###TUGAS 2 DATA MINING & BUSINESS INTELLIGENCE Nama:
Almas Fauzia Wibawa (17/409427/PA/17734)
Bryan Dwison (17/409430/PA/17737)
Fajar Abdul Malik (17/409432/PA/17739)
Faqih Ethana Prabandaru (17/409433/PA/17740)

Import Data

In [ ]:

```
import pandas as pd

df_video = pd.read_excel(r'Video-Store.xls',sheet_name='Sheet1')
df_video = df_video.drop(columns=['Unnamed: 8','Unnamed: 9', 'Unnamed: 10'])
df_video.head()
```

Out[205]:

	Cust ID	Gender	Income	Age	Rentals	Avg Per Visit	Incidentals	Genre
0	1	M	45000	25	27	2.5	Yes	Action
1	2	F	54000	33	12	3.4	No	Drama
2	3	F	32000	20	42	1.6	No	Comedy
3	4	F	59000	70	16	4.2	Yes	Drama
4	5	M	37000	35	25	3.2	Yes	Action

A. Data Exploration

a. Hitung nilai rata-rata, median, dan mode dari atribut Income dan Age. Dari ketiga nilai tersbeut, tentukan distribusi atribut Income dan Age miring ke kanan (*positively skewed*), miring ke kiri (*negatively skewed*), atau simetri.

In [ ]:

```
income_mean = df_video.Income.mean()
income_median = df_video.Income.median()

age_mean = df_video.Age.mean()
age_median = df_video.Age.median()
print('Rata-rata income: ' + str(income_mean))
print('Median income: ' + str(income_median))
print()
print('Rata-rata age: ' + str(age_mean))
print('Median age: ' + str(age_median))
```

```
Rata-rata income: 42300.0
Median income: 41000.0

Rata-rata age: 31.56
Median age: 30.0
```

In [ ]:

```
df_video.Income.value_counts()
```

Out[163]:

```
29000    3
41000    3
74000    2
57000    2
68000    2
45000    2
47000    2
56000    2
32000    2
17000    2
24000    2
49000    2
89000    1
83000    1
36000    1
2000     1
35000    1
18000    1
52000    1
26000    1
23000    1
38000    1
37000    1
25000    1
31000    1
15000    1
1000     1
65000    1
69000    1
79000    1
54000    1
62000    1
59000    1
50000    1
12000    1
6000     1
Name: Income, dtype: int64
```

In [ ]:

```
df_video.Age.value_counts()
```

Out[164]:

```
25    7
35    6
16    3
33    3
20    3
30    2
47    2
46    2
38    2
19    2
52    2
21    2
22    2
24    1
15    1
18    1
56    1
28    1
29    1
32    1
36    1
40    1
43    1
45    1
70    1
Name: Age, dtype: int64
```

In [ ]:

```
df_video.Income.skew()
```

Out[165]:

```
0.12035642908466405
```

Dari kode di atas, dapat kita simpulkan bahwa:

1. Rata-rata atribut Income = **42300**
2. Median atribut Income = **41000**
3. Modus atribut Income = **29000**
4. Rata-rata atribut Age = **31.56**
5. Median atribut Age = **30**
6. Modus atribut Age = **25**

Dari nilai rata-rata, median, dan modus dari atribut Income dan Age tersebut, dapat disimpulkan pula bahwa distribusi atribut Income dan Age bersifat **miring ke kanan (*positively skewed*)** karena modus < median < rata-rata.

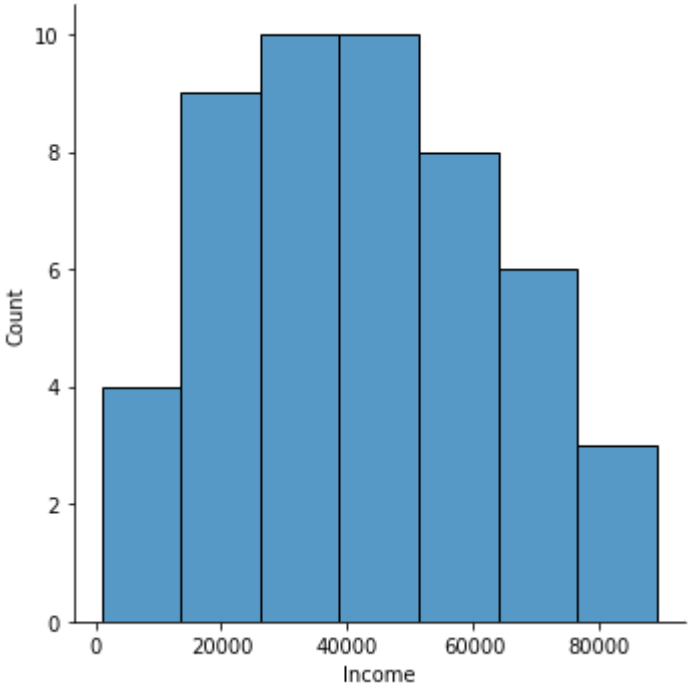
b. Gambarkan histogram dari atribut Income dan Age. Jelaskan apakah distribusi dari atribut Income dan Age berdasarkan histogram sama dengan hasil di soal (a).

In [ ]:

```
import seaborn as sns

sns.displot(df_video.Income)
# sns.distplot(df_video.Income)
```

Out[166]:

```
<seaborn.axisgrid.FacetGrid at 0x7f313bf1c390>
```

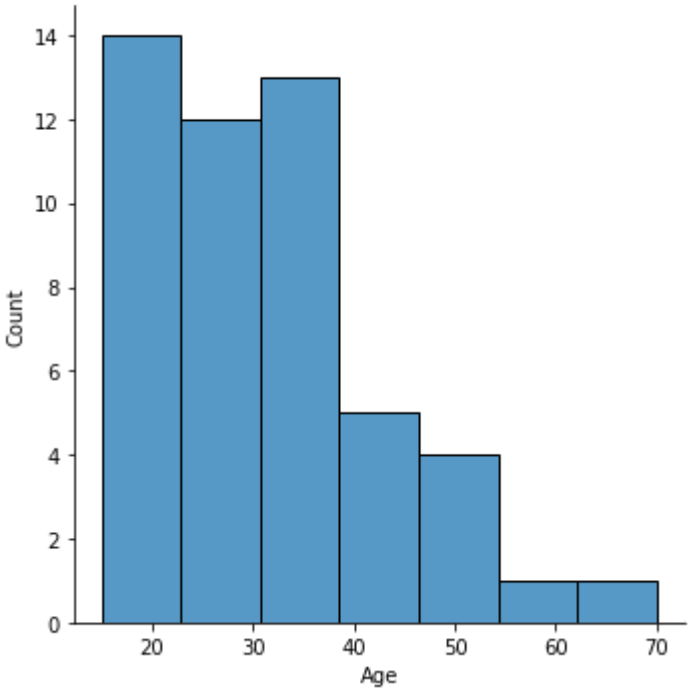


In [ ]:

```
sns.displot(df_video.Age)
```

Out[167]:

```
<seaborn.axisgrid.FacetGrid at 0x7f313beb0a58>
```



Dari dua gambar di atas, terlihat bahwa distribusi atribut Income dan Age sama dengan jawaban nomor (a), yaitu **miring ke kanan (*positively skewed*)**. Hal ini karena **data di atribut Income dan Age lebih banyak tersebar di sebelah kiri grafik**.

c. Hitung nilai minimum, maksimum, Q1, Q2, Q3, dari atribut Income dan Age.

In [ ]:

```
df_video.Income.describe()
```

Out[168]:

```
count       50.000000
mean     42300.000000
std      21409.753642
min       1000.000000
25%      26750.000000
50%      41000.000000
75%      56750.000000
max      89000.000000
Name: Income, dtype: float64
```

In [ ]:

```
df_video.Age.describe()
```

Out[169]:

```
count    50.000000
mean     31.560000
std      12.000272
min      15.000000
25%      22.000000
50%      30.000000
75%      37.500000
max      70.000000
Name: Age, dtype: float64
```

Dari keluaran di atas, dapat diperoleh informasi bahwa:

1. Nilai minimum atribut Income = **1000**
2. Nilai maksimum atribut Income = **89000**
3. Nilai Q1 atribut Income = **26750**
4. Nilai Q2 atribut Income = **41000**
5. Nilai Q3 atribut Income = **56750**
6. Nilai minimum atribut Age = **15**
7. Nilai maksimum atribut Age = **70**
8. Nilai Q1 atribut Age = **22**
9. Nilai Q2 atribut Age = **30**
10. Nilai Q3 atribut Age = **37.5**

d. Gambarkan boxplot dari Income dan Age

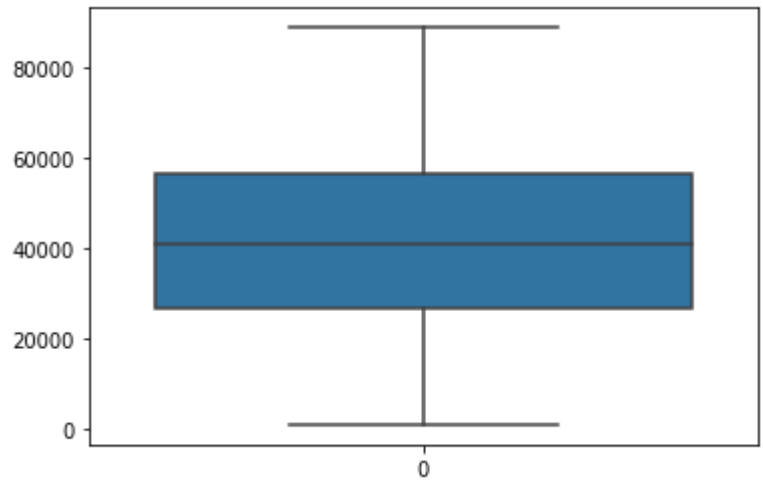
Boxplot atribut Income:

In [ ]:

```
sns.boxplot(data=df_video.Income)
```

Out[170]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f313d6544e0>
```



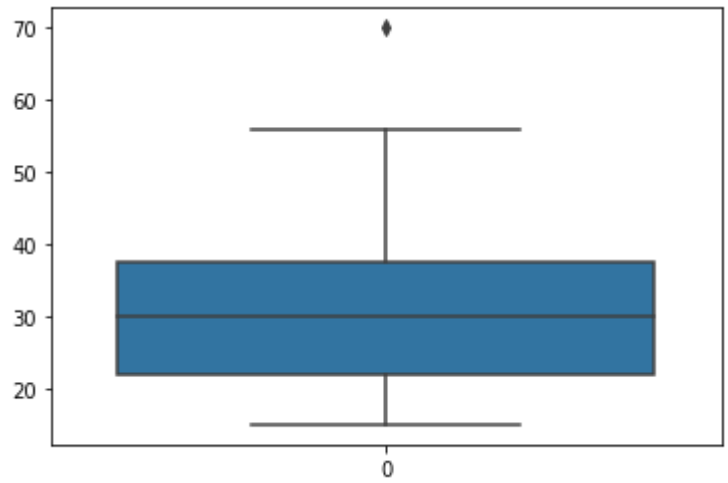
Boxplot atribut Age:

In [ ]:

```
sns.boxplot(data=df_video.Age)
```

Out[171]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f313d9d1358>
```



e. Hitung rata-rata Income pelanggan pria (M) dan pelanggan wanita (F). Rata-rata Income mana yang lebih tinggi dari rata-rata keseluruhan pelanggan?

In [ ]:

```
income_M = [df_video.Income[i] for i in range(len(df_video)) if df_video.Gender[i]=='M']
income_F = [df_video.Income[i] for i in range(len(df_video)) if df_video.Gender[i]=='F']

income_mean_M = sum(income_M) / len(income_M)
income_mean_F = sum(income_F) / len(income_F)

print('Rata-rata Income pelanggan pria (M) = ' + str(income_mean_M))
print('Rata-rata Income pelanggan wanita (F) = ' + str(income_mean_F))
```

```
Rata-rata Income pelanggan pria (M) = 41000.0
Rata-rata Income pelanggan wanita (F) = 43708.333333333336
```

Dari keluaran di atas, dapat dilihat bahwa:

1. Rata-rata Income pelanggan pria (M) = **41000**
2. Rata-rata Income pelanggan wanita (F) = **43708.33**

Jika dibandingkan, **rata-rata Income pelanggan wanita lebih tinggi dibandingkan rata-rata keseluruhan pelanggan**.

f. Tentukan jenis film (Genre) yang paling banyak dipinjam oleh pelanggan pria dan pelanggan wanita.

In [ ]:

```
from matplotlib import pyplot as plt
import matplotlib.patches as mpatches
import seaborn as sns
```

In [ ]:

```
x = set(df_video.Genre)
xlist = list(x)
```

In [ ]:

```
nGenreM = [0,0,0]
nGenreF = [0,0,0]
for i in range(len(df_video.Genre)):
  if df_video.Gender[i] == 'M':
    if df_video.Genre[i] == 'Drama':
      nGenreM[0]+=1
    elif df_video.Genre[i] == 'Comedy':
      nGenreM[1]+=1
    else:
      nGenreM[2]+=1
  else:
    if df_video.Genre[i] == 'Drama':
      nGenreF[0]+=1
    elif df_video.Genre[i] == 'Comedy':
      nGenreF[1]+=1
    else:
      nGenreF[2]+=1
nGenreM,nGenreF
```

Out[175]:

```
([7, 6, 13], [13, 6, 5])
```

In [ ]:

```
mostMGenre = xlist[nGenreM.index(max(nGenreM))]
mostFGenre = xlist[nGenreF.index(max(nGenreF))]
print("Genre yang paling banyak dipinjam pelanggan pria: "+ mostMGenre)
print("Genre yang paling banyak dipinjam pelanggan wanita: "+ mostFGenre)
```

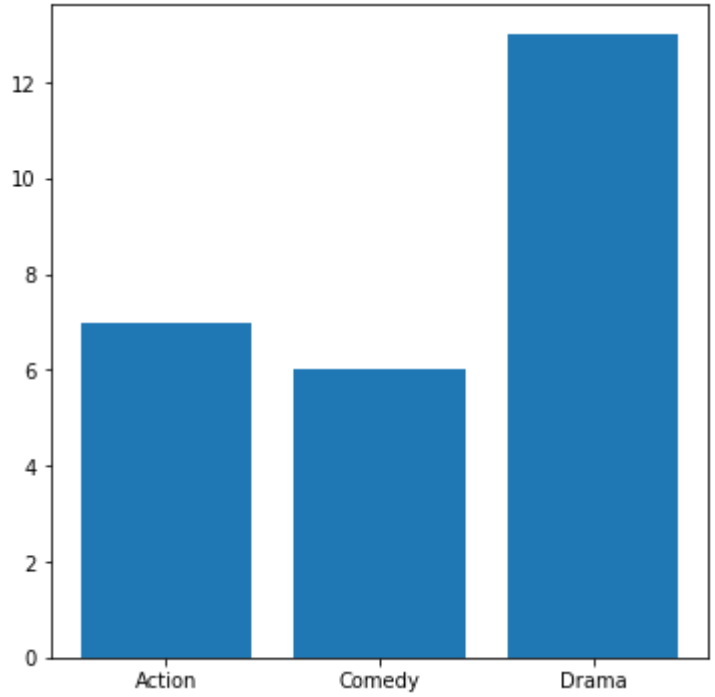
```
Genre yang paling banyak dipinjam pelanggan pria: Drama
Genre yang paling banyak dipinjam pelanggan wanita: Action
```

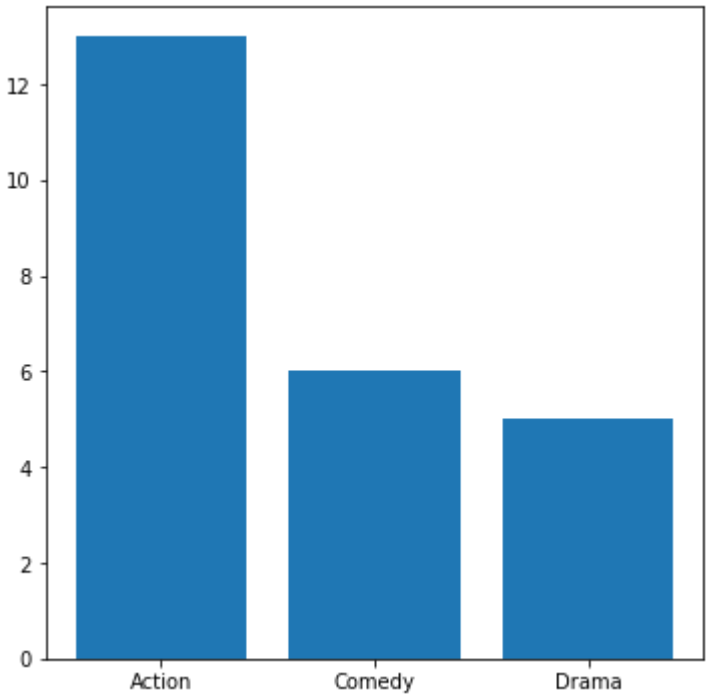
In [ ]:

```
plt.figure(figsize=(20, 6))
plt.subplot(131)
plt.bar(xlist, nGenreM)
plt.figure(figsize=(20, 6))
plt.subplot(132)
plt.bar(xlist, nGenreF)
```

Out[177]:

```
<BarContainer object of 3 artists>
```





g. Bandingkan rata-rata Income pelanggan pria dan wanita yang menyukai film berjenis komedi (comedy).

In [ ]:

```
sumIncomeM = 0
sumIncomeF = 0
nM = 0
nF = 0

for i in range(len(df_video)):
  if df_video.Genre[i] == 'Comedy':
    if df_video.Gender[i] == 'M':
      nM += 1
      sumIncomeM += df_video.Income[i]
    else:
      nF += 1
      sumIncomeF += df_video.Income[i]
```

In [ ]:

```
avgM = sumIncomeM/nM
avgF = sumIncomeF/nF
avgIncome=[]
avgIncome.append(avgM)
avgIncome.append(avgF)
avgIncome
avgg = zip(list(set(df_video.Gender)),avgIncome)
print(tuple(avgg))
```

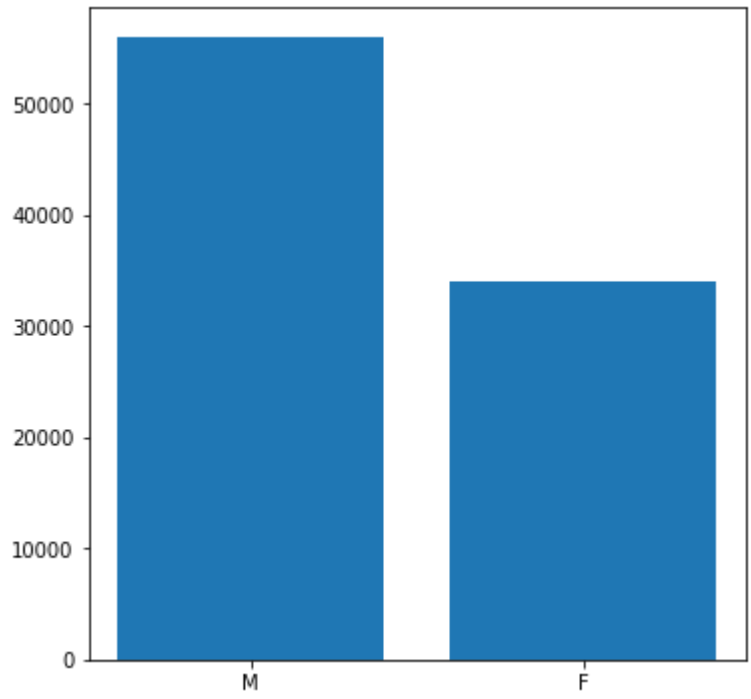
```
(('M', 56000.0), ('F', 34000.0))
```

In [ ]:

```
plt.figure(figsize=(20, 6))
plt.subplot(131)
plt.bar(list(set(df_video.Gender)), avgIncome)
```

Out[180]:

```
<BarContainer object of 2 artists>
```



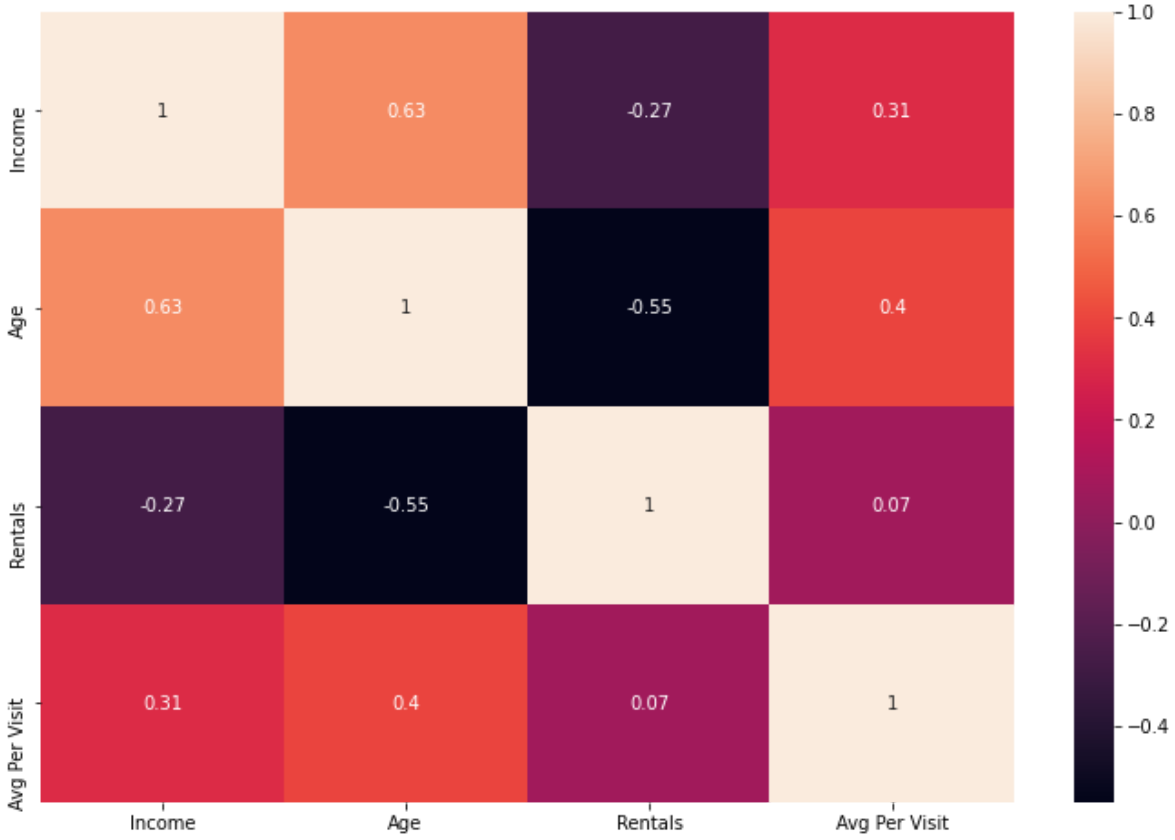
h. Hitung koefisien korelasi antar atribut numerik dalam data (Income, Age, Rentals, Avg per Visit). Tentukan atribut mana yang memiliki koefisien korelasi terbesar dan terkecil, dan jelaskan maknanya.

In [ ]:

```
xCor = df_video.drop('Cust ID', 1)
xCor = xCor.drop('Unnamed: 8', 1)
xCor = xCor.drop('Unnamed: 9', 1)
```

In [ ]:

```
plt.figure(figsize=(12, 8))

vg_corr = xCor.corr()
sns.heatmap(vg_corr,
            xticklabels = vg_corr.columns.values,
            yticklabels = vg_corr.columns.values,
            annot = True);
```



Koefisien Korelasi Terbesar : Age dengan Income
Berarti bahwa semakin besar age/umur dari customer maka semakin besar juga income dari customer tersebut
Koefisien Korelasi Terkecil : Age dengan Rentals
Berarti bahwa semakin besar age/number dari customer justru semakin berkurang waktu rental customer tersebut

In [ ]:

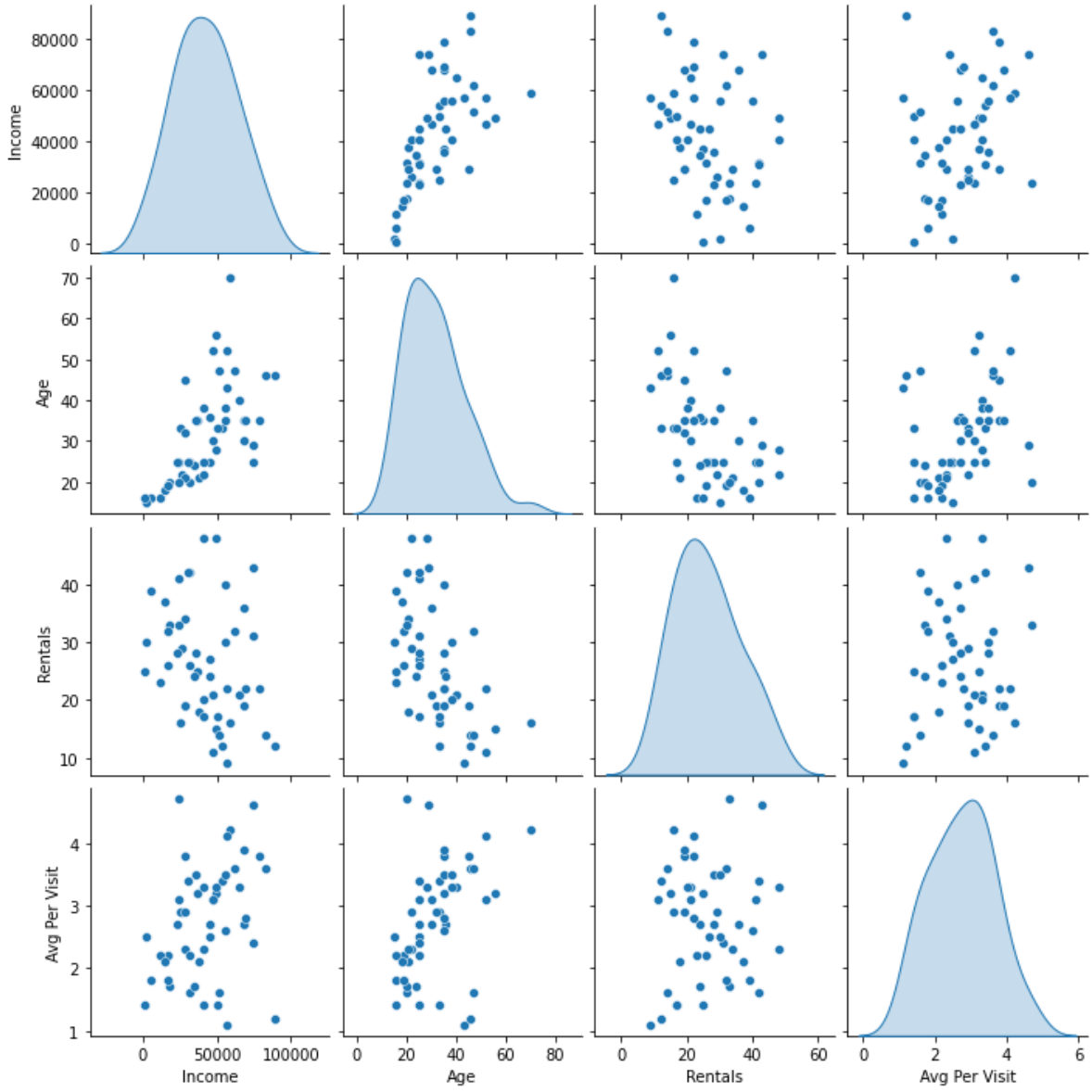
```

```

i. Visualisasikan scatter plot antar keempat atribut, yaitu Income, Age, Rentals, Avg per Visit. bold text

In [ ]:

```
plt.figure(figsize=(14, 14))

sns.pairplot(xCor, diag_kind='kde');
```

```
<Figure size 1008x1008 with 0 Axes>
```



j. Pilih customer yang berharga (valued customer), yaitu customer yang memiliki nilai tinggi pada atribut Rentals >= 30. Bandingkan karakteristik dari customer ini dengan karakteristik dari customer lain (yang bukan valued customer) dalam hal Income dan Age

In [ ]:

```
custValued = [i for i in range(len(df_video)) if df_video.Rentals[i] >= 30]
custValued
```

Out[184]:

```
[2, 5, 7, 10, 14, 17, 18, 22, 25, 26, 28, 34, 35, 39, 42, 44, 48, 49]
```

In [ ]:

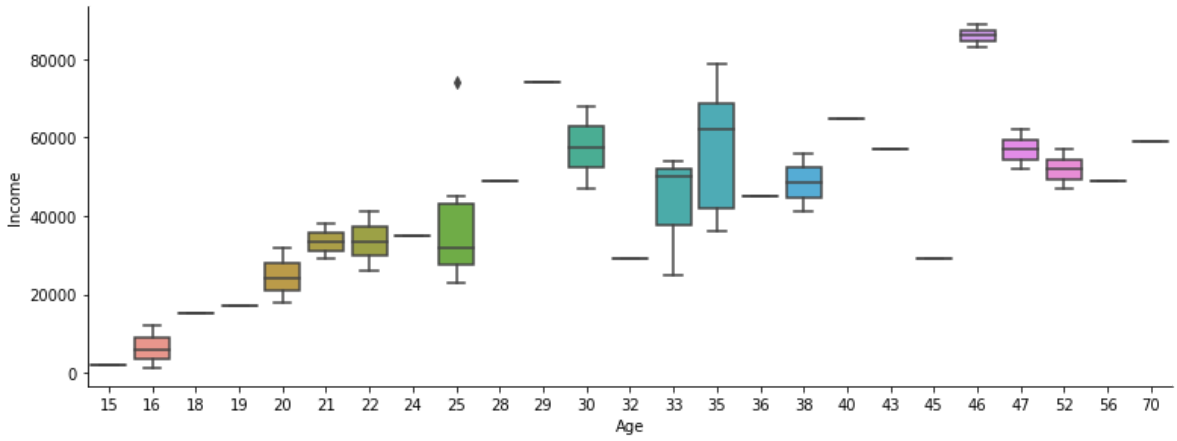
```
IncomeValued = []
IncomeNoValued = []
AgeValued = []
AgeNoValued = []
for i in range(len(df_video)):
  if i in custValued:
    IncomeValued.append(df_video.Income[i])
    AgeValued.append(df_video.Age[i])
  else:
    IncomeNoValued.append(df_video.Income[i])
    AgeNoValued.append(df_video.Age[i])
```

In [ ]:

```

```

In [ ]:

```
g = sns.catplot(x="Age", y="Income",

                data=df_video, kind="box",

                height=4, aspect=2.7);
```



In [ ]:

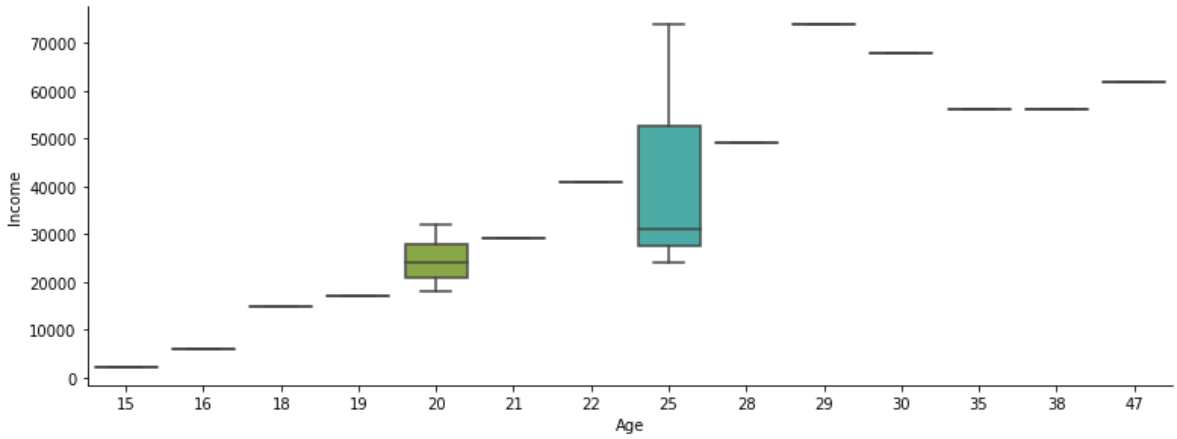
```
dfValued = df_video[df_video.Rentals >= 30]
```

In [ ]:

```
dfNoValued = df_video[df_video.Rentals < 30]
```

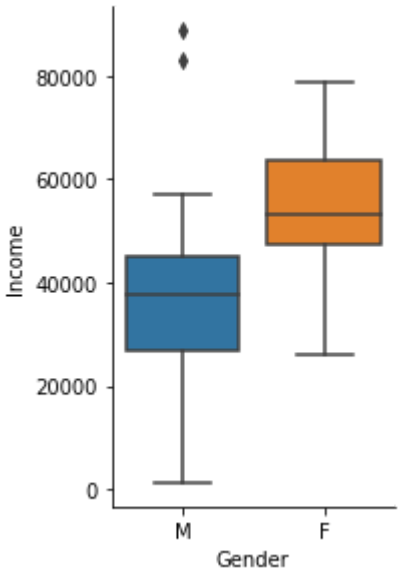
In [ ]:

```
g = sns.catplot(x="Age", y="Income",
                data=dfValued, kind="box",
                height=4, aspect=2.7);
```



In [ ]:

```
h = sns.catplot(x="Gender", y="Income",
                data=dfNoValued, kind="box",
                height=4, aspect=.7);
```



In [ ]:

```
from statistics import mean
avgIncomeValued = round(mean(dfValued.Income), 2)
avgIncomeNoValued = round(mean(dfValued.Income), 2)
```

###B. Data Preprocessing **Lakukan data preprocessing pada atribut dari file Video-Store.xls. Tambahkan kolom baru dalam file excel sebagai hasil dari preprocessing soal (a), (b), (c), (d), (e).**

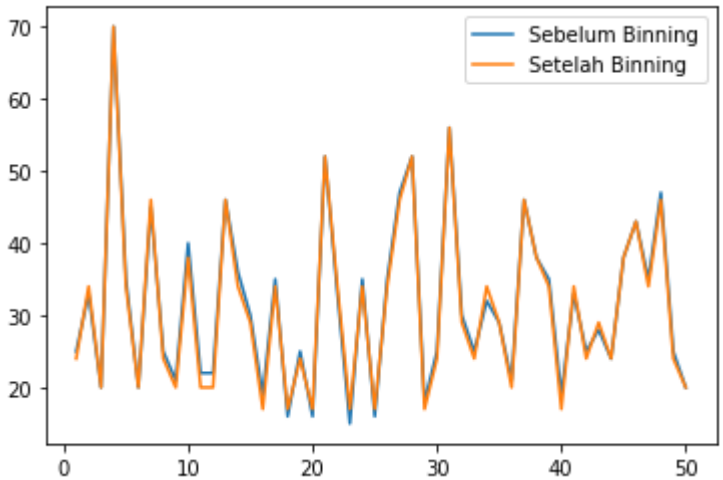
a. Lakukan smoothing dengan metode binning rata-rata pada atribute Age. Gunakan bin berukuran 4 (bin depth). Buat grafik atribut Age sebelum dan sesudah smoothing.

In [ ]:

```
from scipy import stats
import matplotlib.pyplot as plt
import math
import warnings
warnings.simplefilter("ignore")

CustID_Age = df_video[['Cust ID', 'Age']]
CustID_Age.sort_values(by='Age', inplace=True)

temp = CustID_Age['Age']
bin_depth = 4
n_bins = math.ceil((len(CustID_Age['Age'])/bin_depth))

bin_means, bin_edges, binnumber = stats.binned_statistic(CustID_Age['Age'], temp, bins=n_bi
x = []

for i in CustID_Age['Age']:
  for j in range(13):
    if i>=bin_edges[j] and i<=bin_edges[j+1]:
      x.append(int(bin_means[j]))
CustID_Age['Age'] = x
CustID_Age.sort_values(by='Cust ID', inplace=True)
df_video['Age-Binned'] = CustID_Age['Age']

plt.figure()
plt.plot(df_video['Cust ID'],df_video['Age'])
plt.plot(df_video['Cust ID'],df_video['Age-Binned'])
plt.legend(['Sebelum Binning', 'Setelah Binning'])
df_video[['Cust ID', 'Age', 'Age-Binned']].head(5)
```

Out[285]:

	Cust ID	Age	Age-Binned
0	1	25	24
1	2	33	34
2	3	20	20
3	4	70	70
4	5	35	34



b. Lakukan min-max normalization untuk mentransformasi nilai atribut Income ke dalam range [0.0-1.0].

In [ ]:

```
income = df_video['Income']
temp = []
for i in income:
  new_income = (i-income.min())/(income.max()-income.min())
  new_income = round(new_income, 2)
  temp.append(new_income)

df_video['Income-normalized'] = temp
df_video[['Cust ID', 'Income', 'Income-normalized']].head()
```

Out[271]:

	Cust ID	Income	Income-normalized
0	1	45000	0.50
1	2	54000	0.60
2	3	32000	0.35
3	4	59000	0.66
4	5	37000	0.41

c. Lakukan z-score normalization untuk menstandarkan nilai atribut Rentals.

In [ ]:

```
import numpy as np

rentals = df_video['Rentals']
temp = []
stdev = np.std(rentals)
mean = np.mean(rentals)

for i in rentals:
  new_rentals = (i-mean)/stdev
  new_rentals = round(new_rentals,2)
  temp.append(new_rentals)

df_video['Rentals-normalized'] = temp
df_video[['Cust ID', 'Rentals', 'Rentals-normalized']].head()
```

Out[272]:

	Cust ID	Rentals	Rentals-normalized
0	1	27	0.08
1	2	12	-1.43
2	3	42	1.59
3	4	16	-1.03
4	5	25	-0.12

d. Ubah atribut Income (yang belum dinormalisasi) menjadi diskrit menggunakan kategori berikut:: High = 60K+; Mid = 25K-59K; Low = kurang dari $25K. Hitung jumlah pelanggan untuk masing-masing

kategori Income.

In [ ]:

```
import pandas as pd
import numpy as np

df_video['Income-discrete'] = pd.cut(df_video['Income'],
                                     bins=[0, 25000, 60000, np.inf],
                                     labels=['Low', 'Mid', 'High'])
df_video[['Cust ID', 'Income', 'Income-discrete']].head(50)
```

Out[282]:

	Cust ID	Income	Income-discrete
0	1	45000	Mid
1	2	54000	Mid
2	3	32000	Mid
3	4	59000	Mid
4	5	37000	Mid
5	6	18000	Low
6	7	29000	Mid
7	8	74000	High
8	9	38000	Mid
9	10	65000	High
10	11	41000	Mid
11	12	26000	Mid
12	13	83000	High
13	14	45000	Mid
14	15	68000	High
15	16	17000	Low
16	17	36000	Mid
17	18	6000	Low
18	19	24000	Low
19	20	12000	Low
20	21	47000	Mid
21	22	25000	Low
22	23	2000	Low
23	24	79000	High
24	25	1000	Low
25	26	56000	Mid
26	27	62000	High
27	28	57000	Mid
28	29	15000	Low

	Cust ID	Income	Income-discrete
29	30	41000	Mid
30	31	49000	Mid
31	32	47000	Mid
32	33	23000	Low
33	34	29000	Mid
34	35	74000	High
35	36	29000	Mid
36	37	89000	High
37	38	41000	Mid
38	39	68000	High
39	40	17000	Low
40	41	50000	Mid
41	42	32000	Mid
42	43	49000	Mid
43	44	35000	Mid
44	45	56000	Mid
45	46	57000	Mid
46	47	69000	High
47	48	52000	Mid
48	49	31000	Mid
49	50	24000	Low

e. Konversi semua atribut kategorik (Gender, Incidentals, Genre) menjadi atribut kontinyu.

In [ ]:

```
categoric = df_video[['Cust ID', 'Gender', 'Incidentals', 'Genre']]
x = pd.get_dummies(categoric, columns=['Gender', 'Incidentals', 'Genre'], prefix=['Gender',
df_video = df_video.merge(x)
df_video.head()
```

Out[274]:

	Cust ID	Gender	Income	Age	Rentals	Avg Per Visit	Incidentals	Genre	Gender_F	Gender_M	Inci
0	1	M	45000	25	27	2.5	Yes	Action	0	1	
1	2	F	54000	33	12	3.4	No	Drama	1	0	
2	3	F	32000	20	42	1.6	No	Comedy	1	0	
3	4	F	59000	70	16	4.2	Yes	Drama	1	0	
4	5	M	37000	35	25	3.2	Yes	Action	0	1	

f. Buat distance matrik dari pelanggan 1 sampai pelanggan 5 (ukuran matrik adalah 5 x 5) sebelum dan

sesudah normalisasi menggunakan min-max (b) dan z-score (c). NOTE: jarak dihitung menggunakan atribut Income dan Rentals. Apakah distance matrix yang dihasilkan sama?

In [ ]:

```
from scipy.spatial.distance import squareform, pdist
df_sebelum = df_video[['Cust ID','Income', 'Rentals']].head(5)
df_setelah = df_video[['Cust ID','Income-normalized', 'Rentals-normalized']].head(5)

distances = pdist(df_minmax.values, metric='euclidean')
dist_matrix = squareform(distances)
print('Distance Matrix sebelum normalisasi')
print(dist_matrix)

distances2 = pdist(df_setelah.values, metric='euclidean')
dist_matrix2 = squareform(distances)
print('Distance Matrix setelah normalisasi')
print(dist_matrix2)
```

```
Distance Matrix sebelum normalisasi
[[0.         1.00498756 2.00561711 3.00426364 4.00101237]
 [1.00498756 0.         1.03077641 2.0008998  3.00601065]
 [2.00561711 1.03077641 0.         1.04694795 2.0008998 ]
 [3.00426364 2.0008998  1.04694795 0.         1.03077641]
 [4.00101237 3.00601065 2.0008998  1.03077641 0.        ]]
Distance Matrix setelah normalisasi
[[0.         1.00498756 2.00561711 3.00426364 4.00101237]
 [1.00498756 0.         1.03077641 2.0008998  3.00601065]
 [2.00561711 1.03077641 0.         1.04694795 2.0008998 ]
 [3.00426364 2.0008998  1.04694795 0.         1.03077641]
 [4.00101237 3.00601065 2.0008998  1.03077641 0.        ]]
```

Kesimpulan: *Distance matrix* yang dihasilkan menggunakan data sebelum dan sesudah normalisasi adalah **SAMA**. Hal ini dikarenakan normalisasi hanya mengubah range representasi tiap nilai menjadi antara 0-1.

g. Dalam sheet2 ada data yang nilainya tidak ada (missing value). Isi data yang tidak ada tersebut dengan nilai yang sesuai.

Sheet2 sebelum *misiing value* diisi

In [ ]:

```
import pandas as pd

df_video2 = pd.read_excel(r'Video-Store.xls', sheet_name='Sheet2')
df_video2 = df_video2.drop(columns=['Unnamed: 8','Unnamed: 9', 'Unnamed: 10'])
df_video2.head(50)
```

Out[283]:

	Cust ID	Gender	Income	Age	Rentals	Avg Per Visit	Incidentals	Genre
0	1	M	45000.0	25.0	27	2.5	Yes	Action
1	2	F	54000.0	33.0	12	3.4	No	Drama
2	3	F	32000.0	20.0	42	1.6	No	Comedy
3	4	F	59000.0	70.0	16	4.2	Yes	Drama
4	5	M	37000.0	35.0	25	3.2	Yes	Action
5	6	M	18000.0	20.0	33	1.7	No	Action
6	7	F	29000.0	45.0	19	3.8	No	Drama
7	8	M	74000.0	25.0	31	2.4	Yes	Action
8	9	M	NaN	21.0	18	2.1	No	Comedy
9	10	F	65000.0	40.0	21	3.3	No	NaN
10	11	F	41000.0	22.0	48	2.3	Yes	Drama
11	12	F	26000.0	22.0	29	2.9	Yes	Action
12	13	M	83000.0	46.0	14	3.6	No	Comedy
13	14	M	45000.0	36.0	24	2.7	No	Drama
14	15	M	68000.0	NaN	36	2.7	Yes	Comedy
15	16	M	17000.0	19.0	26	2.2	Yes	Action
16	17	M	36000.0	35.0	28	3.5	Yes	Drama
17	18	F	NaN	16.0	39	1.8	Yes	Action
18	19	F	24000.0	25.0	41	3.1	No	Comedy
19	20	M	12000.0	16.0	23	2.2	Yes	NaN
20	21	F	47000.0	52.0	11	3.1	No	Drama
21	22	M	25000.0	33.0	16	2.9	Yes	Drama
22	23	F	2000.0	15.0	30	2.5	No	Comedy
23	24	F	79000.0	35.0	22	3.8	Yes	Drama
24	25	M	1000.0	16.0	25	1.4	Yes	Comedy
25	26	F	56000.0	35.0	40	2.6	Yes	Action
26	27	F	62000.0	47.0	32	3.6	No	Drama
27	28	M	57000.0	52.0	22	4.1	No	Comedy
28	29	F	15000.0	18.0	37	2.1	Yes	Action
29	30	M	41000.0	25.0	17	1.4	Yes	Action
30	31	F	49000.0	56.0	15	3.2	No	Comedy
31	32	M	47000.0	30.0	21	3.1	Yes	Drama

	Cust ID	Gender	Income	Age	Rentals	Avg Per Visit	Incidentals	Genre
32	33	M	23000.0	25.0	28	2.7	No	Action
33	34	F	29000.0	32.0	19	2.9	Yes	Action
34	35	M	74000.0	29.0	43	4.6	Yes	Action
35	36	F	29000.0	21.0	34	2.3	No	Comedy
36	37	M	89000.0	46.0	12	1.2	No	Comedy
37	38	M	41000.0	38.0	20	3.3	Yes	Drama
38	39	F	68000.0	35.0	19	3.9	No	Comedy
39	40	M	17000.0	19.0	32	1.8	No	Action
40	41	F	50000.0	33.0	17	1.4	No	Drama
41	42	M	32000.0	25.0	26	2.2	Yes	Action
42	43	F	49000.0	28.0	48	3.3	Yes	Drama
43	44	M	35000.0	24.0	24	1.7	No	Drama
44	45	M	56000.0	NaN	30	3.5	Yes	Drama
45	46	F	57000.0	43.0	9	1.1	No	Drama
46	47	F	69000.0	35.0	22	2.8	Yes	Drama
47	48	F	52000.0	47.0	14	1.6	No	Drama
48	49	M	31000.0	NaN	42	3.4	Yes	Action
49	50	M	24000.0	20.0	33	4.7	No	Action

Sheet2 setelah *missing value* diisi dengan nilai yang paling sering muncul pada tiap kolomnya.

In [ ]:

```
df_video2 = df_video2.fillna(df_video2.mode().iloc[0])
df_video2.head(50)
```

Out[284]:

	Cust ID	Gender	Income	Age	Rentals	Avg Per Visit	Incidentals	Genre
0	1	M	45000.0	25.0	27	2.5	Yes	Action
1	2	F	54000.0	33.0	12	3.4	No	Drama
2	3	F	32000.0	20.0	42	1.6	No	Comedy
3	4	F	59000.0	70.0	16	4.2	Yes	Drama
4	5	M	37000.0	35.0	25	3.2	Yes	Action
5	6	M	18000.0	20.0	33	1.7	No	Action
6	7	F	29000.0	45.0	19	3.8	No	Drama
7	8	M	74000.0	25.0	31	2.4	Yes	Action
8	9	M	29000.0	21.0	18	2.1	No	Comedy
9	10	F	65000.0	40.0	21	3.3	No	Drama
10	11	F	41000.0	22.0	48	2.3	Yes	Drama
11	12	F	26000.0	22.0	29	2.9	Yes	Action
12	13	M	83000.0	46.0	14	3.6	No	Comedy
13	14	M	45000.0	36.0	24	2.7	No	Drama
14	15	M	68000.0	25.0	36	2.7	Yes	Comedy
15	16	M	17000.0	19.0	26	2.2	Yes	Action
16	17	M	36000.0	35.0	28	3.5	Yes	Drama
17	18	F	29000.0	16.0	39	1.8	Yes	Action
18	19	F	24000.0	25.0	41	3.1	No	Comedy
19	20	M	12000.0	16.0	23	2.2	Yes	Drama
20	21	F	47000.0	52.0	11	3.1	No	Drama
21	22	M	25000.0	33.0	16	2.9	Yes	Drama
22	23	F	2000.0	15.0	30	2.5	No	Comedy
23	24	F	79000.0	35.0	22	3.8	Yes	Drama
24	25	M	1000.0	16.0	25	1.4	Yes	Comedy
25	26	F	56000.0	35.0	40	2.6	Yes	Action
26	27	F	62000.0	47.0	32	3.6	No	Drama
27	28	M	57000.0	52.0	22	4.1	No	Comedy
28	29	F	15000.0	18.0	37	2.1	Yes	Action
29	30	M	41000.0	25.0	17	1.4	Yes	Action
30	31	F	49000.0	56.0	15	3.2	No	Comedy
31	32	M	47000.0	30.0	21	3.1	Yes	Drama
32	33	M	23000.0	25.0	28	2.7	No	Action
33	34	F	29000.0	32.0	19	2.9	Yes	Action

	Cust ID	Gender	Income	Age	Rentals	Avg Per Visit	Incidentals	Genre
34	35	M	74000.0	29.0	43	4.6	Yes	Action
35	36	F	29000.0	21.0	34	2.3	No	Comedy
36	37	M	89000.0	46.0	12	1.2	No	Comedy
37	38	M	41000.0	38.0	20	3.3	Yes	Drama
38	39	F	68000.0	35.0	19	3.9	No	Comedy
39	40	M	17000.0	19.0	32	1.8	No	Action
40	41	F	50000.0	33.0	17	1.4	No	Drama
41	42	M	32000.0	25.0	26	2.2	Yes	Action
42	43	F	49000.0	28.0	48	3.3	Yes	Drama
43	44	M	35000.0	24.0	24	1.7	No	Drama
44	45	M	56000.0	25.0	30	3.5	Yes	Drama
45	46	F	57000.0	43.0	9	1.1	No	Drama
46	47	F	69000.0	35.0	22	2.8	Yes	Drama
47	48	F	52000.0	47.0	14	1.6	No	Drama
48	49	M	31000.0	25.0	42	3.4	Yes	Action
49	50	M	24000.0	20.0	33	4.7	No	Action