Homework #6

Problem 9.2.1)

PartA)

For A and B

Dot product = 1*(3.06*2.68) + a^2*(500*320) + b^2*(6*4)

= 8.2008 + 160000*a^2 + 24*b^2

Length of A = sqrt((1*3.06)^2 + (500*a)^2 + (6*b)

= sqrt(9.3636 + 250000a^2 + 36b^2)

Length of B = sqrt(7.1824 + 102400a^2 + 16b^2)

Cosine Similarity = (|a|*|b|)/(magnitude(a)* magnitude(b))

Cosine A&B = 8.2008 + 160000*a^2 + 24*b^2/ sqrt(9.3636 + 250000*a^2 + 36*b^2)* sqrt(7.1824 + 102400*a^2 + 16*b^2)

Similarly:

Cosine B&C = 7.8256 + 204800*a^2 + 24*b^2/ sqrt(7.1824+102400*a^2+16*b^2)*sqrt(8.5264+409600*a^2+36*b^2)

Cosine A&C = 8.9352+320000*a^2+36*b^2/ sqrt(9.3636+250000*a^2+36*b^2) * sqrt(8.5264+409600*a^2+36*b^2)

Part B) If a = b = 1

Cos(A,B) = (8.2008+160000+24)/sqrt(9.3636+250000+36)*sqrt(7.1824+102400+16)

Cos(A,B) = 0.99999733

Angle(A,B) = cos^(-1)(0. 99999733) = 0.132 (in degree)

Similarly:

Cos(B,C) = 0.99998785

Angle(B,C) = 0.282 (in degree)

Cos(A,C) = 0.99999534

Angle(A,C) = 0.174 (in degree)

Part C)

If a = 0.01, b = 0.5

Cos(A,B) = 0.9908

Angle(A,B) = 7.74 (in degree)

Cos(B,C) = 0.9691

Angle(B,C) = 14.26 (in degree)

Cos(A,C) = 0.9915

Angle(A,C) = 7.45 (in degree)

Part D)

a = 1/{(500+320+640)/3} = 0.00205

b = 1/{(6+4+6)/3} = 0.1875

Cos(A,B) = 0.99883

Angle(A,B) = 2.77 (in degree)

Cos(B,C) = 0.99477

Angle(B,C) = 5.86 (in degree)

Cos(A,C) = 0.99913

Angle(A,C) = 2.39 (in degree)

Problem 9.2.3)

Part A)

Avg = (4+2+5)/3 = 11/3

A = 4-11/3 = 1/3

B = 2-11/3 = -5/3

C = 5-11/3 = 4/3

Part B)

Processor Speed:

= 3.06*(1/3) + 2.68*(-5/3) + 2.92*(4/3)

= 0.4467

Disk Size:

= 500*1/3 + 320*(-5/3) + 640*(4/3)

= 486.6667

Main Memory Size:

= 6*(-1/3) + 4*(-5/3) + 6*4/3

= 3.333

Problem 9.3.1)

Part A)

Jaccard(A,B) = 4/8 = 1/2 => 1- 1/2 = 1/2

Jaccard(A,C) = 4/8 = 1/2 => 1- 1/2 = 1/2

Jaccard(B,C) = 4/8 = 1/2 => 1- 1/2 = 1/2

Part B)

Cos(A,B) = (5*3 + 5*3 + 1*1 + 3*1)/(sqrt(4^2 + 5^2 + 5^2 + 1^2 + 3^2 + 2^2)*sqrt(3^2 + 4^2 + 3^2 + 1^2 + 2^2 + 1^2))

Cos(A,B) = 0.601

Cos(A,C) = 0.615

Cos(B,C) = 0.435

Part C)

J(A,B) = 2/5

J(B,C) = 1/6

J(C,A) = 2/6

Part D)

Cos(A,B) = (1+1)/(sqrt(1^2+1^2+1^2+1^2)*sqrt(1^2+1^2+1^2))

Cos(A,B) = 0.5773

Cos(B,C) = 0.2886

Cos(A,C) = 0.5

Part E)

Avg A = 20/6 = 10/3

Avg B = 14/6 = 7/3

Avg C = 18/6 = 3

	a	b	c	d	e	f	g	h
A	2/3	5/3		5/3	-7/3		-1/3	-4/3
B		2/3	5/3	2/3	-4/3	-1/3	-4/3	
C	-1		-2	0		1	2	0

Part F)

Cos(A,B) = 0.584

Cos(B,C) = 0.739

Cos(A,C) = -0.1154

Problem 9.4.1)

5	2	4	4	3
3	1	2	4	1
2		3	1	4
2	5	4	3	5
4	4	5	4	

Part A) U-32

1	1
1	1
1	X
1	1
1	1

2	2	2	2	2
2	2	2	2	2
X+1	X+1	X+1	X+1	X+1
2	2	2	2	2
2	2	2	2	2

(x-1)^2 + (x-2)^2 + x^2 + (x-3)^2 = 0

d/dx((x-1)^2 + (x-2)^2 + x^2 + (x-3)^2) = 0

2(x-1) + 2(x-2) + 2x + 2(x-3) =

4x-6=0

X = 1.5

2	2	2	2	2
2	2	2	2	2
2.5	2.5	2.5	2.5	2.5
2	2	2	2	2
2	2	2	2	2

Part B) V-41

1	1	1	X	1
1	1	1	1	1

1	1
1	1
1	1
1	1
1	1

2	2	2	Y+1	2
2	2	2	Y+1	2
2	2	2	Y+1	2
2	2	2	Y+1	2
2	2	2	Y+1	2

(y-3)^2 + (y-3)^2 + y^2 + (y-2)^2 + (y-3)^2

d/dy((y-3)^2 + (y-3)^2 + y^2 + (y-2)^2 + (y-3)^2) = 0

2(y-3) + 2(y-3) + 2y + 2(y-2) + 2(y-3) =

5y – 11 = 0

Y = 2.2

2	2	2	3.2	2
2	2	2	3.2	2
2	2	2	3.2	2
2	2	2	3.2	2
2	2	2	3.2	2

Problem 1

In [2]:
```
import numpy as np
import pandas as pd
from sklearn import metrics
```

In [3]:
```
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

In [4]:
```
from google.colab import files
uploaded = files.upload()
```

Choose Files No file chosen

Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.

```
Saving ml-100k.zip to ml-100k.zip
```

In [5]:
```
!unzip -q "ml-100k.zip"
```

In [6]:
```
user_cols = ['user_id',
             'age',
             'gender',
             'occupation',
             'zip_code']

users = pd.read_csv('ml-100k/u.user',
                    sep='|',
                    names=user_cols)
```

In [7]:
```
users.head()
```

Out[7]:

	user_id	age	gender	occupation	zip_code
0	1	24	M	technician	85711
1	2	53	F	other	94043
2	3	23	M	writer	32067
3	4	24	M	technician	43537
4	5	33	F	other	15213

In [8]:
```
rating_cols = ['user_id',
               'movie_id',
               'rating',
               'timestamp']

ratings = pd.read_csv('ml-100k/u.data',
                      sep='\t',
                      names=rating_cols)
```

In [9]:
```
ratings.head()
```

Out[9]:

	user_id	movie_id	rating	timestamp
0	196	242	3	881250949
1	186	302	3	891717742
2	22	377	1	878887116
3	244	51	2	880606923
4	166	346	1	886397596

In [10]:
```
item_cols = ['movie id',
             'movie title',
             'release date',
             'video release date',
             'IMDb URL',
             'Unknown',
             'Action',
             'Adventure',
             'Animation',
             'Childrens',
             'Comedy',
             'Crime',
             'Documentary',
             'Drama',
             'Fantasy',
             'FilmNoir',
             'Horror',
             'Musical',
             'Mystery',
             'Romance',
             'SciFi',
             'Thriller',
             'War',
             'Western']

items = pd.read_csv('ml-100k/u.item',
                    sep='|',
                    names=item_cols,
                    encoding='latin-1')
```

In [11]:
```
items.head(3)
```

Out[11]:

	movie id	movie title	release date	video release date	IMDb URL	Unknown	Action	Adven
0	1	Toy Story (1995)	01-Jan-1995	NaN	http://us.imdb.com/M/title-exact?Toy%20Story%2...	0	0	0
1	2	GoldenEye (1995)	01-Jan-1995	NaN	http://us.imdb.com/M/title-exact?GoldenEye%20(...	0	1	1
2	3	Four Rooms (1995)	01-Jan-1995	NaN	http://us.imdb.com/M/title-exact?Four%20Rooms%...	0	0	0

In [12]:
```
items95 = items.iloc[94:95]
items95.head()
```

Out[12]:

	movie id	movie title	release date	video release date	IMDb URL	Unknown	Action	Adventu
94	95	Aladdin (1992)	01-Jan-1992	NaN	http://us.imdb.com/M/title-exact?Aladdin%20(1992)	0	0	0

In [13]:
```
users15 = users.iloc[14:15]

print(users15.head())

users200 = users.iloc[199:200]

print(users200.head())
```

```
    user_id  age gender occupation zip_code
14       15   49      F   educator    97301
     user_id  age gender  occupation zip_code
199      200   40      M  programmer    93402
```

In [14]:
```
utility = ratings.pivot(index= "user_id",
              columns = "movie_id",
              values = "rating")

utility.head()
utility.iloc[194:197, 240:243]
```

Out[14]:

movie_id	241	242	243
user_id			
195	NaN	4.0	NaN
196	NaN	3.0	NaN
197	3.0	NaN	NaN

In [15]:
```
user_means = utility.mean(axis=1)

utility_centered = utility - user_means

utility_centered = utility_centered.where((pd.notnull(utility_centered)),0)
utility_centered.head()
```

Out[15]:

	1	2	3	4	5	6	7	8
user_id								
1	1.389706	-0.709677	1.203704	-1.333333	0.125714	1.364929	0.034739	-2.79661
2	0.389706	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.00000
3	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.00000
4	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.00000
5	0.389706	-0.709677	0.000000	0.000000	0.000000	0.000000	0.000000	0.00000

5 rows × 1682 columns

In [16]:
```
user200 = utility_centered.iloc[199:200]
user200.head()
```

Out[16]:

	1	2	3	4	5	6	7	8	9	10	11
user_id											
200	1.389706	0.290323	0.0	0.0	0.0	0.0	0.034739	0.20339	-0.272727	0.0	1.535912

1 rows × 1682 columns

In [17]:
```
user15 = utility_centered.iloc[14:15]
user15.head()
```

Out[17]:

	1	2	3	4	5	6	7	8	9	10	11	12	13
user_id													
15	-2.610294	0.0	0.0	0.0	0.0	0.0	-2.965261	0.0	-0.272727	0.0	0.0	0.0	-2.097484

1 rows × 1682 columns

In [19]:
```
cos_Similarity_200 = metrics.pairwise.cosine_similarity(user200.iloc[:,5:24],
items95.iloc[:,5:24])
cos_Distance_200 = metrics.pairwise.cosine_distances(user200.iloc[:,5:24], ite
ms95.iloc[:,5:24])
cos_Similarity_15 = metrics.pairwise.cosine_similarity(user15.iloc[:,5:24], it
ems95.iloc[:,5:24])
cos_Distance_15 = metrics.pairwise.cosine_distances(user15.iloc[:,5:24], items
95.iloc[:,5:24])


print("The Cosine similarity of user200 is",cos_Similarity_200)
print("The Cosine distance of user200 is",cos_Distance_200)
print("The Cosine similarity of user15 is",cos_Similarity_15)
print("The Cosine distance of user200 is",cos_Distance_15)
```

```
The Cosine similarity of user200 is [[0.20426227]]
The Cosine distance of user200 is [[0.79573773]]
The Cosine similarity of user15 is [[-0.32985583]]
The Cosine distance of user200 is [[1.32985583]]
```

Now we can see that we have user_200 with higher similarity 0.20426 as compare to user_15 (-0.329). So the user_200 has the highest similarity and lower distance, our recommendation model recommends these movies to **user_200**.

Problem 2

In [26]:
```
import heapq
```

In [27]:
```
new_dict = dict()
cosine_similarity_list = []
similar_users_list = []
```

In [28]:
```
user_1 = utility_centered[0:1]

for index, row in utility_centered.iterrows():
  numpy_row = np.array(row)
  numpy_row.resize(1,1682)

  if index !=1:
    list = metrics.pairwise.cosine_similarity(user_1,numpy_row)
    cosine_similarity_list.append(list)
    new_dict.__setitem__(index, list)

similar_users_top_ten = heapq.nlargest(10,new_dict,key=new_dict.get)
user_columns = ['user id','508']

data = 0

for i in similar_users_top_ten:

  print("User :", i);
  print("Cosine Similarity :", new_dict.get(i));
  similar_users_list.append(utility_centered.loc[i, 508])
```

```
User : 738
Cosine Similarity : [[0.29148679]]
User : 592
Cosine Similarity : [[0.27840172]]
User : 276
Cosine Similarity : [[0.26815054]]
User : 267
Cosine Similarity : [[0.26476147]]
User : 643
Cosine Similarity : [[0.2640026]]
User : 757
Cosine Similarity : [[0.26236785]]
User : 457
Cosine Similarity : [[0.26233704]]
User : 606
Cosine Similarity : [[0.26084701]]
User : 916
Cosine Similarity : [[0.25562438]]
User : 44
Cosine Similarity : [[0.2529544]]
```

In [29]:
```
numpy_array_similar_user = np.array(similar_users_list)
```

In [30]:
```
print("Expected rating of user_1 for item 508 : ", numpy_array_similar_user.me
an())
```

```
Expected rating of user_1 for item 508 :  0.26896551724137935
```

Here we have the cosine similarity of user_1 using user ratings data and find top 10 most similar users for user_1 based on it, using metrics cosine similarity() function and store it in list. We find the top 10 similar user_1 and item_508 based on their cosine similarity as items: 738, 592, 276, 267, 643, 757, 457, 606, 916, 44. From this list of similar users we get the expected rating of user-1 for item-508 as 0.2689.