In [161…
```
#import basic libraries
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

Import the three datasets

In [162…
```
#LOADING FIRST DATA
df_user=pd.read_csv('users.dat',sep="::",names=['UserID','Gender','Age','Occupatio
```

In [163…
```
df_user
```

Out[163]:

	UserID	Gender	Age	Occupation	Zip Code
0	1	F	1	10	48067
1	2	M	56	16	70072
2	3	M	25	15	55117
3	4	M	45	7	02460
4	5	M	25	20	55455
...	...	...	...	...	...
6035	6036	F	25	15	32603
6036	6037	F	45	1	76006
6037	6038	F	56	1	14706
6038	6039	F	45	0	01060
6039	6040	M	25	6	11106

6040 rows × 5 columns

In [164…
```
#LOADING SECOND DATA
df_movies=pd.read_csv('movies.dat',sep="::",names=['MovieID','Title','Generes'],en
```

In [165…
```
df_movies
```

Out[165]:

	MovieID	Title	Generes
0	1	Toy Story (1995)	Animation|Children's|Comedy
1	2	Jumanji (1995)	Adventure|Children's|Fantasy
2	3	Grumpier Old Men (1995)	Comedy|Romance
3	4	Waiting to Exhale (1995)	Comedy|Drama
4	5	Father of the Bride Part II (1995)	Comedy
...	...	...	...
3878	3948	Meet the Parents (2000)	Comedy
3879	3949	Requiem for a Dream (2000)	Drama
3880	3950	Tigerland (2000)	Drama
3881	3951	Two Family House (2000)	Drama
3882	3952	Contender, The (2000)	Drama|Thriller

3883 rows × 3 columns

In [166…

```
#LOADING 3RD DATA
df_ratings=pd.read_csv('ratings.dat',sep="::",names=['UserID','MovieID','Rating','
df_ratings
```

Out[166]:

	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 × 4 columns

In [167…

```
df_user.shape
```

Out[167]:

```
(6040, 5)
```

In [168…

```
df_movies.shape
```

Out[168]:

```
(3883, 3)
```

In [169…

```
df_ratings.shape
```

Out[169]: (1000209, 4)

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)

In [170…
```
dfMovieRatings=df_movies.merge(df_ratings,on='MovieID',how='inner')
```

In [171…
```
dfMovieRatings
```

Out[171]:

	MovieID	Title	Generes	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
...	...	...	...	...	...	...
1000204	3952	Contender, The (2000)	Drama|Thriller	5812	4	992072099
1000205	3952	Contender, The (2000)	Drama|Thriller	5831	3	986223125
1000206	3952	Contender, The (2000)	Drama|Thriller	5837	4	1011902656
1000207	3952	Contender, The (2000)	Drama|Thriller	5927	1	979852537
1000208	3952	Contender, The (2000)	Drama|Thriller	5998	4	1001781044

1000209 rows × 6 columns

In [172…
```
dfMaster=dfMovieRatings.merge(df_user,on='UserID',how='inner')
```

In [173…
```
dfMaster
```

Out[173]:

	MovieID	Title	Generes	UserID	Rating	Timestamp
0	1	Toy Story (1995)	Animation|Children's|Comedy	1	5	978824268
1	48	Pocahontas (1995)	Animation|Children's|Musical|Romance	1	5	978824351
2	150	Apollo 13 (1995)	Drama	1	5	978301777
3	260	Star Wars: Episode IV - A New Hope (1977)	Action|Adventure|Fantasy|Sci-Fi	1	4	978300760
4	527	Schindler's List (1993)	Drama|War	1	5	978824195
...	...	...	...	...	...	...
1000204	3513	Rules of Engagement (2000)	Drama|Thriller	5727	4	958489970
1000205	3535	American Psycho (2000)	Comedy|Horror|Thriller	5727	2	958489970
1000206	3536	Keeping the Faith (2000)	Comedy|Romance	5727	5	958489902
1000207	3555	U-571 (2000)	Action|Thriller	5727	3	958490699
1000208	3578	Gladiator (2000)	Action|Drama	5727	5	958490171

1000209 rows × 10 columns

In [174…
```
#saving the file to csv
dfMaster.to_csv('Master Data.csv')
```

In [175…
```
dfMaster.isnull().sum().any()
```

Out[175]:
```
False
```

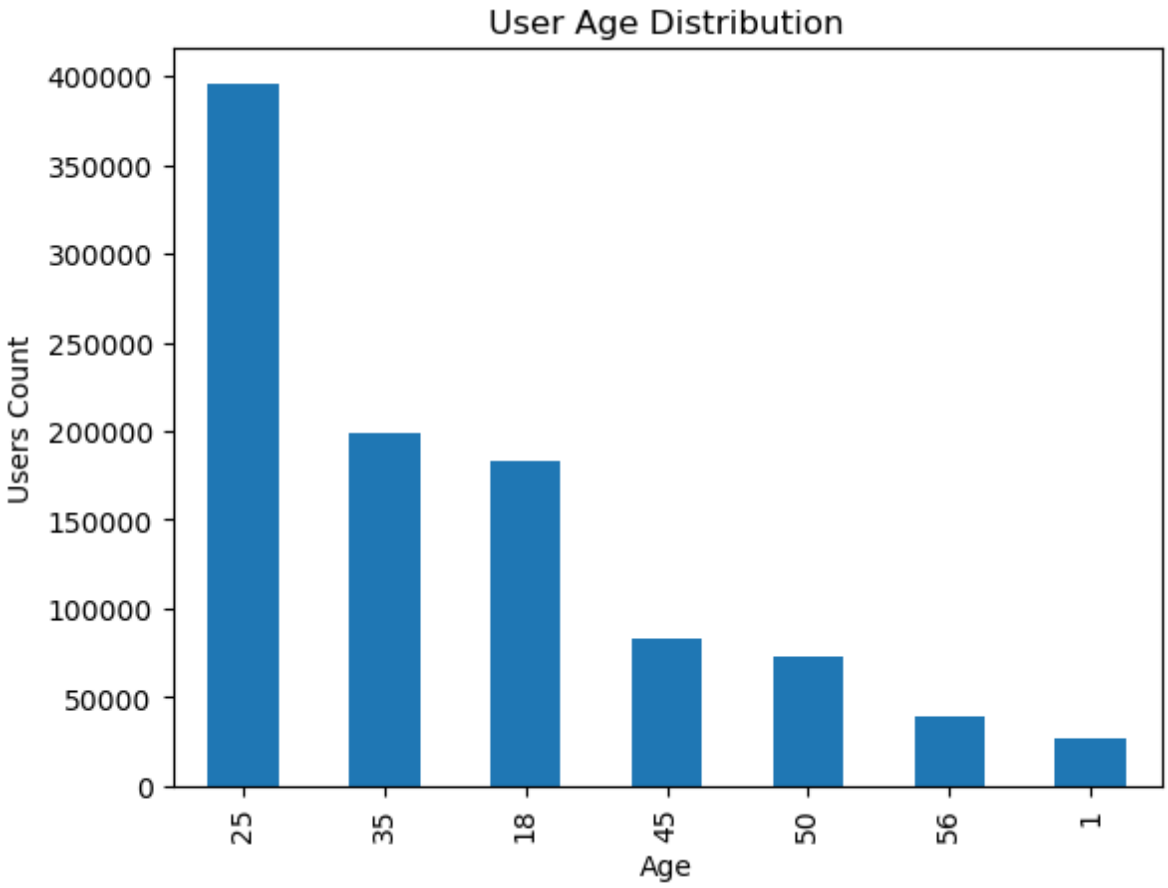
Explore the datasets using visual representations (graphs or tables), also include your comments on the following:

User Age Distribution

In [176…
```
# WE CAN PLOT THE CHARTS BY VALUE_COUNTS
dfMaster['Age'].value_counts().plot(kind='bar')
plt.title('User Age Distribution')
plt.ylabel('Users Count')
```

```
plt.xlabel('Age')
plt.show()

#conclusion = we can see user count mostly belongs to 18-25 age group
```



User rating of the movie “Toy Story”

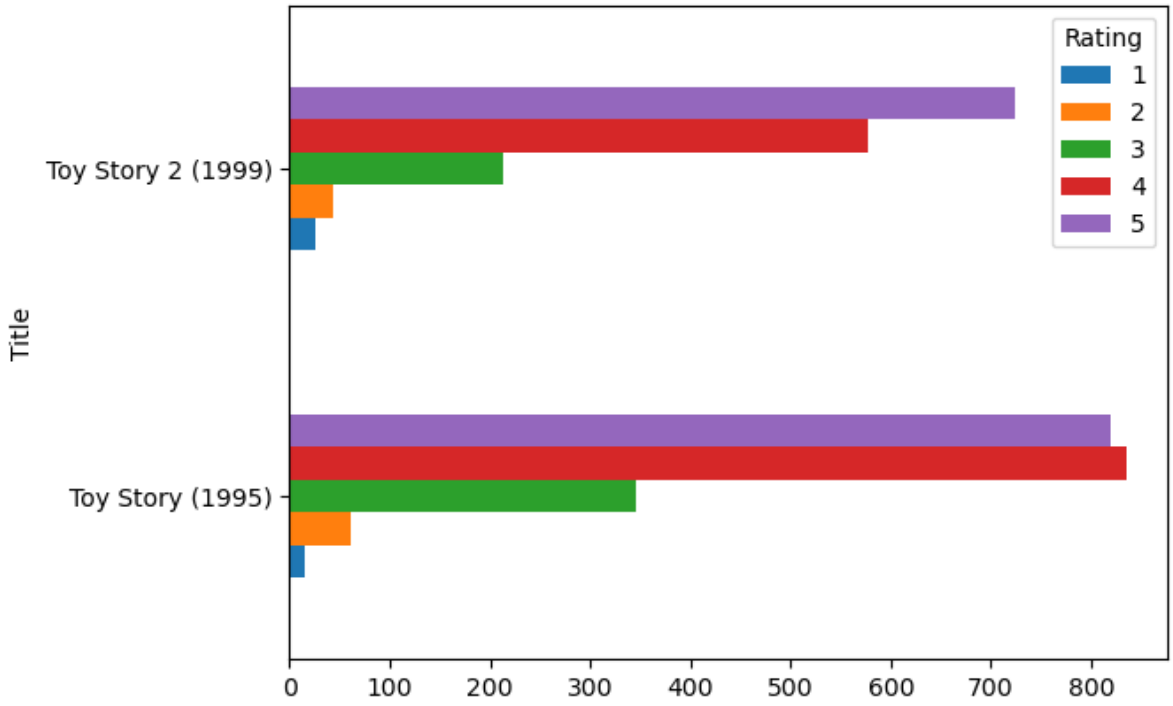
In [177…
```
toystory=dfMaster[dfMaster['Title'].str.contains('Toy Story')==True]
```

In [178…
```
toystory.groupby(['Title','Rating']).size()
```

Out[178]:
```
Title               Rating
Toy Story (1995)    1          16
                    2          61
                    3         345
                    4         835
                    5         820
Toy Story 2 (1999)  1          25
                    2          44
                    3         214
                    4         578
                    5         724
dtype: int64
```

In [179…
```
toystory.groupby(['Title','Rating']).size().unstack().plot(kind='barh')
```

Out[179]:
```
<AxesSubplot:ylabel='Title'>
```



Top 25 movies by viewership rating

In [180…
```
dfMovieRatings
```

Out[180]:

	MovieID	Title	Generes	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
...	...	...	...	...	...	...
1000204	3952	Contender, The (2000)	Drama|Thriller	5812	4	992072099
1000205	3952	Contender, The (2000)	Drama|Thriller	5831	3	986223125
1000206	3952	Contender, The (2000)	Drama|Thriller	5837	4	1011902656
1000207	3952	Contender, The (2000)	Drama|Thriller	5927	1	979852537
1000208	3952	Contender, The (2000)	Drama|Thriller	5998	4	1001781044

1000209 rows × 6 columns

In [181…
```
#Top 25 movies by viewership rating
dfTop25=dfMaster.groupby('Title').size().sort_values(ascending=False)[:25]
dfTop25
```

Out[181]:

Title	
American Beauty (1999)	3428
Star Wars: Episode IV - A New Hope (1977)	2991
Star Wars: Episode V - The Empire Strikes Back (1980)	2990
Star Wars: Episode VI - Return of the Jedi (1983)	2883
Jurassic Park (1993)	2672
Saving Private Ryan (1998)	2653
Terminator 2: Judgment Day (1991)	2649
Matrix, The (1999)	2590
Back to the Future (1985)	2583
Silence of the Lambs, The (1991)	2578
Men in Black (1997)	2538
Raiders of the Lost Ark (1981)	2514
Fargo (1996)	2513
Sixth Sense, The (1999)	2459
Braveheart (1995)	2443
Shakespeare in Love (1998)	2369
Princess Bride, The (1987)	2318
Schindler's List (1993)	2304
L.A. Confidential (1997)	2288
Groundhog Day (1993)	2278
E.T. the Extra-Terrestrial (1982)	2269
Star Wars: Episode I - The Phantom Menace (1999)	2250
Being John Malkovich (1999)	2241
Shawshank Redemption, The (1994)	2227
Godfather, The (1972)	2223

dtype: int64

In [182…

```
userId = 2696
userRatingById = dfMaster[dfMaster["UserID"] == userId]
userRatingById
```

Out[182]:

	MovieID	Title	Generes	UserID	Rating	Timestamp	Gender
991035	350	Client, The (1994)	Drama|Mystery|Thriller	2696	3	973308886	M
991036	800	Lone Star (1996)	Drama|Mystery	2696	5	973308842	M
991037	1092	Basic Instinct (1992)	Mystery|Thriller	2696	4	973308886	M
991038	1097	E.T. the Extra-Terrestrial (1982)	Children's|Drama|Fantasy|Sci-Fi	2696	3	973308690	M
991039	1258	Shining, The (1980)	Horror	2696	4	973308710	M
991040	1270	Back to the Future (1985)	Comedy|Sci-Fi	2696	2	973308676	M
991041	1589	Cop Land (1997)	Crime|Drama|Mystery	2696	3	973308865	M
991042	1617	L.A. Confidential (1997)	Crime|Film-Noir|Mystery|Thriller	2696	4	973308842	M
991043	1625	Game, The (1997)	Mystery|Thriller	2696	4	973308842	M
991044	1644	I Know What You Did Last Summer (1997)	Horror|Mystery|Thriller	2696	2	973308920	M
991045	1645	Devil's Advocate, The (1997)	Crime|Horror|Mystery|Thriller	2696	4	973308904	M
991046	1711	Midnight in the Garden of Good and Evil (1997)	Comedy|Crime|Drama|Mystery	2696	4	973308904	M
991047	1783	Palmetto (1998)	Film-Noir|Mystery|Thriller	2696	4	973308865	M
991048	1805	Wild Things (1998)	Crime|Drama|Mystery|Thriller	2696	4	973308886	M
991049	1892	Perfect Murder, A (1998)	Mystery|Thriller	2696	4	973308904	M
991050	2338	I Still Know What You Did Last Summer (1998)	Horror|Mystery|Thriller	2696	2	973308920	M

	MovieID	Title	Generes	UserID	Rating	Timestamp	Gender
991051	2389	Psycho (1998)	Crime|Horror|Thriller	2696	4	973308710	M
991052	2713	Lake Placid (1999)	Horror|Thriller	2696	1	973308710	M
991053	3176	Talented Mr. Ripley, The (1999)	Drama|Mystery|Thriller	2696	4	973308865	M
991054	3386	JFK (1991)	Drama|Mystery	2696	1	973308842	M

Feature Engineering:

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)

In [183…
```
dfGeneres= dfMaster['Generes'].str.split('|')
```

In [184…
```
listgeneres=set() #constructor method
for genre in dfGeneres:
    listgeneres=listgeneres.union(set(genre))
```

In [185…
```
len(listgeneres)
```

Out[185]:
```
18
```

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.

In [186…
```
#non-splited element and doing encoding by column and row wise
GeneresOnehot=dfMaster['Generes'].str.get_dummies('|')
GeneresOnehot
```

Out[186]:

	Action	Adventure	Animation	Children's	Comedy	Crime	Documentary	Drama	Fant
0	0	0	1	1	1	0	0	0	
1	0	0	1	1	0	0	0	0	
2	0	0	0	0	0	0	0	1	
3	1	1	0	0	0	0	0	0	
4	0	0	0	0	0	0	0	1	
...	...	...	...	...	...	...	...	...	
1000204	0	0	0	0	0	0	0	1	
1000205	0	0	0	0	1	0	0	0	
1000206	0	0	0	0	1	0	0	0	
1000207	1	0	0	0	0	0	0	0	
1000208	1	0	0	0	0	0	0	1	

1000209 rows × 18 columns

In [187…

```
dfMaster=pd.concat([dfMaster,GeneresOnehot],axis=1)
dfMaster
```

Out[187]:

	MovieID	Title	Generes	UserID	Rating	Timestamp
0	1	Toy Story (1995)	Animation|Children's|Comedy	1	5	978824268
1	48	Pocahontas (1995)	Animation|Children's|Musical|Romance	1	5	978824351
2	150	Apollo 13 (1995)	Drama	1	5	978301777
3	260	Star Wars: Episode IV - A New Hope (1977)	Action|Adventure|Fantasy|Sci-Fi	1	4	978300760
4	527	Schindler's List (1993)	Drama|War	1	5	978824195
...	...	...	...	...	...	...
1000204	3513	Rules of Engagement (2000)	Drama|Thriller	5727	4	958489970
1000205	3535	American Psycho (2000)	Comedy|Horror|Thriller	5727	2	958489970
1000206	3536	Keeping the Faith (2000)	Comedy|Romance	5727	5	958489902
1000207	3555	U-571 (2000)	Action|Thriller	5727	3	958490699
1000208	3578	Gladiator (2000)	Action|Drama	5727	5	958490171

1000209 rows × 28 columns

In [188…
```
dfMaster['Gender']=dfMaster['Gender'].replace('M','0')
dfMaster['Gender']=dfMaster['Gender'].replace('F','1')
```

In [189…
```
dfMaster['Gender'].astype(int)
```

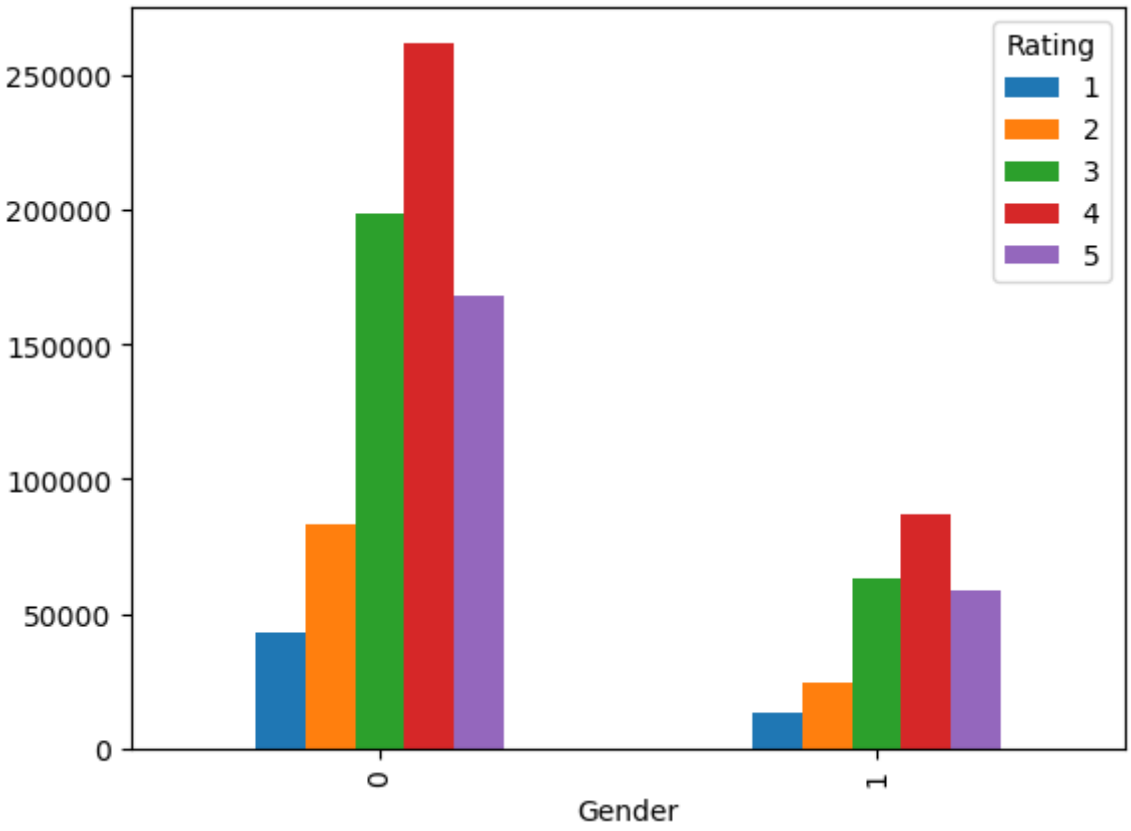
Out[189]:
```
0          1
1          1
2          1
3          1
4          1
          ..
1000204    0
1000205    0
1000206    0
1000207    0
1000208    0
Name: Gender, Length: 1000209, dtype: int32
```

In [190…
```
GenderAffecting=dfMaster.groupby('Gender').size().sort_values(ascending=False)
GenderAffecting
```

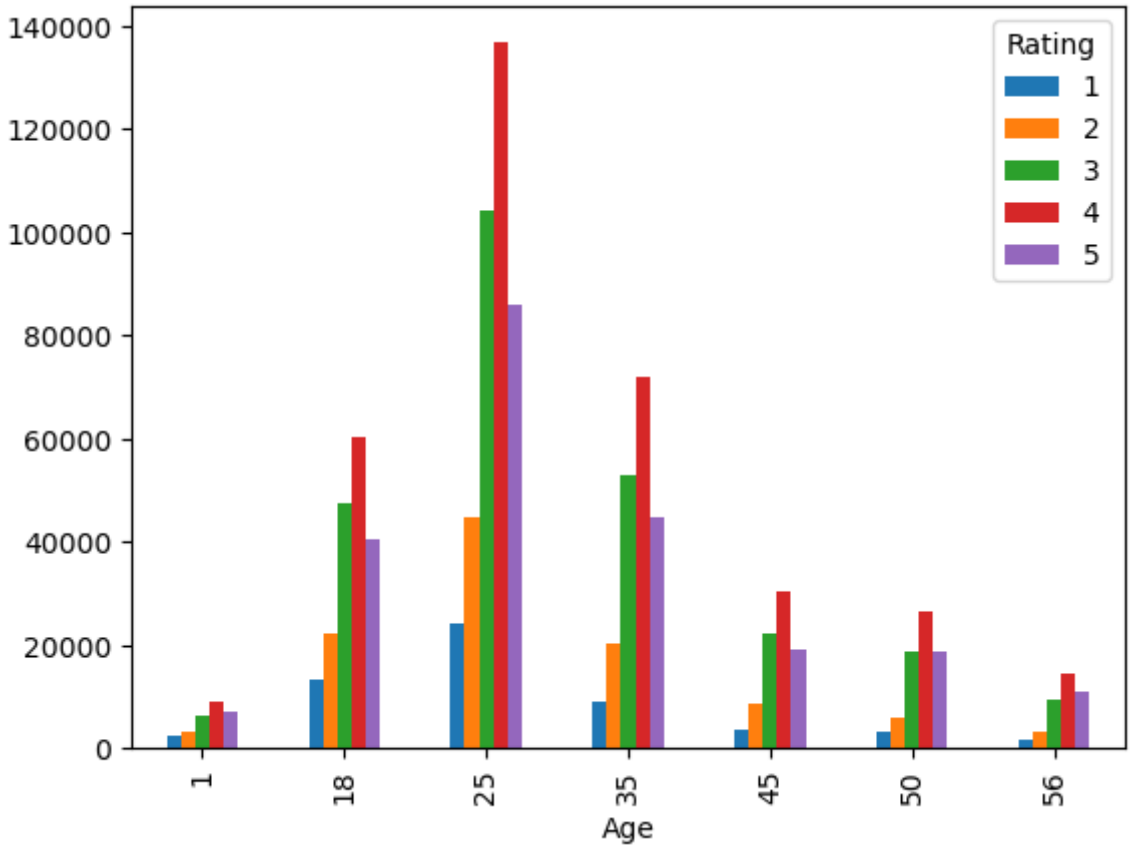
Out[190]:
```
Gender
0    753769
1    246440
dtype: int64
```

Determine the features affecting the ratings of any particular movie.

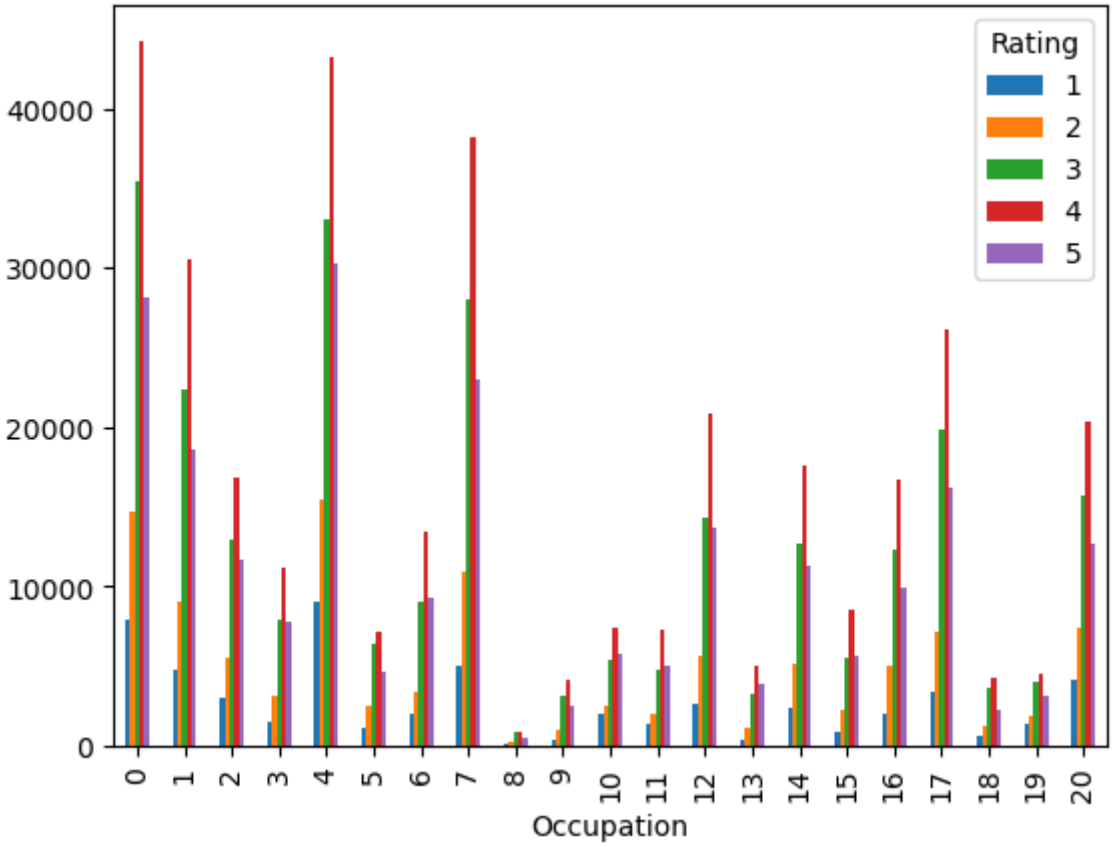
In [191…
```
#relation between rating and gender
dfMaster.groupby(['Gender','Rating']).size().unstack().plot(kind='bar',legend=True)
plt.show()
```



In [192…
```
dfMaster.groupby(['Age','Rating']).size().unstack().plot(kind='bar',legend=True)
plt.show()
```



In [193…
```
dfMaster.groupby(['Occupation','Rating']).size().unstack().plot(kind='bar',legend='
plt.show()
```



Develop an appropriate model to predict the movie ratings

In [194…
```
#model building
# first 500 records
new_data=dfMaster[:500]
```

In [195…
```
features=new_data[['MovieID','Age','Occupation']].values
```

In [196…
```
labels=new_data[['Rating']].values
```

In [197…
```
from sklearn.model_selection import train_test_split
#Create train and test data set
train, test, train_labels, test_labels = train_test_split(features,labels,test_size
```

In [198…
```
from sklearn.ensemble import RandomForestClassifier

random_forest = RandomForestClassifier(n_estimators=100)
random_forest.fit(train, train_labels)
Y_pred = random_forest.predict(test)
random_forest.score(train, train_labels)
acc_random_forest = round(random_forest.score(train, train_labels) * 100, 2)
acc_random_forest
```

```
C:\Users\rutuj\AppData\Local\Temp\ipykernel_19488\19447186.py:4: DataConversionWar
ning: A column-vector y was passed when a 1d array was expected. Please change the
shape of y to (n_samples,), for example using ravel().
  random_forest.fit(train, train_labels)
```

Out[198]:
```
100.0
```

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]: