### 0.0.1 We are performing an EDA and feature engineering on the movielens dataset,using the following packages.

[1]:
```
#importing the required libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from matplotlib import style
import seaborn as sns
%matplotlib inline
import warnings
warnings.filterwarnings('ignore')
```

- **Import the three datasets**

[2]:
```
#Movies.dat
#Format - MovieID::Title::Genres
#Format -  UserID::Gender::Age::Occupation::Zip-code
#Ratings.dat
#Format - UserID::MovieID::Rating::Timestamp



movies_df = pd.read_csv(
    'movies.dat',
    sep='::',
    names=['MovieID','Title','Genres'],
    header=None
)
users_df = pd.read_csv(
    'users.dat',
    sep='::',
    names=['UserID','Gender','Age', 'Occupation', 'zip-code'],
    header=None
)
ratings_df = pd.read_csv(
    'ratings.dat',
    sep='::',
```

```
    names=['UserID','MovieID','Rating', 'Timestamp'],
    parse_dates=['Timestamp'],
    header=None
)
```

[3]:
```
ratings_df
```

[3]:
```
         UserID  MovieID  Rating  Timestamp
0             1     1193       5  978300760
1             1      661       3  978302109
2             1      914       3  978301968
3             1     3408       4  978300275
4             1     2355       5  978824291
...         ...      ...     ...        ...
1000204    6040     1091       1  956716541
1000205    6040     1094       5  956704887
1000206    6040      562       5  956704746
1000207    6040     1096       4  956715648
1000208    6040     1097       4  956715569

[1000209 rows x 4 columns]
```

[4]:
```
movies_df = pd.read_csv(
    'movies.dat',
    sep='::',
    names=['MovieID','Title','Genres'],
    header=None
)
```

[5]:
```
movies_df
```

[5]:
```
      MovieID                               Title  \
0           1                    Toy Story (1995)
1           2                      Jumanji (1995)
2           3             Grumpier Old Men (1995)
3           4            Waiting to Exhale (1995)
4           5  Father of the Bride Part II (1995)
...       ...                                 ...
3878     3948             Meet the Parents (2000)
3879     3949          Requiem for a Dream (2000)
3880     3950                  Tigerland (2000)
3881     3951            Two Family House (2000)
3882     3952              Contender, The (2000)

                        Genres
0  Animation|Children's|Comedy
1  Adventure|Children's|Fantasy
```

```
2                    Comedy|Romance
3                      Comedy|Drama
4                            Comedy
...                             ...
3878                         Comedy
3879                          Drama
3880                          Drama
3881                          Drama
3882                 Drama|Thriller

[3883 rows x 3 columns]
```

[6]:
```
movies_df.shape
```

[6]:
```
(3883, 3)
```

[7]:
```
users_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6040 entries, 0 to 6039
Data columns (total 5 columns):
 #   Column      Non-Null Count  Dtype 
---  ------      --------------  ----- 
 0   UserID      6040 non-null   int64 
 1   Gender      6040 non-null   object
 2   Age         6040 non-null   int64 
 3   Occupation  6040 non-null   int64 
 4   zip-code    6040 non-null   object
dtypes: int64(3), object(2)
memory usage: 236.1+ KB
```

### 0.0.2 • Create a new dataset [Master_Data] with the following columns MovieID Title UserID Age Gender Occupation Rating. (Hint: (i) Merge two tables at a time. (ii) Merge the tables using two primary keys MovieID & UserId)

[8]:
```
First_merge = pd.merge(movies_df,ratings_df,on="MovieID")
First_merge.head()
```

[8]:

| | MovieID | Title | Genres | UserID | Rating | Timestamp |
|---|---|---|---|---|---|---|
| 0 | 1 | Toy Story (1995) | Animation\|Children's\|Comedy | 1 | 5 | 978824268 |
| 1 | 1 | Toy Story (1995) | Animation\|Children's\|Comedy | 6 | 4 | |
| 2 | 1 | Toy Story (1995) | Animation\|Children's\|Comedy | 8 | 4 | |
| 3 | 1 | Toy Story (1995) | Animation\|Children's\|Comedy | 9 | 5 | |
| 4 | 1 | Toy Story (1995) | Animation\|Children's\|Comedy | 10 | 5 | |

```
1  978237008
2  978233496
3  978225952
4  978226474
```

[9]:
```
second_merge = pd.merge(users_df,First_merge,on="UserID")
second_merge.head()
```

[9]:
```
   UserID Gender  Age  Occupation zip-code  MovieID  \
0       1      F    1          10    48067        1
1       1      F    1          10    48067       48
2       1      F    1          10    48067      150
3       1      F    1          10    48067      260
4       1      F    1          10    48067      527

                                  Title  \
0                     Toy Story (1995)
1                     Pocahontas (1995)
2                     Apollo 13 (1995)
3  Star Wars: Episode IV - A New Hope (1977)
4                     Schindler's List (1993)

                             Genres  Rating  Timestamp
0           Animation|Children's|Comedy       5  978824268
1  Animation|Children's|Musical|Romance       5  978824351
2                             Drama       5  978301777
3      Action|Adventure|Fantasy|Sci-Fi       4  978300760
4                           Drama|War       5  978824195
```

[10]:
```
#drop the zipcode and timestamp
master_data = second_merge.drop(['zip-code','Timestamp'],axis=1) #axis=1 means columns
master_data.head()
```

[10]:
```
   UserID Gender  Age  Occupation  MovieID  \
0       1      F    1          10        1
1       1      F    1          10       48
2       1      F    1          10      150
3       1      F    1          10      260
4       1      F    1          10      527

                                  Title  \
0                     Toy Story (1995)
1                     Pocahontas (1995)
2                     Apollo 13 (1995)
3  Star Wars: Episode IV - A New Hope (1977)
4                     Schindler's List (1993)
```

| | Genres | Rating |
|---|---|---|
| 0 | Animation\|Children's\|Comedy | 5 |
| 1 | Animation\|Children's\|Musical\|Romance | 5 |
| 2 | Drama | 5 |
| 3 | Action\|Adventure\|Fantasy\|Sci-Fi | 4 |
| 4 | Drama\|War | 5 |

[11]:
```
## The second_merge file has the desired variables as per q2
```

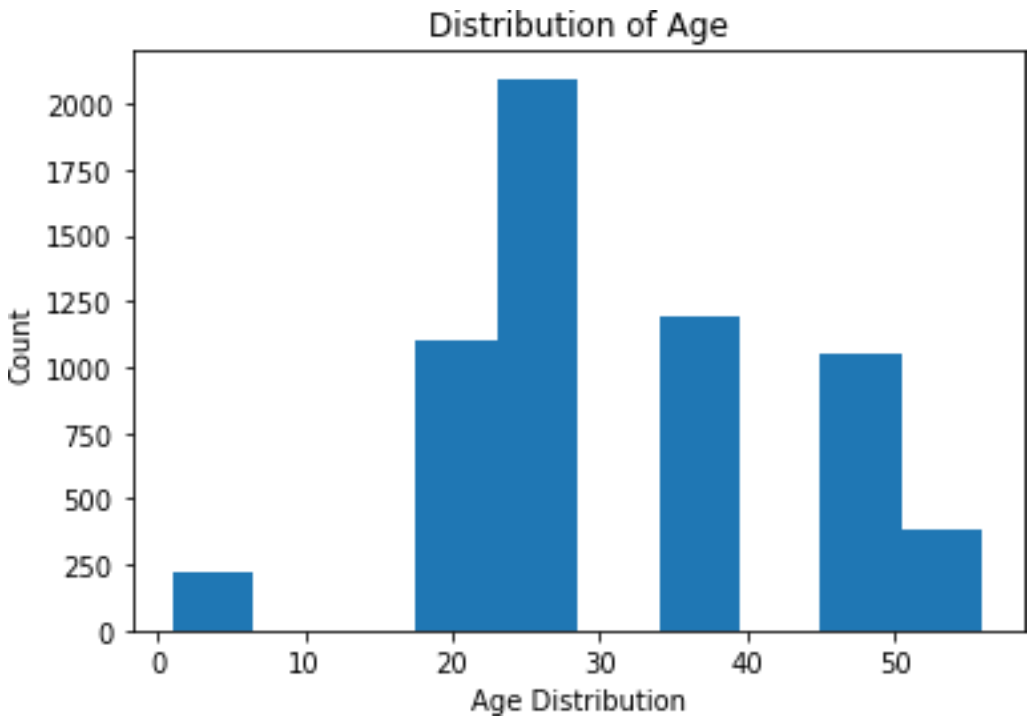
[12]:
```
#Explore the datasets using visual representations (graphs or tables), also include your comments on the following:
#User Age Distribution

users_df['Age'].value_counts()
```

[12]:
```
25    2096
35    1193
18    1103
45     550
50     496
56     380
1      222
Name: Age, dtype: int64
```

[13]:
```
#creating a histogram for all unique ages:
plt.hist(users_df['Age'])
plt.xlabel('Age Distribution')
plt.title('Distribution of Age')
plt.ylabel('Count')
```

[13]:
```
Text(0, 0.5, 'Count')
```




[14]:
```
#2.User rating of the movie "Toy Story"
First_merge.head()
```

[14]:

| | MovieID | Title | Genres | UserID | Rating | Timestamp |
|---|---|---|---|---|---|---|
| 0 | 1 | Toy Story (1995) | Animation\|Children's\|Comedy | 1 | 5 | 978824268 |
| 1 | 1 | Toy Story (1995) | Animation\|Children's\|Comedy | 6 | 4 | 978237008 |
| 2 | 1 | Toy Story (1995) | Animation\|Children's\|Comedy | 8 | 4 | 978233496 |
| 3 | 1 | Toy Story (1995) | Animation\|Children's\|Comedy | 9 | 5 | 978225952 |
| 4 | 1 | Toy Story (1995) | Animation\|Children's\|Comedy | 10 | 5 | 978226474 |

[15]:
```
group = First_merge.groupby('Title')
```

[16]:
```
group.head()
```

[16]:

| | MovieID | Title | Genres | UserID |
|---|---|---|---|---|
| 0 | 1 | Toy Story (1995) | Animation\|Children's\|Comedy | 1 |

```
1              1        Toy Story (1995)   Animation|Children's|Comedy       6
2              1        Toy Story (1995)   Animation|Children's|Comedy       8
3              1        Toy Story (1995)   Animation|Children's|Comedy       9
4              1        Toy Story (1995)   Animation|Children's|Comedy      10
...          ...                     ...                          ...     ...
999821      3952  Contender, The (2000)                Drama|Thriller      23
999822      3952  Contender, The (2000)                Drama|Thriller      36
999823      3952  Contender, The (2000)                Drama|Thriller      52
999824      3952  Contender, The (2000)                Drama|Thriller      72
999825      3952  Contender, The (2000)                Drama|Thriller     102

        Rating    Timestamp
0            5    978824268
1            4    978237008
2            4    978233496
3            5    978225952
4            5    978226474
...        ...          ...
999821       4    978461000
999822       5    978062904
999823       4    977947102
999824       5    977868330
999825       3   1039274093

[17678 rows x 6 columns]
```

[17]:
```
toy_story= group.get_group('Toy Story (1995)')
```

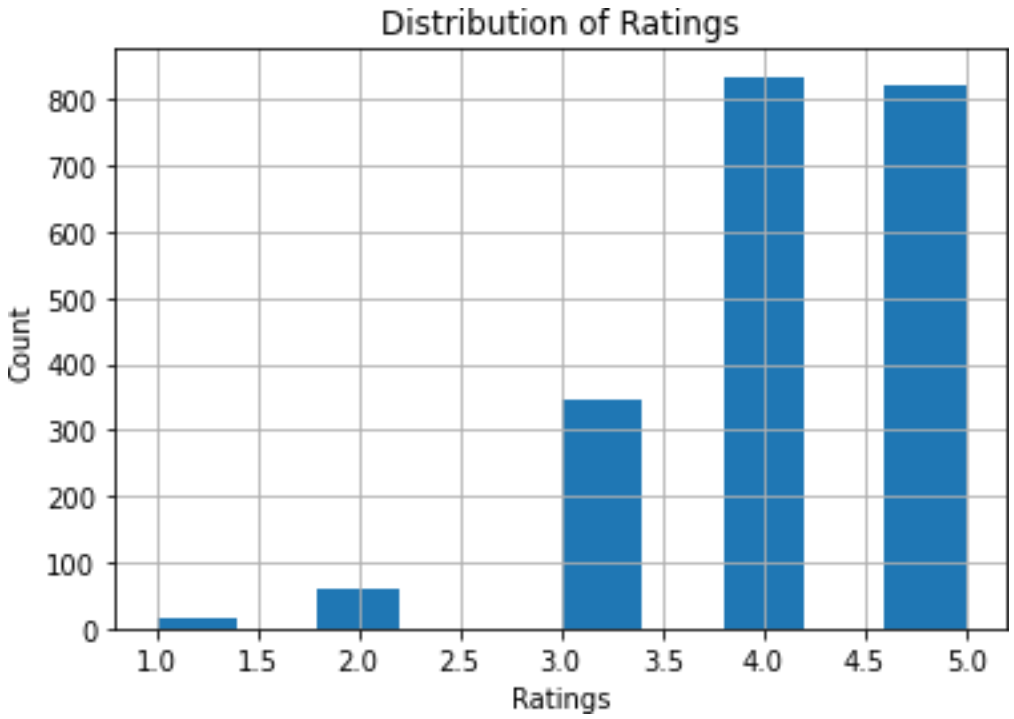
[18]:
```
toy_story.head()
```

[18]:
```
   MovieID              Title                        Genres  UserID  Rating  \
0        1  Toy Story (1995)  Animation|Children's|Comedy       1       5
1        1  Toy Story (1995)  Animation|Children's|Comedy       6       4
2        1  Toy Story (1995)  Animation|Children's|Comedy       8       4
3        1  Toy Story (1995)  Animation|Children's|Comedy       9       5
4        1  Toy Story (1995)  Animation|Children's|Comedy      10       5

   Timestamp
0  978824268
1  978237008
2  978233496
3  978225952
4  978226474
```

[19]:
```
toy_story['Rating'].hist()
plt.xlabel('Ratings')
plt.title('Distribution of Ratings')
```

```
plt.ylabel('Count')
plt.show()
```




[20]:
```
#3.          Top 25 movies by viewership rating
average_rating = First_merge.groupby('Title')['Rating'].mean()
average_rating.head(2)
```

[20] :
```
Title
$1,000,000 Duck (1971)     3.027027
'Night Mother (1986)       3.371429
Name: Rating, dtype: float64
```

[21]:
```
#to find movies with highest ratings
average_rating = average_rating.sort_values(ascending=False)
average_rating.head(25)
```

[21] :
```
Title
Ulysses (Ulisse) (1954)                5.000000
Lured (1947)                           5.000000
Follow the Bitch (1998)                5.000000
Bittersweet Motel (2000)               5.000000
Song of Freedom (1936)                 5.000000
One Little Indian (1973)               5.000000
```

```
Smashing Time (1967)                                                    5.000000
Schlafes Bruder (Brother of Sleep) (1995)                               5.000000
Gate of Heavenly Peace, The (1995)                                      5.000000
Baby, The (1973)                                                        5.000000
I Am Cuba (Soy Cuba/Ya Kuba) (1964)                                     4.800000
Lamerica (1994)                                                         4.750000
Apple, The (Sib) (1998)                                                 4.666667
Sanjuro (1962)                                                          4.608696
Seven Samurai (The Magnificent Seven) (Shichinin no samurai) (1954)     4.560510
Shawshank Redemption, The (1994)                                        4.554558
Godfather, The (1972)                                                   4.524966
Close Shave, A (1995)                                                   4.520548
Usual Suspects, The (1995)                                              4.517106
Schindler's List (1993)                                                 4.510417
Wrong Trousers, The (1993)                                              4.507937
Dry Cleaning (Nettoyage  sec) (1997)                                    4.500000
Inheritors, The (Die Siebtelbauern) (1998)                              4.500000
Mamma Roma (1962)                                                       4.500000
Bells, The (1926)                                                       4.500000
Name: Rating, dtype: float64
```

[22]:
```
#4.         Find the ratings for all the movies reviewed by for a particular
↪user of user id = 2696
rattings_all_2696 = second_merge[second_merge["UserID"]==2696]
```

[23]:
```
rattings_all_2696
```

[23]:

| | UserID | Gender | Age | Occupation | zip-code | MovieID | \ |
|---|---|---|---|---|---|---|---|
| 440667 | 2696 | M | 25 | 7 | 24210 | 350 | |
| 440668 | 2696 | M | 25 | 7 | 24210 | 800 | |
| 440669 | 2696 | M | 25 | 7 | 24210 | 1092 | |
| 440670 | 2696 | M | 25 | 7 | 24210 | 1097 | |
| 440671 | 2696 | M | 25 | 7 | 24210 | 1258 | |
| 440672 | 2696 | M | 25 | 7 | 24210 | 1270 | |
| 440673 | 2696 | M | 25 | 7 | 24210 | 1589 | |
| 440674 | 2696 | M | 25 | 7 | 24210 | 1617 | |
| 440675 | 2696 | M | 25 | 7 | 24210 | 1625 | |
| 440676 | 2696 | M | 25 | 7 | 24210 | 1644 | |
| 440677 | 2696 | M | 25 | 7 | 24210 | 1645 | |
| 440678 | 2696 | M | 25 | 7 | 24210 | 1711 | |
| 440679 | 2696 | M | 25 | 7 | 24210 | 1783 | |
| 440680 | 2696 | M | 25 | 7 | 24210 | 1805 | |
| 440681 | 2696 | M | 25 | 7 | 24210 | 1892 | |
| 440682 | 2696 | M | 25 | 7 | 24210 | 2338 | |
| 440683 | 2696 | M | 25 | 7 | 24210 | 2389 | |
| 440684 | 2696 | M | 25 | 7 | 24210 | 2713 | |
| 440685 | 2696 | M | 25 | 7 | 24210 | 3176 | |

```
440686   2696      M   25           7    24210     3386

                                                 Title  \
440667                                 Client, The (1994)
440668                                   Lone Star (1996)
440669                                Basic Instinct (1992)
440670                  E.T. the Extra-Terrestrial (1982)
440671                                 Shining, The (1980)
440672                          Back to the Future (1985)
440673                                     Cop Land (1997)
440674                                L.A. Confidential (1997)
440675                                     Game, The (1997)
440676             I Know What You Did Last Summer (1997)
440677                          Devil's Advocate, The (1997)
440678  Midnight in the Garden of Good and Evil (1997)
440679                                     Palmetto (1998)
440680                                   Wild Things (1998)
440681                              Perfect Murder, A (1998)
440682     I Still Know What You Did Last Summer (1998)
440683                                       Psycho (1998)
440684                                   Lake Placid (1999)
440685                    Talented Mr. Ripley, The (1999)
440686                                          JFK (1991)

                                    Genres  Rating  Timestamp
440667                 Drama|Mystery|Thriller       3  973308886
440668                          Drama|Mystery       5  973308842
440669                       Mystery|Thriller       4  973308886
440670  Children's|Drama|Fantasy|Sci-Fi       3  973308690
440671                                Horror       4  973308710
440672                        Comedy|Sci-Fi       2  973308676
440673                   Crime|Drama|Mystery       3  973308865
440674  Crime|Film-Noir|Mystery|Thriller       4  973308842
440675                       Mystery|Thriller       4  973308842
440676                 Horror|Mystery|Thriller       2  973308920
440677          Crime|Horror|Mystery|Thriller       4  973308904
440678            Comedy|Crime|Drama|Mystery       4  973308904
440679             Film-Noir|Mystery|Thriller       4  973308865
440680          Crime|Drama|Mystery|Thriller       4  973308886
440681                       Mystery|Thriller       4  973308904
440682                 Horror|Mystery|Thriller       2  973308920
440683                   Crime|Horror|Thriller       4  973308710
440684                         Horror|Thriller       1  973308710
440685                 Drama|Mystery|Thriller       4  973308865
440686                          Drama|Mystery       1  973308842
```

[24]:
```
#1.          Find out all the unique genres
#(Hint: split the data in column genre making a list and then process the data
 ↪to find out
#only the unique categories of genres)
First_merge["Genres"].value_counts().head(10)
```

[24]:
```
Comedy                      116883
Drama                       111423
Comedy|Romance               42712
Comedy|Drama                 42245
Drama|Romance                29170
Action|Thriller              26759
Horror                       22563
Drama|Thriller               18248
Thriller                     17851
Action|Adventure|Sci-Fi      17783
Name: Genres, dtype: int64
```

[25]:
```
First_merge["Genres"].unique()
```

[25]:
```
array(["Animation|Children's|Comedy", "Adventure|Children's|Fantasy",
       'Comedy|Romance', 'Comedy|Drama', 'Comedy',
       'Action|Crime|Thriller', "Adventure|Children's", 'Action',
       'Action|Adventure|Thriller', 'Comedy|Drama|Romance',
       'Comedy|Horror', "Animation|Children's", 'Drama',
       'Action|Adventure|Romance', 'Drama|Thriller', 'Drama|Romance',
       'Thriller', 'Action|Comedy|Drama', 'Crime|Drama|Thriller',
       'Drama|Sci-Fi', 'Romance', 'Adventure|Sci-Fi', 'Adventure|Romance',
       "Children's|Comedy|Drama", 'Documentary', 'Drama|War',
       'Action|Crime|Drama', 'Action|Adventure', 'Crime|Thriller',
       "Animation|Children's|Musical|Romance", "Children's|Comedy",
       'Drama|Mystery', 'Sci-Fi|Thriller',
       'Action|Comedy|Crime|Horror|Thriller', 'Drama|Musical',
       'Crime|Drama|Romance', 'Adventure|Drama', 'Action|Thriller',
       "Adventure|Children's|Comedy|Musical", 'Action|Drama|War',
       'Action|Adventure|Crime', 'Crime', 'Drama|Mystery|Romance',
       'Action|Drama', 'Drama|Romance|War', 'Horror',
       'Action|Adventure|Comedy|Crime', 'Comedy|War',
       'Action|Adventure|Mystery|Sci-Fi', 'Drama|Thriller|War',
       'Action|Romance|Thriller', 'Crime|Film-Noir|Mystery|Thriller',
       'Action|Adventure|Drama|Romance', "Adventure|Children's|Drama",
       'Action|Sci-Fi|Thriller', 'Action|Adventure|Sci-Fi',
       "Action|Children's", 'Horror|Sci-Fi', 'Action|Crime|Sci-Fi',
       'Western', "Animation|Children's|Comedy|Romance",
       "Children's|Drama", 'Crime|Drama',
       'Drama|Fantasy|Romance|Thriller', 'Drama|Horror', 'Comedy|Sci-Fi',
       'Mystery|Thriller', "Adventure|Children's|Comedy|Fantasy|Romance",
```

'Action|Adventure|Fantasy|Sci-Fi', 'Drama|Romance|War|Western',
'Action|Drama|Thriller', 'Crime|Drama|Romance|Thriller',
'Action|Adventure|Western', 'Horror|Thriller',
"Children's|Comedy|Fantasy", 'Film-Noir|Thriller',
'Action|Comedy|Musical|Sci-Fi', "Children's",
'Drama|Mystery|Thriller', 'Comedy|Romance|War', 'Action|Comedy',
"Adventure|Children's|Romance", "Animation|Children's|Musical",
'Comedy|Crime|Fantasy', 'Action|Comedy|Western', 'Action|Sci-Fi',
'Action|Adventure|Comedy|Romance', 'Comedy|Thriller',
'Horror|Sci-Fi|Thriller', 'Mystery|Romance|Thriller',
'Comedy|Western', 'Drama|Western',
'Action|Adventure|Crime|Thriller', 'Action|Comedy|War',
'Comedy|Mystery', 'Comedy|Mystery|Romance', 'Comedy|Drama|War',
'Action|Drama|Mystery', 'Comedy|Crime|Horror', 'Film-Noir|Sci-Fi',
'Comedy|Romance|Thriller', "Action|Adventure|Children's|Sci-Fi",
"Children's|Comedy|Musical", 'Action|Adventure|Comedy',
'Action|Crime|Romance',
"Action|Adventure|Animation|Children's|Fantasy",
"Animation|Children's|Comedy|Musical", 'Adventure|Drama|Western',
'Action|Adventure|Crime|Drama',
'Action|Adventure|Animation|Horror|Sci-Fi', 'Action|Horror|Sci-Fi',
'War', "Action|Adventure|Mystery', 'Mystery',
'Action|Adventure|Fantasy',
"Adventure|Animation|Children's|Comedy|Fantasy", 'Sci-Fi',
'Documentary|Drama', 'Action|Adventure|Comedy|War',
'Crime|Film-Noir|Thriller', 'Animation',
'Action|Adventure|Romance|Thriller', 'Animation|Sci-Fi',
'Animation|Comedy|Thriller', 'Film-Noir', 'Sci-Fi|War',
'Adventure', 'Comedy|Crime', 'Action|Sci-Fi|War',
'Comedy|Fantasy|Romance|Sci-Fi', 'Fantasy',
'Action|Mystery|Thriller', 'Comedy|Musical',
'Action|Adventure|Sci-Fi|Thriller', "Children's|Drama|Fantasy",
'Adventure|War', 'Musical|Romance', 'Comedy|Musical|Romance',
'Comedy|Mystery|Romance|Thriller', 'Film-Noir|Mystery', 'Musical',
"Adventure|Children's|Drama|Musical",
'Drama|Mystery|Sci-Fi|Thriller', 'Romance|Thriller',
'Film-Noir|Romance|Thriller', 'Crime|Film-Noir|Mystery',
'Adventure|Comedy', 'Action|Adventure|Romance|War', 'Romance|War',
'Action|Drama|Western', 'Action|Crime',
"Children's|Comedy|Western", "Adventure|Children's|Comedy",
"Children's|Comedy|Mystery", "Adventure|Children's|Fantasy|Sci-Fi",
"Adventure|Animation|Children's|Musical",
"Adventure|Children's|Musical", 'Crime|Film-Noir',
"Adventure|Children's|Comedy|Fantasy",
"Children's|Drama|Fantasy|Sci-Fi", 'Action|Romance',
'Adventure|Western', 'Comedy|Fantasy', 'Animation|Comedy',
'Crime|Drama|Film-Noir', 'Action|Adventure|Drama|Sci-Fi|War',

'Action|Sci-Fi|Thriller|War', 'Action|Western',
"Action|Animation|Children's|Sci-Fi|Thriller|War",
'Action|Adventure|Romance|Sci-Fi|War',
'Action|Horror|Sci-Fi|Thriller',
'Action|Adventure|Comedy|Horror|Sci-Fi', 'Action|Comedy|Musical',
'Mystery|Sci-Fi', 'Film-Noir|Mystery|Thriller',
'Adventure|Comedy|Drama', 'Action|Adventure|Comedy|Horror',
'Action|Drama|Mystery|Romance|Thriller', 'Comedy|Mystery|Thriller',
'Adventure|Animation|Sci-Fi|Thriller', 'Action|Drama|Romance',
'Action|Adventure|Drama', 'Comedy|Drama|Musical',
'Documentary|War', 'Drama|Musical|War', 'Action|Horror',
'Horror|Romance', 'Action|Comedy|Sci-Fi|War', 'Crime|Drama|Sci-Fi',
'Action|Romance|War', 'Action|Comedy|Crime|Drama',
'Action|Drama|Thriller|War', "Action|Adventure|Children's",
"Action|Adventure|Children's|Fantasy",
"Adventure|Animation|Children's|Comedy|Musical",
'Action|Adventure|Comedy|Sci-Fi', "Children's|Fantasy",
'Crime|Drama|Mystery', 'Action|Mystery|Sci-Fi|Thriller',
'Action|Mystery|Romance|Thriller', 'Adventure|Thriller',
'Action|Thriller|War', 'Action|Crime|Mystery',
'Horror|Mystery|Thriller', 'Crime|Horror|Mystery|Thriller',
'Comedy|Drama|Thriller', 'Drama|Sci-Fi|Thriller',
'Drama|Romance|Thriller', 'Action|Adventure|Sci-Fi|War',
'Comedy|Crime|Drama|Mystery', 'Comedy|Crime|Mystery|Thriller',
'Film-Noir|Sci-Fi|Thriller', 'Adventure|Sci-Fi|Thriller',
'Crime|Drama|Mystery|Thriller', 'Comedy|Crime|Drama',
'Comedy|Documentary', 'Documentary|Musical',
'Action|Drama|Sci-Fi|Thriller',
"Adventure|Animation|Children's|Fantasy",
'Adventure|Comedy|Romance', 'Mystery|Sci-Fi|Thriller',
'Action|Comedy|Crime', "Animation|Children's|Fantasy|War",
'Action|Crime|Drama|Thriller', 'Comedy|Sci-Fi|Western',
"Children's|Fantasy|Musical", 'Fantasy|Sci-Fi',
"Children's|Comedy|Sci-Fi", "Action|Adventure|Children's|Comedy",
"Adventure|Children's|Drama|Romance",
"Adventure|Children's|Sci-Fi",
"Adventure|Children's|Comedy|Fantasy|Sci-Fi",
"Animation|Children's|Comedy|Musical|Romance",
"Children's|Musical", 'Drama|Fantasy',
"Animation|Children's|Fantasy|Musical", 'Adventure|Comedy|Musical',
"Children's|Sci-Fi", "Children's|Horror", 'Comedy|Fantasy|Romance',
'Comedy|Crime|Thriller', "Adventure|Animation|Children's|Sci-Fi",
'Action|Crime|Mystery|Thriller', 'Adventure|Musical',
"Animation|Children's|Drama|Fantasy", "Children's|Fantasy|Sci-Fi",
'Adventure|Fantasy|Romance', 'Crime|Horror',
'Action|Adventure|Horror', 'Adventure|Fantasy|Sci-Fi',
'Drama|Film-Noir|Thriller', 'Action|Comedy|Fantasy',

'Sci-Fi|Thriller|War', 'Action|Adventure|Sci-Fi|Thriller|War',
'Action|Adventure|Drama|Thriller', 'Crime|Horror|Thriller',
'Animation|Musical', 'Action|War',
'Action|Comedy|Romance|Thriller', 'Comedy|Horror|Thriller',
'Drama|Horror|Thriller', 'Action|Sci-Fi|Thriller|Western',
'Drama|Romance|Sci-Fi', 'Action|Adventure|Horror|Thriller',
'Comedy|Film-Noir|Thriller', 'Comedy|Horror|Musical|Sci-Fi',
'Comedy|Romance|Sci-Fi', 'Action|Comedy|Sci-Fi|Thriller',
'Action|Sci-Fi|Western', 'Comedy|Horror|Musical', 'Crime|Mystery',
'Animation|Mystery', 'Action|Horror|Thriller',
'Action|Drama|Fantasy|Romance', 'Horror|Mystery',
"Adventure|Animation|Children's", 'Musical|Romance|War',
'Adventure|Drama|Romance', 'Adventure|Animation|Film-Noir',
'Action|Adventure|Animation', 'Comedy|Drama|Western',
'Adventure|Comedy|Sci-Fi', 'Drama|Romance|Western',
'Comedy|Drama|Sci-Fi', 'Action|Drama|Romance|Thriller',
'Adventure|Romance|Sci-Fi', 'Film-Noir|Horror',
'Crime|Drama|Film-Noir|Thriller', 'Action|Adventure|War',
'Romance|Western', "Action|Children's|Fantasy",
'Adventure|Drama|Thriller', 'Adventure|Fantasy', 'Musical|War',
'Adventure|Musical|Romance', 'Action|Romance|Sci-Fi',
'Drama|Film-Noir', 'Comedy|Horror|Sci-Fi',
'Adventure|Drama|Romance|Sci-Fi', 'Adventure|Animation|Sci-Fi',
'Adventure|Crime|Sci-Fi|Thriller'], dtype=object)

[26]:
```
#2.        Create a separate column for each genre category with a one-hot
 ↪encoding ( 1 and 0)
#whether or not the movie belongs to that genre.
genrecol =First_merge["Genres"]
genre= genrecol.str.get_dummies().add_prefix("NEW")
```

[27]:
```
genrecol.head()
```

[27]:
```
0    Animation|Children's|Comedy
1    Animation|Children's|Comedy
2    Animation|Children's|Comedy
3    Animation|Children's|Comedy
4    Animation|Children's|Comedy
Name: Genres, dtype: object
```

[28]:
```
First_merge.head()
```

[28]:

| | MovieID | Title | Genres | UserID | Rating | \ |
|---|---|---|---|---|---|---|
| 0 | 1 | Toy Story (1995) | Animation\|Children's\|Comedy | 1 | 5 | |
| 1 | 1 | Toy Story (1995) | Animation\|Children's\|Comedy | 6 | 4 | |
| 2 | 1 | Toy Story (1995) | Animation\|Children's\|Comedy | 8 | 4 | |
| 3 | 1 | Toy Story (1995) | Animation\|Children's\|Comedy | 9 | 5 | |

```
4         1  Toy Story (1995)  Animation|Children's|Comedy      10        5

   Timestamp
0  978824268
1  978237008
2  978233496
3  978225952
4  978226474
```

[29]:
```
final_data = pd.concat([First_merge,genre],axis=1)
```

[30]:
```
final_data.head()
```

[30]:
```
   MovieID             Title                        Genres  UserID  Rating  \
0        1  Toy Story (1995)  Animation|Children's|Comedy       1       5
1        1  Toy Story (1995)  Animation|Children's|Comedy       6       4
2        1  Toy Story (1995)  Animation|Children's|Comedy       8       4
3        1  Toy Story (1995)  Animation|Children's|Comedy       9       5
4        1  Toy Story (1995)  Animation|Children's|Comedy      10       5

   Timestamp  NEWAction  NEWAdventure  NEWAnimation  NEWChildren's  ...  \
0  978824268          0             0             1              1  ...
1  978237008          0             0             1              1  ...
2  978233496          0             0             1              1  ...
3  978225952          0             0             1              1  ...
4  978226474          0             0             1              1  ...

   NEWFantasy  NEWFilm-Noir  NEWHorror  NEWMusical  NEWMystery  NEWRomance  \
0           0             0          0           0           0           0
1           0             0          0           0           0           0
2           0             0          0           0           0           0
3           0             0          0           0           0           0
4           0             0          0           0           0           0

   NEWSci-Fi  NEWThriller  NEWWar  NEWWestern
0          0            0       0           0
1          0            0       0           0
2          0            0       0           0
3          0            0       0           0
4          0            0       0           0

[5 rows x 24 columns]
```

[31]:
```
final_data.columns
```

[31]:
```
Index(['MovieID', 'Title', 'Genres', 'UserID', 'Rating', 'Timestamp',
       'NEWAction', 'NEWAdventure', 'NEWAnimation', 'NEWChildren's',
```

```
       'NEWComedy', 'NEWCrime', 'NEWDocumentary', 'NEWDrama', 'NEWFantasy',
       'NEWFilm-Noir', 'NEWHorror', 'NEWMusical', 'NEWMystery', 'NEWRomance',
       'NEWSci-Fi', 'NEWThriller', 'NEWWar', 'NEWWestern'],
      dtype='object')
```

[32]:
```
final_data.to_csv('final_data.csv')
```

[56]:
```
master_data.columns
```

[56]:
```
Index(['UserID', 'Gender', 'Age', 'Occupation', 'MovieID', 'Title', 'Genres',
       'Rating'],
      dtype='object')
```

[72]:
```
#4.Develop an appropriate model to predict the movie ratings
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
```

[73]:
```
X = final_data.
↪drop(['Rating','Title','MovieID','UserID','Timestamp','Genres'],axis=1)
y= final_data['Rating']
```

[74]:
```
X.columns
```

[74]:
```
Index(['NEWAction', 'NEWAdventure', 'NEWAnimation', 'NEWChildren's',
       'NEWComedy', 'NEWCrime', 'NEWDocumentary', 'NEWDrama', 'NEWFantasy',
       'NEWFilm-Noir', 'NEWHorror', 'NEWMusical', 'NEWMystery', 'NEWRomance',
       'NEWSci-Fi', 'NEWThriller', 'NEWWar', 'NEWWestern'],
      dtype='object')
```

[75]:
```
X_train,X_test,y_train,y_test = train_test_split(X,y,test_size=0.
↪20,random_state=0)
```

[76]:
```
linear_reg=LinearRegression()
```

[77]:
```
linear_reg.fit(X_train,y_train)
```

[77]:
```
LinearRegression()
```

[78]:
```
#predictions
y_pred= linear_reg.predict(X_test)
```

[79]:
```
#to find the most important features
linear_reg.coef_
```

[79]:
```
array([-0.10259894,  0.01484072,  0.36292453, -0.3215857 , -0.01624118,
        0.09633733,  0.40940027,  0.23650902,  0.06420402,  0.4467839 ,
       -0.28896811,  0.16319447,  0.01412358, -0.00614819, -0.02582426,
```

```
0.05733083,  0.29962257,  0.12561833])
```

[ ]:

[71]:
```
from sklearn.metrics import r2_score
r2 = r2_score(y_test,y_pred)
print(r2)
```

```
0.036632192843985
```

[52]:
```
#3.          Determine the features affecting the ratings of any particular movie.
data=pd.DataFrame(master_data)
```

[54]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1000209 entries, 0 to 1000208
Data columns (total 8 columns):
 #   Column      Non-Null Count    Dtype 
---  ------      --------------    ----- 
 0   UserID      1000209 non-null  int64 
 1   Gender      1000209 non-null  object
 2   Age         1000209 non-null  int64 
 3   Occupation  1000209 non-null  int64 
 4   MovieID     1000209 non-null  int64 
 5   Title       1000209 non-null  object
 6   Genres      1000209 non-null  object
 7   Rating      1000209 non-null  int64 
dtypes: int64(5), object(3)
memory usage: 68.7+ MB
```

[55]:
```
print(data.corr()) #pearson correlation coefficient
```

| | UserID | Age | Occupation | MovieID | Rating |
|---|---|---|---|---|---|
| UserID | 1.000000 | 0.034688 | -0.026698 | -0.017739 | 0.012303 |
| Age | 0.034688 | 1.000000 | 0.078371 | 0.027575 | 0.056869 |
| Occupation | -0.026698 | 0.078371 | 1.000000 | 0.008585 | 0.006753 |
| MovieID | -0.017739 | 0.027575 | 0.008585 | 1.000000 | -0.064042 |
| Rating | 0.012303 | 0.056869 | 0.006753 | -0.064042 | 1.000000 |

[ ]: