INTRODUCTION TO WEB SCIENCES:
Assignment 8

Babitha Bokka

15 November 2014

Contents

1 Question 1:

What 5 movies have the highest average ratings? Show the movies and their ratings sorted by their average ratings.

1.1 Approach

1. To solve this question the straight forward approach is to read u.data file get movie id, corresponding movie ratings and read u.item file to get movie id and corresponding movie name.
2. Extract the movie id and ratings. For each movie id build a dictionary that has all the ratings.
3. Now, for each each movie id calculate the average a using sum() and len() functions.
4. For each movie id get the movie name and append it to the dictionary which has average.
5. Now, sort the dictionary and print the top 5 movies with highest average ratings
6. Program averageMovies.py produces the top 5 movies with highest average ratings.
7. Figure 1 is the output showing the results of the program are limited to top 5 it can get more average ratings by changing a conditional statement.

1.2 Source Code

1.2.1 averageMovies.py

```python
#!/usr/bin/env python

import sys

def main():
    # Create a dictionary
    movie_id_ratings    = {}
    movie_id_avg        = {}
    count = 1
    # Read the input files.
    readData = open('/home/bbokka/cs594/A8/u.data', 'r')
    readItem = open('/home/bbokka/cs594/A8/u.item','r')
    # reading the u.data file for item and ratings.
    for line in readData:
        split_input_line = line.strip().split('\t')
        movie_id         =  split_input_line[1]
        movie_rating     =  float(split_input_line[2])
        try:
            movie_id_ratings[movie_id].append(movie_rating)
        except KeyError:
            movie_id_ratings[movie_id] = list()
            movie_id_ratings[movie_id].append(movie_rating)
    readData.close()
    # Calculating the average.
    for key in movie_id_ratings:
        avg = sum(movie_id_ratings[key])/len(movie_id_ratings[key])
        movie_id_avg[key] = [float(avg)]

    # Reading the u.item file for movie name.
    for each_line in readItem:
        split_each_line  = each_line.strip().split('|')
        movie_item_id    = split_each_line[0]
        movie_name_split = split_each_line[1].split('(1')
        movie_name       = movie_name_split[0]

        movie_id_avg[movie_item_id].append(movie_name)
    readItem.close()
    print "*" * 60
    print "Top 5 movies having the highest average ratings"
    print "*" * 60
    print "Movie Name\t\t\t\t\t","Avg Rating"
    print "-" * 60
    # sorting the movies from highest to lowest based on the average value.
    for key, value in sorted(movie_id_avg.items(), key=lambda e: e[1], reverse=True):
        if (count <= 5):
            print '{:<50}{:<0.1f} '  .format(value[1],value[0])
            count += 1

if __name__ == "__main__":
    try:
        main()
    except KeyboardInterrupt:
        sys.exit(1)
```

1.3 Input Files

1.3.1 u.data

```
userid itemid rating timestamp

196      242      3    881250949
186      302      3    891717742
22       377      1    878887116
244      51       2    880606923
166      346      1    886397596
298      474      4    884182806
115      265      2    881171488
```

1.3.2 u.item

```
movie id | movie title | release date | video release date | IMDb URL | unknown |
    Action | Adventure | Animation |Children's | Comedy | Crime | Documentary | Drama
    | Fantasy | Film-Noir | Horror | Musical | Mystery | Romance | Sci-Fi | Thriller |
     War | Western |

1|Toy Story (1995)|01-Jan-1995||http://us.imdb.com/M/title-exact?Toy%20Story%20(1995)
    |0|0|0|1|1|1|0|0|0|0|0|0|0|0|0|0|0|0|0
2|GoldenEye (1995)|01-Jan-1995||http://us.imdb.com/M/title-exact?GoldenEye%20(1995)
    |0|1|1|0|0|0|0|0|0|0|0|0|0|0|0|1|0|0
3|Four Rooms (1995)|01-Jan-1995||http://us.imdb.com/M/title-exact?Four%20Rooms
    %20(1995)|0|0|0|0|0|0|0|0|0|0|0|0|0|0|0|1|0|0
4|Get Shorty (1995)|01-Jan-1995||http://us.imdb.com/M/title-exact?Get%20Shorty
    %20(1995)|0|1|0|0|0|1|0|0|1|0|0|0|0|0|0|0|0|0
5|Copycat (1995)|01-Jan-1995||http://us.imdb.com/M/title-exact?Copycat%20(1995)
    |0|0|0|0|0|0|1|0|1|0|0|0|0|0|0|0|1|0|0
6|Shanghai Triad (Yao a yao yao dao waipo qiao) (1995)|01-Jan-1995||http://us.imdb.com
    /Title?Yao+a+yao+yao+dao+waipo+qiao+(1995)|0|0|0|0|0|0|0|0|1|0|0|0|0|0|0|0|0|0
7|Twelve Monkeys (1995)|01-Jan-1995||http://us.imdb.com/M/title-exact?Twelve%20Monkeys
    %20(1995)|0|0|0|0|0|0|0|0|1|0|0|0|0|0|0|1|0|0|0
```

1.4 Output Files

1.4.1 averageMovies.png

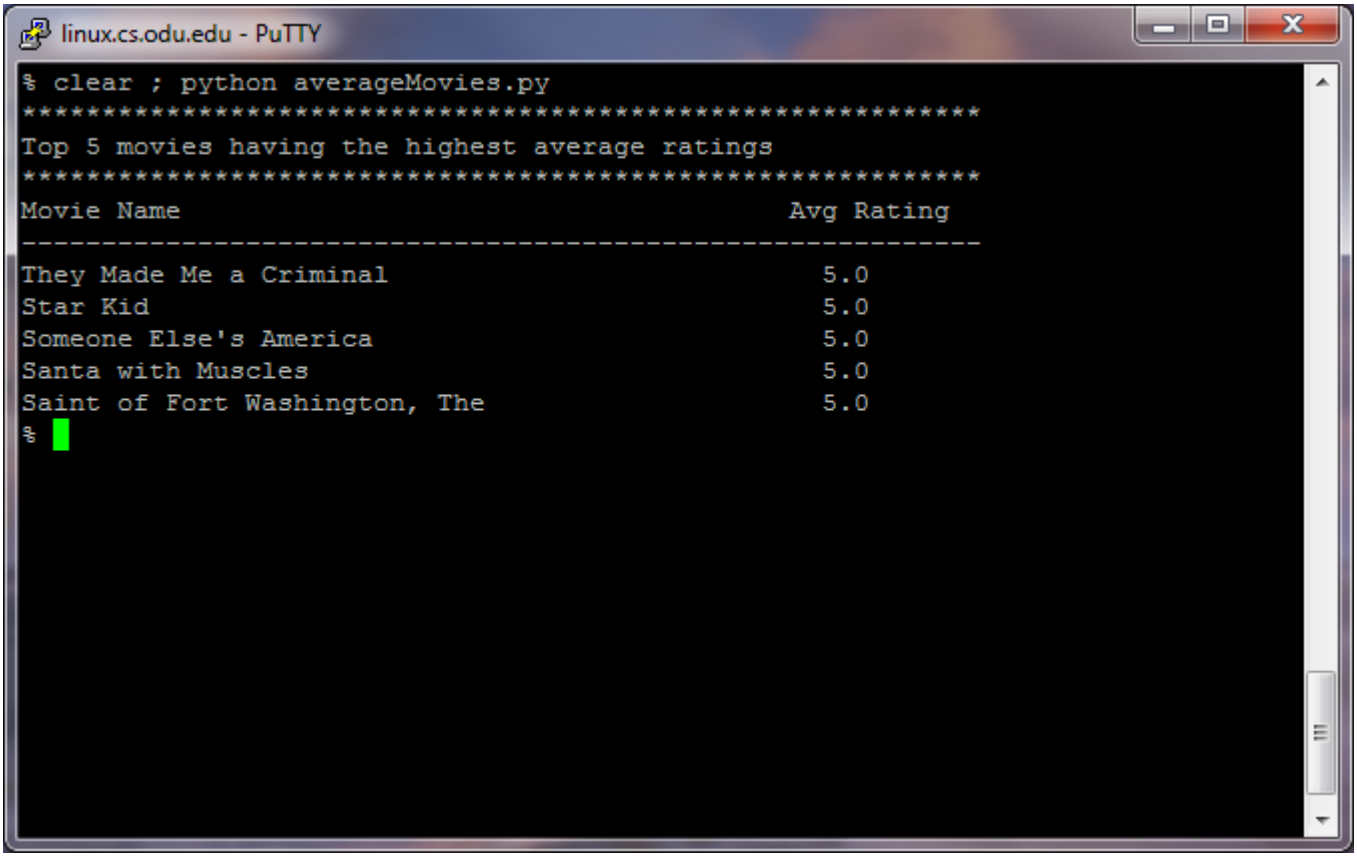


Figure 1: Top 5 movies with highest average ratings.

2 Question 2:

What 5 movies received the most ratings? Show the movies and the number of ratings sorted by number of ratings.

2.1 Approach

1. The approach is same as question 1.
2. The only difference is we count the number of ratings for each movie and display the corresponding movie name and number of ratings for each movie.
3. The output displays only top 5 movies and their number of ratings.
4. Figure 2 is the output for the program.

2.2 Source Code

2.2.1 mostRatingMovies.py

```python
#!/usr/bin/env python

import sys

def main():
    # Create a dictionary
    movie_id_ratings    = {}
    movie_id_avg        = {}
    count = 1
    # Read the input files.
    readData = open('/home/bbokka/cs594/A8/u.data', 'r')
    readItem = open('/home/bbokka/cs594/A8/u.item','r')
    # reading the u.data file for item and ratings.
    for line in readData:
        split_input_line = line.strip().split('\t')
        movie_id         =  split_input_line[1]
        movie_rating     =  int(split_input_line[2])
        try:
            movie_id_ratings[movie_id].append(movie_rating)
        except KeyError:
            movie_id_ratings[movie_id] = list()
            movie_id_ratings[movie_id].append(movie_rating)
    readData.close()
    # Calculating the average.
    for key in movie_id_ratings:
        length = len(movie_id_ratings[key])
        movie_id_avg[key] = [int(length)]

    # Reading the u.item file for movie name.
    for each_line in readItem:
        split_each_line  = each_line.strip().split('|')
        movie_item_id    = split_each_line[0]
        movie_name_split = split_each_line[1].split('(1')
        movie_name       = movie_name_split[0]

        movie_id_avg[movie_item_id].append(movie_name)
    readItem.close()
    print '*' * 60
    print "Top 5 movies received the most ratings"
    print "*" * 60
    print "Movie Name\t\t\t\t\t","Number of ratings"
    print "-" * 60
    # sorting the movies from highest to lowest based on the average value.
    for key, value in sorted(movie_id_avg.items(), key=lambda e: e[1], reverse=True):
        if (count <= 5):
            print '{:<50}{} '  .format(value[1],value[0])
            count += 1

if __name__ == "__main__":
    try:
        main()
    except KeyboardInterrupt:
        sys.exit(1)
```

2.3 Input Files

2.3.1 u.data

```
userid itemid rating timestamp

196      242      3    881250949
186      302      3    891717742
22       377      1    878887116
244      51       2    880606923
166      346      1    886397596
298      474      4    884182806
115      265      2    881171488
```

2.3.2 u.item

```
movie id | movie title | release date | video release date | IMDb URL | unknown |
    Action | Adventure | Animation |Children's | Comedy | Crime | Documentary | Drama
    | Fantasy | Film-Noir | Horror | Musical | Mystery | Romance | Sci-Fi | Thriller |
     War | Western |

1|Toy Story (1995)|01-Jan-1995||http://us.imdb.com/M/title-exact?Toy%20Story%20(1995)
    |0|0|0|1|1|1|0|0|0|0|0|0|0|0|0|0|0|0
2|GoldenEye (1995)|01-Jan-1995||http://us.imdb.com/M/title-exact?GoldenEye%20(1995)
    |0|1|1|0|0|0|0|0|0|0|0|0|0|0|0|1|0|0
3|Four Rooms (1995)|01-Jan-1995||http://us.imdb.com/M/title-exact?Four%20Rooms
    %20(1995)|0|0|0|0|0|0|0|0|0|0|0|0|0|0|0|1|0|0
4|Get Shorty (1995)|01-Jan-1995||http://us.imdb.com/M/title-exact?Get%20Shorty
    %20(1995)|0|1|0|0|0|1|0|0|1|0|0|0|0|0|0|0|0|0
5|Copycat (1995)|01-Jan-1995||http://us.imdb.com/M/title-exact?Copycat%20(1995)
    |0|0|0|0|0|0|1|0|1|0|0|0|0|0|0|0|1|0|0
6|Shanghai Triad (Yao a yao yao dao waipo qiao) (1995)|01-Jan-1995||http://us.imdb.com
    /Title?Yao+a+yao+yao+dao+waipo+qiao+(1995)|0|0|0|0|0|0|0|0|1|0|0|0|0|0|0|0|0|0
7|Twelve Monkeys (1995)|01-Jan-1995||http://us.imdb.com/M/title-exact?Twelve%20Monkeys
    %20(1995)|0|0|0|0|0|0|0|0|1|0|0|0|0|0|0|1|0|0|0
```

2.4 Output Files

2.4.1 mostRatingMovies.png

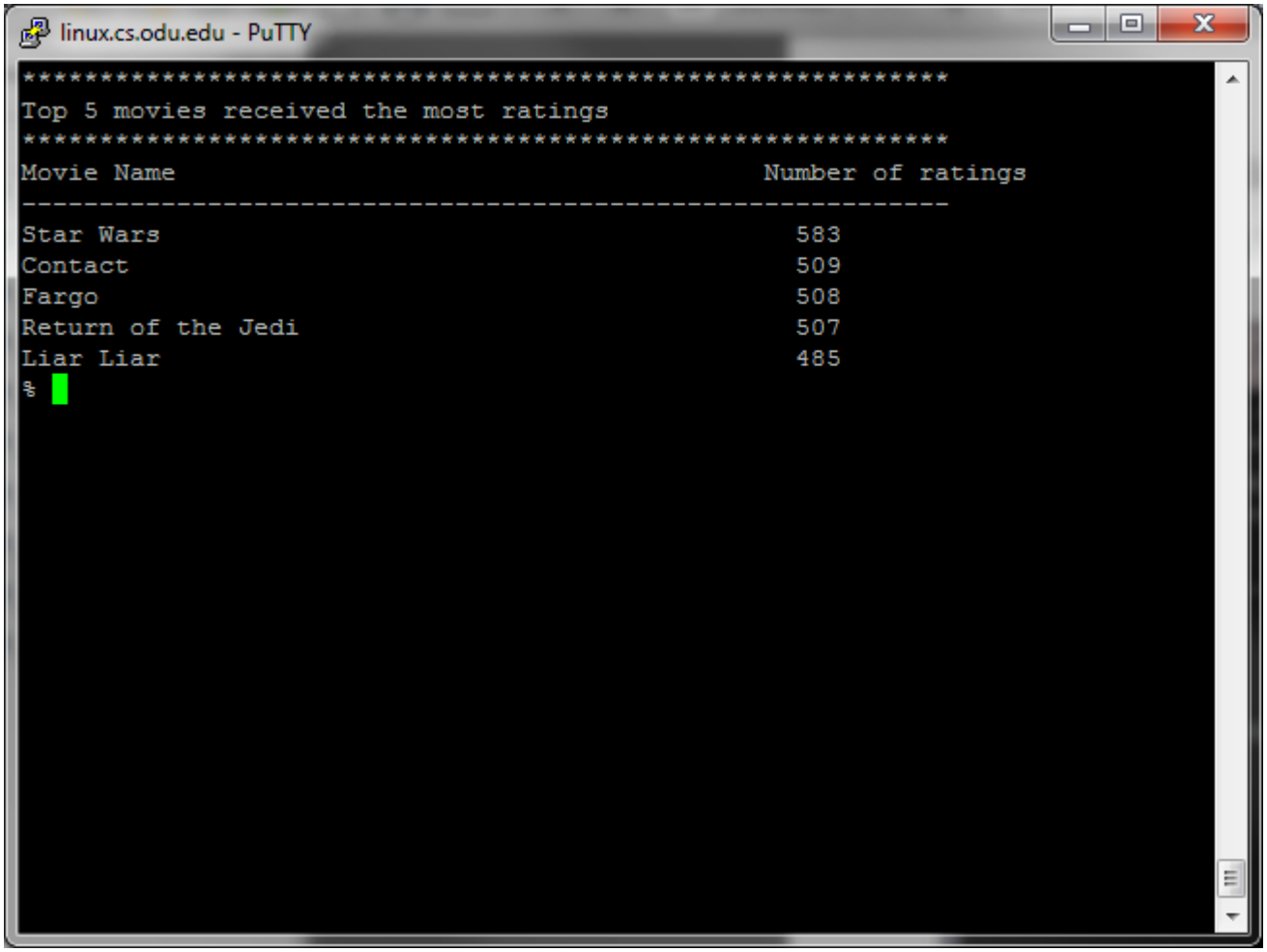


Figure 2: Top 5 movies received the most ratings.

3 Question 3:

What 5 movies were rated the highest on average by women? Show the movies and their ratings sorted by ratings.

3.1 Approach

1. To solve this question the straight forward approach is to read u.user file get the user id, gender. Read u.data file get movie id, corresponding movie ratings. Read u.item file to get movie id and corresponding movie name.

2. From u.user file extract the user id and gender store them in a dictionary.

3. Now, Extract the movie id and ratings. For each user id build a dictionary that has all the ratings for the respective gender.

4. Now, for each each movie id calculate the average a using sum() and len() functions.

5. For each movie id get the movie name and append it to the dictionary which has average.

6. Now, sort the dictionary and print the top 5 movies rated the highest on average by women.

7. Program averageGender.py is designed in such a that it can get the highest averages by men and women just by changing the arguments to F or M while executing the program.

8. Figure 3 is the output showing top 5 movies rated the highest on average by women the results of the program are limited to top 5 it can be modified to get more average ratings.

3.2 Source Code

3.2.1 averageMovies.py

```
#!/usr/bin/env python


# program to get the highest average ratings by men and women
import sys
import operator
def main():
    # take the arguments from the command line
    numOfArgs=len(sys.argv)
    if numOfArgs<2 or numOfArgs>2:

        print'Usage: averageGender.py <F or M> '
        print'e.g. : averageGender.py F'
        sys.exit(1)
    gender           = sys.argv[1]
    # Create a dictionary
    movie_userid_avg = {}
    movie_id_ratings = {}
    movie_userid_avg = {}
    users            = {}
    count            = 1

    readData = open('/home/bbokka/cs594/A8/u.data', 'r')
    readItem = open('/home/bbokka/cs594/A8/u.item','r')
    readUser = open('/home/bbokka/cs594/A8/u.user','r')

    # Reading u.user file to get the male or female
    for line in readUser:
        split_input_line = line.strip().split('|')
        user_id_from_user= split_input_line[0]
        user_gender      = split_input_line[2]

        if(gender.upper() == user_gender):
            users[user_id_from_user] = [user_gender]

    readUser.close()

    # reading the u.data file for item and ratings.
    for line in readData:
        split_input_line  = line.strip().split('\t')
        user_id_from_data = split_input_line[0]
        movie_id_from_data=  split_input_line[1]
        movie_rating      =  float(split_input_line[2])

        if user_id_from_data in users:
            try:
                movie_id_ratings[movie_id_from_data].append(movie_rating)
            except KeyError:
                movie_id_ratings[movie_id_from_data] = list()
                movie_id_ratings[movie_id_from_data].append(movie_rating)
    readData.close()

    # Calculating the average.
```

```
    for key in movie_id_ratings:
        avg = sum(movie_id_ratings[key])/len(movie_id_ratings[key])
        movie_userid_avg[key] = [float(avg)]

    # Reading the u.item file for movie name.
    for each_line in readItem:
        split_each_line  = each_line.strip().split('|')
        movie_item_id    = split_each_line[0]
        movie_name_split = split_each_line[1].split('(1')
        movie_name       = movie_name_split[0]

        try:
            movie_userid_avg[movie_item_id].append(movie_name)
        except KeyError:
            pass
    readItem.close()
    print "*" * 60
    print "Top 5 movies rated the highest on average by women"
    print "*" * 60
    print "Movie Name\t\t\t\t\t","Avg Rating"
    print "-" * 60
    # sorting the movies from highest to lowest based on the average value.
    for key, value in sorted(movie_userid_avg.items(), key=lambda e: e[1], reverse=True):
        if (count <=5 ):
            print '{:<50}{:<0.1f} '  .format(value[1],value[0])
            count += 1

if __name__ == "__main__":
    try:
        main()
    except KeyboardInterrupt:
        sys.exit(1)
```

3.3 Input Files

3.3.1 u.data

```
userid itemid rating timestamp

196      242      3    881250949
186      302      3    891717742
22       377      1    878887116
244      51       2    880606923
166      346      1    886397596
298      474      4    884182806
115      265      2    881171488
```

3.3.2 u.item

```
movie id | movie title | release date | video release date | IMDb URL | unknown |
    Action | Adventure | Animation |Children's | Comedy | Crime | Documentary | Drama
    | Fantasy | Film-Noir | Horror | Musical | Mystery | Romance | Sci-Fi | Thriller |
     War | Western |

1|Toy Story (1995)|01-Jan-1995||http://us.imdb.com/M/title-exact?Toy%20Story%20(1995)
    |0|0|0|1|1|1|0|0|0|0|0|0|0|0|0|0|0|0|0
2|GoldenEye (1995)|01-Jan-1995||http://us.imdb.com/M/title-exact?GoldenEye%20(1995)
    |0|1|1|0|0|0|0|0|0|0|0|0|0|0|0|1|0|0
3|Four Rooms (1995)|01-Jan-1995||http://us.imdb.com/M/title-exact?Four%20Rooms
    %20(1995)|0|0|0|0|0|0|0|0|0|0|0|0|0|0|0|1|0|0
4|Get Shorty (1995)|01-Jan-1995||http://us.imdb.com/M/title-exact?Get%20Shorty
    %20(1995)|0|1|0|0|0|1|0|0|1|0|0|0|0|0|0|0|0|0
5|Copycat (1995)|01-Jan-1995||http://us.imdb.com/M/title-exact?Copycat%20(1995)
    |0|0|0|0|0|0|1|0|1|0|0|0|0|0|0|0|1|0|0
6|Shanghai Triad (Yao a yao yao dao waipo qiao) (1995)|01-Jan-1995||http://us.imdb.com
    /Title?Yao+a+yao+yao+dao+waipo+qiao+(1995)|0|0|0|0|0|0|0|0|1|0|0|0|0|0|0|0|0|0
7|Twelve Monkeys (1995)|01-Jan-1995||http://us.imdb.com/M/title-exact?Twelve%20Monkeys
    %20(1995)|0|0|0|0|0|0|0|0|1|0|0|0|0|0|0|1|0|0|0
```

3.3.3 u.user

```
userid  age    gender    occupation    zip code

1      24        M          technician        85711
2      53        F     other             94043
3      23        M     writer            32067
4      24        M     technician        43537
5      33        F     other             15213
6      42        M     executive         98101
7      57        M     administrator     91344
```

3.4 Output Files

3.4.1 averageGenderF.png

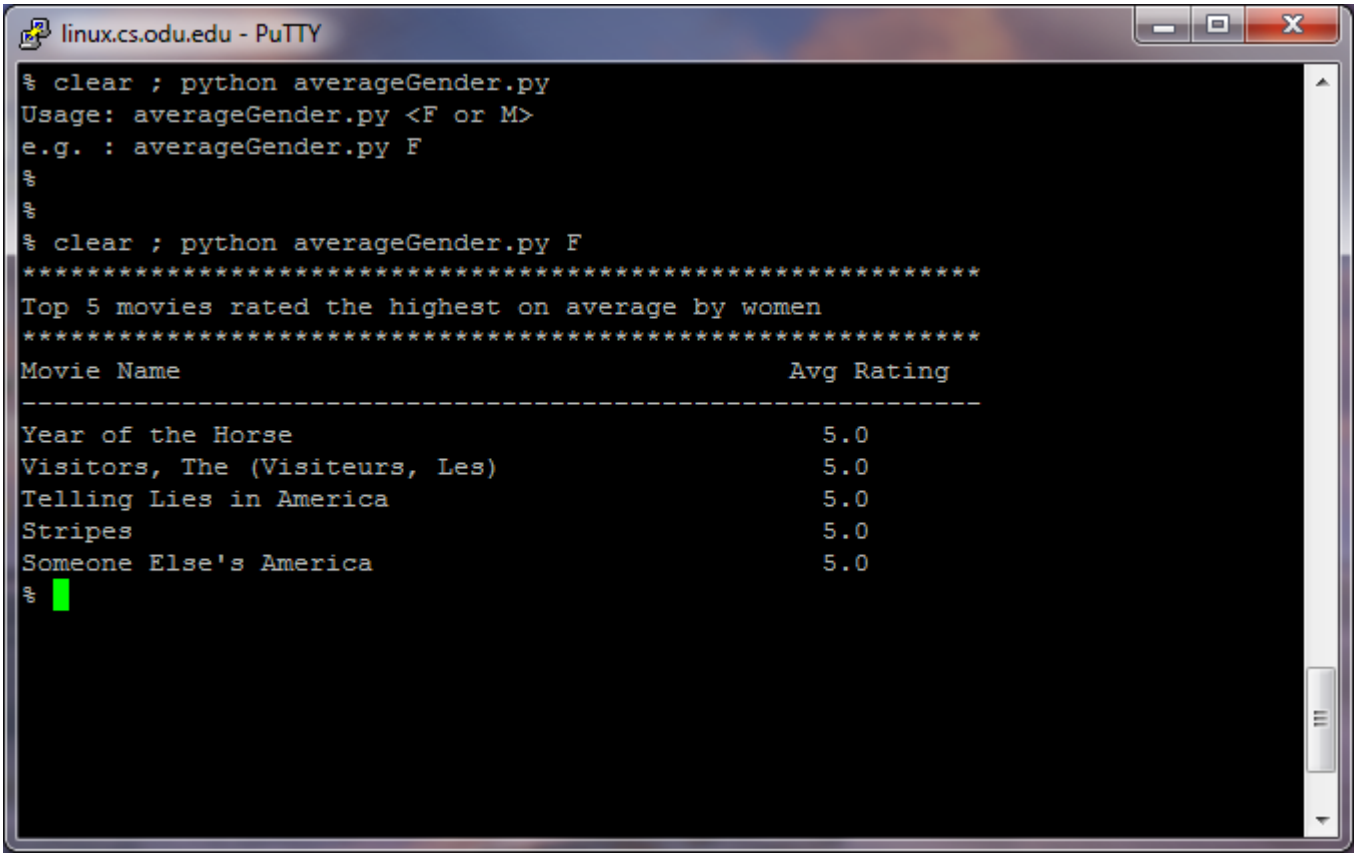


Figure 3: Top 5 movies rated the highest on average by women.

4 Question 4:

What 5 movies were rated the highest on average by men? Show the movies and their ratings sorted by ratings.

4.1 Approach

1. The approach is same as question 3.
2. The only difference is changing the command line argument Which can be seen in the output produced.
3. The output displays only top 5 movies rated the highest on average by men.
4. Figure 4 is the output for the program.

4.2 Source Code

4.2.1 averageMovies.py

```
#!/usr/bin/env python


# program to get the highest average ratings by men and women
import sys
import operator
def main():
    # take the arguments from the command line
    numOfArgs=len(sys.argv)
    if numOfArgs<2 or numOfArgs>2:

        print'Usage: averageGender.py <F or M> '
        print'e.g. : averageGender.py F'
        sys.exit(1)
    gender           = sys.argv[1]
    # Create a dictionary
    movie_userid_avg = {}
    movie_id_ratings = {}
    movie_userid_avg = {}
    users            = {}
    count            = 1

    readData = open('/home/bbokka/cs594/A8/u.data', 'r')
    readItem = open('/home/bbokka/cs594/A8/u.item','r')
    readUser = open('/home/bbokka/cs594/A8/u.user','r')

    # Reading u.user file to get the male or female
    for line in readUser:
        split_input_line = line.strip().split('|')
        user_id_from_user= split_input_line[0]
        user_gender      = split_input_line[2]

        if(gender.upper() == user_gender):
            users[user_id_from_user] = [user_gender]

    readUser.close()

    # reading the u.data file for item and ratings.
    for line in readData:
        split_input_line  = line.strip().split('\t')
        user_id_from_data = split_input_line[0]
        movie_id_from_data=  split_input_line[1]
        movie_rating      =  float(split_input_line[2])

        if user_id_from_data in users:
            try:
                movie_id_ratings[movie_id_from_data].append(movie_rating)
            except KeyError:
                movie_id_ratings[movie_id_from_data] = list()
                movie_id_ratings[movie_id_from_data].append(movie_rating)
    readData.close()

    # Calculating the average.
```

```
    for key in movie_id_ratings:
        avg = sum(movie_id_ratings[key])/len(movie_id_ratings[key])
        movie_userid_avg[key] = [float(avg)]

    # Reading the u.item file for movie name.
    for each_line in readItem:
        split_each_line  = each_line.strip().split('|')
        movie_item_id    = split_each_line[0]
        movie_name_split = split_each_line[1].split('(1')
        movie_name       = movie_name_split[0]

        try:
            movie_userid_avg[movie_item_id].append(movie_name)
        except KeyError:
            pass
    readItem.close()
    print "*" * 60
    print "Top 5 movies rated the highest on average by women"
    print "*" * 60
    print "Movie Name\t\t\t\t\t","Avg Rating"
    print "-" * 60
    # sorting the movies from highest to lowest based on the average value.
    for key, value in sorted(movie_userid_avg.items(), key=lambda e: e[1], reverse=True):
        if (count <=5 ):
            print '{:<50}{:<0.1f} '  .format(value[1],value[0])
            count += 1

if __name__ == "__main__":
    try:
        main()
    except KeyboardInterrupt:
        sys.exit(1)
```

4.3 Input Files

4.3.1 u.data

```
userid itemid rating timestamp

196      242      3    881250949
186      302      3    891717742
22       377      1    878887116
244      51       2    880606923
166      346      1    886397596
298      474      4    884182806
115      265      2    881171488
```

4.3.2 u.item

```
movie id | movie title | release date | video release date | IMDb URL | unknown |
    Action | Adventure | Animation |Children's | Comedy | Crime | Documentary | Drama
    | Fantasy | Film-Noir | Horror | Musical | Mystery | Romance | Sci-Fi | Thriller |
     War | Western |

1|Toy Story (1995)|01-Jan-1995||http://us.imdb.com/M/title-exact?Toy%20Story%20(1995)
    |0|0|0|1|1|1|0|0|0|0|0|0|0|0|0|0|0|0|0
2|GoldenEye (1995)|01-Jan-1995||http://us.imdb.com/M/title-exact?GoldenEye%20(1995)
    |0|1|1|0|0|0|0|0|0|0|0|0|0|0|0|1|0|0
3|Four Rooms (1995)|01-Jan-1995||http://us.imdb.com/M/title-exact?Four%20Rooms
    %20(1995)|0|0|0|0|0|0|0|0|0|0|0|0|0|0|0|1|0|0
4|Get Shorty (1995)|01-Jan-1995||http://us.imdb.com/M/title-exact?Get%20Shorty
    %20(1995)|0|1|0|0|0|1|0|0|1|0|0|0|0|0|0|0|0|0
5|Copycat (1995)|01-Jan-1995||http://us.imdb.com/M/title-exact?Copycat%20(1995)
    |0|0|0|0|0|0|1|0|1|0|0|0|0|0|0|0|1|0|0
6|Shanghai Triad (Yao a yao yao dao waipo qiao) (1995)|01-Jan-1995||http://us.imdb.com
    /Title?Yao+a+yao+yao+dao+waipo+qiao+(1995)|0|0|0|0|0|0|0|0|1|0|0|0|0|0|0|0|0|0
7|Twelve Monkeys (1995)|01-Jan-1995||http://us.imdb.com/M/title-exact?Twelve%20Monkeys
    %20(1995)|0|0|0|0|0|0|0|0|1|0|0|0|0|0|0|1|0|0|0
```

4.3.3 u.user

```
userid  age    gender    occupation    zip code

1      24        M          technician        85711
2      53        F      other             94043
3      23        M      writer            32067
4      24        M      technician        43537
5      33        F      other             15213
6      42        M      executive         98101
7      57        M      administrator     91344
```

4.4 Output Files

4.4.1 averageGenderM.png

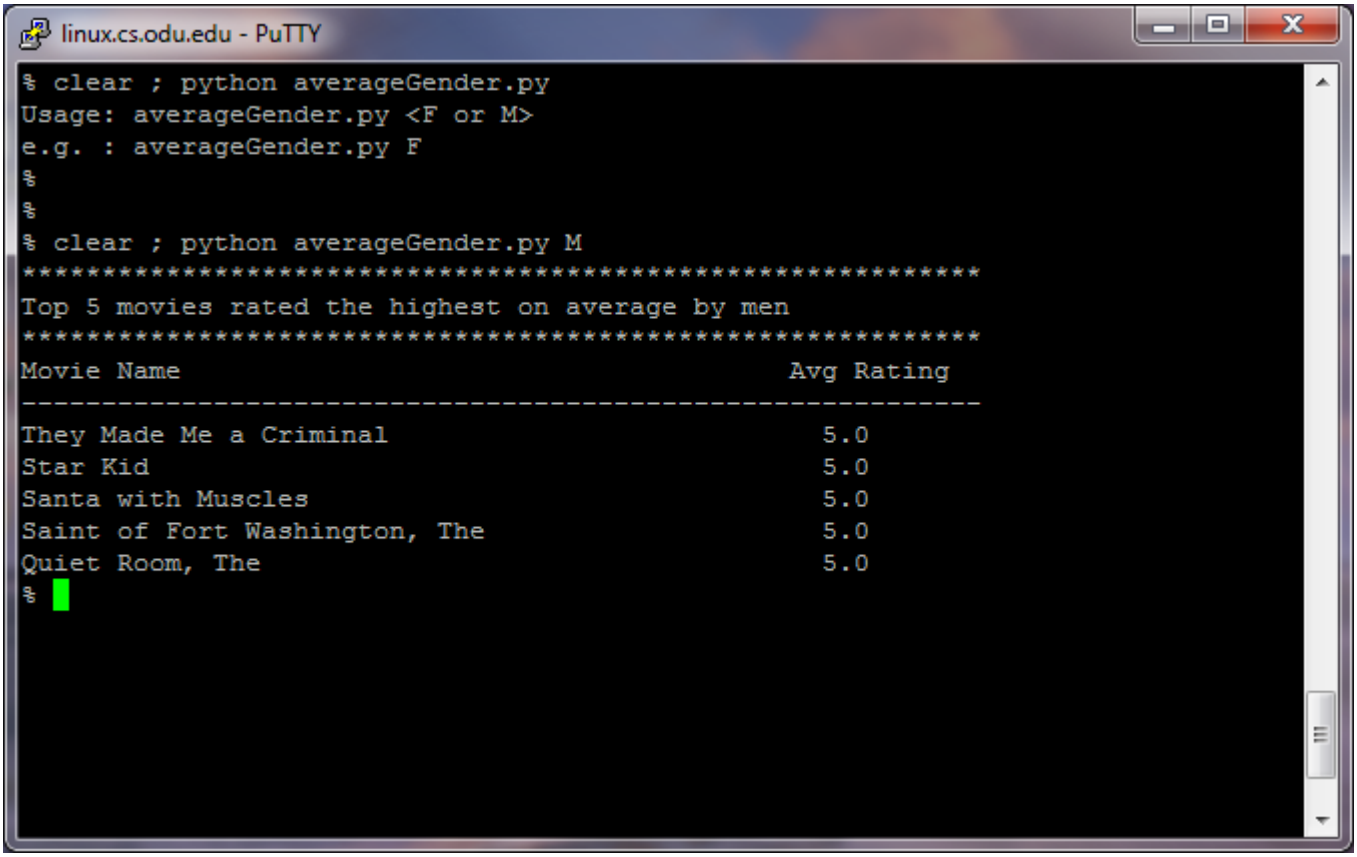


Figure 4: Top 5 movies rated the highest on average by men.

5 Question 5:

What movie received ratings most like Top Gun? Which movie received ratings that were least like Top Gun (negative correlation)?

5.1 Approach

5.2 Approach

1.

5.3 Source Code

5.3.1 recommendations.py

```
def calculateSimilarItems(prefs,n=80):
  # Create a dictionary of items showing which other items they
  # are most similar to.
  result={}
  item1 = 'Top Gun (1986)'
  # Invert the preference matrix to be item-centric
  itemPrefs=transformPrefs(prefs)
  c=0
  for item in itemPrefs:
    # Status updates for large datasets
    # c+=1
    # if c%100==0: print "%d / %d" % (c,len(itemPrefs))
    # Find the most similar items to this one
    scores=topMatches(itemPrefs,item1,n=n,similarity=sim_pearson)
    result[item]=scores
  return result


def loadMovieLens():
  # Get movie titles
  readData = open('/home/bbokka/cs594/A8/u.data', 'r')
  readItem = open('/home/bbokka/cs594/A8/u.item','r')
  movies={}
  for line in readItem:
    (id,title)=line.split('|')[0:2]
    movies[id]=title
  # Load data
  prefs={}
  for line in readData:
    (user,movieid,rating,ts)=line.split('\t')
    prefs.setdefault(user,{})
    prefs[user][movies[movieid]]=float(rating)

  return prefs
```

5.3.2 ratingsTopGun.py

```
#!/usr/bin/env python

import sys
import recommendations

def main():
    count = 'Top Gun (1986)'
    resultsOfloadMovieLens          = recommendations.loadMovieLens()
    resultsOfcalculateSimilarItems = recommendations.calculateSimilarItems(
   resultsOfloadMovieLens,n=80)
    print "*" * 60
    print "Movies received ratings most like Top Gun"
    print "*" * 60
    print "value\t\t\t\t\t","Movie Name"
    print "-" * 60

    for key, value in sorted(resultsOfcalculateSimilarItems.items(), key=lambda e: e
   [1], reverse=True):
        variable = key
        if (count == variable):
            for value , movie in value:
                print  value , movie

if __name__ == "__main__":
    try:
        main()
    except KeyboardInterrupt:
        sys.exit(1)
```

5.4 Input Files

5.4.1 u.data

```
userid itemid rating timestamp

196      242      3    881250949
186      302      3    891717742
22       377      1    878887116
244      51       2    880606923
166      346      1    886397596
298      474      4    884182806
115      265      2    881171488
```

5.4.2 u.item

```
movie id | movie title | release date | video release date | IMDb URL | unknown |
    Action | Adventure | Animation |Children's | Comedy | Crime | Documentary | Drama
    | Fantasy | Film-Noir | Horror | Musical | Mystery | Romance | Sci-Fi | Thriller |
     War | Western |

1|Toy Story (1995)|01-Jan-1995||http://us.imdb.com/M/title-exact?Toy%20Story%20(1995)
    |0|0|0|1|1|1|0|0|0|0|0|0|0|0|0|0|0|0|0
2|GoldenEye (1995)|01-Jan-1995||http://us.imdb.com/M/title-exact?GoldenEye%20(1995)
    |0|1|1|0|0|0|0|0|0|0|0|0|0|0|0|1|0|0
3|Four Rooms (1995)|01-Jan-1995||http://us.imdb.com/M/title-exact?Four%20Rooms
    %20(1995)|0|0|0|0|0|0|0|0|0|0|0|0|0|0|0|1|0|0
4|Get Shorty (1995)|01-Jan-1995||http://us.imdb.com/M/title-exact?Get%20Shorty
    %20(1995)|0|1|0|0|0|1|0|0|1|0|0|0|0|0|0|0|0|0
5|Copycat (1995)|01-Jan-1995||http://us.imdb.com/M/title-exact?Copycat%20(1995)
    |0|0|0|0|0|0|1|0|1|0|0|0|0|0|0|0|1|0|0
6|Shanghai Triad (Yao a yao yao dao waipo qiao) (1995)|01-Jan-1995||http://us.imdb.com
    /Title?Yao+a+yao+yao+dao+waipo+qiao+(1995)|0|0|0|0|0|0|0|0|1|0|0|0|0|0|0|0|0|0
7|Twelve Monkeys (1995)|01-Jan-1995||http://us.imdb.com/M/title-exact?Twelve%20Monkeys
    %20(1995)|0|0|0|0|0|0|0|0|1|0|0|0|0|0|0|1|0|0|0
```

5.5 Output Files

6 Question 6:

Which 5 raters rated the most films? Show the raters' IDs and the number of films each rated.

6.1 Approach

1. The approach is to read U.data file get user id and for the corresponding user id store all the ratings .
2. Now, we count the number of ratings for each user by using the len() function.
3. The output displays only top 5 users and their number of ratings.
4. Figure 5 is the output for the program.

6.2 Source Code

6.2.1 ratersMostRatedMovies.py

```
#!/usr/bin/env python

import sys

def main():
    # Create a dictionary
    movie_userid_ratings = {}
    movie_userid_len     = {}
    count                = 1
    # Read the input files.
    readData = open('/home/bbokka/cs594/A8/u.data', 'r')
     # reading the u.data file for item and ratings.
    for line in readData:
        split_input_line  = line.strip().split('\t')
        user_id_from_data = split_input_line[0]
        movie_rating      =  float(split_input_line[2])

        try:
            movie_userid_ratings[user_id_from_data].append(movie_rating)
        except KeyError:
            movie_userid_ratings[user_id_from_data] = list()
            movie_userid_ratings[user_id_from_data].append(movie_rating)
    readData.close()
    # Calculating the number of movies each user rated.
    for key in movie_userid_ratings:
        length = len(movie_userid_ratings[key])
        movie_userid_len[key] = int(length)


    print '*' * 60
    print "Top 5 raters rated movies."
    print "*" * 60
    print "User id\t\t","Number of movies rated"
    print "-" * 60
    # sorting the movies from highest to lowest based on the number of
    # ratings for each movie.
    for key, value in sorted(movie_userid_len.items(), key=lambda e: e[1], reverse=
    True):
        if (count <= 5):
            print '{:<20}{} '  .format(key,value)
            count += 1

if __name__ == "__main__":
    try:
        main()
    except KeyboardInterrupt:
        sys.exit(1)
```

6.3 Input Files

6.3.1 u.data

```
userid itemid rating timestamp

196      242      3    881250949
186      302      3    891717742
22       377      1    878887116
244      51       2    880606923
166      346      1    886397596
298      474      4    884182806
115      265      2    881171488
```

6.4 Output Files

6.4.1 ratersMostRatedMovies.png

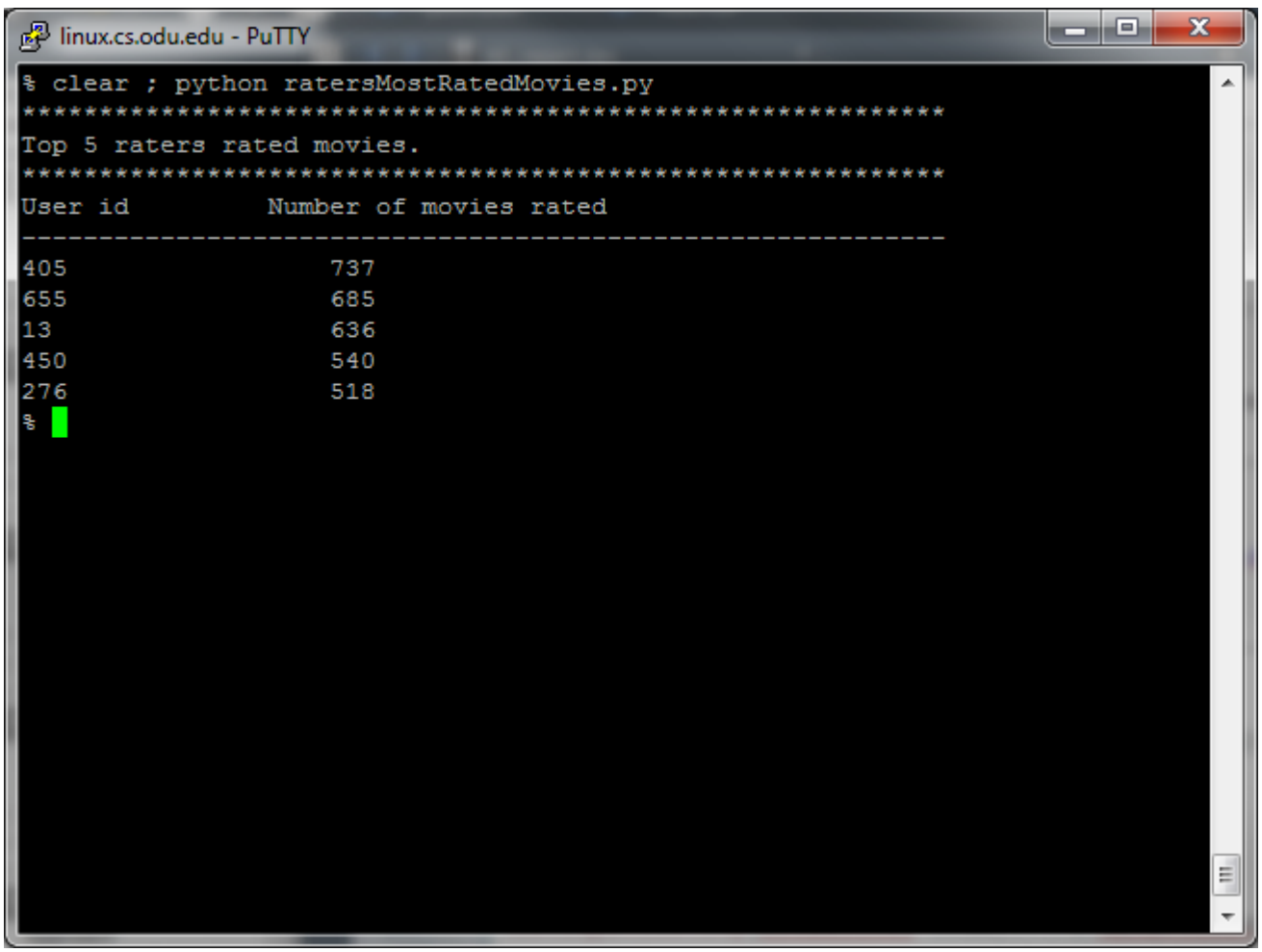


Figure 5: Top 5 raters rated movies.

7 Question 7:

Which 5 raters most agreed with each other? Show the raters' IDs and Pearson's r, sorted by r.

7.1 Approach

7.2 Description of

7.3 Source Code

7.4 Output Files

8 Question 8:

Which 5 raters most disagreed with each other (negative correlation)? Show the raters' IDs and Pearson's r, sorted by r.

8.1 Approach

8.2 Description of

8.3 Source Code

8.4 Output Files

9 Question 9:

What movie was rated highest on average by men over 40? By men under 40?

9.1 Approach

1. To solve this question the straight forward approach is to read u.user file get the user id, gender, age of each user. Read u.data file get movie id, corresponding movie ratings. Read u.item file to get movie id and corresponding movie name.

2. Program averageOverUnderGender.py is designed in such a way that it can get the highest averages by men and women over and under 40 just by changing the arguments to F or M and "over" or "under" while executing the program.

3. From u.user file extract the user id, gender, age based on arguments given from command line either men over 40 or men under 40 or women over 40 or women under 40 is stored into dictionary.

4. Now, Extract the movie id and ratings. For each user id build a dictionary that has all the ratings for the respective gender with over or under condition.

5. Now, for each each movie id calculate the average a using sum() and len() functions.

6. For each movie id get the movie name and append it to the dictionary which has average.

7. Now, sort the dictionary and get the movies which are rated highest on average by men or women over 40 or under 40.

8. Figure 6 is the output showing how to execute the program and shows the top 5 movies which are rated highest on average by men "over" 40.

9. Figure 7 is the output showing how to execute the program and shows the top 5 movies which are rated highest on average by men "under" 40.

9.2 Source Code

9.2.1 averageOverUnderGender.py

```
#!/usr/bin/env python


# program to get the highest average ratings by men under and over 40.
#
import sys
import operator
def main():
    # Take the arguments from the command line
    numOfArgs=len(sys.argv)
    if numOfArgs<3 or numOfArgs>3:

        print'Usage: averageOverUnderGender.py <F or M> <under or over>'
        print'e.g. : averageOverUnderGender.py F over'
        sys.exit(1)
    gender            = sys.argv[1]
    gender_under_over= sys.argv[2]

    # Create a dictionary
    movie_userid_avg = {}
    movie_id_ratings = {}
    movie_userid_avg = {}
    users            = {}
    count            = 1

    # Read the files
    readData = open('/home/bbokka/cs594/A8/u.data', 'r')
    readItem = open('/home/bbokka/cs594/A8/u.item','r')
    readUser = open('/home/bbokka/cs594/A8/u.user','r')

    # Reading u.user file to get the male or female
    for line in readUser:
        split_input_line = line.strip().split('|')
        user_id_from_user= split_input_line[0]
        user_age         = int(split_input_line[1])
        user_gender      = split_input_line[2]

        if gender_under_over == 'over' :
            if(gender.upper() == user_gender and user_age > 40):
                users[user_id_from_user] = [user_age]

        if gender_under_over == 'under' :
            if(gender.upper() == user_gender and user_age < 40):
                users[user_id_from_user] = [user_age]

    readUser.close()

    # reading the u.data file for item and ratings.
    for line in readData:
        split_input_line  = line.strip().split('\t')
        user_id_from_data = split_input_line[0]
        movie_id_from_data=  split_input_line[1]
        movie_rating      =  float(split_input_line[2])
```

```

        if user_id_from_data in users:
            try:
                movie_id_ratings[movie_id_from_data].append(movie_rating)
            except KeyError:
                movie_id_ratings[movie_id_from_data] = list()
                movie_id_ratings[movie_id_from_data].append(movie_rating)
    readData.close()

    # Calculating the average.
    for key in movie_id_ratings:
        avg = sum(movie_id_ratings[key])/len(movie_id_ratings[key])
        movie_userid_avg[key] = [float(avg)]

    # Reading the u.item file for movie name.
    for each_line in readItem:
        split_each_line  = each_line.strip().split('|')

        movie_item_id    = split_each_line[0]
        movie_name_split = split_each_line[1].split('(1')
        movie_name       = movie_name_split[0]

        try:
            movie_userid_avg[movie_item_id].append(movie_name)
        except KeyError:
            pass
    readItem.close()
    print '*' * 55
    print "Movies rated highest on average by men"
    print "*" * 55
    print "Movie Name\t\t\t\t\t","Avg Rating"
    print "-" * 55
    # sorting the movies from highest to lowest based on the average value.
    for key, value in sorted(movie_userid_avg.items(), key=lambda e: e[1], reverse=True):
        if (count <=5 ):
            print '{:<50}{:<0.1f} '  .format(value[1],value[0])
            count += 1

if __name__ == "__main__":
    try:
        main()
    except KeyboardInterrupt:
        sys.exit(1)
```

9.3 Input Files

9.3.1 u.data

```
userid itemid rating timestamp

196      242      3    881250949
186      302      3    891717742
22       377      1    878887116
244      51       2    880606923
166      346      1    886397596
298      474      4    884182806
115      265      2    881171488
```

9.3.2 u.item

```
movie id | movie title | release date | video release date | IMDb URL | unknown |
    Action | Adventure | Animation |Children's | Comedy | Crime | Documentary | Drama
    | Fantasy | Film-Noir | Horror | Musical | Mystery | Romance | Sci-Fi | Thriller |
     War | Western |

1|Toy Story (1995)|01-Jan-1995||http://us.imdb.com/M/title-exact?Toy%20Story%20(1995)
    |0|0|0|1|1|1|0|0|0|0|0|0|0|0|0|0|0|0|0
2|GoldenEye (1995)|01-Jan-1995||http://us.imdb.com/M/title-exact?GoldenEye%20(1995)
    |0|1|1|0|0|0|0|0|0|0|0|0|0|0|0|1|0|0
3|Four Rooms (1995)|01-Jan-1995||http://us.imdb.com/M/title-exact?Four%20Rooms
    %20(1995)|0|0|0|0|0|0|0|0|0|0|0|0|0|0|0|1|0|0
4|Get Shorty (1995)|01-Jan-1995||http://us.imdb.com/M/title-exact?Get%20Shorty
    %20(1995)|0|1|0|0|0|1|0|0|1|0|0|0|0|0|0|0|0|0
5|Copycat (1995)|01-Jan-1995||http://us.imdb.com/M/title-exact?Copycat%20(1995)
    |0|0|0|0|0|0|1|0|1|0|0|0|0|0|0|0|1|0|0
6|Shanghai Triad (Yao a yao yao dao waipo qiao) (1995)|01-Jan-1995||http://us.imdb.com
    /Title?Yao+a+yao+yao+dao+waipo+qiao+(1995)|0|0|0|0|0|0|0|0|1|0|0|0|0|0|0|0|0|0
7|Twelve Monkeys (1995)|01-Jan-1995||http://us.imdb.com/M/title-exact?Twelve%20Monkeys
    %20(1995)|0|0|0|0|0|0|0|0|1|0|0|0|0|0|0|1|0|0|0
```

9.3.3 u.user

```
userid  age    gender    occupation    zip code

1      24        M          technician        85711
2      53        F      other             94043
3      23        M      writer            32067
4      24        M      technician        43537
5      33        F      other             15213
6      42        M      executive         98101
7      57        M      administrator     91344
```

9.4 Output Files

9.4.1 averageOverM.png

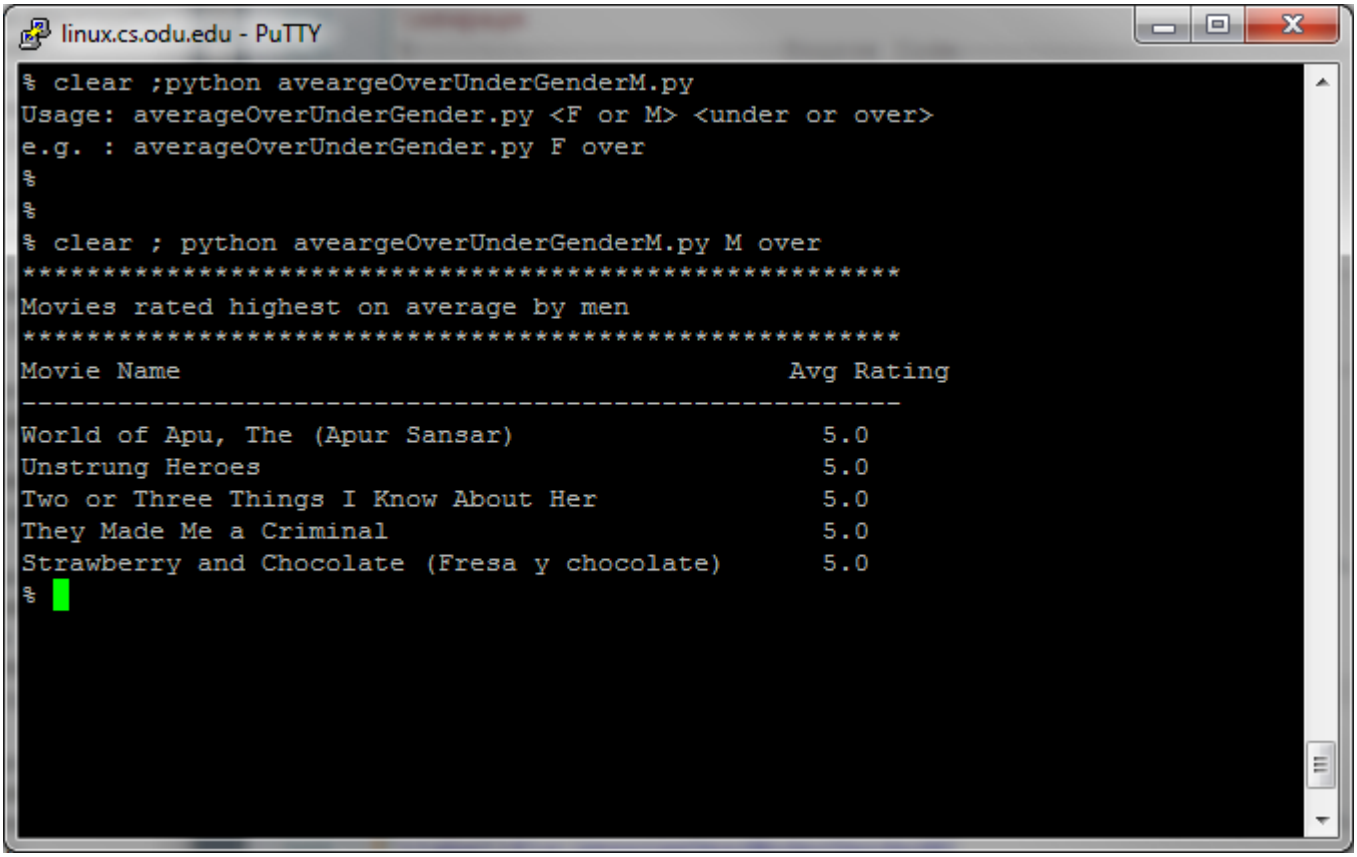


Figure 6: Movies rated highest on average by men over 40.

9.4.2 averageUnderM.png

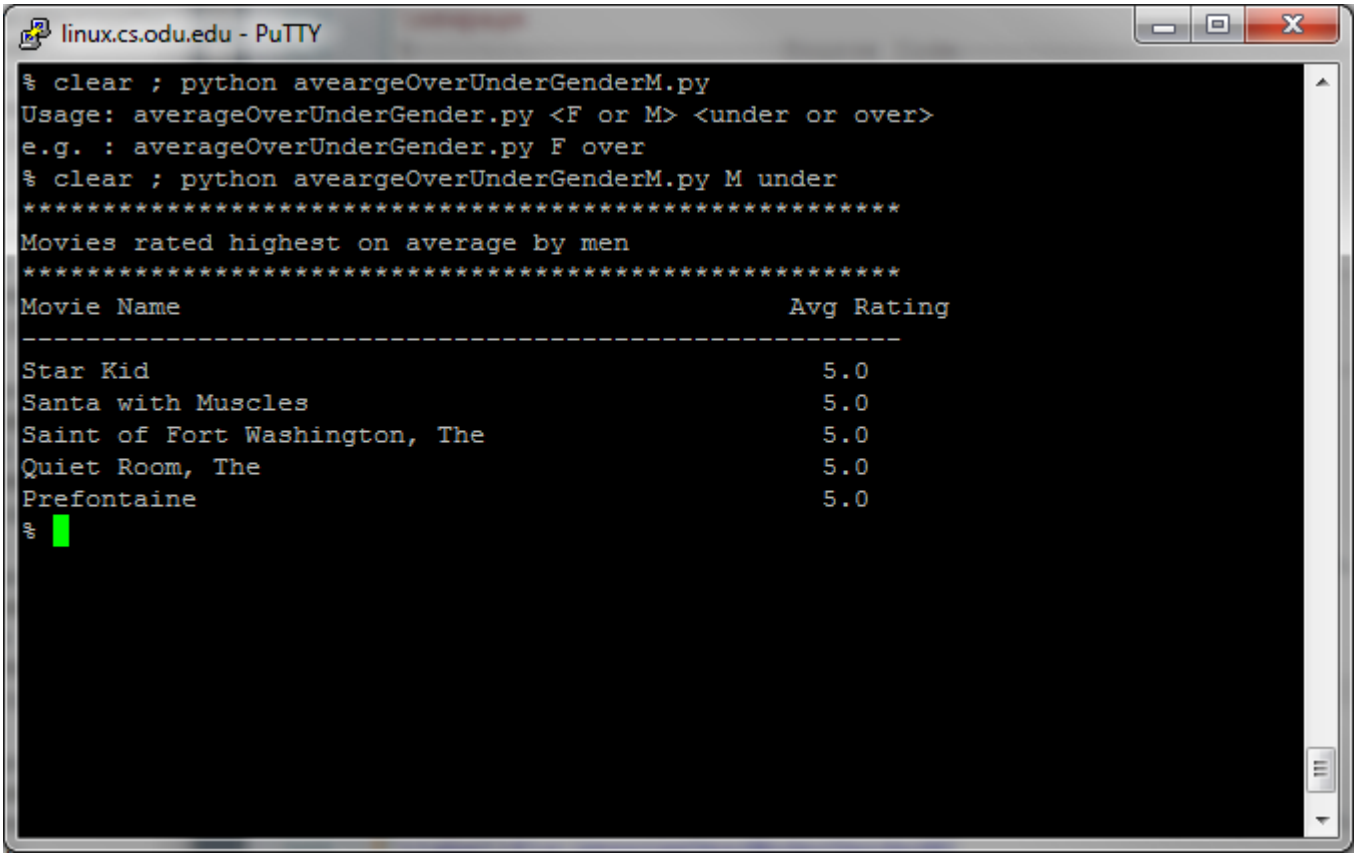


Figure 7: Movies rated highest on average by men under 40.

10 Question 10:

What movie was rated highest on average by women over 40? By women under 40?

10.1 Approach

1. The approach is same as question 9.
2. To get the corresponding output for this question it will still use averageOverUnderGender.py but the only difference is changing the arguments.
3. Figure 8 is the output showing how to execute the program and shows the top 5 movies which are rated highest on average by men "over" 40.
4. Figure 9 is the output showing how to execute the program and shows the top 5 movies which are rated highest on average by men "under" 40.

10.2 Source Code

10.2.1 averageOverUnderGender.py

```
#!/usr/bin/env python


# program to get the highest average ratings by women under and over 40.
import sys
import operator
def main():
    # Take the arguments from the command line
    numOfArgs=len(sys.argv)
    if numOfArgs<3 or numOfArgs>3:

        print'Usage: averageOverUnderGender.py <F or M> <under or over>'
        print'e.g. : averageOverUnderGender.py F over'
        sys.exit(1)
    gender            = sys.argv[1]
    gender_under_over= sys.argv[2]

    # Create a dictionary
    movie_userid_avg = {}
    movie_id_ratings = {}
    movie_userid_avg = {}
    users            = {}
    count            = 1

    # Read the files
    readData = open('/home/bbokka/cs594/A8/u.data', 'r')
    readItem = open('/home/bbokka/cs594/A8/u.item','r')
    readUser = open('/home/bbokka/cs594/A8/u.user','r')

    # Reading u.user file to get the male or female
    for line in readUser:
        split_input_line = line.strip().split('|')
        user_id_from_user= split_input_line[0]
        user_age         = int(split_input_line[1])
        user_gender      = split_input_line[2]

        if gender_under_over == 'over' :
            if(gender.upper() == user_gender and user_age > 40):
                users[user_id_from_user] = [user_age]

        if gender_under_over == 'under' :
            if(gender.upper() == user_gender and user_age < 40):
                users[user_id_from_user] = [user_age]

    readUser.close()

    # reading the u.data file for item and ratings.
    for line in readData:
        split_input_line  = line.strip().split('\t')
        user_id_from_data = split_input_line[0]
        movie_id_from_data=  split_input_line[1]
        movie_rating      =  float(split_input_line[2])

```

```
        if user_id_from_data in users:
            try:
                movie_id_ratings[movie_id_from_data].append(movie_rating)
            except KeyError:
                movie_id_ratings[movie_id_from_data] = list()
                movie_id_ratings[movie_id_from_data].append(movie_rating)
    readData.close()

    # Calculating the average.
    for key in movie_id_ratings:
        avg = sum(movie_id_ratings[key])/len(movie_id_ratings[key])
        movie_userid_avg[key] = [float(avg)]

    # Reading the u.item file for movie name.
    for each_line in readItem:
        split_each_line  = each_line.strip().split('|')

        movie_item_id    = split_each_line[0]
        movie_name_split = split_each_line[1].split('(1')
        movie_name       = movie_name_split[0]

        try:
            movie_userid_avg[movie_item_id].append(movie_name)
        except KeyError:
            pass
    readItem.close()
    print '*' * 55
    print "Movies rated highest on average by women"
    print "*" * 55
    print "Movie Name\t\t\t\t\t","Avg Rating"
    print "-" * 55
    # sorting the movies from highest to lowest based on the average value.
    for key, value in sorted(movie_userid_avg.items(), key=lambda e: e[1], reverse=
   True):
        if (count <=5 ):
            print '{:<50}{:<0.1f} '  .format(value[1],value[0])
            count += 1

if __name__ == "__main__":
    try:
        main()
    except KeyboardInterrupt:
        sys.exit(1)
```

10.3 Input Files

10.3.1 u.data

```
userid itemid rating timestamp

196      242      3    881250949
186      302      3    891717742
22       377      1    878887116
244      51       2    880606923
166      346      1    886397596
298      474      4    884182806
115      265      2    881171488
```

10.3.2 u.item

```
movie id | movie title | release date | video release date | IMDb URL | unknown |
    Action | Adventure | Animation |Children's | Comedy | Crime | Documentary | Drama
    | Fantasy | Film-Noir | Horror | Musical | Mystery | Romance | Sci-Fi | Thriller |
     War | Western |

1|Toy Story (1995)|01-Jan-1995||http://us.imdb.com/M/title-exact?Toy%20Story%20(1995)
    |0|0|0|1|1|1|0|0|0|0|0|0|0|0|0|0|0|0|0
2|GoldenEye (1995)|01-Jan-1995||http://us.imdb.com/M/title-exact?GoldenEye%20(1995)
    |0|1|1|0|0|0|0|0|0|0|0|0|0|0|0|1|0|0
3|Four Rooms (1995)|01-Jan-1995||http://us.imdb.com/M/title-exact?Four%20Rooms
    %20(1995)|0|0|0|0|0|0|0|0|0|0|0|0|0|0|0|1|0|0
4|Get Shorty (1995)|01-Jan-1995||http://us.imdb.com/M/title-exact?Get%20Shorty
    %20(1995)|0|1|0|0|0|1|0|0|1|0|0|0|0|0|0|0|0|0
5|Copycat (1995)|01-Jan-1995||http://us.imdb.com/M/title-exact?Copycat%20(1995)
    |0|0|0|0|0|0|1|0|1|0|0|0|0|0|0|0|1|0|0
6|Shanghai Triad (Yao a yao yao dao waipo qiao) (1995)|01-Jan-1995||http://us.imdb.com
    /Title?Yao+a+yao+yao+dao+waipo+qiao+(1995)|0|0|0|0|0|0|0|0|1|0|0|0|0|0|0|0|0|0
7|Twelve Monkeys (1995)|01-Jan-1995||http://us.imdb.com/M/title-exact?Twelve%20Monkeys
    %20(1995)|0|0|0|0|0|0|0|0|1|0|0|0|0|0|0|1|0|0|0
```

10.3.3 u.user

```
userid  age    gender    occupation    zip code

1      24        M          technician        85711
2      53        F     other             94043
3      23        M     writer            32067
4      24        M     technician        43537
5      33        F     other             15213
6      42        M     executive         98101
7      57        M     administrator     91344
```

10.4 Output Files

10.4.1 averageOverF.png

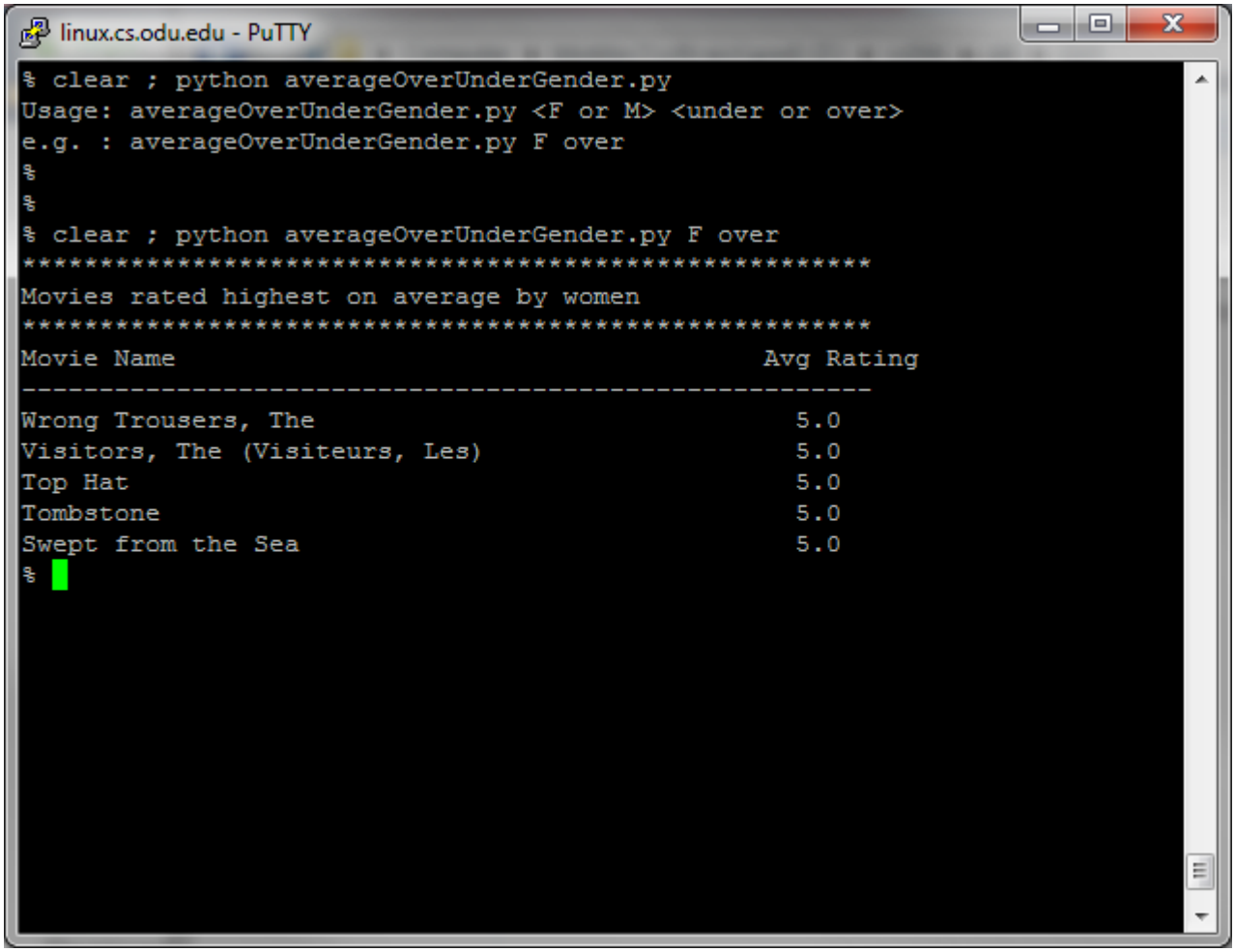
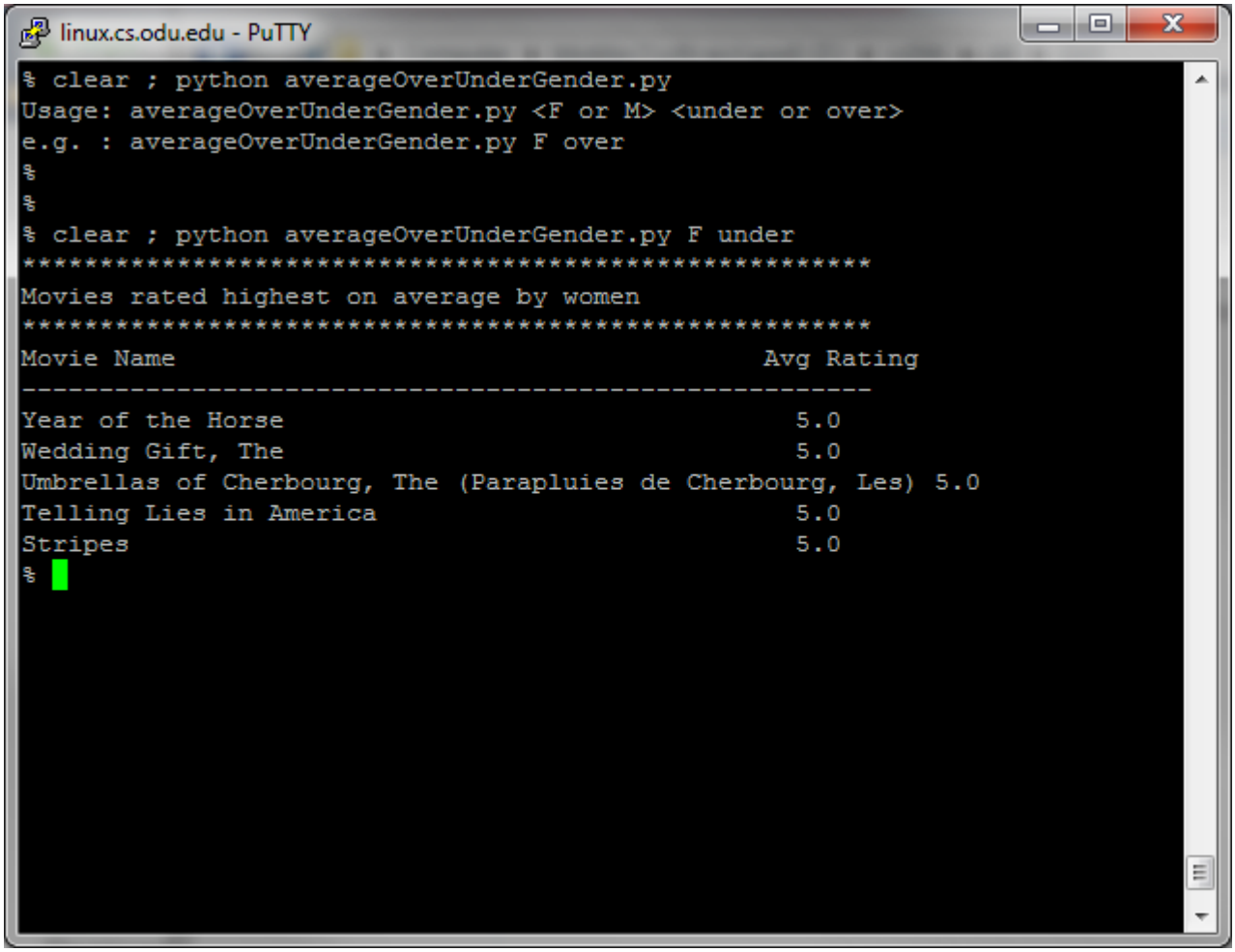


Figure 8: Movies rated highest on average by women over 40.

10.4.2 averageUnderF.png



```
% clear ; python averageOverUnderGender.py
Usage: averageOverUnderGender.py <F or M> <under or over>
e.g. : averageOverUnderGender.py F over
%
%
% clear ; python averageOverUnderGender.py F under
*******************************************************
Movies rated highest on average by women
*******************************************************
Movie Name                                        Avg Rating
-------------------------------------------------------
Year of the Horse                                   5.0
Wedding Gift, The                                   5.0
Umbrellas of Cherbourg, The (Parapluies de Cherbourg, Les) 5.0
Telling Lies in America                             5.0
Stripes                                             5.0
%
```

Figure 9: Movies rated highest on average by women under 40.

References

[1] Data files used for assignment. http://files.grouplens.org/datasets/movielens/ml-100k/, April 1998. Data collected by GroupLens Research Project.

[2] K. Arthur Endsley. recommendations.py program. https://github.com/arthur-e/Programming-Collective-Intelligence/blob/master/chapter2/recommendations.py, December 2012.

[3] Frederick Hamidi. Python: Run function from the command line. http://stackoverflow.com/questions/3987041/python-run-function-from-the-command-line, October 2010.

[4] Devin Jeanpierre. Sort a python dictionary by value. http://stackoverflow.com/questions/613183/sort-a-python-dictionary-by-value, March 2009.

[5] Toby Segaran. *Programming Collective Intelligence*. O'Reilly Media, August 2007.

[]