In [75]:
```
import numpy as np
import pandas as pd
import warnings
warnings.filterwarnings('ignore')
```

In [76]:
```
df1 = pd.read_csv('user_demographics.csv')
df2 = pd.read_csv('User_product_purchase_details_p2.csv')
```

In [77]:
```
df1
```

Out[77]:

	User_ID	Gender	Age	Occupation
0	1000001	F	0-17	10
1	1000002	M	55+	16
2	1000003	M	26-35	15
3	1000004	M	46-50	7
4	1000005	M	26-35	20
...	...	...	...	...
5886	1004588	F	26-35	4
5887	1004871	M	18-25	12
5888	1004113	M	36-45	17
5889	1005391	M	26-35	7
5890	1001529	M	18-25	4

5891 rows × 4 columns

In [112...
```
df1['Age'].unique()
```

Out[112...
```
array(['0-17', '55+', '26-35', '46-50', '51-55', '36-45', '18-25'],
      dtype=object)
```

In [78]:
```
df2
```

Out[78]:

	User_ID	Product_ID	City_Category	Stay_In_Current_City_Years	Marital_Status	Prc
0	1000001	P00069042	A	2	0	
1	1000001	P00248942	A	2	0	
2	1000001	P00087842	A	2	0	
3	1000001	P00085442	A	2	0	
4	1000002	P00285442	C	4+	0	
...	...	...	...	...	...	
550063	1006033	P00372445	B	1	1	
550064	1006035	P00375436	C	3	0	
550065	1006036	P00375436	B	4+	1	
550066	1006038	P00375436	C	2	0	
550067	1006039	P00371644	B	4+	1	

550068 rows × 9 columns

In [79]:
```
modified_df = df2.groupby('User_ID').agg({
    'City_Category': 'first',
    'Stay_In_Current_City_Years': 'first',
    'Marital_Status': 'first',
    'Product_Category_1': 'sum',
    'Product_Category_2': 'sum',
    'Product_Category_3': 'sum',
    'Purchase': 'sum'
}).reset_index()
```

In [80]:
```
modified_df
```

Out[80]:

	User_ID	City_Category	Stay_In_Current_City_Years	Marital_Status	Product_Category_
0	1000001	A	2	0	21
1	1000002	C	4+	0	35
2	1000003	A	3	0	9
3	1000004	B	2	1	3
4	1000005	A	1	1	65
...	...	...	...	...	
5886	1006036	B	4+	1	320
5887	1006037	C	4+	0	93
5888	1006038	C	2	0	8
5889	1006039	B	4+	1	43
5890	1006040	B	2	0	114

5891 rows × 8 columns

In [81]:
```
merged_df = pd.merge(df1, modified_df, on='User_ID')
```

In [82]:
```
merged_df
```

Out[82]:

	User_ID	Gender	Age	Occupation	City_Category	Stay_In_Current_City_Years	Marita
0	1000001	F	0-17	10	A	2	
1	1000002	M	55+	16	C	4+	
2	1000003	M	26-35	15	A	3	
3	1000004	M	46-50	7	B	2	
4	1000005	M	26-35	20	A	1	
...	...	...	...	...	...	...	
5886	1004588	F	26-35	4	C	0	
5887	1004871	M	18-25	12	C	2	
5888	1004113	M	36-45	17	C	3	
5889	1005391	M	26-35	7	A	0	
5890	1001529	M	18-25	4	C	4+	

5891 rows × 11 columns

In [83]:
```
sample_df = merged_df
```

In [84]:
```
from sklearn.preprocessing import LabelEncoder
```

In [85]:
```
label_encoder = LabelEncoder()
sample_df['Gender'] = label_encoder.fit_transform(sample_df['Gender'])
sample_df['Age'] = label_encoder.fit_transform(sample_df['Age'])
sample_df['City_Category'] = label_encoder.fit_transform(sample_df['City_Category'])
```

In [86]:
```
sample_df
```

Out[86]:

	User_ID	Gender	Age	Occupation	City_Category	Stay_In_Current_City_Years	Marita
0	1000001	0	0	10	0	2	
1	1000002	1	6	16	2	4+	
2	1000003	1	2	15	0	3	
3	1000004	1	4	7	1	2	
4	1000005	1	2	20	0	1	
...	...	...	...	...	...	...	
5886	1004588	0	2	4	2	0	
5887	1004871	1	1	12	2	2	
5888	1004113	1	3	17	2	3	
5889	1005391	1	2	7	0	0	
5890	1001529	1	1	4	2	4+	

5891 rows × 11 columns

In [87]:

```
# Calculate Q1 (25th percentile) and Q3 (75th percentile)
Q1 = sample_df['Purchase'].quantile(0.25)
Q3 = sample_df['Purchase'].quantile(0.75)
IQR = Q3 - Q1

# Define the bounds for outliers
lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR

# Filter out outliers
df_no_outliers = sample_df[(sample_df['Purchase'] >= lower_bound) & (sample_df['Pur

df_no_outliers
```

Out[87]:

	User_ID	Gender	Age	Occupation	City_Category	Stay_In_Current_City_Years	Marita
0	1000001	0	0	10	0	2	
1	1000002	1	6	16	2	4+	
2	1000003	1	2	15	0	3	
3	1000004	1	4	7	1	2	
4	1000005	1	2	20	0	1	
...	...	...	...	...	...	...	
5886	1004588	0	2	4	2	0	
5887	1004871	1	1	12	2	2	
5888	1004113	1	3	17	2	3	
5889	1005391	1	2	7	0	0	
5890	1001529	1	1	4	2	4+	

5482 rows × 11 columns

In [88]:
```
df_no_outliers.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 5482 entries, 0 to 5890
Data columns (total 11 columns):
 #   Column                      Non-Null Count  Dtype  
---  ------                      --------------  -----  
 0   User_ID                     5482 non-null   int64  
 1   Gender                      5482 non-null   int64  
 2   Age                         5482 non-null   int64  
 3   Occupation                  5482 non-null   int64  
 4   City_Category               5482 non-null   int64  
 5   Stay_In_Current_City_Years  5482 non-null   object 
 6   Marital_Status              5482 non-null   int64  
 7   Product_Category_1          5482 non-null   int64  
 8   Product_Category_2          5482 non-null   float64
 9   Product_Category_3          5482 non-null   float64
 10  Purchase                    5482 non-null   int64  
dtypes: float64(2), int64(8), object(1)
memory usage: 513.9+ KB
```

In [89]:
```
df_no_outliers['Stay_In_Current_City_Years'] = df_no_outliers['Stay_In_Current_City
```

In [90]:
```
df_no_outliers['Product_Category_1'] = df_no_outliers['Product_Category_1'].fillna(
df_no_outliers['Product_Category_2'] = df_no_outliers['Product_Category_2'].fillna(
df_no_outliers['Product_Category_3'] = df_no_outliers['Product_Category_3'].fillna(
```

In [91]:
```
df_no_outliers
```

Out[91]:

	User_ID	Gender	Age	Occupation	City_Category	Stay_In_Current_City_Years	Marita
0	1000001	0	0	10	0	2	
1	1000002	1	6	16	2	4	
2	1000003	1	2	15	0	3	
3	1000004	1	4	7	1	2	
4	1000005	1	2	20	0	1	
...	...	...	...	...	...	...	
5886	1004588	0	2	4	2	0	
5887	1004871	1	1	12	2	2	
5888	1004113	1	3	17	2	3	
5889	1005391	1	2	7	0	0	
5890	1001529	1	1	4	2	4	

5482 rows × 11 columns

In [114...
```
df_no_outliers.to_csv('model.csv', index=False)
```

In [92]:
```
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestRegressor
from sklearn.tree import DecisionTreeRegressor

from sklearn.metrics import mean_squared_error, r2_score, mean_absolute_error
```

In [93]:
```
x = df_no_outliers[['Age', 'City_Category', 'Stay_In_Current_City_Years', 'Product_
y = df_no_outliers['Purchase']
```

In [94]:
```
xtrain, xtest, ytrain, ytest = train_test_split(x, y, test_size = 0.2, random_state
```

In [95]:
```
dt = DecisionTreeRegressor()
```

In [96]:
```
dt.fit(xtrain, ytrain)
```

Out[96]:
```
DecisionTreeRegressor
DecisionTreeRegressor()
```

In [97]:
```
ypred_dt = dt.predict(xtest)
```

In [98]:
```
mean_squared_error(ytest, ypred_dt, squared = False)
```

Out[98]:
```
np.float64(183346.83728026864)
```

In [99]:
```
r2_score(ytest, ypred_dt)
```

Out[99]:
```
0.8942172263012904
```

In [100…
```
mean_absolute_error(ytest, ypred_dt)
```

Out[100…
```
np.float64(124439.05104831359)
```

In [109…
```
dt.predict([[2,1,2,380,628,485]])
```

Out[109…
```
array([1015469.])
```

In [119…
```
xtest[:7]
```

Out[119…

	Age	City_Category	Stay_In_Current_City_Years	Product_Category_1	Product_Categor
343	2	1	2	380	
33	4	2	4	718	
8	2	2	0	355	
2257	6	2	3	114	
2265	2	2	1	114	
3020	6	2	3	75	
3882	2	0	4	149	

In [111…
```
ytest[:7]
```

Out[111…
```
343     920708
33      821303
8       594099
2257    243214
2265    144223
3020    186272
3882    287340
Name: Purchase, dtype: int64
```

In [104…
```
rf = RandomForestRegressor(n_estimators = 500, random_state=42)
```

In [105…
```
rf.fit(xtrain, ytrain)
```

Out[105…

RandomForestRegressor

```
RandomForestRegressor(n_estimators=500, random_state=42)
```

In [106…
```
ypred_rf = rf.predict(xtest)
```

In [107…
```
mean_squared_error(ytest, ypred_rf, squared=False)
```

Out[107…

```
np.float64(139399.04023725065)
```

In [108…

```
r2_score(ytest, ypred_rf)
```

Out[108…

```
0.9388512375552188
```

In [110…

```
rf.predict([[2,0,4,149,140,90]])
```

Out[110…

```
array([253897.838])
```

In [ ]:

```

```