Black Friday is a colloquial term for the Friday after Thanksgiving in the United States. It traditionally marks the start of the Christmas shopping season in the United States. Many stores offer highly promoted sales at discounted prices and often open early, sometimes as early as midnight or even on Thanksgiving.

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

In [2]:
```
#importing dataset
df_train = pd.read_csv("C:/Users/Asus/Documents/Projects/Python/A data analysis resume p
df_test = pd.read_csv("C:/Users/Asus/Documents/Projects/Python/A data analysis resume pr
```

In [3]:
```
df_train.shape
```

Out[3]:
```
(550068, 12)
```

In [4]:
```
df_test.shape
```

Out[4]:
```
(233599, 11)
```

In [5]:
```
df_train.head()
```

Out[5]:

	User_ID	Product_ID	Gender	Age	Occupation	City_Category	Stay_In_Current_City_Years	Marital_Status	Prod
0	1000001	P00069042	F	0-17	10	A	2	0	
1	1000001	P00248942	F	0-17	10	A	2	0	
2	1000001	P00087842	F	0-17	10	A	2	0	
3	1000001	P00085442	F	0-17	10	A	2	0	
4	1000002	P00285442	M	55+	16	C	4+	0	

In [6]:
```
##Merge train and test dataset

df = df_train.append(df_test)
```

```
C:\Users\Asus\AppData\Local\Temp\ipykernel_7304\3203966175.py:3: FutureWarning: The fram
e.append method is deprecated and will be removed from pandas in a future version. Use p
andas.concat instead.
  df = df_train.append(df_test)
```

In [7]:
```
df.head()
```

Out[7]:

	User_ID	Product_ID	Gender	Age	Occupation	City_Category	Stay_In_Current_City_Years	Marital_Status	Prod
0	1000001	P00069042	F	0-17	10	A	2	0	
1	1000001	P00248942	F	0-17	10	A	2	0	
2	1000001	P00087842	F	0-17	10	A	2	0	

3	1000001	P00085442	F	0-17	10	A	2	0	
4	1000002	P00285442	M	55+	16	C	4+	0	

In [8]:
```
df.shape
```

Out[8]:
```
(783667, 12)
```

In [9]:
```
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 [10]:
```
df.describe()
```

Out[10]:

	User_ID	Occupation	Marital_Status	Product_Category_1	Product_Category_2	Product_Category_3
count	7.836670e+05	783667.000000	783667.000000	783667.000000	537685.000000	237858.000000
mean	1.003029e+06	8.079300	0.409777	5.366196	9.844506	12.668605
std	1.727267e+03	6.522206	0.491793	3.878160	5.089093	4.125510
min	1.000001e+06	0.000000	0.000000	1.000000	2.000000	3.000000
25%	1.001519e+06	2.000000	0.000000	1.000000	5.000000	9.000000
50%	1.003075e+06	7.000000	0.000000	5.000000	9.000000	14.000000
75%	1.004478e+06	14.000000	1.000000	8.000000	15.000000	16.000000
max	1.006040e+06	20.000000	1.000000	20.000000	18.000000	18.000000

In [11]:
```
df.drop(['User_ID'],axis=1, inplace = True)
```

In [12]:
```
df.head(2)
```

Out[12]:

	Product_ID	Gender	Age	Occupation	City_Category	Stay_In_Current_City_Years	Marital_Status	Product_Categ
0	P00069042	F	0-17	10	A	2	0	
1	P00248942	F	0-17	10	A	2	0	

```
df
```

In [13]:

Out[13]:

	Product_ID	Gender	Age	Occupation	City_Category	Stay_In_Current_City_Years	Marital_Status	Product_
0	P00069042	F	0-17	10	A	2	0	
1	P00248942	F	0-17	10	A	2	0	
2	P00087842	F	0-17	10	A	2	0	
3	P00085442	F	0-17	10	A	2	0	
4	P00285442	M	55+	16	C	4+	0	
...	...	...	...	...	...	...	...	
233594	P00118942	F	26-35	15	B	4+	1	
233595	P00254642	F	26-35	15	B	4+	1	
233596	P00031842	F	26-35	15	B	4+	1	
233597	P00124742	F	46-50	1	C	4+	0	
233598	P00316642	F	46-50	0	B	4+	1	

783667 rows × 11 columns

In [14]:
```
pd.get_dummies(df['Gender'], drop_first = 1)
```

Out[14]:

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

783667 rows × 1 columns

In [15]:
```
#handling categorical feature Gender
df['Gender'] = df['Gender'].map({'F':0, 'M':1})
df.head()
```

Out[15]:

	Product_ID	Gender	Age	Occupation	City_Category	Stay_In_Current_City_Years	Marital_Status	Product_Categ
0	P00069042	0	0-17	10	A	2	0	
1	P00248942	0	0-17	10	A	2	0	
2	P00087842	0	0-17	10	A	2	0	
3	P00085442	0	0-17	10	A	2	0	
4	P00285442	1	55+	16	C	4+	0	

In [16]:
```
#handling categorical feature Age

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

Out[16]:
```
array(['0-17', '55+', '26-35', '46-50', '51-55', '36-45', '18-25'],
      dtype=object)
```

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

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

Out[18]:

	Product_ID	Gender	Age	Occupation	City_Category	Stay_In_Current_City_Years	Marital_Status	Product_Categ
0	P00069042	0	1	10	A	2	0	
1	P00248942	0	1	10	A	2	0	
2	P00087842	0	1	10	A	2	0	
3	P00085442	0	1	10	A	2	0	
4	P00285442	1	7	16	C	4+	0	

In [19]:
```
df_city = pd.get_dummies(df['City_Category'], drop_first= True)
```

In [20]:
```
df_city.head()
```

Out[20]:

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

In [21]:
```
df = pd.concat([df, df_city], axis=1)
df.head()
```

Out[21]:

	Product_ID	Gender	Age	Occupation	City_Category	Stay_In_Current_City_Years	Marital_Status	Product_Categ
0	P00069042	0	1	10	A	2	0	
1	P00248942	0	1	10	A	2	0	
2	P00087842	0	1	10	A	2	0	

3	P00085442	0	1	10	A	2	0
4	P00285442	1	7	16	C	4+	0

In [24]:
```
df.drop('City_Category', axis = 1, inplace = True)
```

In [25]:
```
df.head(2)
```

Out[25]:

	Product_ID	Gender	Age	Occupation	Stay_In_Current_City_Years	Marital_Status	Product_Category_1	Product_
0	P00069042	0	1	10	2	0	3	
1	P00248942	0	1	10	2	0	1	

In [26]:
```
##Missing Values
df.isnull().sum()
```

Out[26]:
```
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 [27]:
```
df['Product_Category_2'].unique()
```

Out[27]:
```
array([nan,  6., 14.,  2.,  8., 15., 16., 11.,  5.,  3.,  4., 12.,  9.,
       10., 17., 13.,  7., 18.])
```

In [28]:
```
df['Product_Category_2'].value_counts()
```

Out[28]:
```
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 [29]:
```
df['Product_Category_2'].mode()
```

Out[29]:
```
0    8.0
Name: Product_Category_2, dtype: float64
```

In [30]:
```
##Replace missing values with mode
```

```
df['Product_Category_2'] = df['Product_Category_2'].fillna(df['Product_Category_2'].mode
```

In [31]:
```
df['Product_Category_2'].isnull().sum()
```

Out[31]:
```
0
```

In [35]:
```
df['Product_Category_3'].value_counts()
```

Out[35]:
```
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 [33]:
```
df['Product_Category_3'].mode()
```

Out[33]:
```
0    16.0
Name: Product_Category_3, dtype: float64
```

In [36]:
```
df['Product_Category_3'] = df['Product_Category_3'].fillna(df['Product_Category_3'].mode
```

In [37]:
```
df['Product_Category_2'].isnull().sum()
```

Out[37]:
```
0
```

In [38]:
```
df.head()
```

Out[38]:

	Product_ID	Gender	Age	Occupation	Stay_In_Current_City_Years	Marital_Status	Product_Category_1	Product_
0	P00069042	0	1	10	2	0	3	
1	P00248942	0	1	10	2	0	1	
2	P00087842	0	1	10	2	0	12	
3	P00085442	0	1	10	2	0	12	
4	P00285442	1	7	16	4+	0	8	

In [39]:
```
df["Stay_In_Current_City_Years"].unique()
```

Out[39]:
```
array(['2', '4+', '3', '1', '0'], dtype=object)
```

In [40]:
```
df["Stay_In_Current_City_Years"] = df["Stay_In_Current_City_Years"].str.replace('+','')
```

```
C:\Users\Asus\AppData\Local\Temp\ipykernel_7304\3223755903.py:1: FutureWarning: The defa
ult value of regex will change from True to False in a future version. In addition, sing
le character regular expressions will *not* be treated as literal strings when regex=Tru
e.
  df["Stay_In_Current_City_Years"] = df["Stay_In_Current_City_Years"].str.replace
('+','')
```

In [41]:
```
df["Stay_In_Current_City_Years"].unique()
```

Out[41]:
```
array(['2', '4', '3', '1', '0'], dtype=object)
```

In [42]:
```
df.head()
```

Out[42]:

	Product_ID	Gender	Age	Occupation	Stay_In_Current_City_Years	Marital_Status	Product_Category_1	Product_
0	P00069042	0	1	10	2	0	3	
1	P00248942	0	1	10	2	0	1	
2	P00087842	0	1	10	2	0	12	
3	P00085442	0	1	10	2	0	12	
4	P00285442	1	7	16	4	0	8	

In [43]:
```
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 [44]:
```
##convet object into integers
df['Stay_In_Current_City_Years'] = df['Stay_In_Current_City_Years'].astype(int)
```

In [45]:
```
##convet uint8 into integers
df['B'] = df['B'].astype(int)
df['C'] = df['C'].astype(int)
```

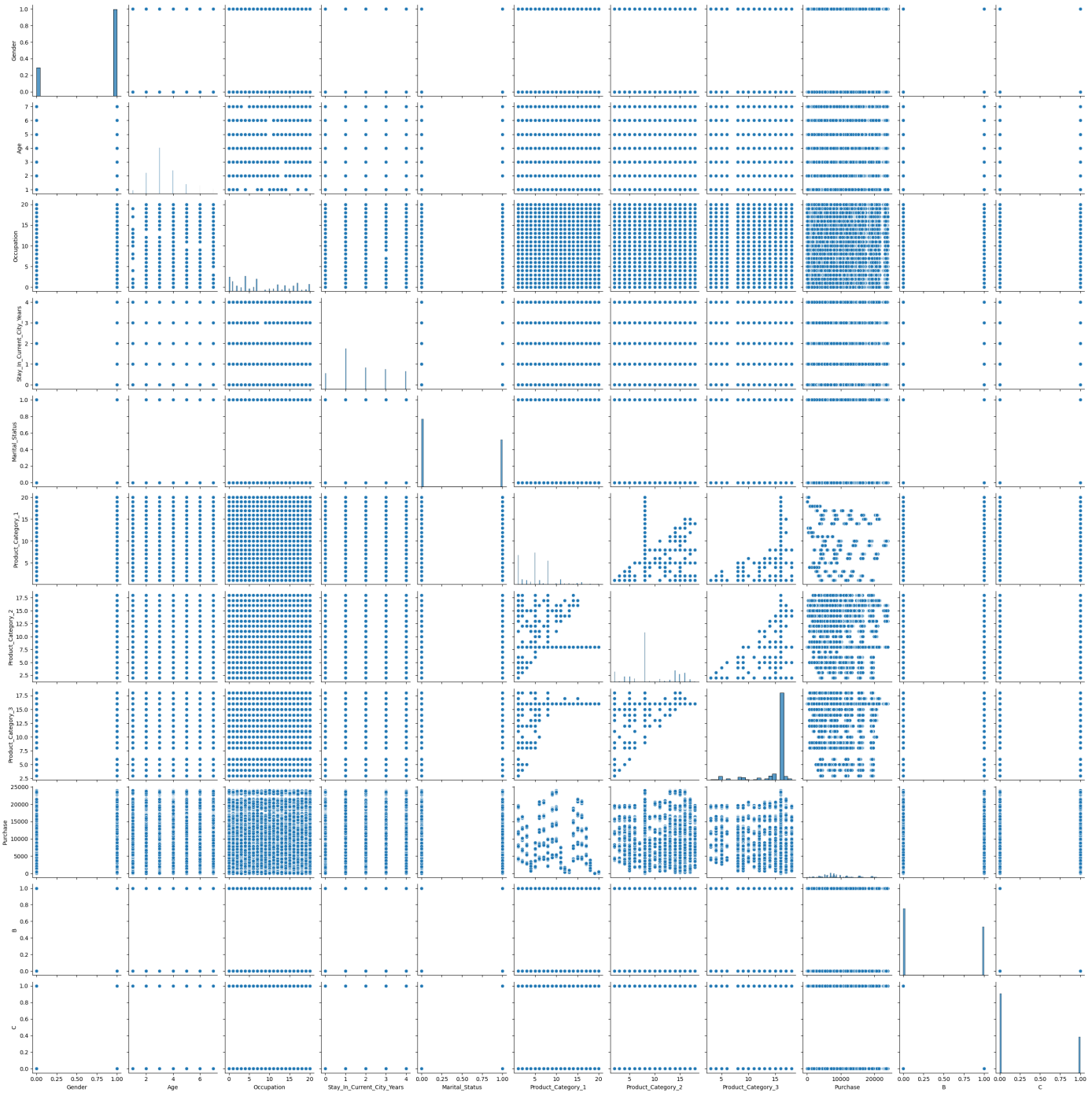
In [46]:
```
df.info()
```

```
<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 [47]:
```
sns.pairplot(df)
```

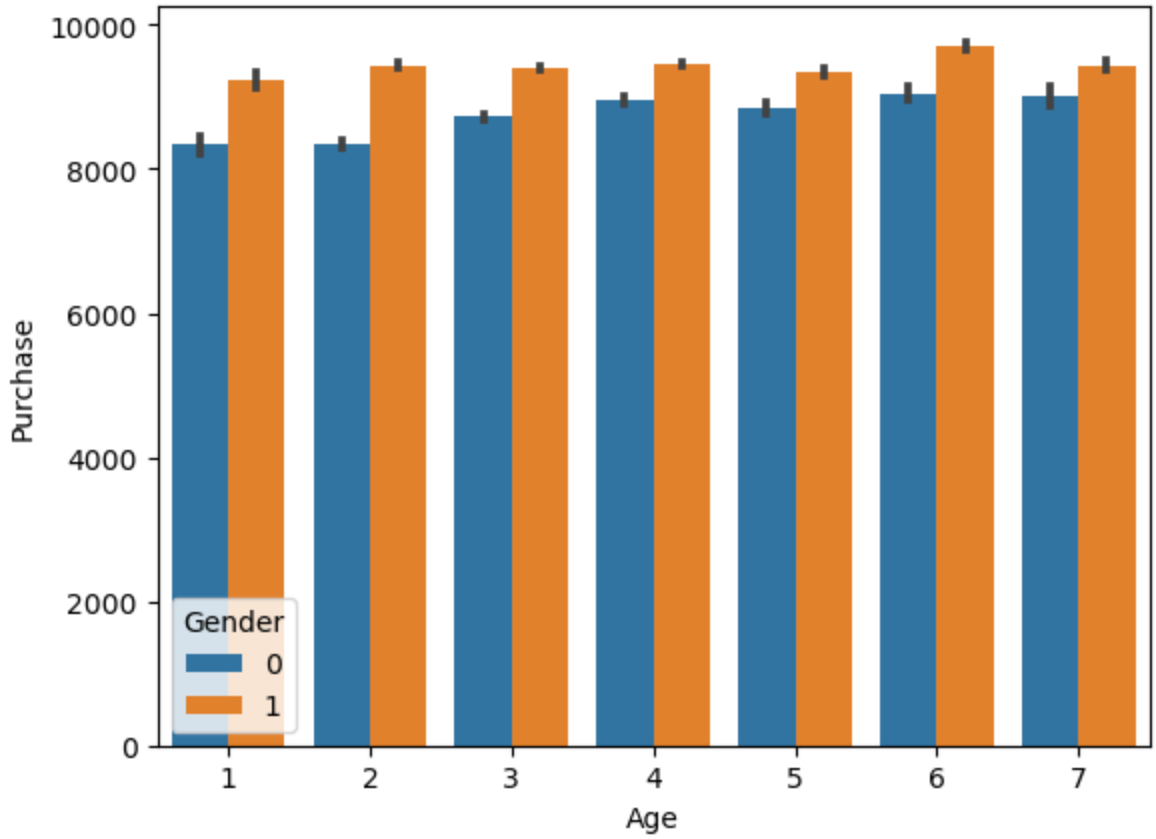
Out[47]:
```
<seaborn.axisgrid.PairGrid at 0x19028b98bb0>
```



In [ ]:
```
```

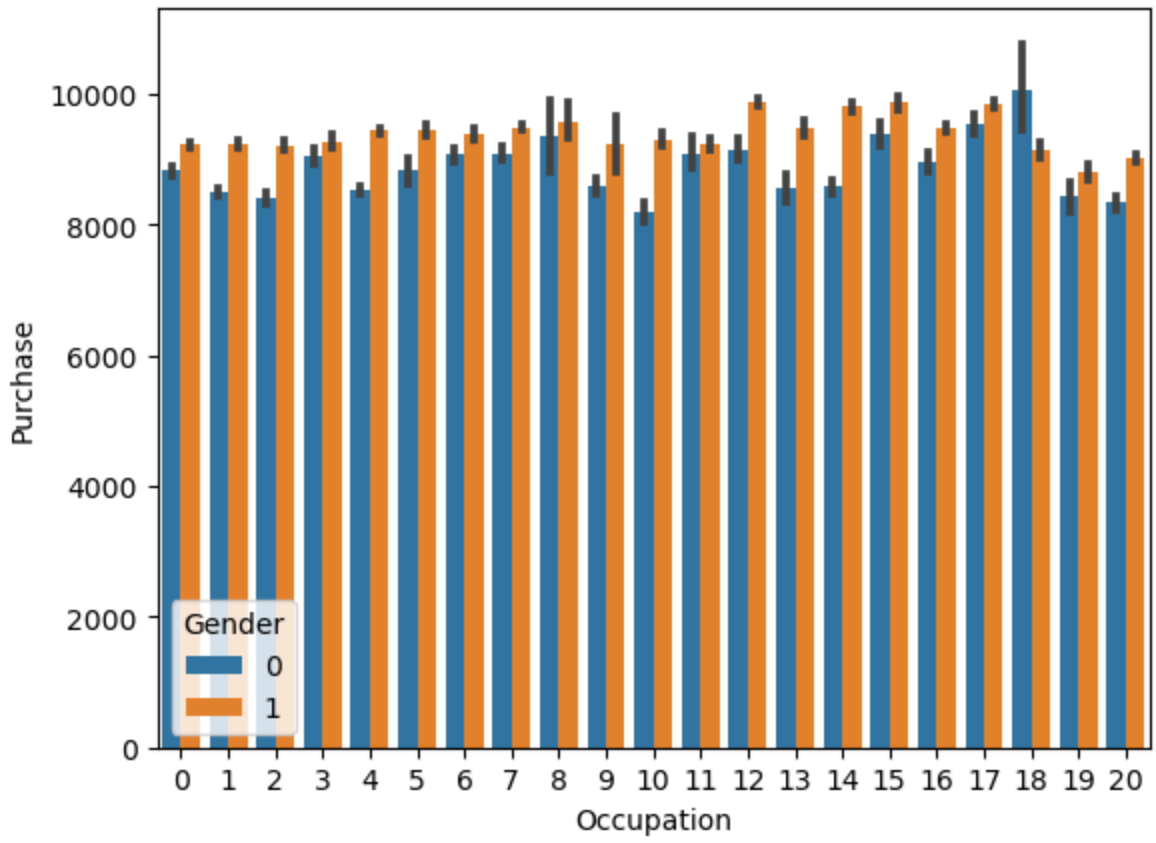
In [51]:
```
## Age VS Purchase
sns.barplot(x='Age',y='Purchase',hue='Gender', data=df)
```

Out[51]:
```
<Axes: xlabel='Age', ylabel='Purchase'>
```



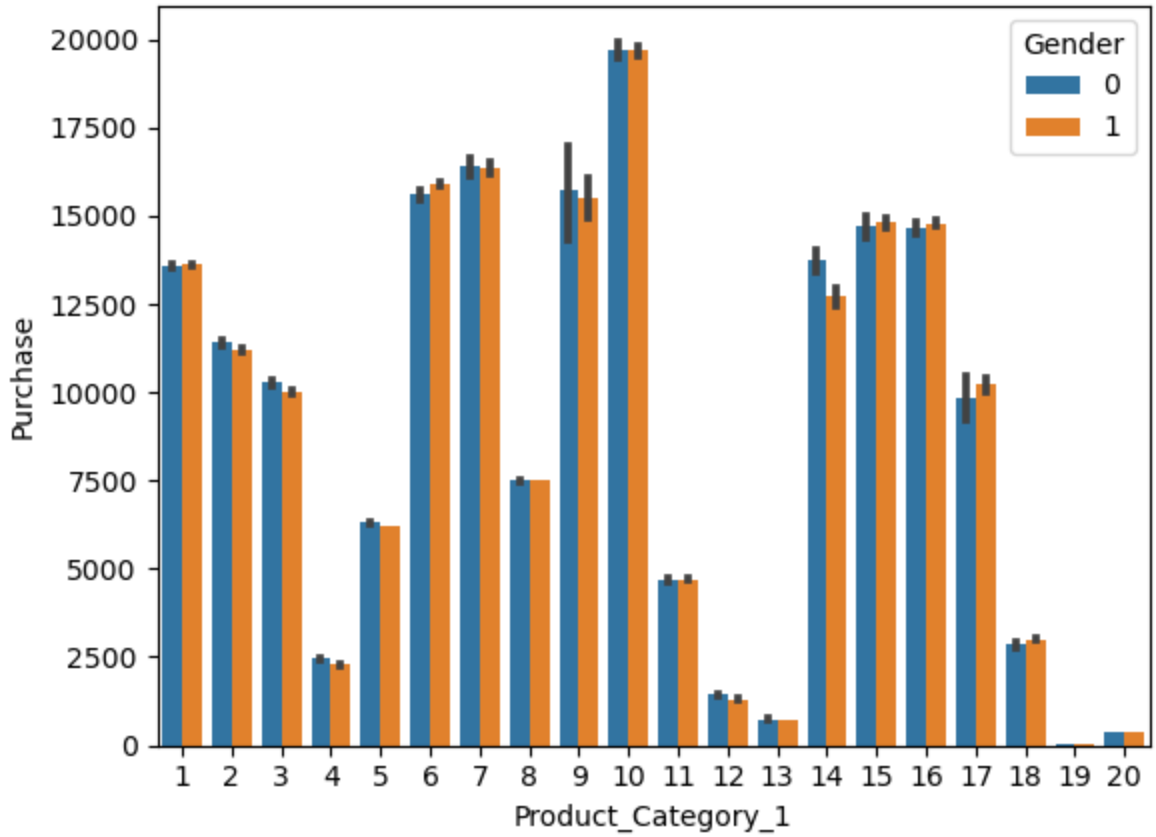
In [52]:
```
sns.barplot(x='Occupation',y='Purchase',hue='Gender', data=df)
```

Out[52]:
```
<Axes: xlabel='Occupation', ylabel='Purchase'>
```



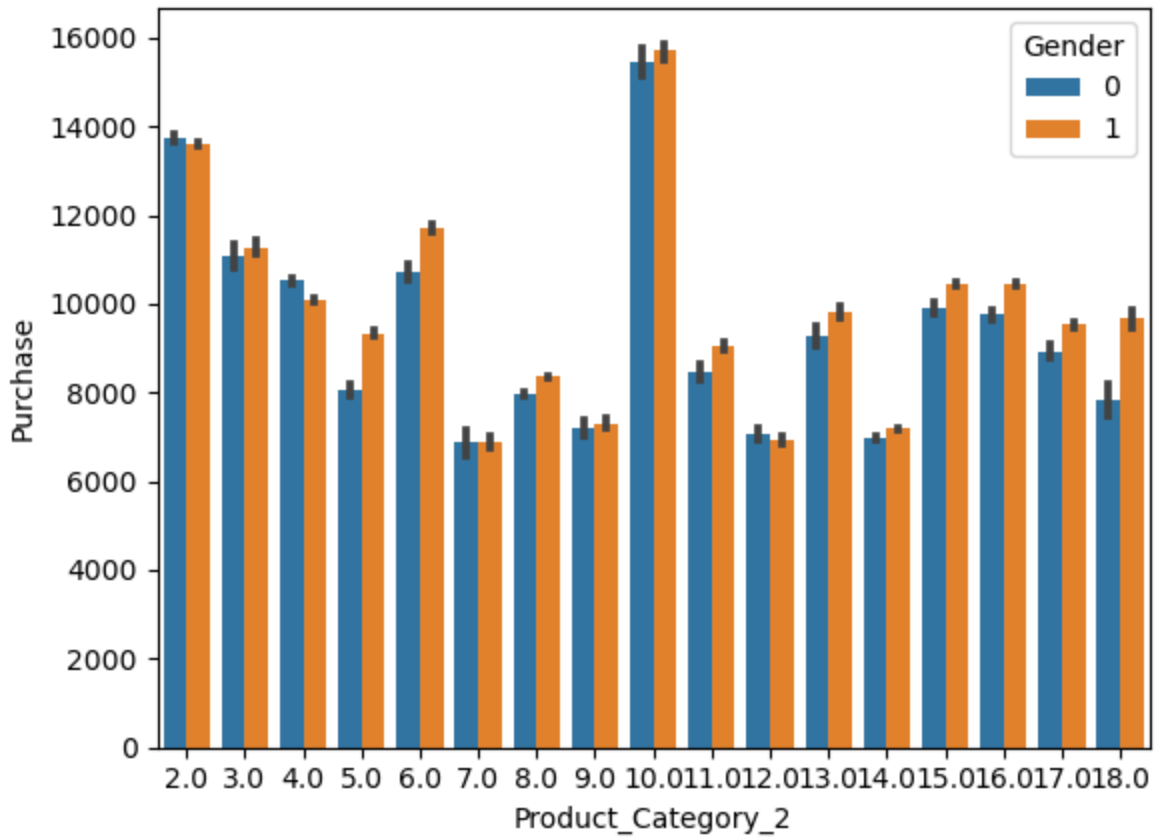
In [53]:
```
sns.barplot(x='Product_Category_1',y='Purchase',hue='Gender', data=df)
```

Out[53]:
```
<Axes: xlabel='Product_Category_1', ylabel='Purchase'>
```



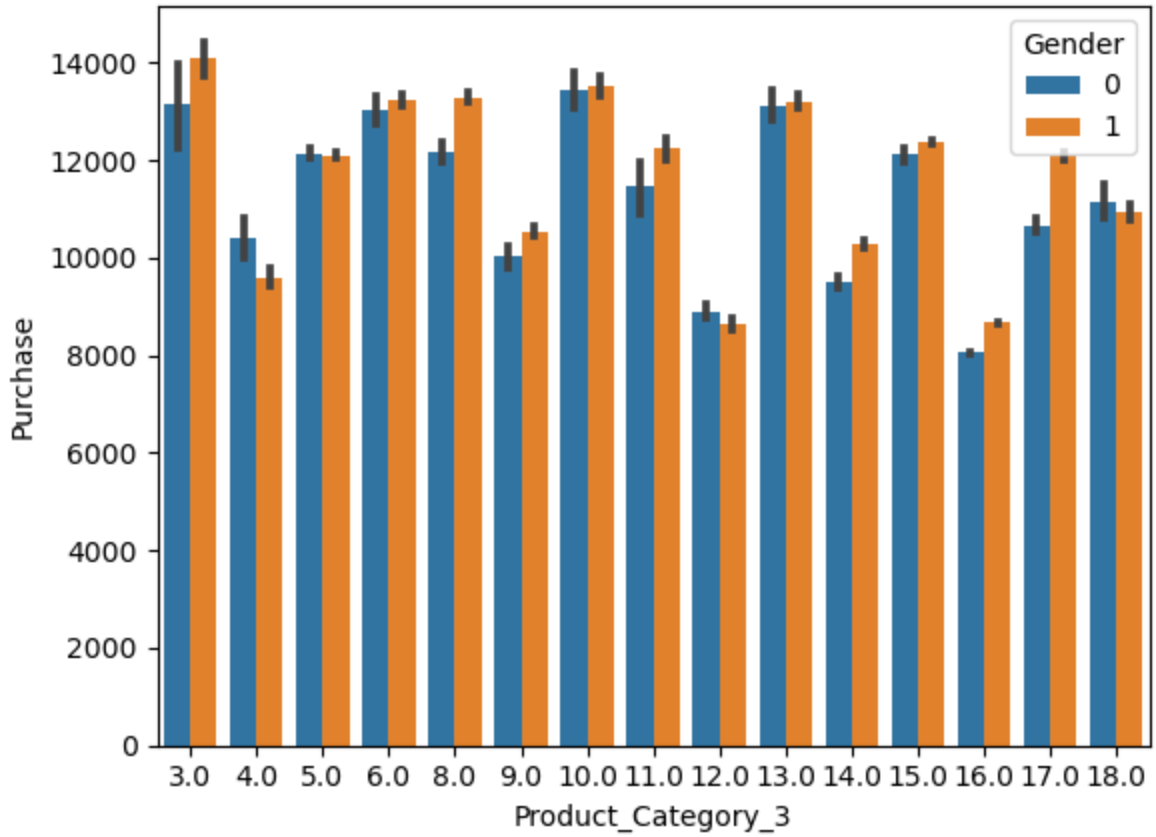
In [55]:
```
sns.barplot(x='Product_Category_2',y='Purchase',hue='Gender', data=df)
```

Out[55]:
```
<Axes: xlabel='Product_Category_2', ylabel='Purchase'>
```



In [56]:
```
sns.barplot(x='Product_Category_3',y='Purchase',hue='Gender', data=df)
```

Out[56]:
```
<Axes: xlabel='Product_Category_3', ylabel='Purchase'>
```


Gender
0
1
Purchase
14000
12000
10000
8000
6000
4000
2000
0
3.0
4.0
5.0
6.0
8.0
9.0
10.0
11.0
12.0
13.0
14.0
15.0
16.0
17.0
18.0
Product_Category_3

In [ ]: