosine_index +␣
↪1} with similarity score {most_similar_cosine_score}")
```

```
Most similar user to User 2 (cosine): User 599 with similarity score
0.8413852739119481
```

[20]:
```python
# index of the most similar user using Manhattan distance
most_similar_manhattan_index = np.argmin(manhattan_distances)
most_similar_manhattan_distance = manhattan_distances[0,␣
↪most_similar_manhattan_index]

print(f"Most similar user to User 2 (manhattan): User␣
↪{most_similar_manhattan_index + 1} with distance␣
↪{most_similar_manhattan_distance}")
```

```
Most similar user to User 2 (manhattan): User 448 with distance 44.0
```

## 0.11 movies rated by the most similar user

[32]:
```python
user_2_recommended_movies = all_users_vector.loc[most_similar_cosine_index]

movies_not_rated_by_user_2 =␣
 ↪user_2_recommended_movies[user_2_recommended_movies == 0].index

# Extract recommended movies information
recommended_movies_info = movies[movies['movieId'].
 ↪isin(movies_not_rated_by_user_2)]
recommended_movies_info = recommended_movies_info[['movieId', 'title']].
 ↪merge(ratings[ratings['userId'] == most_similar_cosine_index + 1],␣
 ↪on='movieId')
recommended_movies_info = recommended_movies_info.rename(columns={'rating':␣
 ↪'user_rating'})

print("Movies rated by the most similar user:")
print(recommended_movies_info)
```

```
Movies rated by the most similar user:
    movieId                                              title  userId  \
0       318                   Shawshank Redemption, The (1994)     599
1       333                                  Tommy Boy (1995)     599
2      1704                          Good Will Hunting (1997)     599
3      3578                                  Gladiator (2000)     599
4      6874                          Kill Bill: Vol. 1 (2003)     599
5      8798                                 Collateral (2004)     599
6     46970  Talladega Nights: The Ballad of Ricky Bobby (2…     599
7     48516                               Departed, The (2006)     599
8     58559                            Dark Knight, The (2008)     599
9     60756                               Step Brothers (2008)     599
10    68157                        Inglourious Basterds (2009)     599
11    71535                                  Zombieland (2009)     599
12    77455                  Exit Through the Gift Shop (2010)     599
13    80489                                   Town, The (2010)     599
14    91529                      Dark Knight Rises, The (2012)     599
15    99114                           Django Unchained (2012)     599
16   106782                     Wolf of Wall Street, The (2013)     599
17   109487                                Interstellar (2014)     599
18   112552                                    Whiplash (2014)     599
19   115713                                 Ex Machina (2015)     599
20   122882                          Mad Max: Fury Road (2015)     599

    user_rating   timestamp
0           4.0  1498498867
1           2.5  1498516187
```

```
2           4.5  1498762601
3           3.5  1498501113
4           5.0  1498457174
5           3.0  1498523618
6           2.5  1498514842
7           3.0  1498522886
8           3.5  1498798185
9           2.5  1498515769
10          3.5  1498500693
11          3.0  1498524922
12          3.5  1498542480
13          3.5  1498542459
14          3.0  1498527139
15          3.5  1498528776
16          3.0  1498528478
17          3.5  1498532289
18          3.0  1498589282
19          3.5  1498528866
20          4.0  1498854698
```

### 0.12 Identify movies not yet rated by User 2

[33]:
```python
recommended_movies_info = movies[movies['movieId'].
 ↪isin(user_2_recommended_movies[user_2_recommended_movies == 0].index)]

print("Movies not yet rated by User 2:")
print(recommended_movies_info)
```

```
Movies not yet rated by User 2:
      movieId                                           title  \
277       318                Shawshank Redemption, The (1994)
291       333                                Tommy Boy (1995)
1284     1704                        Good Will Hunting (1997)
2674     3578                                Gladiator (2000)
4615     6874                        Kill Bill: Vol. 1 (2003)
5305     8798                               Collateral (2004)
6253    46970  Talladega Nights: The Ballad of Ricky Bobby (2…
6315    48516                           Departed, The (2006)
6710    58559                         Dark Knight, The (2008)
6801    60756                            Step Brothers (2008)
7010    68157                     Inglourious Basterds (2009)
7154    71535                               Zombieland (2009)
7258    74458                           Shutter Island (2010)
7323    77455                Exit Through the Gift Shop (2010)
7415    80489                                Town, The (2010)
7436    80906                               Inside Job (2010)
7590    86345                      Louis C.K.: Hilarious (2010)
7697    89774                                  Warrior (2011)
```

```
7768    91529                          Dark Knight Rises, The (2012)
7776    91658             Girl with the Dragon Tattoo, The (2011)
8063    99114                               Django Unchained (2012)
8305   106782                        Wolf of Wall Street, The (2013)
8376   109487                                    Interstellar (2014)
8466   112552                                        Whiplash (2014)
8509   114060                                        The Drop (2014)
8550   115713                                      Ex Machina (2015)
8681   122882                              Mad Max: Fury Road (2015)
8828   131724  The Jinx: The Life and Deaths of Robert Durst …

                                genres
277                        Crime|Drama
291                             Comedy
1284                     Drama|Romance
2674            Action|Adventure|Drama
4615             Action|Crime|Thriller
5305       Action|Crime|Drama|Thriller
6253                     Action|Comedy
6315              Crime|Drama|Thriller
6710           Action|Crime|Drama|IMAX
6801                            Comedy
7010                 Action|Drama|War
7154              Action|Comedy|Horror
7258            Drama|Mystery|Thriller
7323                Comedy|Documentary
7415              Crime|Drama|Thriller
7436                       Documentary
7590                            Comedy
7697                             Drama
7768       Action|Adventure|Crime|IMAX
7776                    Drama|Thriller
8063              Action|Drama|Western
8305                Comedy|Crime|Drama
8376                       Sci-Fi|IMAX
8466                             Drama
8509              Crime|Drama|Thriller
8550             Drama|Sci-Fi|Thriller
8681  Action|Adventure|Sci-Fi|Thriller
8828                       Documentary
```

This represents movies that User 2 has not yet rated, and they are recommended based on the preferences of the most similar user. The movies are presented in a DataFrame with columns, `movieId: The unique identifier for each movie.` `title: The title of the movie.` `genres: The genre or genres associated with the movie.`

How we achieved this

Calculate Similarity Scores - Cosine similarity scores are calculated between User 2 and all other users. The user with the highest similarity score (excluding User 2 itself) is identified.

Retrieve Recommended Movies - We then retrieve the movies that were rated by the most similar user but not yet rated by User 2. This is achieved by comparing the movie ratings of User 2 with the most similar user and selecting the movies where User 2 has a rating of 0 (indicating that the movie has not been rated).

Get Movie Details - Then we get the details (movieId, title, genres) for each recommended movie from the 'movies' dataset.

### 0.13 Retrieve and print recommended movies for User 2

[35]:
```
# Get movies rated by the most similar user but not yet rated by User 2
movies_not_rated_by_user_2 =␣
↪user_2_recommended_movies[user_2_recommended_movies == 0].index

# Extract recommended movies information
recommended_movies_info = movies[movies['movieId'].
↪isin(movies_not_rated_by_user_2)]

# Merge with ratings of the most similar user
recommended_movies_info = recommended_movies_info.merge(
    ratings[ratings['userId'] == most_similar_cosine_index + 1],
    on='movieId',
    how='left'
)

# Filter only movies that are highly rated by the most similar user
threshold_rating = 3.5
recommended_movies_info =␣
↪recommended_movies_info[recommended_movies_info['rating'] >=␣
↪threshold_rating]

print("Recommended movies for User 2:")
print(recommended_movies_info[['movieId', 'title', 'genres', 'rating']])
```

```
Recommended movies for User 2:
    movieId                               title  \
0       318    Shawshank Redemption, The (1994)
2      1704            Good Will Hunting (1997)
3      3578                    Gladiator (2000)
4      6874            Kill Bill: Vol. 1 (2003)
8     58559             Dark Knight, The (2008)
10    68157         Inglourious Basterds (2009)
13    77455  Exit Through the Gift Shop (2010)
14    80489                    Town, The (2010)
20    99114             Django Unchained (2012)
22   109487                 Interstellar (2014)
25   115713                   Ex Machina (2015)
```

```
26   122882          Mad Max: Fury Road (2015)

                                genres  rating
0                          Crime|Drama     4.0
2                        Drama|Romance     4.5
3             Action|Adventure|Drama     3.5
4              Action|Crime|Thriller     5.0
8            Action|Crime|Drama|IMAX     3.5
10                  Action|Drama|War     3.5
13                Comedy|Documentary     3.5
14              Crime|Drama|Thriller     3.5
20              Action|Drama|Western     3.5
22                       Sci-Fi|IMAX     3.5
25             Drama|Sci-Fi|Thriller     3.5
26  Action|Adventure|Sci-Fi|Thriller     4.0
```

The recommendations are providing a more reasonable list of movies for User 2. Each recommended movie has a rating equal to or above the threshold of 3.5, ensuring they are highly rated by the most similar user. This approach allows for a more personalized recommendation based on the preferences of similar users.

The recommendations from this method make sense since there is a high cosine similarity with the most similar user. The method identifies movies highly rated by the most similar user that User 2 has not yet watched. The assumption is that users with similar movie preferences will continue to have similar preferences, and the threshold of 3.5 indicates a preference for well-rated movies. The final output provides movies, along with their details and ratings by the most similar user.

## 0.14 Analysis

In this collaborative filtering recommendation analysis using cosine similarity, we've employed a metric that effectively measures user similarity by considering both the direction and magnitude of preference vectors. Cosine similarity, a commonly used metric in recommendation systems, proves suitable for our purpose of gauging user similarity based on movie ratings.

Upon scrutinizing the recommended movies, it is evident that they resonate with the presumed preferences of User 2. This alignment is substantiated by the high ratings, predominantly falling in the range of 3.5 and above, given by the most similar user. The fundamental idea behind this method is to propose movies that have been well-received by the similar user while excluding those that User 2 has already watched.