In [1]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
```

In [2]:

```
movies=pd.read_table(r"C:\Users\Admin\Desktop\movie lens project\movies.dat",
                     engine='python',sep='::',names=['MovieID','Title','GENRES'])
ratings=pd.read_table(r"C:\Users\Admin\Desktop\movie lens project\ratings.dat",
                      engine='python',sep='::',names=['User ID','Movie ID','Rating','Time S
users=pd.read_table(r"C:\Users\Admin\Desktop\movie lens project\users.dat",
                    engine='python',sep='::',names=['User ID','Gender','Age','Occuption','Z
```

In [3]:

```
movies=pd.DataFrame(movies)
ratings=pd.DataFrame(ratings)
users=pd.DataFrame(users)
print(type(movies))
print(type(ratings))
print(type(users))
```

```
<class 'pandas.core.frame.DataFrame'>
<class 'pandas.core.frame.DataFrame'>
<class 'pandas.core.frame.DataFrame'>
```

In [4]:

```
movies.head()
```

Out[4]:

	MovieID	Title	GENRES
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

In [5]:

```
ratings.head()
```

Out[5]:

	User ID	Movie ID	Rating	Time Stamp
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

In [6]:

```
users.head()
```

Out[6]:

	User ID	Gender	Age	Occuption	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

In [7]:

```
#Doing Inner join to make master Data
master_data=pd.concat([movies.MovieID,movies.Title,users['User ID'],users.Age,users.Gender,
                       axis=1,
                       join='inner'
                      )
```

In [8]:

```
master_data.head()
```

Out[8]:

	MovieID	Title	User ID	Age	Gender	Occupation	Rating
0	1	Toy Story (1995)	1	1	F	10	5
1	2	Jumanji (1995)	2	56	M	16	3
2	3	Grumpier Old Men (1995)	3	25	M	15	3
3	4	Waiting to Exhale (1995)	4	45	M	7	4
4	5	Father of the Bride Part II (1995)	5	25	M	20	5

In [9]:

```
# Assigning to df for simplification
df=master_data
```

In [10]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3883 entries, 0 to 3882
Data columns (total 7 columns):
MovieID      3883 non-null int64
Title        3883 non-null object
User ID      3883 non-null int64
Age          3883 non-null int64
Gender       3883 non-null object
Occuption    3883 non-null int64
Rating       3883 non-null int64
dtypes: int64(5), object(2)
memory usage: 212.4+ KB
```

In [11]:

```
df.describe()
```

Out[11]:

	MovieID	User ID	Age	Occuption	Rating
count	3883.000000	3883.000000	3883.000000	3883.000000	3883.000000
mean	1986.049446	1942.000000	30.429307	8.207314	3.561937
std	1146.778349	1121.069876	13.062302	6.330827	1.095616
min	1.000000	1.000000	1.000000	0.000000	1.000000
25%	982.500000	971.500000	25.000000	3.000000	3.000000
50%	2010.000000	1942.000000	25.000000	7.000000	4.000000
75%	2980.500000	2912.500000	35.000000	14.000000	4.000000
max	3952.000000	3883.000000	56.000000	20.000000	5.000000

In [12]:

```
df.isna().sum()
```

Out[12]:

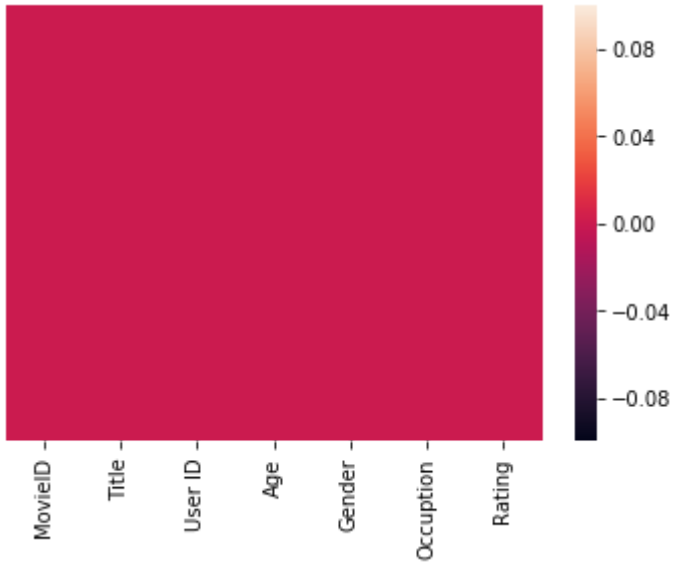
```
MovieID      0
Title        0
User ID      0
Age          0
Gender       0
Occuption    0
Rating       0
dtype: int64
```

In [13]:

```
sns.heatmap(df.isna(),yticklabels=False)
```

Out[13]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x195a32fe4e0>
```

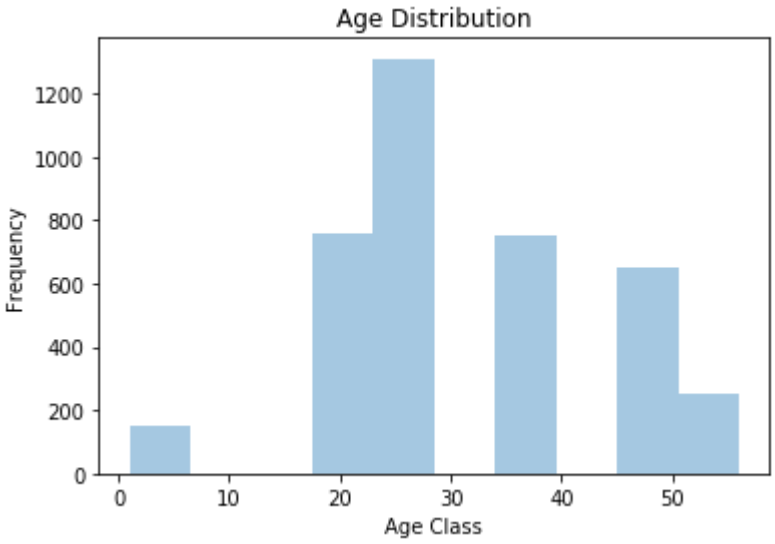


In [14]:

```
sns.distplot(df['Age'],bins=10,kde=False)
plt.xlabel("Age Class")
plt.ylabel("Frequency")
plt.title("Age Distribution")
```

Out[14]:

```
Text(0.5, 1.0, 'Age Distribution')
```



1: User Age Distribution

Answer: Most of the users lying between age group 18 and 28

In [15]:

```
df.head()
```

Out[15]:

	MovieID	Title	User ID	Age	Gender	Occuption	Rating
0	1	Toy Story (1995)	1	1	F	10	5
1	2	Jumanji (1995)	2	56	M	16	3
2	3	Grumpier Old Men (1995)	3	25	M	15	3
3	4	Waiting to Exhale (1995)	4	45	M	7	4
4	5	Father of the Bride Part II (1995)	5	25	M	20	5

In [16]:

```
master_data[master_data['Title']=='Toy Story (1995)']
```

Out[16]:

	MovieID	Title	User ID	Age	Gender	Occuption	Rating
0	1	Toy Story (1995)	1	1	F	10	5

2: User rating of the movie “Toy Story”

3: Top 25 movies by viewership rating

In [17]:

```
master_data.sort_values('Rating',ascending=False)[1:26]
```

Out[17]:

	MovieID	Title	User ID	Age	Gender	Occuption	Rating
1396	1420	Message to Love: The Isle of Wight Festival (1...	1397	35	M	7	5
1478	1513	Romy and Michele's High School Reunion (1997)	1479	50	F	2	5
1477	1511	A Chef in Love (1996)	1478	1	M	10	5
1470	1502	Kissed (1996)	1471	25	M	17	5
1469	1501	Keys to Tulsa (1997)	1470	18	M	4	5
1460	1490	B*A*P*S (1997)	1461	45	F	0	5
1439	1466	Donnie Brasco (1997)	1440	35	M	12	5
1431	1458	Touch (1997)	1432	35	M	1	5
1412	1437	Cement Garden, The (1993)	1413	18	M	3	5
1387	1410	Evening Star, The (1996)	1388	56	M	20	5
2007	2076	Blue Velvet (1986)	2008	25	M	3	5
1381	1404	Night Falls on Manhattan (1997)	1382	45	F	1	5
1380	1401	Ghosts of Mississippi (1996)	1381	18	M	11	5
1376	1397	Bastard Out of Carolina (1996)	1377	35	M	16	5
1372	1393	Jerry Maguire (1996)	1373	25	F	1	5
1365	1386	Terror in a Texas Town (1958)	1366	1	M	10	5
1352	1373	Star Trek V: The Final Frontier (1989)	1353	1	M	10	5
1351	1372	Star Trek VI: The Undiscovered Country (1991)	1352	35	M	20	5
1347	1368	Forbidden Christ, The (Cristo proibito, Il) (1...	1348	1	F	10	5
1483	1518	Breakdown (1997)	1484	25	M	12	5
1484	1519	Broken English (1996)	1485	25	F	9	5
1520	1559	Next Step, The (1995)	1521	25	F	7	5
1601	1647	Playing God (1997)	1602	25	F	7	5
1749	1814	Price Above Rubies, A (1998)	1750	50	M	5	5
1747	1811	Niagara, Niagara (1997)	1748	50	M	1	5

4: Find the ratings for all the movies reviewed by for a particular user of user id = 2696?

In [18]:

```
master_data[master_data['User ID']==2696]
```

Out[18]:

	MovieID	Title	User ID	Age	Gender	Occuption	Rating
2695	2764	Thomas Crown Affair, The (1968)	2696	25	M	7	3

In [19]:

```
unique=pd.DataFrame(movies['GENRES'].unique())
unique
```

Out[19]:

	0
0	Animation|Children's|Comedy
1	Adventure|Children's|Fantasy
2	Comedy|Romance
3	Comedy|Drama
4	Comedy
5	Action|Crime|Thriller
6	Adventure|Children's
7	Action
8	Action|Adventure|Thriller
9	Comedy|Drama|Romance
10	Comedy|Horror
11	Animation|Children's
12	Drama
13	Action|Adventure|Romance
14	Drama|Thriller
15	Drama|Romance
16	Thriller
17	Action|Comedy|Drama
18	Crime|Drama|Thriller
19	Drama|Sci-Fi
20	Romance
21	Adventure|Sci-Fi
22	Adventure|Romance
23	Children's|Comedy|Drama
24	Documentary
25	Drama|War
26	Action|Crime|Drama
27	Action|Adventure
28	Crime|Thriller
29	Animation|Children's|Musical|Romance
...	...
271	Animation|Mystery
272	Action|Horror|Thriller

	0
273	Action|Drama|Fantasy|Romance
274	Horror|Mystery
275	Adventure|Animation|Children's
276	Musical|Romance|War
277	Adventure|Drama|Romance
278	Adventure|Animation|Film-Noir
279	Action|Adventure|Animation
280	Comedy|Drama|Western
281	Adventure|Comedy|Sci-Fi
282	Drama|Romance|Western
283	Comedy|Drama|Sci-Fi
284	Action|Drama|Romance|Thriller
285	Adventure|Romance|Sci-Fi
286	Film-Noir|Horror
287	Crime|Drama|Film-Noir|Thriller
288	Action|Adventure|War
289	Romance|Western
290	Action|Children's|Fantasy
291	Adventure|Drama|Thriller
292	Adventure|Fantasy
293	Musical|War
294	Adventure|Musical|Romance
295	Action|Romance|Sci-Fi
296	Drama|Film-Noir
297	Comedy|Horror|Sci-Fi
298	Adventure|Drama|Romance|Sci-Fi
299	Adventure|Animation|Sci-Fi
300	Adventure|Crime|Sci-Fi|Thriller

301 rows × 1 columns

In [20]:

```
genre=pd.get_dummies(movies['GENRES'],drop_first=True)
genre
```

Out[20]:

	Action|Adventure	Action|Adventure|Animation	Action|Adventure|Animation|Children's|Fantas
0	0	0	
1	0	0	
2	0	0	
3	0	0	
4	0	0	
5	0	0	
6	0	0	
7	0	0	
8	0	0	
9	0	0	
10	0	0	
11	0	0	
12	0	0	
13	0	0	
14	0	0	
15	0	0	
16	0	0	
17	0	0	
18	0	0	
19	0	0	
20	0	0	
21	0	0	
22	0	0	
23	0	0	
24	0	0	
25	0	0	
26	0	0	
27	0	0	
28	0	0	
29	0	0	
...	...	...	
3853	0	0	
3854	0	0	

	Action|Adventure	Action|Adventure|Animation	Action|Adventure|Animation|Children's|Fantas
3855	0	0	
3856	0	0	
3857	0	0	
3858	0	0	
3859	0	0	
3860	0	0	
3861	0	0	
3862	0	0	
3863	0	0	
3864	0	0	
3865	0	0	
3866	0	0	
3867	0	0	
3868	0	0	
3869	0	0	
3870	0	0	
3871	0	0	
3872	0	0	
3873	0	0	
3874	0	0	
3875	0	0	
3876	0	0	
3877	0	0	
3878	0	0	
3879	0	0	
3880	0	0	
3881	0	0	
3882	0	0	

3883 rows × 300 columns

In [21]:

```
master_data.corr()
```

Out[21]:

	MovieID	User ID	Age	Occuption	Rating
MovieID	1.000000	0.999944	-0.004667	0.003366	-0.153298
User ID	0.999944	1.000000	-0.005309	0.003216	-0.153561
Age	-0.004667	-0.005309	1.000000	0.075108	-0.017635
Occuption	0.003366	0.003216	0.075108	1.000000	-0.001090
Rating	-0.153298	-0.153561	-0.017635	-0.001090	1.000000

In [22]:

```
master_data.head()
```

Out[22]:

	MovieID	Title	User ID	Age	Gender	Occuption	Rating
0	1	Toy Story (1995)	1	1	F	10	5
1	2	Jumanji (1995)	2	56	M	16	3
2	3	Grumpier Old Men (1995)	3	25	M	15	3
3	4	Waiting to Exhale (1995)	4	45	M	7	4
4	5	Father of the Bride Part II (1995)	5	25	M	20	5

In [23]:

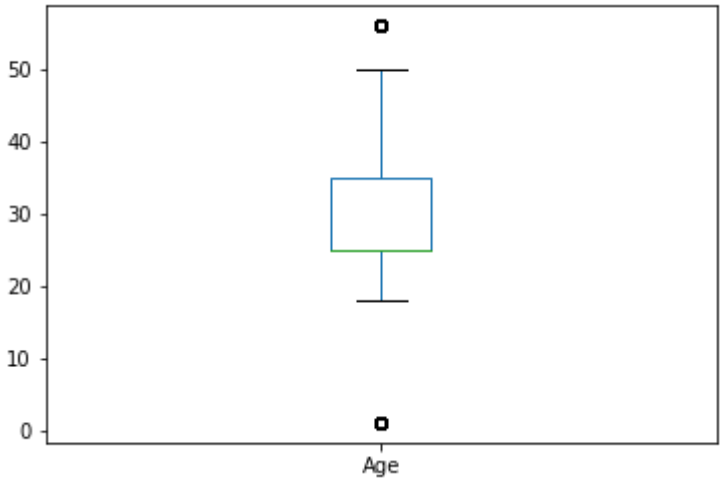
```
df=master_data
```

In [24]:

```
df['Age'].plot.box()
```

Out[24]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1959204b9b0>
```



In [25]:

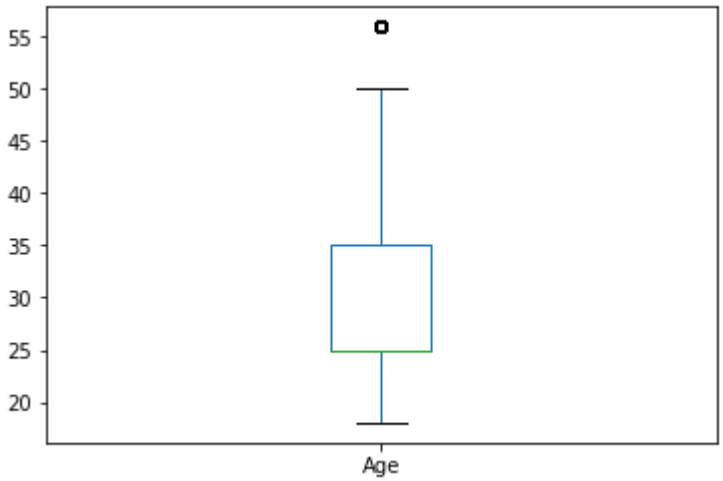
```
df=df[df['Age']!=1]
```

In [26]:

```
df['Age'].plot.box()
```

Out[26]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x19592057b38>
```



In [27]:

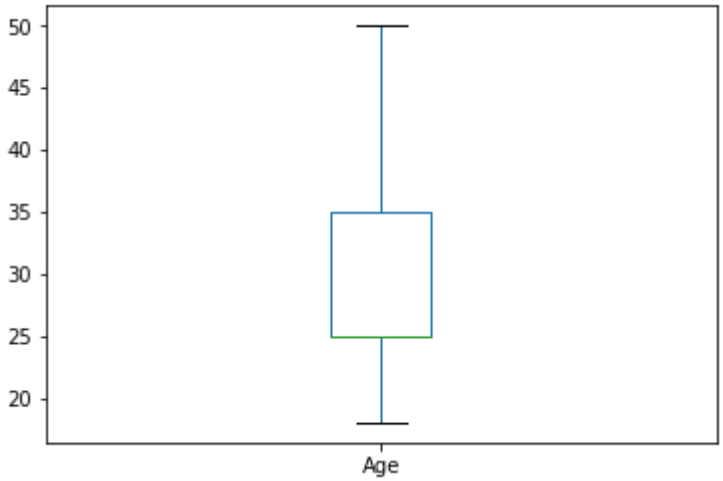
```
df=df[df['Age']!=56]
```

In [28]:

```
df['Age'].plot.box()
```

Out[28]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1959213cf28>
```



In [29]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 3476 entries, 2 to 3882
Data columns (total 7 columns):
MovieID      3476 non-null int64
Title        3476 non-null object
User ID      3476 non-null int64
Age          3476 non-null int64
Gender       3476 non-null object
Occuption    3476 non-null int64
Rating       3476 non-null int64
dtypes: int64(5), object(2)
memory usage: 217.2+ KB
```

In [30]:

```
df.head()
```

Out[30]:

	MovieID	Title	User ID	Age	Gender	Occuption	Rating
2	3	Grumpier Old Men (1995)	3	25	M	15	3
3	4	Waiting to Exhale (1995)	4	45	M	7	4
4	5	Father of the Bride Part II (1995)	5	25	M	20	5
5	6	Heat (1995)	6	50	F	9	3
6	7	Sabrina (1995)	7	35	M	1	5

In [31]:

```
sex=pd.get_dummies(df['Gender'],drop_first=True)
```

In [32]:

```
df_enc=pd.concat([df,sex],axis=1)
```

In [33]:

```
df_enc.head()
```

Out[33]:

	MovieID	Title	User ID	Age	Gender	Occuption	Rating	M
2	3	Grumpier Old Men (1995)	3	25	M	15	3	1
3	4	Waiting to Exhale (1995)	4	45	M	7	4	1
4	5	Father of the Bride Part II (1995)	5	25	M	20	5	1
5	6	Heat (1995)	6	50	F	9	3	0
6	7	Sabrina (1995)	7	35	M	1	5	1

In [34]:

```
df_final=df_enc.drop('Gender',axis=1)
```

In [35]:

```
df_final
```

Out[35]:

	MovieID	Title	User ID	Age	Occuption	Rating	M
2	3	Grumpier Old Men (1995)	3	25	15	3	1
3	4	Waiting to Exhale (1995)	4	45	7	4	1
4	5	Father of the Bride Part II (1995)	5	25	20	5	1
5	6	Heat (1995)	6	50	9	3	0
6	7	Sabrina (1995)	7	35	1	5	1
7	8	Tom and Huck (1995)	8	25	12	5	1
8	9	Sudden Death (1995)	9	25	17	4	1
9	10	GoldenEye (1995)	10	35	1	4	0
10	11	American President, The (1995)	11	25	1	5	0
11	12	Dracula: Dead and Loving It (1995)	12	25	12	4	1
12	13	Balto (1995)	13	45	1	4	1
13	14	Nixon (1995)	14	35	0	4	1
14	15	Cutthroat Island (1995)	15	25	7	5	1
15	16	Casino (1995)	16	35	0	4	0
16	17	Sense and Sensibility (1995)	17	50	1	3	1
17	18	Four Rooms (1995)	18	18	3	4	0
19	20	Money Train (1995)	20	25	14	4	1
20	21	Get Shorty (1995)	21	18	16	3	1
21	22	Copycat (1995)	22	18	15	3	1
22	23	Assassins (1995)	23	35	0	5	1
23	24	Powder (1995)	24	25	7	5	0
24	25	Leaving Las Vegas (1995)	25	18	4	3	1
25	26	Othello (1995)	26	25	7	5	1
26	27	Now and Then (1995)	27	25	11	4	1
27	28	Persuasion (1995)	28	25	1	4	0
28	29	City of Lost Children, The (1995)	29	35	7	4	1
29	30	Shanghai Triad (Yao a yao yao dao waipo qiao) ...	30	35	7	3	0
31	32	Twelve Monkeys (1995)	32	25	0	4	0
32	33	Wings of Courage (1995)	33	45	3	4	1
33	34	Babe (1995)	34	18	0	4	0
...	...	...	...	...	...	...	...
3851	3921	Beach Party (1963)	3852	50	16	3	1
3852	3922	Bikini Beach (1964)	3853	25	14	5	1

	MovieID	Title	User ID	Age	Occupation	Rating	M
3853	3923	Return of the Fly (1959)	3854	25	14	5	1
3854	3924	Pajama Party (1964)	3855	25	17	5	1
3855	3925	Stranger Than Paradise (1984)	3856	35	20	3	0
3856	3926	Voyage to the Bottom of the Sea (1961)	3857	18	14	3	1
3857	3927	Fantastic Voyage (1966)	3858	45	9	3	0
3858	3928	Abbott and Costello Meet Frankenstein (1948)	3859	25	0	4	1
3859	3929	Bank Dick, The (1940)	3860	45	17	4	1
3860	3930	Creature From the Black Lagoon, The (1954)	3861	18	4	4	1
3861	3931	Giant Gila Monster, The (1959)	3862	25	17	5	1
3862	3932	Invisible Man, The (1933)	3863	25	0	3	1
3863	3933	Killer Shrews, The (1959)	3864	50	20	4	1
3865	3935	Kronos (1973)	3866	45	14	3	1
3866	3936	Phantom of the Opera, The (1943)	3867	35	3	5	1
3867	3937	Runaway (1984)	3868	18	12	5	1
3868	3938	Slumber Party Massacre, The (1982)	3869	18	17	2	1
3869	3939	Slumber Party Massacre II, The (1987)	3870	25	17	4	1
3870	3940	Slumber Party Massacre III, The (1990)	3871	18	4	3	1
3871	3941	Sorority House Massacre (1986)	3872	25	1	5	1
3872	3942	Sorority House Massacre II (1990)	3873	45	2	4	0
3873	3943	Bamboozled (2000)	3874	25	7	4	1
3874	3944	Bootmen (2000)	3875	25	0	4	1
3875	3945	Digimon: The Movie (2000)	3876	25	0	3	1
3876	3946	Get Carter (2000)	3877	45	1	4	1
3877	3947	Get Carter (1971)	3878	35	16	4	1
3878	3948	Meet the Parents (2000)	3879	25	3	4	1
3879	3949	Requiem for a Dream (2000)	3880	25	7	2	1
3880	3950	Tigerland (2000)	3881	18	2	3	1
3882	3952	Contender, The (2000)	3883	50	16	4	1

3476 rows × 7 columns

In [36]:

```
x1=df_final.drop('Title',axis=1)
```

In [37]:

```
x=x1.drop('Rating',axis=1)
```

In [38]:

```
x.head()
```

Out[38]:

	MovieID	User ID	Age	Occuption	M
2	3	3	25	15	1
3	4	4	45	7	1
4	5	5	25	20	1
5	6	6	50	9	0
6	7	7	35	1	1

In [39]:

```
x=np.array(x)
x
```

Out[39]:

```
array([[   3,    3,   25,   15,    1],
       [   4,    4,   45,    7,    1],
       [   5,    5,   25,   20,    1],
       ...,
       [3949, 3880,   25,    7,    1],
       [3950, 3881,   18,    2,    1],
       [3952, 3883,   50,   16,    1]], dtype=int64)
```

In [40]:

```
y=np.array(df_final.Rating)
```

In [41]:

```
y
```

Out[41]:

```
array([3, 4, 5, ..., 2, 3, 4], dtype=int64)
```

In [42]:

```
from sklearn.model_selection import train_test_split as tts
```

In [43]:

```
x_train,x_test,y_train,y_test=tts(x,y,test_size=0.20,random_state=0)
```

In [44]:

```
from sklearn.linear_model import LinearRegression as LR
```

In [45]:

```
model=LR()
```

In [46]:

```
x_train[:10,:]
```

Out[46]:

```
array([[2579, 2511,   25,    7,    1],
       [ 591,  588,   25,   11,    0],
       [2559, 2491,   35,    7,    1],
       [2291, 2223,   25,    2,    1],
       [3471, 3403,   35,    5,    1],
       [1256, 1237,   18,    4,    1],
       [2589, 2521,   35,   17,    1],
       [2645, 2577,   25,    7,    1],
       [3915, 3846,   35,    0,    1],
       [ 783,  774,   18,    4,    1]], dtype=int64)
```

In [47]:

```
model.fit(x_train,y_train)
```

Out[47]:

```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None,
         normalize=False)
```

In [48]:

```
y_pred=model.predict(x_test)
```

In [49]:

```
from sklearn.metrics import mean_squared_error as mse
```

In [50]:

```
mse(y_test,y_pred)
```

Out[50]:

```
1.1918151942033821
```

MSE=1.19 Which is Good Value to predict Model

Linechart of Actual VS Predicted Value

In [51]:

```
fig=plt.figure(figsize=(8,4),dpi=100)
axes=fig.add_axes([0,0,1,1])
axes.plot(y_test[1:20],label='Actual')
axes.plot(y_pred[1:20],label='Predicted')
axes.legend()
```

Out[51]:

```
<matplotlib.legend.Legend at 0x19592154b00>
```