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

data = pd.read_csv(r'C:\Users\sharm\Downloads\
2725Swisspix.RarExtractorRarFileOpenerSimpleUnrarS_q68sgvev02mx6!App\
Black Friday Sales Prediction\Black Friday Sales.csv')

data.info
```

```
<bound method DataFrame.info of         User_ID Product_ID Gender
Age  Occupation City_Category  \
0       1000001  P00069042      F   0-17          10             A
1       1000001  P00248942      F   0-17          10             A
2       1000001  P00087842      F   0-17          10             A
3       1000001  P00085442      F   0-17          10             A
4       1000002  P00285442      M    55+          16             C
...         ...        ...    ...    ...         ...           ...
550063  1006033  P00372445      M  51-55          13             B
550064  1006035  P00375436      F  26-35           1             C
550065  1006036  P00375436      F  26-35          15             B
550066  1006038  P00375436      F    55+           1             C
550067  1006039  P00371644      F  46-50           0             B

       Stay_In_Current_City_Years  Marital_Status  Product_Category_1
\
0                              2               0                   3

1                              2               0                   1

2                              2               0                  12

3                              2               0                  12

4                             4+               0                   8

...                          ...             ...                 ...

550063                         1               1                  20

550064                         3               0                  20

550065                        4+               1                  20

550066                         2               0                  20

550067                        4+               1                  20


        Product_Category_2  Product_Category_3  Purchase
```

```
0                      NaN                  NaN      8370
1                      6.0                 14.0     15200
2                      NaN                  NaN      1422
3                     14.0                  NaN      1057
4                      NaN                  NaN      7969
...                    ...                  ...       ...
550063                 NaN                  NaN       368
550064                 NaN                  NaN       371
550065                 NaN                  NaN       137
550066                 NaN                  NaN       365
550067                 NaN                  NaN       490

[550068 rows x 12 columns]>
```

```
data.shape
```

```
(550068, 10)
```

```
data.describe()
```

```
               Gender            Age     Occupation  City_Category  \
count  550068.000000  550068.000000  550068.000000  550068.000000
mean        0.753105       2.496430       8.076707       1.042640
std         0.431205       1.353632       6.522660       0.760211
min         0.000000       0.000000       0.000000       0.000000
25%         1.000000       2.000000       2.000000       0.000000
50%         1.000000       2.000000       7.000000       1.000000
75%         1.000000       3.000000      14.000000       2.000000
max         1.000000       6.000000      20.000000       2.000000

       Marital_Status  Product_Category_1  Product_Category_2  \
count   550068.000000       550068.000000       550068.000000
mean         0.409653            5.404270            9.828967
std          0.491770            3.936211            4.207898
min          0.000000            1.000000            2.000000
25%          0.000000            1.000000            8.000000
50%          0.000000            5.000000            9.800000
75%          1.000000            8.000000           14.000000
max          1.000000           20.000000           18.000000

       Product_Category_3       Purchase
count       550068.000000  550068.000000
mean            12.662500    9263.968713
std              2.271833    5023.065394
min              3.000000      12.000000
25%             12.660000    5823.000000
50%             12.660000    8047.000000
75%             12.660000   12054.000000
max             18.000000   23961.000000
```

```
data.isnull().sum()
```

```
Gender                        0
Age                           0
Occupation                    0
City_Category                 0
Stay_In_Current_City_Years    0
Marital_Status                0
Product_Category_1            0
Product_Category_2            0
Product_Category_3            0
Purchase                      0
dtype: int64
```

```
data.dtypes
```

```
Gender                         int32
Age                            int32
Occupation                     int64
City_Category                  int32
Stay_In_Current_City_Years    object
Marital_Status                 int64
Product_Category_1             int64
Product_Category_2           float64
Product_Category_3           float64
Purchase                       int64
dtype: object
```

```
data['Product_Category_2'].mean()
```

```
9.828967327675853
```

```
 data['Product_Category_3'].mean()
```

```
12.662499945461288
```

```
data['Product_Category_2'].fillna(9.8, inplace=True)
data['Product_Category_3'].fillna(12.66, inplace=True)
```

```
data.isnull().sum()
```

```
Gender                        0
Age                           0
Occupation                    0
City_Category                 0
Stay_In_Current_City_Years    0
Marital_Status                0
Product_Category_1            0
Product_Category_2            0
Product_Category_3            0
Purchase                      0
dtype: int64
```

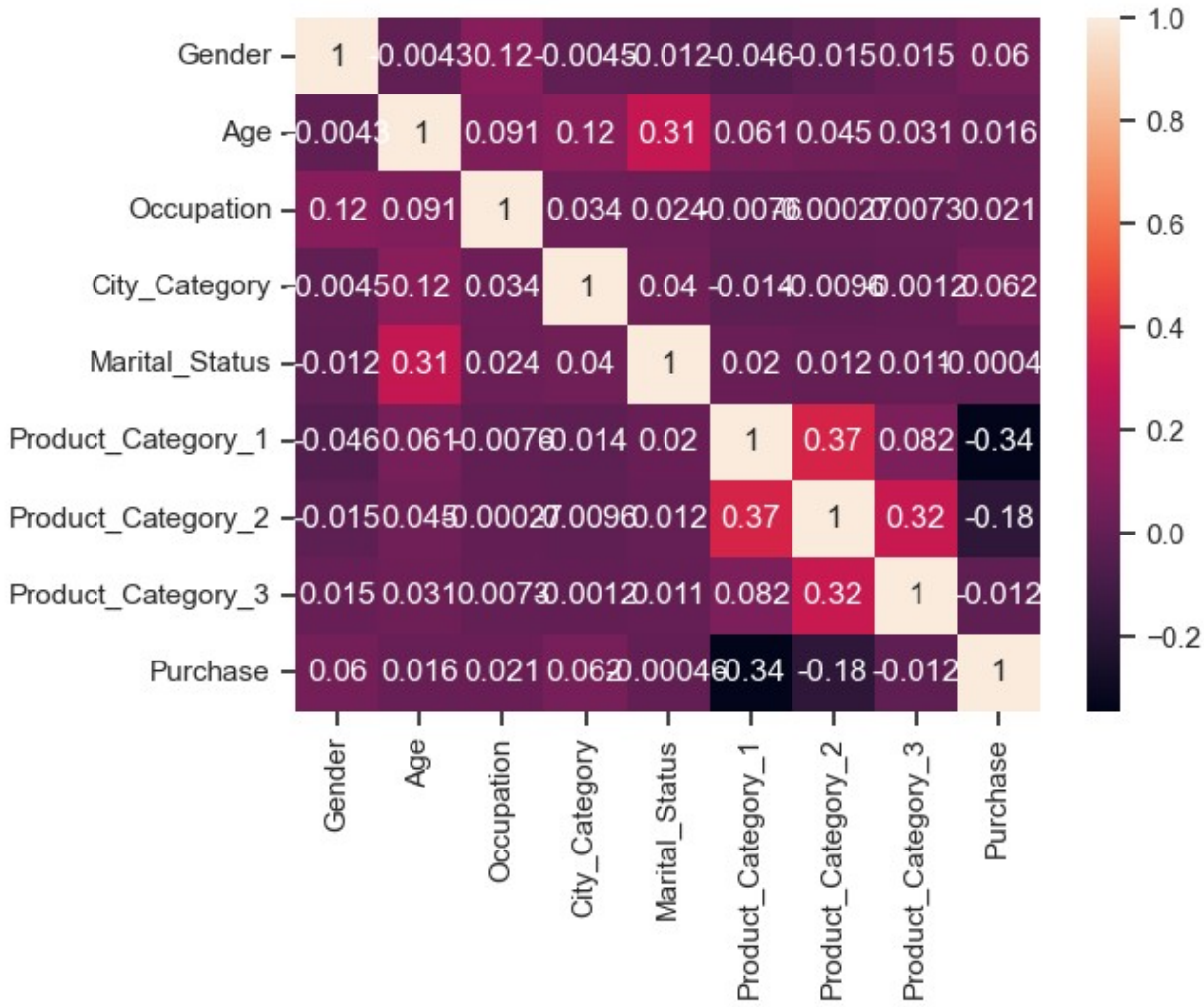
```
data.drop(['User_ID','Product_ID'], axis=1, inplace=True)

---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
Cell In[37], line 1
----> 1 data.drop(['User_ID','Product_ID'], axis=1, inplace=True)

File ~\New folder\lib\site-packages\pandas\util\_decorators.py:331, in deprecate_nonkeyword_arguments.<locals>.decorate.<locals>.wrapper(*args, **kwargs)
    325 if len(args) > num_allow_args:
    326     warnings.warn(
    327         msg.format(arguments=_format_argument_list(allow_args)),
    328         FutureWarning,
    329         stacklevel=find_stack_level(),
    330     )
--> 331 return func(*args, **kwargs)

File ~\New folder\lib\site-packages\pandas\core\frame.py:5399, in DataFrame.drop(self, labels, axis, index, columns, level, inplace, errors)
   5251 @deprecate_nonkeyword_arguments(version=None, allowed_args=["self", "labels"])
   5252 def drop(  # type: ignore[override]
   5253     self,
   (...)
   5260     errors: IgnoreRaise = "raise",
   5261 ) -> DataFrame | None:
   5262     """
   5263     Drop specified labels from rows or columns.
   5264
   (...)
   5397             weight  1.0     0.8
   5398     """
-> 5399     return super().drop(
   5400         labels=labels,
   5401         axis=axis,
   5402         index=index,
   5403         columns=columns,
   5404         level=level,
   5405         inplace=inplace,
   5406         errors=errors,
   5407     )

File ~\New folder\lib\site-packages\pandas\util\_decorators.py:331, in deprecate_nonkeyword_arguments.<locals>.decorate.<locals>.wrapper(*args, **kwargs)
```

```
    325 if len(args) > num_allow_args:
    326     warnings.warn(
    327
msg.format(arguments=_format_argument_list(allow_args)),
    328         FutureWarning,
    329         stacklevel=find_stack_level(),
    330     )
--> 331 return func(*args, **kwargs)

File ~\New folder\lib\site-packages\pandas\core\generic.py:4505, in
NDFrame.drop(self, labels, axis, index, columns, level, inplace,
errors)
   4503 for axis, labels in axes.items():
   4504     if labels is not None:
-> 4505         obj = obj._drop_axis(labels, axis, level=level,
errors=errors)
   4507 if inplace:
   4508     self._update_inplace(obj)

File ~\New folder\lib\site-packages\pandas\core\generic.py:4546, in
NDFrame._drop_axis(self, labels, axis, level, errors, only_slice)
   4544         new_axis = axis.drop(labels, level=level,
errors=errors)
   4545     else:
-> 4546         new_axis = axis.drop(labels, errors=errors)
   4547     indexer = axis.get_indexer(new_axis)
   4549 # Case for non-unique axis
   4550 else:

File ~\New folder\lib\site-packages\pandas\core\indexes\base.py:6934,
in Index.drop(self, labels, errors)
   6932 if mask.any():
   6933     if errors != "ignore":
-> 6934         raise KeyError(f"{list(labels[mask])} not found in
axis")
   6935     indexer = indexer[~mask]
   6936 return self.delete(indexer)

KeyError: "['User_ID', 'Product_ID'] not found in axis"
```

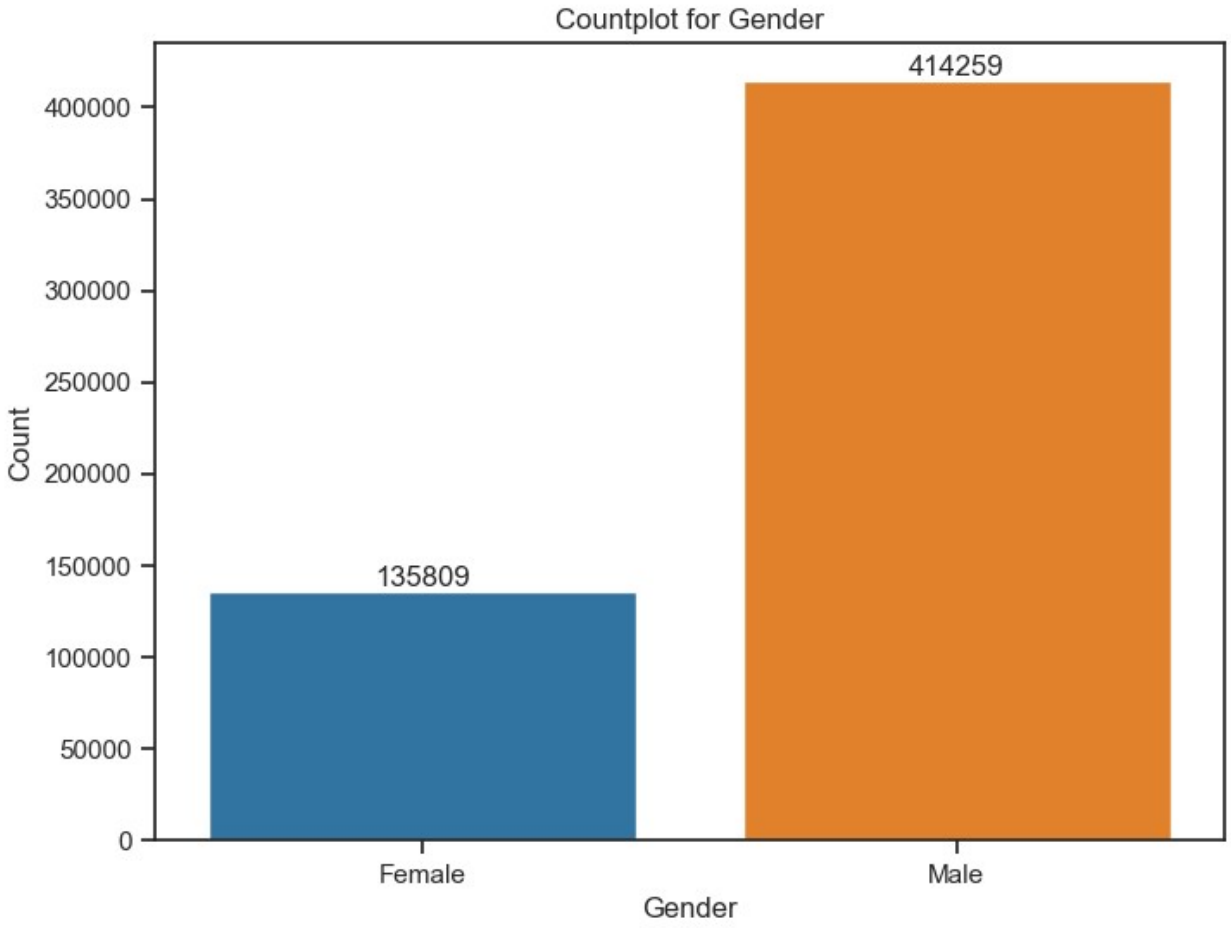
```
corr1 = data.corr()
sns.heatmap(corr1,annot=True)
```

```
C:\Users\sharm\AppData\Local\Temp\ipykernel_13708\1251888686.py:1:
FutureWarning: The default value of numeric_only in DataFrame.corr is
deprecated. In a future version, it will default to False. Select only
valid columns or specify the value of numeric_only to silence this
warning.
  corr1 = data.corr()
```

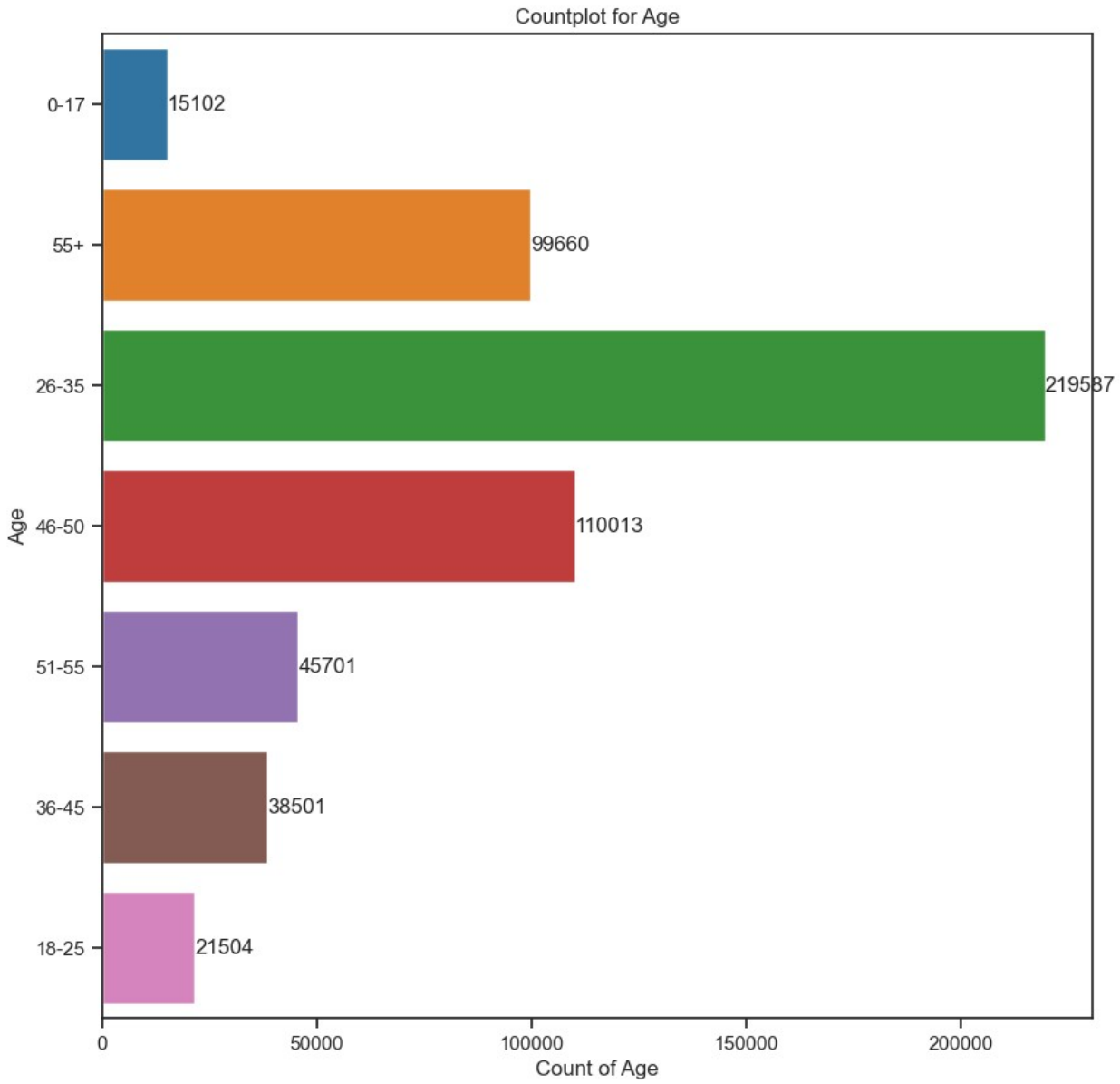
```
<Axes: >
```



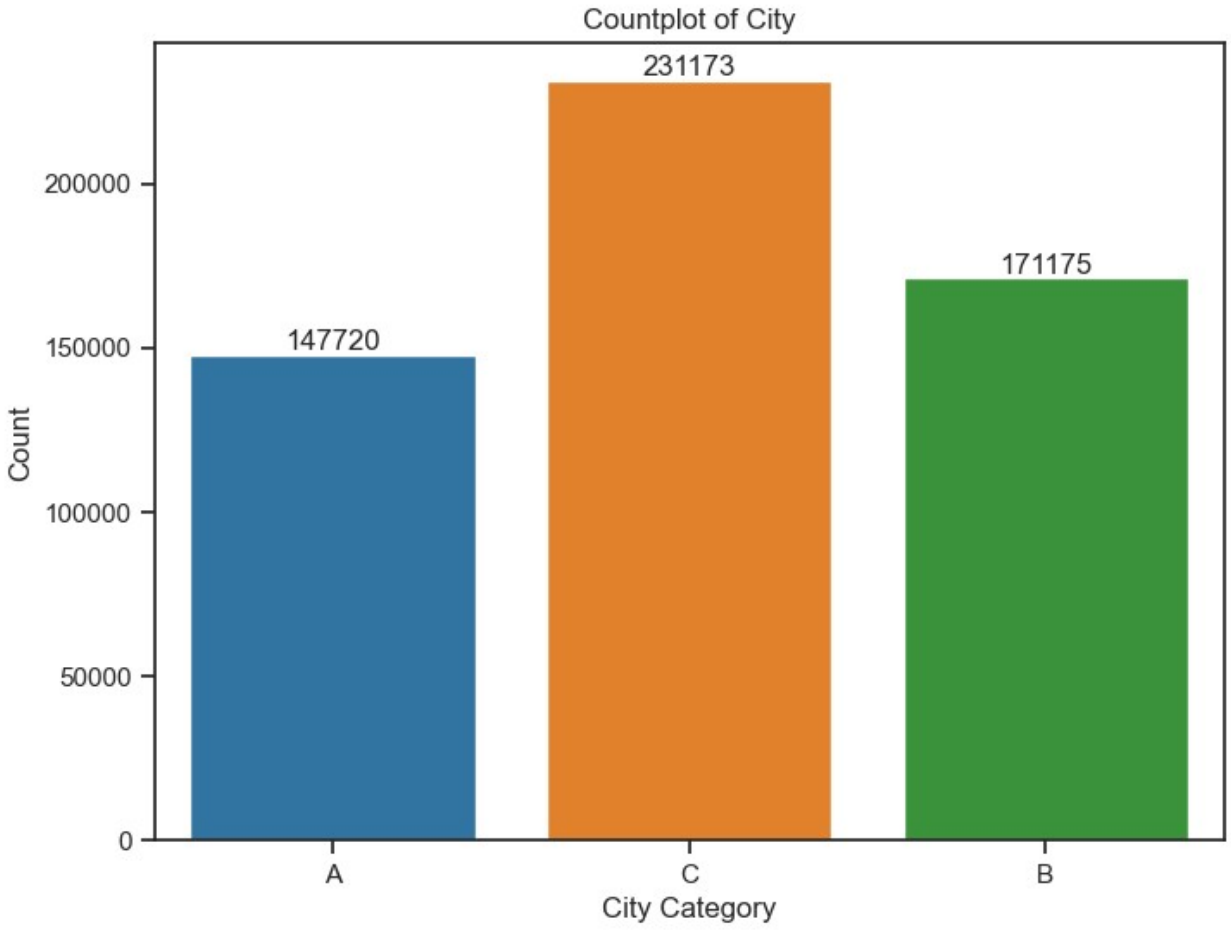

```
sns.set_theme(style='ticks',palette=None)
plt.figure(figsize=(8, 6))
ax = sns.countplot(x="Gender", data=data)
plt.title('Countplot for Gender')
plt.xlabel('Gender')
plt.ylabel('Count')
# Adding value labels to the bars
for bars in ax.containers:
    ax.bar_label(bars)
    ax.set_xticklabels(['Female', 'Male'])

plt.show()
```




```
sns.set_theme(style='ticks',palette=None)
plt.figure(figsize=(10, 10))
ax = sns.countplot(y="Age", data=data)
plt.title('Countplot for Age')
plt.xlabel('Count of Age')
plt.ylabel('Age')

for bars in ax.containers:
    ax.bar_label(bars)
    ax.set_yticklabels(['0-17','55+','26-35','46-50','51-55','36-
45','18-25'])
plt.show()
```




```
sns.set_theme(style='ticks',palette=None)
plt.figure(figsize=(8, 6))
ax = sns.countplot(x="City_Category", data=data)
plt.title('Countplot of City')
plt.xlabel('City Category')
plt.ylabel('Count')

for bars in ax.containers:
    ax.bar_label(bars)
    ax.set_xticklabels(['A','C','B'])


plt.show()
```




```
cat_var =
['Gender','Age','Occupation','City_Category','Stay_In_Current_City_Yea
rs','Marital_Status']

for x in cat_var:
    print('***************',x,'*****************')
    print(data[x].value_counts())
    print('****************************************')
```

```
*************** Gender *****************
1    414259
0    135809
Name: Gender, dtype: int64
****************************************
*************** Age *****************
2    219587
3    110013
1     99660
4     45701
5     38501
6     21504
```

```
0     15102
Name: Age, dtype: int64
**************************************
*************** Occupation ****************
4     72308
0     69638
7     59133
1     47426
17    40043
20    33562
12    31179
14    27309
2     26588
16    25371
6     20355
3     17650
10    12930
5     12177
15    12165
11    11586
19     8461
13     7728
18     6622
9      6291
8      1546
Name: Occupation, dtype: int64
**************************************
*************** City_Category ****************
1    231173
2    171175
0    147720
Name: City_Category, dtype: int64
**************************************
*************** Stay_In_Current_City_Years ****************
1    193821
2    101838
3     95285
4     84726
0     74398
Name: Stay_In_Current_City_Years, dtype: int64
**************************************
*************** Marital_Status ****************
0    324731
1    225337
Name: Marital_Status, dtype: int64
**************************************

 data['Stay_In_Current_City_Years'] =
data['Stay_In_Current_City_Years'].replace(to_replace="4+",value="4")
```

```
data.value_counts()

Gender  Age  Occupation  City_Category  Stay_In_Current_City_Years
Marital_Status  Product_Category_1  Product_Category_2
Product_Category_3  Purchase
1       2    2           0              0                           0
8                   9.8                 12.66                 8005
4
             1           1              1                           1
5                   9.8                 12.66                 6893
4
        0    10          1              4                           0
5                   9.8                 12.66                 8693
4
        2    2           1              1                           1
8                   9.8                 12.66                 5843
4
        1    4           2              1                           0
5                   9.8                 12.66                 6998
3

..
                         1              4                           0
20                  9.8                 12.66                 250
1

236         1

126         1

121         1
        6    20          2              3                           1
18                  9.8                 12.66                 3833
1
Length: 544807, dtype: int64

data.head()

   Gender  Age  Occupation  City_Category
Stay_In_Current_City_Years  \
0       0    0           10             0                          2

1       0    0           10             0                          2

2       0    0           10             0                          2

3       0    0           10             0                          2

4       1    6           16             2                          4
```

```
   Marital_Status  Product_Category_1  Product_Category_2
Product_Category_3  \
0               0                   3                 9.8
12.66
1               0                   1                 6.0
14.00
2               0                  12                 9.8
12.66
3               0                  12                14.0
12.66
4               0                   8                 9.8
12.66

   Purchase
0      8370
1     15200
2      1422
3      1057
4      7969
```

```
X = data.drop("Purchase",axis=1)
y = data["Purchase"]
```

```
 cat_var = ['Gender','Age','City_Category']
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
for var in cat_var:
    data[var] = le.fit_transform(data[var])
```

```
data.head(5)
```

```
   Gender  Age  Occupation  City_Category
Stay_In_Current_City_Years  \
0       0    0          10              0                           2

1       0    0          10              0                           2

2       0    0          10              0                           2

3       0    0          10              0                           2

4       1    6          16              2                           4


   Marital_Status  Product_Category_1  Product_Category_2
Product_Category_3  \
0               0                   3                 9.8
12.66
1               0                   1                 6.0
14.00
2               0                  12                 9.8
```

```
12.66
3               0                   12                 14.0
12.66
4               0                    8                  9.8
12.66
```

```
   Purchase
0      8370
1     15200
2      1422
3      1057
4      7969
```

```
X.shape
```

```
(550068, 9)
```

```
y.shape
```

```
(550068,)
```

```
from sklearn.preprocessing import StandardScaler
scale = StandardScaler()
X_transform = scale.fit_transform(X)
```

```
 from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2,random_state=27)
```

```
from sklearn.linear_model import LinearRegression
LR = LinearRegression()
LR.fit(X_train,y_train)
```

```
LinearRegression()
```

```
y_pred = LR.predict(X_test)
from sklearn.metrics import
mean_absolute_error,mean_squared_error,r2_score
error_score_test1 = np.sqrt(mean_squared_error(y_test,y_pred))
print('The mean squared error is: ',error_score_test1)
error_score_test2 = np.sqrt(mean_absolute_error(y_test,y_pred))
print('The mean absolute error is: ',error_score_test2)
error_score_test3 = np.sqrt(r2_score(y_test,y_pred))
print('The r2 score error is: ',error_score_test3)
```

```
The mean squared error is:  4694.357280165304
The mean absolute error is:  59.96477022706394
The r2 score error is:  0.360653650744712
```