Dharavath Ramdas

```
Code Github link: https://github.com/dharavathramdas101
linkedin link: https://www.linkedin.com/in/dharavath-ramdas-a283aa213/
```

Black Friday Dataset EDA and Feature engineering

Cleaning and preparing the data for model training

Importing required libraries

In [1]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
import warnings
warnings.filterwarnings('ignore')
```

Problem Statement

    Perform Black Friday Dataset EDA and Feature engineering

importing the train dataset

In [2]:

```
df_train = pd.read_csv(r"C:\Users\DHARAVATH RAMDAS\Downloads\archive (3)\train.csv")
df_train.head()
```

Out[2]:

	User_ID	Product_ID	Gender	Age	Occupation	City_Category	Stay_In_Current_City_Years	Marital_Status	Product_Category_1	Product_Category_2	Pı
0	1000001	P00069042	F	0-17	10	A	2	0	3	NaN	
1	1000001	P00248942	F	0-17	10	A	2	0	1	6.0	
2	1000001	P00087842	F	0-17	10	A	2	0	12	NaN	
3	1000001	P00085442	F	0-17	10	A	2	0	12	14.0	
4	1000002	P00285442	M	55+	16	C	4+	0	8	NaN	

import the test data

In [3]:

```
df_test = pd.read_csv(r"C:\Users\DHARAVATH RAMDAS\Downloads\archive (3)\test.csv")
df_test
```

Out[3]:

	User_ID	Product_ID	Gender	Age	Occupation	City_Category	Stay_In_Current_City_Years	Marital_Status	Product_Category_1	Product_Category_
0	1000004	P00128942	M	46-50	7	B	2	1	1	11
1	1000009	P00113442	M	26-35	17	C	0	0	3	5
2	1000010	P00288442	F	36-45	1	B	4+	1	5	14
3	1000010	P00145342	F	36-45	1	B	4+	1	4	9
4	1000011	P00053842	F	26-35	1	C	1	0	4	5
...	...	...	...	...	...	...	...	...	...	
233594	1006036	P00118942	F	26-35	15	B	4+	1	8	Na
233595	1006036	P00254642	F	26-35	15	B	4+	1	5	8
233596	1006036	P00031842	F	26-35	15	B	4+	1	1	5
233597	1006037	P00124742	F	46-50	1	C	4+	0	10	16
233598	1006039	P00316642	F	46-50	0	B	4+	1	4	5

233599 rows × 11 columns

Merge the both train and test data

In [4]:

```
df=df_train.append(df_test)
df.head()
```

Out[4]:

	User_ID	Product_ID	Gender	Age	Occupation	City_Category	Stay_In_Current_City_Years	Marital_Status	Product_Category_1	Product_Category_2	Pı
0	1000001	P00069042	F	0-17	10	A	2	0	3	NaN	
1	1000001	P00248942	F	0-17	10	A	2	0	1	6.0	
2	1000001	P00087842	F	0-17	10	A	2	0	12	NaN	
3	1000001	P00085442	F	0-17	10	A	2	0	12	14.0	
4	1000002	P00285442	M	55+	16	C	4+	0	8	NaN	

see information

In [5]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 783667 entries, 0 to 233598
Data columns (total 12 columns):
 #   Column                      Non-Null Count   Dtype  
---  ------                      --------------   -----  
 0   User_ID                     783667 non-null  int64  
 1   Product_ID                  783667 non-null  object 
 2   Gender                      783667 non-null  object 
 3   Age                         783667 non-null  object 
 4   Occupation                  783667 non-null  int64  
 5   City_Category               783667 non-null  object 
 6   Stay_In_Current_City_Years  783667 non-null  object 
 7   Marital_Status              783667 non-null  int64  
 8   Product_Category_1          783667 non-null  int64  
 9   Product_Category_2          537685 non-null  float64
 10  Product_Category_3          237858 non-null  float64
 11  Purchase                    550068 non-null  float64
dtypes: float64(3), int64(4), object(5)
memory usage: 77.7+ MB
```

user_id column is of no use so i will remove it

In [6]:

```
df.drop('User_ID',axis=1,inplace=True)
```

Describe for stats analysis

In [7]:

```
df.describe()
```

Out[7]:

	Occupation	Marital_Status	Product_Category_1	Product_Category_2	Product_Category_3	Purchase
count	783667.000000	783667.000000	783667.000000	537685.000000	237858.000000	550068.000000
mean	8.079300	0.409777	5.366196	9.844506	12.668605	9263.968713
std	6.522206	0.491793	3.878160	5.089093	4.125510	5023.065394
min	0.000000	0.000000	1.000000	2.000000	3.000000	12.000000
25%	2.000000	0.000000	1.000000	5.000000	9.000000	5823.000000
50%	7.000000	0.000000	5.000000	9.000000	14.000000	8047.000000
75%	14.000000	1.000000	8.000000	15.000000	16.000000	12054.000000
max	20.000000	1.000000	20.000000	18.000000	18.000000	23961.000000

In [8]:

```
df.head()
```

Out[8]:

	Product_ID	Gender	Age	Occupation	City_Category	Stay_In_Current_City_Years	Marital_Status	Product_Category_1	Product_Category_2	Product_Cat
0	P00069042	F	0-17	10	A	2	0	3	NaN	
1	P00248942	F	0-17	10	A	2	0	1	6.0	
2	P00087842	F	0-17	10	A	2	0	12	NaN	
3	P00085442	F	0-17	10	A	2	0	12	14.0	
4	P00285442	M	55+	16	C	4+	0	8	NaN	

Total number of categorical attributes

In [9]:

```
cat_col = df.select_dtypes(exclude=['int64','float64']).columns.size
print("total number of categorical attributes are :", cat_col)
```

```
total number of categorical attributes are : 5
```

Total number of numerical attributes

In [10]:

```
num_col = df.select_dtypes(exclude=['object']).columns.size
print("total number of numerical attributes are :", num_col)
```

```
total number of numerical attributes are : 6
```

In [11]:

```
pd.get_dummies(df['Gender'])
```

Out[11]:

	F	M
0	1	0
1	1	0
2	1	0
3	1	0
4	0	1
...	...	...
233594	1	0
233595	1	0
233596	1	0
233597	1	0
233598	1	0

783667 rows × 2 columns

In [12]:

```
##  feature
```

In [13]:

```
df['Gender'] = df['Gender'].map({'F':0,'M':1})
df.head()
```

Out[13]:

	Product_ID	Gender	Age	Occupation	City_Category	Stay_In_Current_City_Years	Marital_Status	Product_Category_1	Product_Category_2	Product_Cat
0	P00069042	0	0-17	10	A	2	0	3	NaN	
1	P00248942	0	0-17	10	A	2	0	1	6.0	
2	P00087842	0	0-17	10	A	2	0	12	NaN	
3	P00085442	0	0-17	10	A	2	0	12	14.0	
4	P00285442	1	55+	16	C	4+	0	8	NaN	

Handiling categorical feature age

In [14]:

```
df['Age'].unique()
```

Out[14]:

```
array(['0-17', '55+', '26-35', '46-50', '51-55', '36-45', '18-25'],
      dtype=object)
```

In [15]:

```
df['Age'] = df['Age'].map({'0-17':1,'18-25':2,'26-35':3,'36-45':4,'46:50':5,'51-55':6,'55+':7})
```

In [16]:

```
#pd.get_dummies(df['Age'],drop_first=True)
```

In [17]:

```
## Second technique
```

In [18]:

```
from sklearn import preprocessing
```

In [19]:

```
df.head()
```

Out[19]:

	Product_ID	Gender	Age	Occupation	City_Category	Stay_In_Current_City_Years	Marital_Status	Product_Category_1	Product_Category_2	Product_Cat
0	P00069042	0	1.0	10	A	2	0	3	NaN	
1	P00248942	0	1.0	10	A	2	0	1	6.0	
2	P00087842	0	1.0	10	A	2	0	12	NaN	
3	P00085442	0	1.0	10	A	2	0	12	14.0	
4	P00285442	1	7.0	16	C	4+	0	8	NaN	

In [20]:

```
df.isnull().sum()
```

Out[20]:

```
Product_ID                         0
Gender                             0
Age                            65278
Occupation                         0
City_Category                      0
Stay_In_Current_City_Years         0
Marital_Status                     0
Product_Category_1                 0
Product_Category_2            245982
Product_Category_3            545809
Purchase                      233599
dtype: int64
```

fill na

In [21]:

```
df.bfill(inplace=True)
```

In [22]:

```
df.ffill(inplace=True)
```

In [23]:

```
df.isnull().sum()
```

Out[23]:

```
Product_ID                    0
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
```

In [24]:

```
df.head()
```

Out[24]:

	Product_ID	Gender	Age	Occupation	City_Category	Stay_In_Current_City_Years	Marital_Status	Product_Category_1	Product_Category_2	Product_Cat
0	P00069042	0	1.0	10	A	2	0	3	6.0	
1	P00248942	0	1.0	10	A	2	0	1	6.0	
2	P00087842	0	1.0	10	A	2	0	12	14.0	
3	P00085442	0	1.0	10	A	2	0	12	14.0	
4	P00285442	1	7.0	16	C	4+	0	8	2.0	

In [ ]:

```

```

EDA

In [25]:

```
df.columns
```

Out[25]:

```
Index(['Product_ID', 'Gender', 'Age', 'Occupation', 'City_Category',
       'Stay_In_Current_City_Years', 'Marital_Status', 'Product_Category_1',
       'Product_Category_2', 'Product_Category_3', 'Purchase'],
      dtype='object')
```

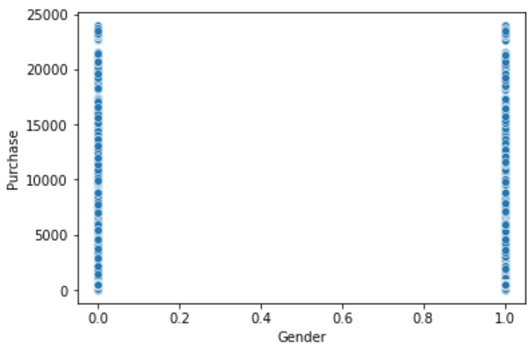
Gender vs Purchase

In [26]:

```
sns.scatterplot(x=df["Gender"],y=df["Purchase"],data=df)
```

Out[26]:

```
<AxesSubplot:xlabel='Gender', ylabel='Purchase'>
```



Gender counting male and female

In [27]:

```
df['Gender'].value_counts()
```

Out[27]:

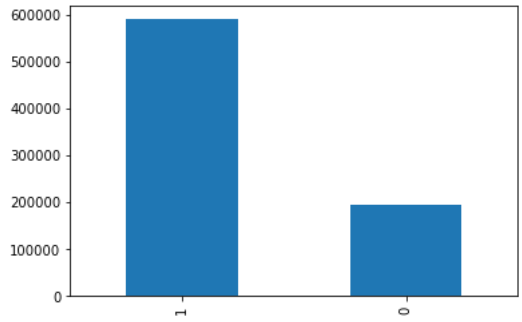
```
1    590031
0    193636
Name: Gender, dtype: int64
```

In [28]:

```
df['Gender'].value_counts().plot.bar()
```

Out[28]:

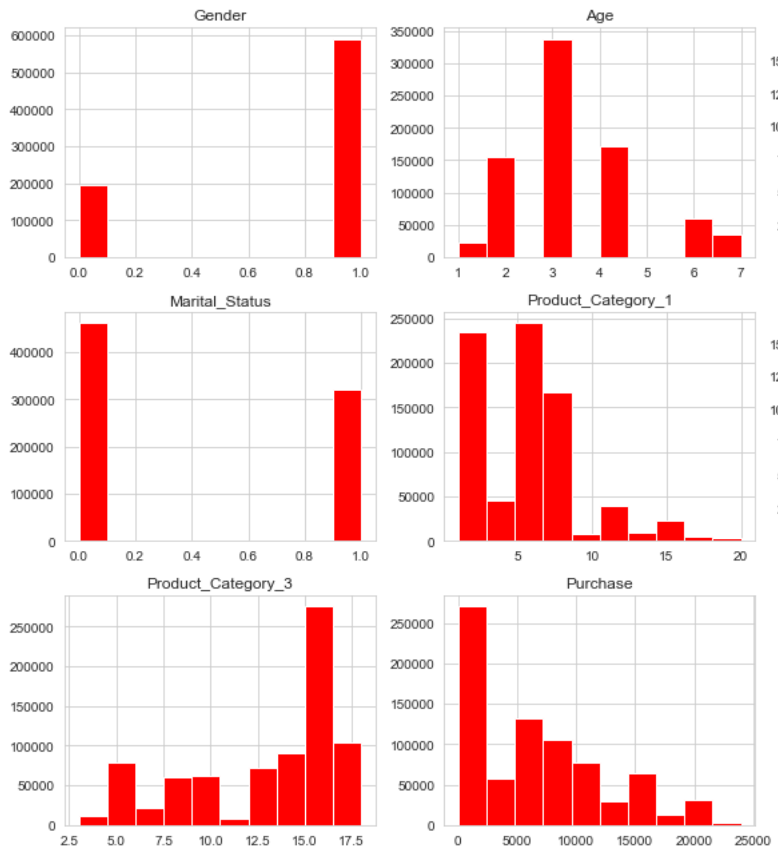
```
<AxesSubplot:>
```



Histogram of feature in dataset

In [29]:

```
sns.set_style('whitegrid')
df.hist(figsize=(12,9),color="r")
plt.tight_layout()
plt.show()
```

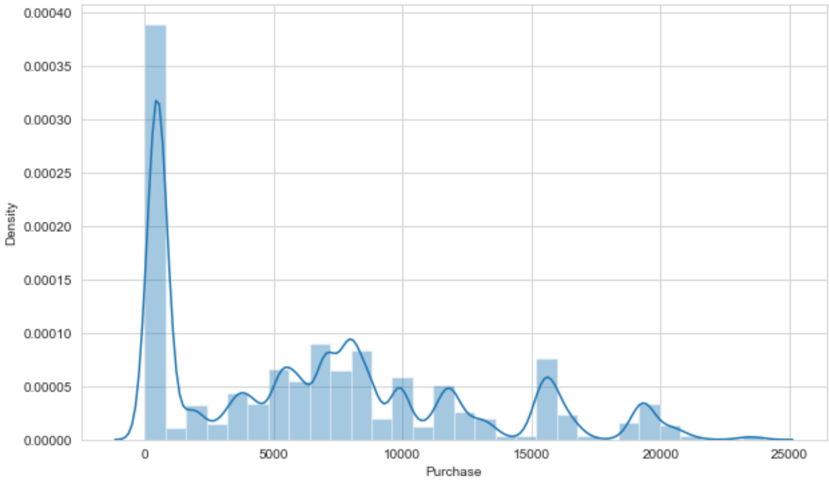


In [ ]:

Distribution of amount purchase

In [30]:

```
plt.figure(figsize=(10,6))
sns.set_style('whitegrid')
sns.distplot(df['Purchase'],kde=True,bins=30)
plt.show()
```



Countplot

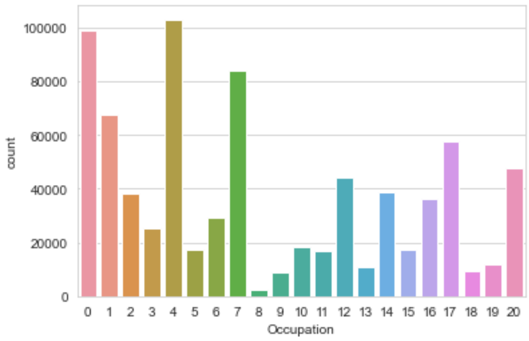
Countplot on Occupation

In [31]:

```
sns.countplot(x='Occupation',data=df)
```

Out[31]:

```
<AxesSubplot:xlabel='Occupation', ylabel='count'>
```



In [32]:

```
df['City_Category'].value_counts()
```

Out[32]:

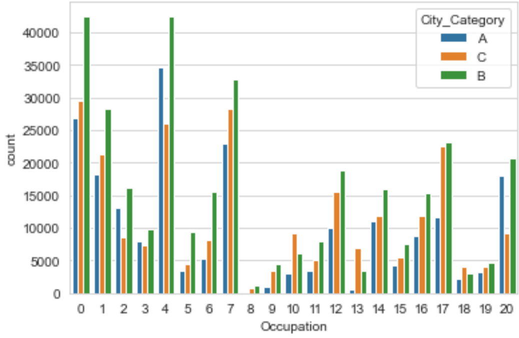
```
B    329739
C    243684
A    210244
Name: City_Category, dtype: int64
```

In [33]:

```
sns.countplot(x='Occupation',hue='City_Category',data=df)
```

Out[33]:

```
<AxesSubplot:xlabel='Occupation', ylabel='count'>
```



In [34]:

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

Out[34]:

```
3.0    337590
4.0    172297
2.0    155085
6.0     60682
7.0     34854
1.0     23159
Name: Age, dtype: int64
```

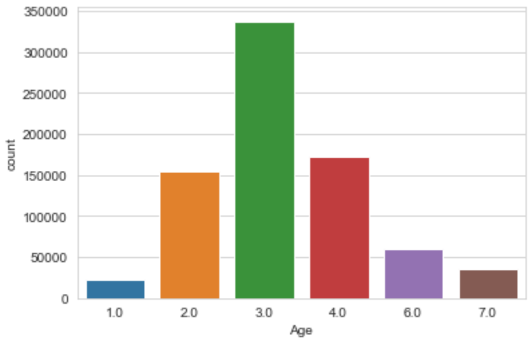
we have seven different groups

In [36]:

```
sns.countplot(x='Age',data=df)
```

Out[36]:

```
<AxesSubplot:xlabel='Age', ylabel='count'>
```



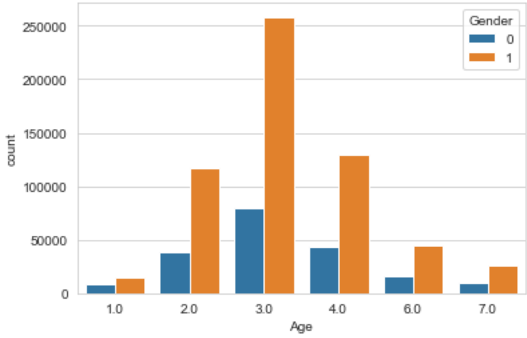
lets see the distribution of gender in agegroup

In [38]:

```
sns.countplot(x='Age',hue='Gender',data=df)
```

Out[38]:

```
<AxesSubplot:xlabel='Age', ylabel='count'>
```



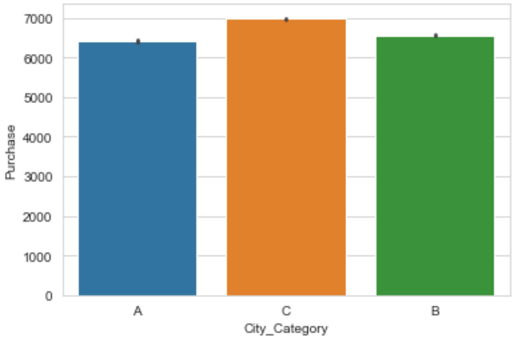
purchase vs city category

In [39]:

```
sns.barplot(x='City_Category',y='Purchase',data=df)
```

Out[39]:

```
<AxesSubplot:xlabel='City_Category', ylabel='Purchase'>
```



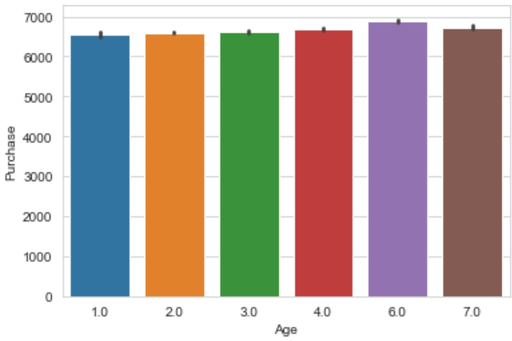
age vs purchase

In [40]:

```
sns.barplot(x='Age',y='Purchase',data=df)
```

Out[40]:

```
<AxesSubplot:xlabel='Age', ylabel='Purchase'>
```



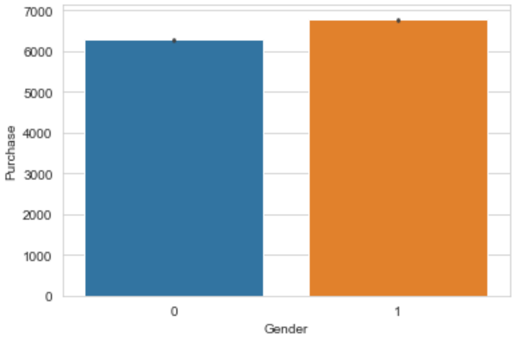
gender vs purchase

In [41]:

```
sns.barplot(x='Gender',y='Purchase',data=df)
```

Out[41]:

```
<AxesSubplot:xlabel='Gender', ylabel='Purchase'>
```



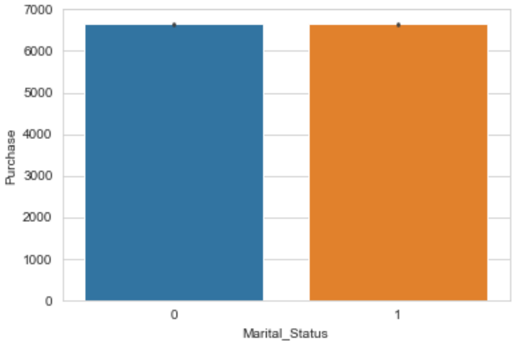
purchse vs marital status

In [42]:

```
sns.barplot(x='Marital_Status',y='Purchase',data=df)
```

Out[42]:

```
<AxesSubplot:xlabel='Marital_Status', ylabel='Purchase'>
```



we are plotting relationship between purchase and various other attribute

Marital_status vs gender

In [43]:

```
df.groupby('Marital_Status').agg({'Purchase':['max','min','mean']})
```

Out[43]:

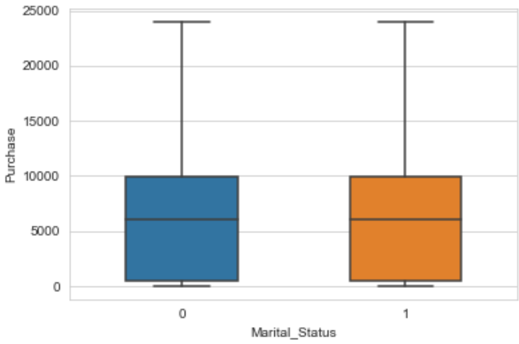
	Purchase		
	max	min	mean
Marital_Status			
0	23961.0	12.0	6651.243524
1	23961.0	12.0	6644.754522

In [44]:

```
sns.boxplot(x='Marital_Status',y='Purchase',data=df,width=0.5)
```

Out[44]:

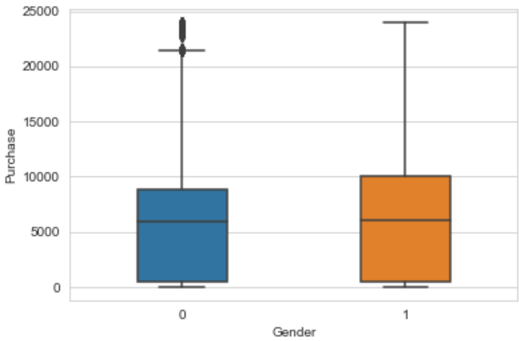
```
<AxesSubplot:xlabel='Marital_Status', ylabel='Purchase'>
```



Gender vs Purchse

In [45]:

```
sns.boxplot(x='Gender',y='Purchase',data=df,width= 0.4)
plt.show()
```



In [46]:

```
df.groupby('Gender').agg({'Purchase':['max','min','mean','median']})
```

Out[46]:

	Purchase			
	max	min	mean	median
Gender				
0	23959.0	12.0	6272.428020	5953.0
1	23961.0	12.0	6772.031266	6101.0

Age vs Purchase

In [47]:

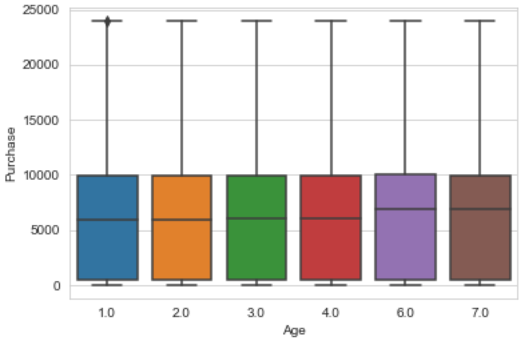
```
df.groupby('Age').agg({'Purchase':['max','min','mean','median']})
```

Out[47]:

	Purchase			
	max	min	mean	median
Age				
1.0	23960.0	12.0	6542.634829	5963.0
2.0	23958.0	12.0	6597.732282	5988.0
3.0	23961.0	12.0	6615.568681	6027.0
4.0	23960.0	12.0	6674.355822	6084.0
6.0	23960.0	12.0	6881.840875	6878.0
7.0	23960.0	12.0	6731.533741	6883.0

In [48]:

```
sns.boxplot(x='Age',y='Purchase',data=df,width=0.8)
plt.show()
```



Occupation vs Purchase

In [49]:

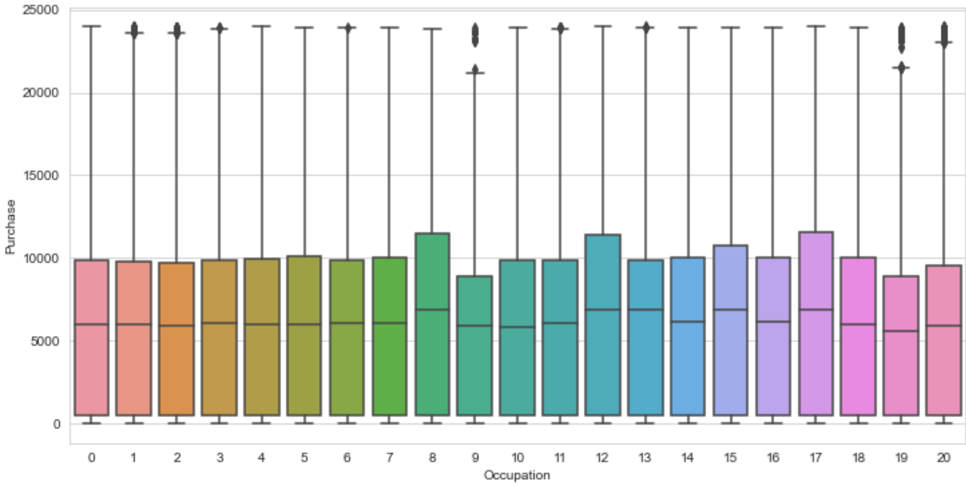
```
df.groupby('Occupation').agg({'Purchase':['max','min']})
```

Out[49]:

	Purchase	
	max	min
Occupation		
0	23961.0	12.0
1	23960.0	12.0
2	23955.0	12.0
3	23914.0	12.0
4	23961.0	12.0
5	23924.0	12.0
6	23951.0	12.0
7	23948.0	12.0
8	23869.0	14.0
9	23943.0	13.0
10	23955.0	12.0
11	23946.0	12.0
12	23960.0	12.0
13	23959.0	12.0
14	23941.0	12.0
15	23949.0	12.0
16	23947.0	12.0
17	23961.0	12.0
18	23894.0	12.0
19	23939.0	12.0
20	23960.0	12.0

In [50]:

```
plt.figure(figsize=(12,6))
sns.boxplot(x='Occupation',y='Purchase',data=df)
plt.show()
```



Purchase vs City_Category

In [51]:

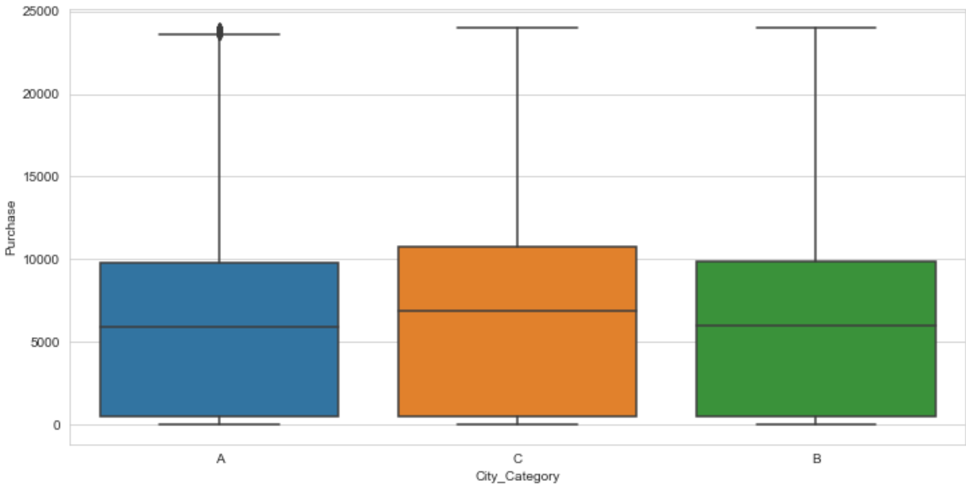
```
df.groupby('City_Category').agg({'Purchase':['max','min']})
```

Out[51]:

	Purchase	
	max	min
City_Category		
A	23961.0	12.0
B	23960.0	12.0
C	23961.0	12.0

In [52]:

```
plt.figure(figsize=(12,6))
sns.boxplot(y='Purchase',x='City_Category',data=df)
plt.show()
```



Purchase vs Marital status

In [53]:

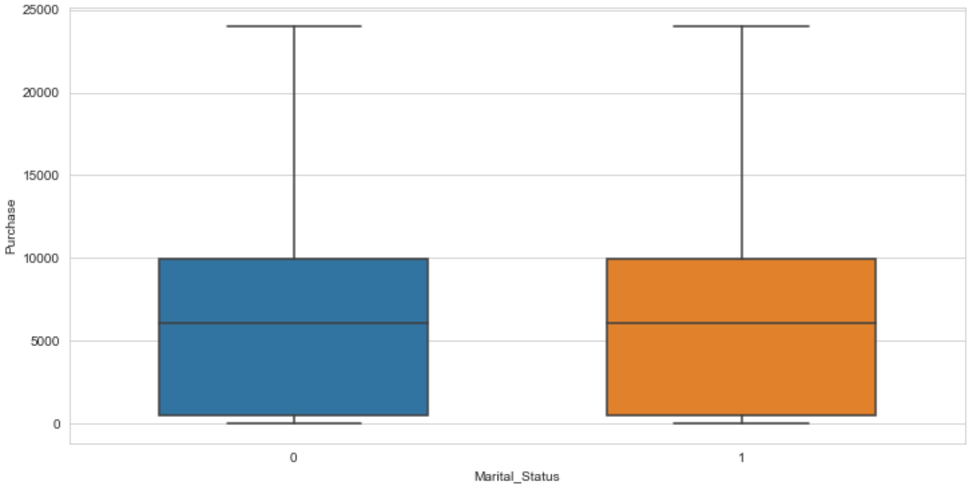
```
df.groupby('Marital_Status').agg({'Purchase':['max','min']})
```

Out[53]:

	Purchase	
	max	min
Marital_Status		
0	23961.0	12.0
1	23961.0	12.0

In [54]:

```
plt.figure(figsize=(12,6))
sns.boxplot(x='Marital_Status',y='Purchase',data=df,width=0.6)
plt.show()
```



Purchase vs Stay_in_current city year

In [55]:

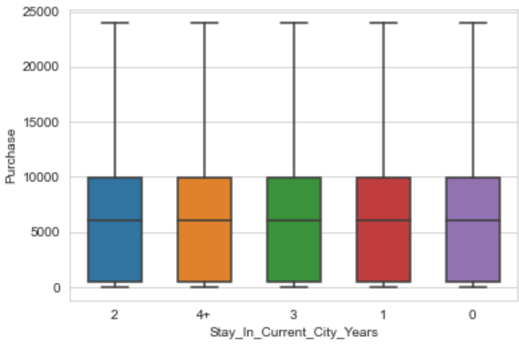
```
df.groupby('Stay_In_Current_City_Years').agg({'Purchase':['max','min']})
```

Out[55]:

	Purchase	
	max	min
Stay_In_Current_City_Years		
0	23960.0	12.0
1	23961.0	12.0
2	23961.0	12.0
3	23961.0	12.0
4+	23958.0	12.0

In [56]:

```
sns.boxplot(x='Stay_In_Current_City_Years',y='Purchase',data=df,width=0.6)
plt.show()
```



In [ ]:

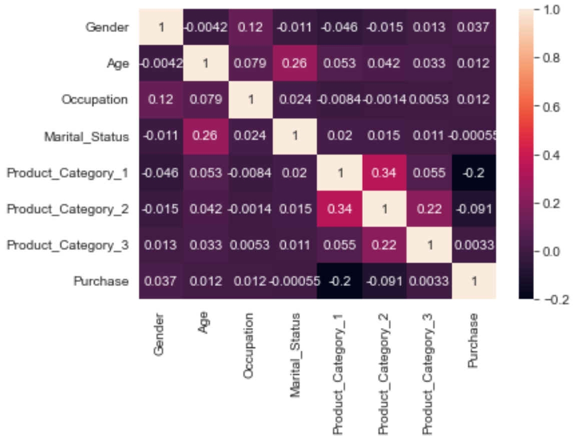
Correlation

In [57]:

```
corr = df.corr()
sns.heatmap(corr,annot=True)
```

Out[57]:

```
<AxesSubplot:>
```



In [ ]:

In [58]:

```
## Label Encoding
```

In [59]:

```
df.head()
```

Out[59]:

	Product_ID	Gender	Age	Occupation	City_Category	Stay_In_Current_City_Years	Marital_Status	Product_Category_1	Product_Category_2	Product_Cat
0	P00069042	0	1.0	10	A	2	0	3	6.0	
1	P00248942	0	1.0	10	A	2	0	1	6.0	
2	P00087842	0	1.0	10	A	2	0	12	14.0	
3	P00085442	0	1.0	10	A	2	0	12	14.0	
4	P00285442	1	7.0	16	C	4+	0	8	2.0	

In [60]:

```
df.drop('Product_ID',axis=1)
```

Out[60]:

	Gender	Age	Occupation	City_Category	Stay_In_Current_City_Years	Marital_Status	Product_Category_1	Product_Category_2	Product_Category_:
0	0	1.0	10	A	2	0	3	6.0	14.(
1	0	1.0	10	A	2	0	1	6.0	14.(
2	0	1.0	10	A	2	0	12	14.0	17.(
3	0	1.0	10	A	2	0	12	14.0	17.(
4	1	7.0	16	C	4+	0	8	2.0	17.(
...	...	...	...	...	...	...	...	...	.
233594	0	3.0	15	B	4+	1	8	8.0	12.(
233595	0	3.0	15	B	4+	1	5	8.0	12.(
233596	0	3.0	15	B	4+	1	1	5.0	12.(
233597	0	3.0	1	C	4+	0	10	16.0	12.(
233598	0	3.0	0	B	4+	1	4	5.0	12.(

783667 rows × 10 columns

Checking shape of data

In [61]:

```
df.shape
```

Out[61]:

```
(783667, 11)
```

In [62]:

```
df_gender = pd.get_dummies(df['Gender'])
df_age = pd.get_dummies(df['Age'])
df_city_category = pd.get_dummies(df['City_Category'])
df_stay_in_current_city_years = pd.get_dummies(df['Stay_In_Current_City_Years'])

df_final = pd.concat([df,df_gender,df_age ,df_city_category,df_stay_in_current_city_years],axis=1)

df_final.head()
```

Out[62]:

	Product_ID	Gender	Age	Occupation	City_Category	Stay_In_Current_City_Years	Marital_Status	Product_Category_1	Product_Category_2	Product_Cat
0	P00069042	0	1.0	10	A	2	0	3	6.0	
1	P00248942	0	1.0	10	A	2	0	1	6.0	
2	P00087842	0	1.0	10	A	2	0	12	14.0	
3	P00085442	0	1.0	10	A	2	0	12	14.0	
4	P00285442	1	7.0	16	C	4+	0	8	2.0	

5 rows × 27 columns

In [63]:

```
df_final = df_final.drop(['Gender','Age','City_Category','Stay_In_Current_City_Years'],axis=1)
df_final
```

Out[63]:

	Product_ID	Occupation	Marital_Status	Product_Category_1	Product_Category_2	Product_Category_3	Purchase	0	1	1.0	...	6.0	7.0	A	B
0	P00069042	10	0	3	6.0	14.0	8370.0	1	0	1	...	0	0	1	0
1	P00248942	10	0	1	6.0	14.0	15200.0	1	0	1	...	0	0	1	0
2	P00087842	10	0	12	14.0	17.0	1422.0	1	0	1	...	0	0	1	0
3	P00085442	10	0	12	14.0	17.0	1057.0	1	0	1	...	0	0	1	0
4	P00285442	16	0	8	2.0	17.0	7969.0	0	1	0	...	0	1	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
233594	P00118942	15	1	8	8.0	12.0	490.0	1	0	0	...	0	0	0	1
233595	P00254642	15	1	5	8.0	12.0	490.0	1	0	0	...	0	0	0	1
233596	P00031842	15	1	1	5.0	12.0	490.0	1	0	0	...	0	0	0	1
233597	P00124742	1	0	10	16.0	12.0	490.0	1	0	0	...	0	0	0	0
233598	P00316642	0	1	4	5.0	12.0	490.0	1	0	0	...	0	0	0	1

783667 rows × 23 columns

In [64]:

```
df_final.drop('Product_ID',axis=1,inplace=True)
```

In [65]:

```
df_final.dtypes
```

Out[65]:

```
Occupation              int64
Marital_Status          int64
Product_Category_1      int64
Product_Category_2    float64
Product_Category_3    float64
Purchase              float64
0                       uint8
1                       uint8
1.0                     uint8
2.0                     uint8
3.0                     uint8
4.0                     uint8
6.0                     uint8
7.0                     uint8
A                       uint8
B                       uint8
C                       uint8
0                       uint8
1                       uint8
2                       uint8
3                       uint8
4+                      uint8
dtype: object
```

In [ ]:

In [ ]:

In [ ]:

dividing dataset into test and train

In [67]:

```
X = df_final.drop('Purchase',axis=1)
y = df_final['Purchase']
```

In [68]:

```
from sklearn.model_selection import train_test_split
```

In [69]:

```
X_train,X_test,y_train,y_test = train_test_split(X,y, test_size=0.2,random_state= 0)
```

In [70]:

```
print(X_train.shape,y_train.shape)
```

```
(626933, 21) (626933,)
```

In [71]:

```
print(X_test.shape,y_test.shape)
```

```
(156734, 21) (156734,)
```

Feature Scaling

In [73]:

```
from sklearn.preprocessing import StandardScaler
```

In [74]:

```
scaler = StandardScaler()
```

In [75]:

```
X_train = scaler.fit_transform(X_train)
```

In [76]:

```
X_test = scaler.transform(X_test)
```

In [ ]:

In [ ]:

In [ ]: