In [1]:
```python
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
```

In [2]:
```python
df_train=pd.read_csv("train.csv")
df_train
```

Out[2]:

	User_ID	Product_ID	Gender	Age	Occupation	City_Category	Stay_In_Current_City_Years	Marital_Status	Product_Category_1	Product_Category_2	Product_Category_3	Purchase
0	1000001	P00069042	F	0-17	10	A	2	0	3	NaN	NaN	8370
1	1000001	P00248942	F	0-17	10	A	2	0	1	6.0	14.0	15200
2	1000001	P00087842	F	0-17	10	A	2	0	12	NaN	NaN	1422
3	1000001	P00085442	F	0-17	10	A	2	0	12	14.0	NaN	1057
4	1000002	P00285442	M	55+	16	C	4+	0	8	NaN	NaN	7969
...	...	...	...	...	...	...	...	...	...	...	...	...
550063	1006033	P00372445	M	51-55	13	B	1	1	20	NaN	NaN	368
550064	1006035	P00375436	F	26-35	1	C	3	0	20	NaN	NaN	371
550065	1006036	P00375436	F	26-35	15	B	4+	1	20	NaN	NaN	137
550066	1006038	P00375436	F	55+	1	C	2	0	20	NaN	NaN	365
550067	1006039	P00371644	F	46-50	0	B	4+	1	20	NaN	NaN	490

550068 rows × 12 columns

In [3]:
```python
df_test=pd.read_csv('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_2	Product_Category_3
0	1000004	P00128942	M	46-50	7	B	2	1	1	11.0	NaN
1	1000009	P00113442	M	26-35	17	C	0	0	3	5.0	NaN
2	1000010	P00288442	F	36-45	1	B	4+	1	5	14.0	NaN
3	1000010	P00145342	F	36-45	1	B	4+	1	4	9.0	NaN
4	1000011	P00053842	F	26-35	1	C	1	0	4	5.0	12.0
...	...	...	...	...	...	...	...	...	...	...	...
233594	1006036	P00118942	F	26-35	15	B	4+	1	8	NaN	NaN
233595	1006036	P00254642	F	26-35	15	B	4+	1	5	8.0	NaN
233596	1006036	P00031842	F	26-35	15	B	4+	1	1	5.0	12.0
233597	1006037	P00124742	F	46-50	1	C	4+	0	10	16.0	NaN
233598	1006039	P00316642	F	46-50	0	B	4+	1	4	5.0	NaN

233599 rows × 11 columns

In [4]:
```python
df=df_train.append([df_test])
```

```
C:\Users\VIVEK PANDEY\AppData\Local\Temp\ipykernel_7612\1970076644.py:1: FutureWarning: The frame.append method is deprecated and will be removed from pandas in a future version. U
se pandas.concat instead.
  df=df_train.append([df_test])
```

In [5]:
```python
df.head()
```

Out[5]:

	User_ID	Product_ID	Gender	Age	Occupation	City_Category	Stay_In_Current_City_Years	Marital_Status	Product_Category_1	Product_Category_2	Product_Category_3	Purchase
0	1000001	P00069042	F	0-17	10	A	2	0	3	NaN	NaN	8370.0
1	1000001	P00248942	F	0-17	10	A	2	0	1	6.0	14.0	15200.0
2	1000001	P00087842	F	0-17	10	A	2	0	12	NaN	NaN	1422.0
3	1000001	P00085442	F	0-17	10	A	2	0	12	14.0	NaN	1057.0
4	1000002	P00285442	M	55+	16	C	4+	0	8	NaN	NaN	7969.0

In [6]:
```python
df.shape
```

Out[6]:
```
(783667, 12)
```

In [7]:
```python
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
```

In [8]:
```python
df.isnull().sum()
```

Out[8]:
```
User_ID                            0
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            245982
Product_Category_3            545809
Purchase                      233599
dtype: int64
```

In [9]:
```python
df['Gender']
```

Out[9]:
```
0         F
1         F
2         F
3         F
4         M
         ..
233594    F
233595    F
233596    F
233597    F
233598    F
Name: Gender, Length: 783667, dtype: object
```

In [10]:
```python
df['Gender']=df['Gender'].map({'F':0,'M':1})
```

In [11]:
```python
df.head()
```

Out[11]:

	User_ID	Product_ID	Gender	Age	Occupation	City_Category	Stay_In_Current_City_Years	Marital_Status	Product_Category_1	Product_Category_2	Product_Category_3	Purchase
0	1000001	P00069042	0	0-17	10	A	2	0	3	NaN	NaN	8370.0
1	1000001	P00248942	0	0-17	10	A	2	0	1	6.0	14.0	15200.0
2	1000001	P00087842	0	0-17	10	A	2	0	12	NaN	NaN	1422.0
3	1000001	P00085442	0	0-17	10	A	2	0	12	14.0	NaN	1057.0
4	1000002	P00285442	1	55+	16	C	4+	0	8	NaN	NaN	7969.0

In [12]:
```python
df['Age'].unique()
```

Out[12]:
```
array(['0-17', '55+', '26-35', '46-50', '51-55', '36-45', '18-25'],
      dtype=object)
```

In [13]:
```python
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 [14]:
```python
df.head()
```

Out[14]:

	User_ID	Product_ID	Gender	Age	Occupation	City_Category	Stay_In_Current_City_Years	Marital_Status	Product_Category_1	Product_Category_2	Product_Category_3	Purchase
0	1000001	P00069042	0	1	10	A	2	0	3	NaN	NaN	8370.0
1	1000001	P00248942	0	1	10	A	2	0	1	6.0	14.0	15200.0
2	1000001	P00087842	0	1	10	A	2	0	12	NaN	NaN	1422.0
3	1000001	P00085442	0	1	10	A	2	0	12	14.0	NaN	1057.0
4	1000002	P00285442	1	7	16	C	4+	0	8	NaN	NaN	7969.0

In [15]:
```python
df_city=pd.get_dummies(df['City_Category'],drop_first=True)
```

In [16]:
```python
df_city.head()
```

Out[16]:

	B	C
0	0	0
1	0	0
2	0	0
3	0	0
4	0	1

In [17]:
```python
df=pd.concat([df,df_city],axis=1)
```

In [18]:
```python
df.head()
```

Out[18]:

	User_ID	Product_ID	Gender	Age	Occupation	City_Category	Stay_In_Current_City_Years	Marital_Status	Product_Category_1	Product_Category_2	Product_Category_3	Purchase	B	C
0	1000001	P00069042	0	1	10	A	2	0	3	NaN	NaN	8370.0	0	0
1	1000001	P00248942	0	1	10	A	2	0	1	6.0	14.0	15200.0	0	0
2	1000001	P00087842	0	1	10	A	2	0	12	NaN	NaN	1422.0	0	0
3	1000001	P00085442	0	1	10	A	2	0	12	14.0	NaN	1057.0	0	0
4	1000002	P00285442	1	7	16	C	4+	0	8	NaN	NaN	7969.0	0	1

In [19]:
```python
df.drop('User_ID',axis=1,inplace=True)
```

In [20]:
```python
df.head()
```

Out[20]:

	Product_ID	Gender	Age	Occupation	City_Category	Stay_In_Current_City_Years	Marital_Status	Product_Category_1	Product_Category_2	Product_Category_3	Purchase	B	C
0	P00069042	0	1	10	A	2	0	3	NaN	NaN	8370.0	0	0
1	P00248942	0	1	10	A	2	0	1	6.0	14.0	15200.0	0	0
2	P00087842	0	1	10	A	2	0	12	NaN	NaN	1422.0	0	0
3	P00085442	0	1	10	A	2	0	12	14.0	NaN	1057.0	0	0
4	P00285442	1	7	16	C	4+	0	8	NaN	NaN	7969.0	0	1

In [21]:
```python
df.drop('City_Category',axis=1,inplace=True)
```

In [22]:
```python
df.head()
```

Out[22]:

	Product_ID	Gender	Age	Occupation	Stay_In_Current_City_Years	Marital_Status	Product_Category_1	Product_Category_2	Product_Category_3	Purchase	B	C
0	P00069042	0	1	10	2	0	3	NaN	NaN	8370.0	0	0
1	P00248942	0	1	10	2	0	1	6.0	14.0	15200.0	0	0
2	P00087842	0	1	10	2	0	12	NaN	NaN	1422.0	0	0
3	P00085442	0	1	10	2	0	12	14.0	NaN	1057.0	0	0
4	P00285442	1	7	16	4+	0	8	NaN	NaN	7969.0	0	1

In [23]:
```python
df.shape
```

Out[23]:
```
(783667, 12)
```

In [24]:
```python
df.isnull().sum()
```

Out[24]:
```
Product_ID                         0
Gender                             0
Age                                0
Occupation                         0
Stay_In_Current_City_Years         0
Marital_Status                     0
Product_Category_1                 0
Product_Category_2            245982
Product_Category_3            545809
Purchase                      233599
B                                  0
C                                  0
dtype: int64
```

In [1]:
```python
#here i found 3 column that have null data.
```

In [25]:
```python
#Handling Missing value in 'Product_Category_2'
df['Product_Category_2'].value_counts()
```

Out[25]:
```
8.0     91317
14.0    78834
2.0     70498
16.0    61687
15.0    54114
5.0     37165
4.0     36705
6.0     23575
11.0    20230
17.0    19104
13.0    15054
9.0      8177
12.0     7801
10.0     4420
3.0      4123
18.0     4027
7.0       854
Name: Product_Category_2, dtype: int64
```

In [26]:
```python
df['Product_Category_2'].mode()
```

Out[26]:
```
0    8.0
Name: Product_Category_2, dtype: float64
```

In [27]:
```python
df['Product_Category_2']=df['Product_Category_2'].fillna(df['Product_Category_2'].mode()[0])
```

In [28]:
```python
df.head()
```

Out[28]:

	Product_ID	Gender	Age	Occupation	Stay_In_Current_City_Years	Marital_Status	Product_Category_1	Product_Category_2	Product_Category_3	Purchase	B	C
0	P00069042	0	1	10	2	0	3	8.0	NaN	8370.0	0	0
1	P00248942	0	1	10	2	0	1	6.0	14.0	15200.0	0	0
2	P00087842	0	1	10	2	0	12	8.0	NaN	1422.0	0	0
3	P00085442	0	1	10	2	0	12	14.0	NaN	1057.0	0	0
4	P00285442	1	7	16	4+	0	8	8.0	NaN	7969.0	0	1

In [29]:
```python
#Handling Missing value in 'Product_Category_3'
df['Product_Category_3'].value_counts()
```

Out[29]:
```
16.0    46469
15.0    39968
14.0    26283
17.0    23818
5.0     23799
8.0     17861
9.0     16532
12.0    13115
13.0     7849
6.0      6888
18.0     6621
4.0      2691
11.0     2585
10.0     2501
3.0       878
Name: Product_Category_3, dtype: int64
```

In [30]:
```python
df['Product_Category_3'].mode()
```

Out[30]:
```
0    16.0
Name: Product_Category_3, dtype: float64
```

In [31]:
```python
df['Product_Category_3']=df['Product_Category_3'].fillna(df['Product_Category_3'].mode()[0])
```

In [32]:
```python
df.head()
```

Out[32]:

	Product_ID	Gender	Age	Occupation	Stay_In_Current_City_Years	Marital_Status	Product_Category_1	Product_Category_2	Product_Category_3	Purchase	B	C
0	P00069042	0	1	10	2	0	3	8.0	16.0	8370.0	0	0
1	P00248942	0	1	10	2	0	1	6.0	14.0	15200.0	0	0
2	P00087842	0	1	10	2	0	12	8.0	16.0	1422.0	0	0
3	P00085442	0	1	10	2	0	12	14.0	16.0	1057.0	0	0
4	P00285442	1	7	16	4+	0	8	8.0	16.0	7969.0	0	1

In [33]:
```python
df.shape
```

Out[33]:
```
(783667, 12)
```

In [34]:
```python
df['Stay_In_Current_City_Years'].unique()
```

Out[34]:
```
array(['2', '4+', '3', '1', '0'], dtype=object)
```

In [35]:
```python
df['Stay_In_Current_City_Years']=df['Stay_In_Current_City_Years'].str.replace('+','')
```

```
C:\Users\VIVEK PANDEY\AppData\Local\Temp\ipykernel_7612\2063355665.py:1: FutureWarning: The default value of regex will change from True to False in a future version. In addition,
single character regular expressions will *not* be treated as literal strings when regex=True.
  df['Stay_In_Current_City_Years']=df['Stay_In_Current_City_Years'].str.replace('+','')
```

In [36]:
```python
df.head()
```

Out[36]:

	Product_ID	Gender	Age	Occupation	Stay_In_Current_City_Years	Marital_Status	Product_Category_1	Product_Category_2	Product_Category_3	Purchase	B	C
0	P00069042	0	1	10	2	0	3	8.0	16.0	8370.0	0	0
1	P00248942	0	1	10	2	0	1	6.0	14.0	15200.0	0	0
2	P00087842	0	1	10	2	0	12	8.0	16.0	1422.0	0	0
3	P00085442	0	1	10	2	0	12	14.0	16.0	1057.0	0	0
4	P00285442	1	7	16	4	0	8	8.0	16.0	7969.0	0	1

In [37]:
```python
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 783667 entries, 0 to 233598
Data columns (total 12 columns):
 #   Column                      Non-Null Count   Dtype  
---  ------                      --------------   -----  
 0   Product_ID                  783667 non-null  object 
 1   Gender                      783667 non-null  int64  
 2   Age                         783667 non-null  int64  
 3   Occupation                  783667 non-null  int64  
 4   Stay_In_Current_City_Years  783667 non-null  object 
 5   Marital_Status              783667 non-null  int64  
 6   Product_Category_1          783667 non-null  int64  
 7   Product_Category_2          783667 non-null  float64
 8   Product_Category_3          783667 non-null  float64
 9   Purchase                    550068 non-null  float64
 10  B                           783667 non-null  uint8  
 11  C                           783667 non-null  uint8  
dtypes: float64(3), int64(5), object(2), uint8(2)
memory usage: 67.3+ MB
```

In [38]:
```python
#Convert Object into Integers
df['Stay_In_Current_City_Years']=df['Stay_In_Current_City_Years'].astype(int)
```

In [39]:
```python
df.head()
```

Out[39]:

	Product_ID	Gender	Age	Occupation	Stay_In_Current_City_Years	Marital_Status	Product_Category_1	Product_Category_2	Product_Category_3	Purchase	B	C
0	P00069042	0	1	10	2	0	3	8.0	16.0	8370.0	0	0
1	P00248942	0	1	10	2	0	1	6.0	14.0	15200.0	0	0
2	P00087842	0	1	10	2	0	12	8.0	16.0	1422.0	0	0
3	P00085442	0	1	10	2	0	12	14.0	16.0	1057.0	0	0
4	P00285442	1	7	16	4	0	8	8.0	16.0	7969.0	0	1

In [40]:
```python
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 783667 entries, 0 to 233598
Data columns (total 12 columns):
 #   Column                      Non-Null Count   Dtype  
---  ------                      --------------   -----  
 0   Product_ID                  783667 non-null  object 
 1   Gender                      783667 non-null  int64  
 2   Age                         783667 non-null  int64  
 3   Occupation                  783667 non-null  int64  
 4   Stay_In_Current_City_Years  783667 non-null  int32  
 5   Marital_Status              783667 non-null  int64  
 6   Product_Category_1          783667 non-null  int64  
 7   Product_Category_2          783667 non-null  float64
 8   Product_Category_3          783667 non-null  float64
 9   Purchase                    550068 non-null  float64
 10  B                           783667 non-null  uint8  
 11  C                           783667 non-null  uint8  
dtypes: float64(3), int32(1), int64(5), object(1), uint8(2)
memory usage: 64.3+ MB
```

In [41]:
```python
df['B']=df['B'].astype(int)
df['C']=df['C'].astype(int)
```

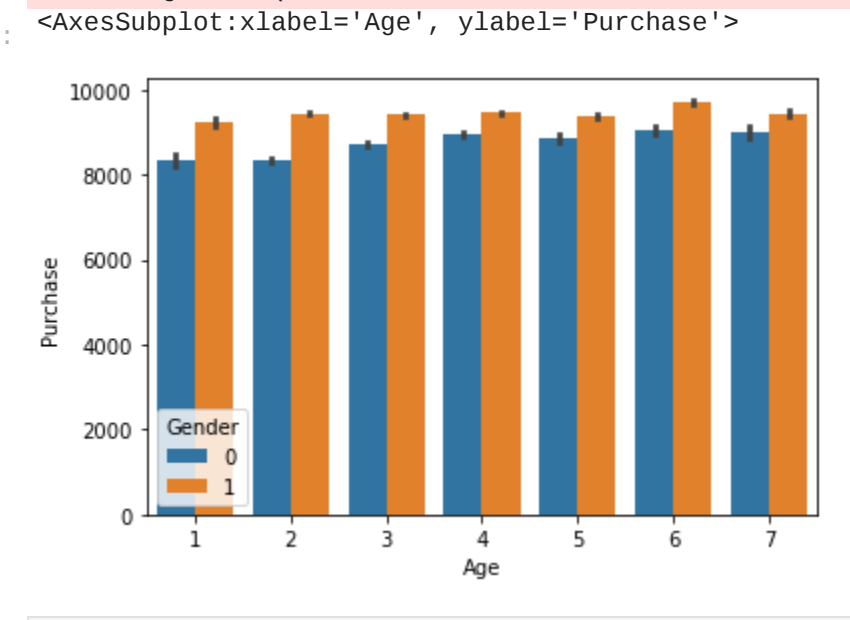
In [42]:
```python
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 783667 entries, 0 to 233598
Data columns (total 12 columns):
 #   Column                      Non-Null Count   Dtype  
---  ------                      --------------   -----  
 0   Product_ID                  783667 non-null  object 
 1   Gender                      783667 non-null  int64  
 2   Age                         783667 non-null  int64  
 3   Occupation                  783667 non-null  int64  
 4   Stay_In_Current_City_Years  783667 non-null  int32  
 5   Marital_Status              783667 non-null  int64  
 6   Product_Category_1          783667 non-null  int64  
 7   Product_Category_2          783667 non-null  float64
 8   Product_Category_3          783667 non-null  float64
 9   Purchase                    550068 non-null  float64
 10  B                           783667 non-null  int32  
 11  C                           783667 non-null  int32  
dtypes: float64(3), int32(3), int64(5), object(1)
memory usage: 68.8+ MB
```

In [43]:
```python
#Visualisation
sns.barplot('Age','Purchase',hue='Gender',data=df)
```

```
C:\Users\VIVEK PANDEY\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variables as keyword args: x, y. From version 0.12, the only valid po
sitional argument will be `data`, and passing other arguments without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```

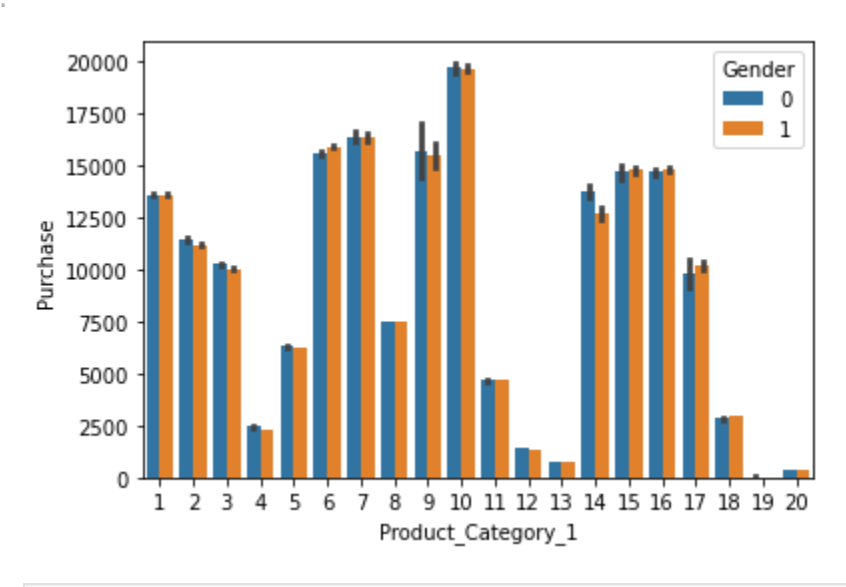
Out[43]:
```
<AxesSubplot:xlabel='Age', ylabel='Purchase'>
```



In [44]:
```python
sns.barplot('Product_Category_1','Purchase',hue='Gender',data=df)
```

```
C:\Users\VIVEK PANDEY\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variables as keyword args: x, y. From version 0.12, the only valid po
sitional argument will be `data`, and passing other arguments without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```

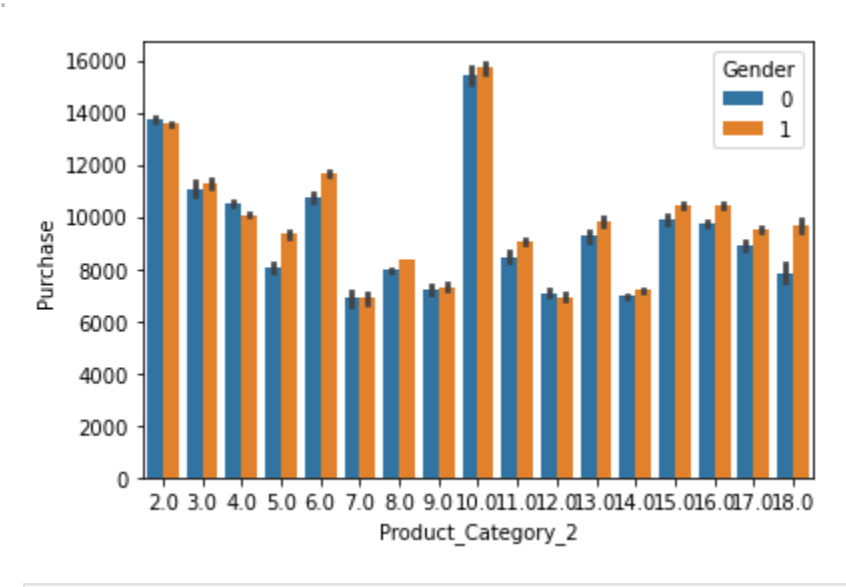
Out[44]:
```
<AxesSubplot:xlabel='Product_Category_1', ylabel='Purchase'>
```



In [45]:
```python
sns.barplot('Product_Category_2','Purchase',hue='Gender',data=df)
```

```
C:\Users\VIVEK PANDEY\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variables as keyword args: x, y. From version 0.12, the only valid po
sitional argument will be `data`, and passing other arguments without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```

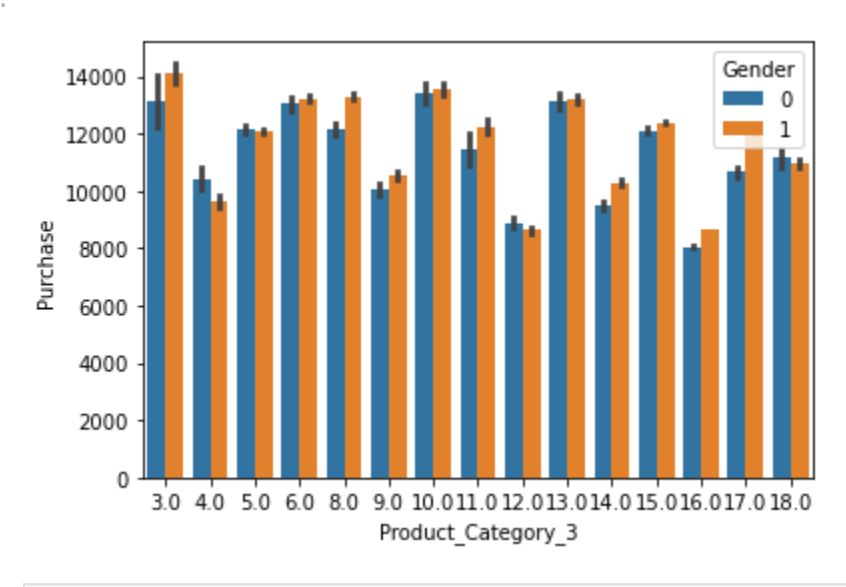
Out[45]:
```
<AxesSubplot:xlabel='Product_Category_2', ylabel='Purchase'>
```



In [46]:
```python
sns.barplot('Product_Category_3','Purchase',hue='Gender',data=df)
```

```
C:\Users\VIVEK PANDEY\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variables as keyword args: x, y. From version 0.12, the only valid po
sitional argument will be `data`, and passing other arguments without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```

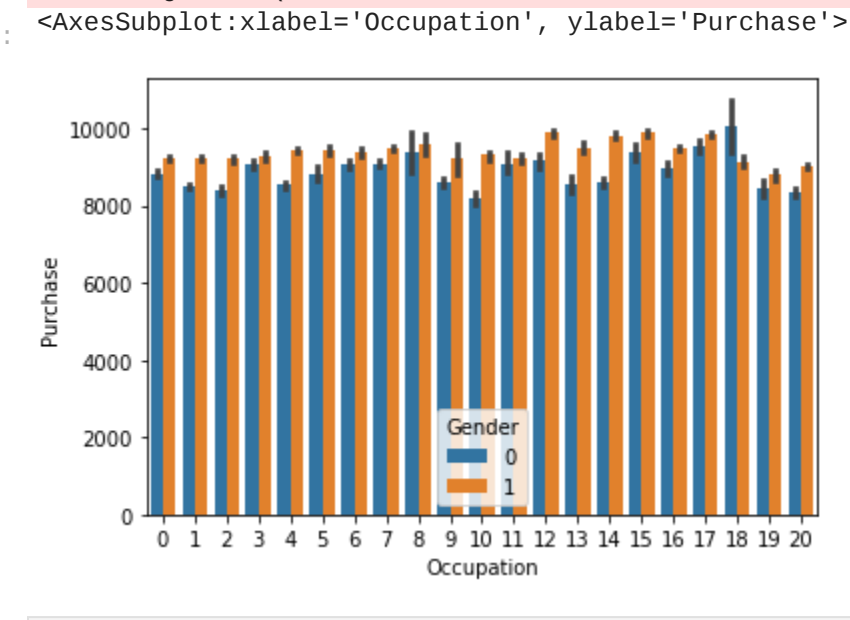
Out[46]:
```
<AxesSubplot:xlabel='Product_Category_3', ylabel='Purchase'>
```



In [47]:
```python
sns.barplot('Occupation','Purchase',hue='Gender',data=df)
```

```
C:\Users\VIVEK PANDEY\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variables as keyword args: x, y. From version 0.12, the only valid po
sitional argument will be `data`, and passing other arguments without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```

Out[47]:
```
<AxesSubplot:xlabel='Occupation', ylabel='Purchase'>
```



In [48]:
```python
#Feature Scaling
df_test=df[df['Purchase'].isnull()]
```

In [49]:
```python
df_train=df[~df['Purchase'].isnull()]
```

In [59]:
```python
X=df_train.drop('Purchase',axis=1)
```

In [68]:
```python
X=df_train.drop('Product_ID',axis=1)
```

In [69]:
```python
X.head()
```

Out[69]:

	Gender	Age	Occupation	Stay_In_Current_City_Years	Marital_Status	Product_Category_1	Product_Category_2	Product_Category_3	Purchase	B	C
0	0	1	10	2	0	3	8.0	16.0	8370.0	0	0
1	0	1	10	2	0	1	6.0	14.0	15200.0	0	0
2	0	1	10	2	0	12	8.0	16.0	1422.0	0	0
3	0	1	10	2	0	12	14.0	16.0	1057.0	0	0
4	1	7	16	4	0	8	8.0	16.0	7969.0	0	1

In [71]:
```python
X.shape
```

Out[71]:
```
(550068, 11)
```

In [72]:
```python
y=df_train['Purchase']
```

In [73]:
```python
y.head()
```

Out[73]:
```
0     8370.0
1    15200.0
2     1422.0
3     1057.0
4     7969.0
Name: Purchase, dtype: float64
```

In [74]:
```python
y.shape
```

Out[74]:
```
(550068,)
```

In [75]:
```python
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test= train_test_split(
    X,y,test_size=0.33,random_state=42)
```

In [76]:
```python
#Feature Scaling
from sklearn.preprocessing import StandardScaler
sc=StandardScaler()
X_train=sc.fit_transform(X_train)
X_test=sc.transform(X_test)
```

In [ ]: