Exploratory Data Analysis for Retailer with customer data, transaction data, and products data

In [37]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from datetime import datetime
```

Import our Data Sets

In [10]:
```
products = pd.read_csv('Product Data Set - Student 2 of 3.csv', sep='|')
```

In [13]:
```
transaction = pd.read_csv('Transaction Data Set - Student 3 of 3.csv', sep = '|')
```

In [7]:
```
customer = pd.read_csv('Customer Data Set - Student 1 of 3.csv')
```

In [11]:
```
products.head()
```

Out[11]:

	PRODUCT CODE	PRODUCT CATEGORY	UNIT LIST PRICE
0	30001	HEALTH & BEAUTY	$7.45
1	30002	HEALTH & BEAUTY	$5.35
2	30003	HEALTH & BEAUTY	$5.49
3	30004	HEALTH & BEAUTY	$6.46
4	30005	HEALTH & BEAUTY	$7.33

In [14]:
```
transaction.head()
```

Out[14]:

	CUSTOMER NUM	PRODUCT NUM	QUANTITY PURCHASED	DISCOUNT TAKEN	TRANSACTION DATE	STOCKOUT
0	10114	30011	4	0.0	1/2/2015	0
1	10217	30016	3	0.0	1/2/2015	0
2	10224	30013	4	0.0	1/2/2015	0
3	10103	30012	3	0.2	1/2/2015	0
4	10037	30010	8	0.0	1/2/2015	0

In [15]:
```
customer.head()
```

	CUSTOMERID	GENDER	AGE	INCOME	EXPERIENCE SCORE	LOYALTY GROUP	ENROLLMENT DATE	HOUSEHOLD SIZE	M
0	10001	0	64	$133,498	5	enrolled	06-03-2013	4	
1	10002	0	42	$94,475	9	notenrolled	NaN	6	
2	10003	0	40	$88,610	9	enrolled	02-09-2010	5	
3	10004	0	38	$84,313	8	enrolled	06-04-2015	1	
4	10005	0	30	$51,498	3	notenrolled	NaN	1	

Quick Data Exploration

In [16]:
```
products.shape
```

Out[16]:
```
(30, 3)
```

In [18]:
```
transaction.shape
```

Out[18]:
```
(10000, 6)
```

In [19]:
```
customer.shape
```

Out[19]:
```
(500, 9)
```

verify dataframe type

In [20]:
```
type(customer)
```

Out[20]:
```
pandas.core.frame.DataFrame
```

In [22]:
```
type(customer['AGE'])
```

Out[22]:
```
pandas.core.series.Series
```

verify data type of each column

In [23]:
```
customer.dtypes
```

Out[23]:
```
CUSTOMERID           int64
GENDER               int64
AGE                  int64
INCOME              object
EXPERIENCE SCORE     int64
LOYALTY GROUP       object
ENROLLMENT DATE     object
HOUSEHOLD SIZE       int64
```

```
MARITAL STATUS      object
dtype: object
```

Data Preparation

In [24]:
```
#Income is string with $ sign and comma sign, convert it into integer
```

In [25]:
```
customer['INCOME'] = customer['INCOME'].map(lambda x: x.replace('$',''))
```

In [26]:
```
customer.head()
```

Out[26]:

	CUSTOMERID	GENDER	AGE	INCOME	EXPERIENCE SCORE	LOYALTY GROUP	ENROLLMENT DATE	HOUSEHOLD SIZE	M
0	10001	0	64	133,498	5	enrolled	06-03-2013	4	
1	10002	0	42	94,475	9	notenrolled	NaN	6	
2	10003	0	40	88,610	9	enrolled	02-09-2010	5	
3	10004	0	38	84,313	8	enrolled	06-04-2015	1	
4	10005	0	30	51,498	3	notenrolled	NaN	1	

In [27]:
```
customer['INCOME'] = customer['INCOME'].map(lambda x: x.replace(',',''))
```

In [28]:
```
customer.head()
```

Out[28]:

	CUSTOMERID	GENDER	AGE	INCOME	EXPERIENCE SCORE	LOYALTY GROUP	ENROLLMENT DATE	HOUSEHOLD SIZE	M
0	10001	0	64	133498	5	enrolled	06-03-2013	4	
1	10002	0	42	94475	9	notenrolled	NaN	6	
2	10003	0	40	88610	9	enrolled	02-09-2010	5	
3	10004	0	38	84313	8	enrolled	06-04-2015	1	
4	10005	0	30	51498	3	notenrolled	NaN	1	

In [29]:
```
customer.dtypes
```

Out[29]:
```
CUSTOMERID           int64
GENDER               int64
AGE                  int64
INCOME              object
EXPERIENCE SCORE     int64
LOYALTY GROUP       object
ENROLLMENT DATE     object
HOUSEHOLD SIZE       int64
MARITAL STATUS      object
```

In [31]:
```
customer['INCOME'] = customer['INCOME'].astype('int')
```

In [32]:
```
customer.dtypes
```

Out[32]:
```
CUSTOMERID           int64
GENDER               int64
AGE                  int64
INCOME               int32
EXPERIENCE SCORE     int64
LOYALTY GROUP       object
ENROLLMENT DATE     object
HOUSEHOLD SIZE       int64
MARITAL STATUS      object
dtype: object
```

Statistical Information

In [34]:
```
customer['MARITAL STATUS'].describe()
```

Out[34]:
```
count         500
unique          4
top       Married
freq          267
Name: MARITAL STATUS, dtype: object
```

In [36]:
```
customer['MARITAL STATUS'].unique()
```

Out[36]:
```
array(['Single', 'Married', 'Divorced', 'Widow/Widower'], dtype=object)
```

In [35]:
```
customer['INCOME'].describe()
```

Out[35]:
```
count       500.000000
mean      85792.482000
std       37157.766304
min       20256.000000
25%       52429.000000
50%       86846.500000
75%      118381.000000
max      149999.000000
Name: INCOME, dtype: float64
```

Date type must be Datetime

In [43]:
```
customer['ENROLLMENT DATE'] = customer['ENROLLMENT DATE'][customer['ENROLLMENT DATE'
```

In [44]:
```
customer.dtypes
```

Out[44]:
```
CUSTOMERID                  int64
GENDER                      int64
AGE                         int64
INCOME                      int32
EXPERIENCE SCORE            int64
LOYALTY GROUP              object
HOUSEHOLD SIZE              int64
```

```
MARITAL STATUS              object
dtype: object
```

In [45]:
```
customer.head()
```

Out[45]:

	CUSTOMERID	GENDER	AGE	INCOME	EXPERIENCE SCORE	LOYALTY GROUP	ENROLLMENT DATE	HOUSEHOLD SIZE	M
0	10001	0	64	133498	5	enrolled	2013-03-06	4	
1	10002	0	42	94475	9	notenrolled	NaT	6	
2	10003	0	40	88610	9	enrolled	2010-09-02	5	
3	10004	0	38	84313	8	enrolled	2015-04-06	1	
4	10005	0	30	51498	3	notenrolled	NaT	1	

check null values

In [47]:
```
customer.isnull().values.any()
```

Out[47]:
```
True
```

In [48]:
```
products.isnull().values.any()
```

Out[48]:
```
False
```

In [49]:
```
transaction.isnull().values.any()
```

Out[49]:
```
False
```

which column has null values in customer data

In [50]:
```
customer.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 500 entries, 0 to 499
Data columns (total 9 columns):
 #   Column            Non-Null Count  Dtype         
---  ------            --------------  -----         
 0   CUSTOMERID        500 non-null    int64         
 1   GENDER            500 non-null    int64         
 2   AGE               500 non-null    int64         
 3   INCOME            500 non-null    int32         
 4   EXPERIENCE SCORE  500 non-null    int64         
 5   LOYALTY GROUP     500 non-null    object        
 6   ENROLLMENT DATE   264 non-null    datetime64[ns]
 7   HOUSEHOLD SIZE    500 non-null    int64         
 8   MARITAL STATUS    500 non-null    object        
dtypes: datetime64[ns](1), int32(1), int64(5), object(2)
```

In [52]:
```
customer.columns
```

Out[52]:
```
Index(['CUSTOMERID', 'GENDER', 'AGE', 'INCOME', 'EXPERIENCE SCORE',
       'LOYALTY GROUP', 'ENROLLMENT DATE', 'HOUSEHOLD SIZE', 'MARITAL STATUS'],
      dtype='object')
```

In [53]:
```
customer.columns[customer.isna().any()].tolist()
```

Out[53]:
```
['ENROLLMENT DATE']
```

drop rows for na values

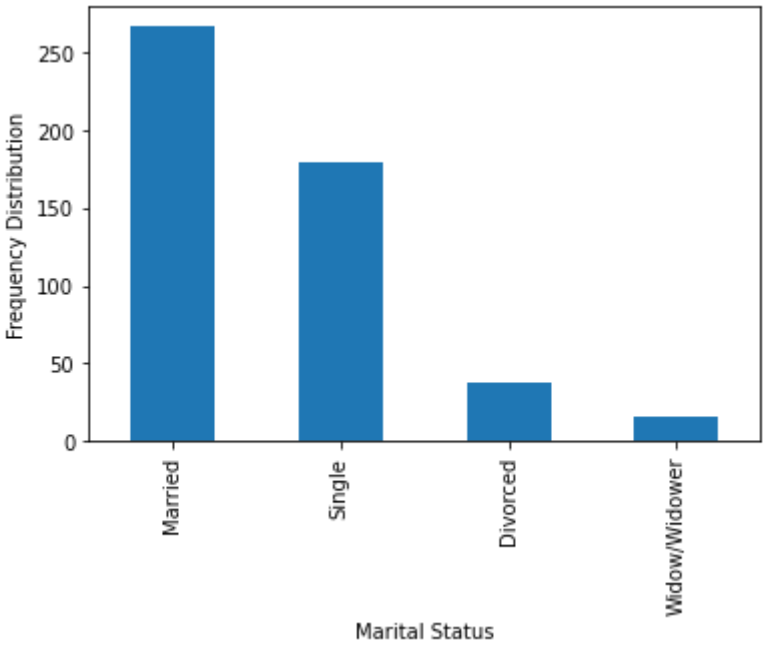
In [59]:
```
new_customer = customer.dropna()
```

In [60]:
```
new_customer.info()
```

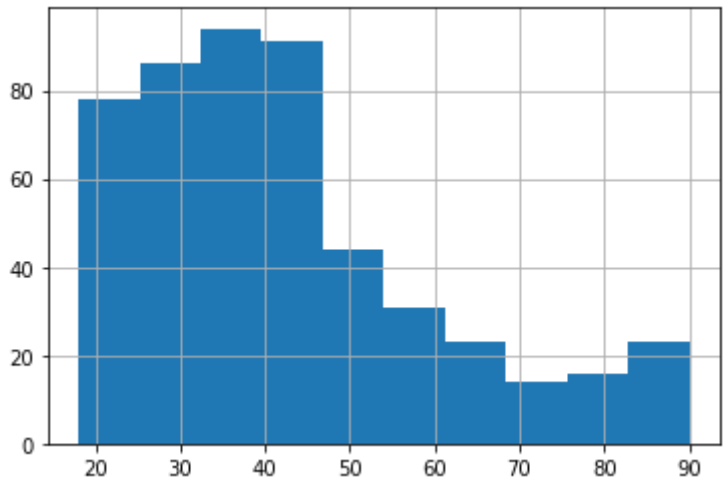
```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 264 entries, 0 to 497
Data columns (total 9 columns):
 #   Column            Non-Null Count  Dtype         
---  ------            --------------  -----         
 0   CUSTOMERID        264 non-null    int64         
 1   GENDER            264 non-null    int64         
 2   AGE               264 non-null    int64         
 3   INCOME            264 non-null    int32         
 4   EXPERIENCE SCORE  264 non-null    int64         
 5   LOYALTY GROUP     264 non-null    object        
 6   ENROLLMENT DATE   264 non-null    datetime64[ns]
 7   HOUSEHOLD SIZE    264 non-null    int64         
 8   MARITAL STATUS    264 non-null    object        
dtypes: datetime64[ns](1), int32(1), int64(5), object(2)
memory usage: 19.6+ KB
```

Graph data

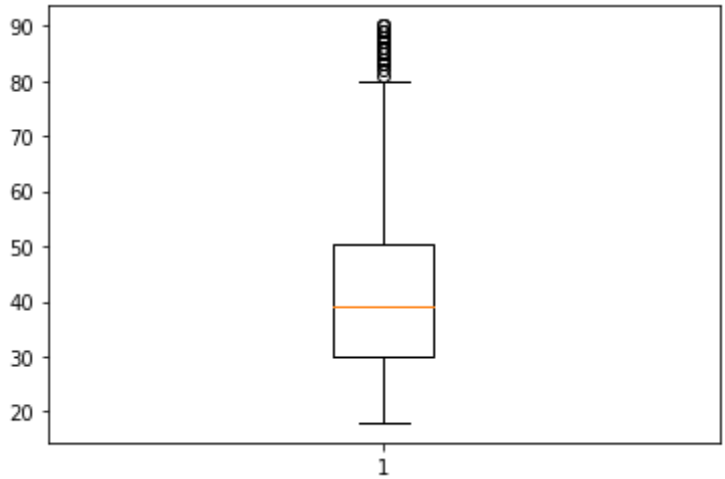
In [65]:
```
customer['MARITAL STATUS'].value_counts().plot(kind='bar')
plt.xlabel('Marital Status')
plt.ylabel('Frequency Distribution')
plt.show()
```



In [66]:
```
customer['AGE'].hist(bins=10)
plt.show()
```



In [67]:
```
plt.boxplot(customer['AGE'])
plt.show()
```



Out[68]:
```
count    500.000000
mean      42.316000
std       17.567509
min       18.000000
25%       30.000000
50%       39.000000
75%       50.250000
max       90.000000
Name: AGE, dtype: float64
```

Join Transcation with Product DataFrames

In [71]:
```
trans_product = transaction.merge(products, how='inner', left_on='PRODUCT NUM', righ
```

In [72]:
```
trans_product.head()
```

Out[72]:

	CUSTOMER NUM	PRODUCT NUM	QUANTITY PURCHASED	DISCOUNT TAKEN	TRANSACTION DATE	STOCKOUT	PRODUCT CODE	PRODU CATEGO
0	10114	30011	4	0.0	1/2/2015	0	30011	APPA
1	10086	30011	6	0.0	1/2/2015	0	30011	APPA
2	10174	30011	10	0.0	1/2/2015	0	30011	APPA
3	10401	30011	12	0.0	1/2/2015	0	30011	APPA
4	10216	30011	12	0.1	1/2/2015	0	30011	APPA

In [75]:
```
trans_product['UNIT LIST PRICE'] = trans_product['UNIT LIST PRICE'].map(lambda x: x.
```

In [77]:
```
trans_product['UNIT LIST PRICE'] = trans_product['UNIT LIST PRICE'].astype('float')
```

In [78]:
```
trans_product.dtypes
```

Out[78]:
```
CUSTOMER NUM            int64
PRODUCT NUM             int64
QUANTITY PURCHASED      int64
DISCOUNT TAKEN        float64
TRANSACTION DATE       object
STOCKOUT                int64
PRODUCT CODE            int64
PRODUCT CATEGORY       object
UNIT LIST PRICE       float64
dtype: object
```

In [79]:
```
trans_product['Total Price'] = (trans_product['UNIT LIST PRICE']*trans_product['QUAN
```

In [80]:
```
trans_product.head()
```

Out[80]:

	CUSTOMER NUM	PRODUCT NUM	QUANTITY PURCHASED	DISCOUNT TAKEN	TRANSACTION DATE	STOCKOUT	PRODUCT CODE	PRODU CATEGO
0	10114	30011	4	0.0	1/2/2015	0	30011	APPA
1	10086	30011	6	0.0	1/2/2015	0	30011	APPA
2	10174	30011	10	0.0	1/2/2015	0	30011	APPA
3	10401	30011	12	0.0	1/2/2015	0	30011	APPA
4	10216	30011	12	0.1	1/2/2015	0	30011	APPA

Grouping each category then find total revenue of each

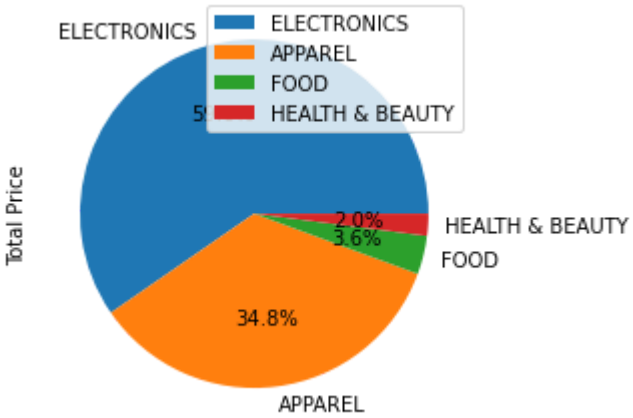
In [84]:
```
Income_by_product = trans_product.groupby('PRODUCT CATEGORY').agg({'Total Price':'su
Income_by_product
```

Out[84]:

PRODUCT CATEGORY	Total Price
ELECTRONICS	1724115.18
APPAREL	1005161.26
FOOD	103257.95
HEALTH & BEAUTY	58979.62

In [88]:
```
Income_by_product['Total Price'].plot(kind='pie', legend = True, autopct='%1.1f%%')
```

Out[88]:
```
<AxesSubplot:ylabel='Total Price'>
```



Calculate this measures for cutomer:

Total spends per category

Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js

Total spends

Most recent transaction dates

Average Discount taken

In [103…
```
Spend_by_category = trans_product.groupby(['CUSTOMER NUM','PRODUCT CATEGORY']).agg({
Spend_by_category
```

Out[103…

CUSTOMER NUM	PRODUCT CATEGORY	Total Price
10001	APPAREL	4522.14
	ELECTRONICS	1677.46
	FOOD	76.22
	HEALTH & BEAUTY	1292.23
10002	APPAREL	2627.51
...	...	...
10498	HEALTH & BEAUTY	331.68
10499	HEALTH & BEAUTY	20.01
10500	APPAREL	1684.16
	ELECTRONICS	3458.33
	FOOD	50.09

1427 rows × 1 columns

In [104…
```
Spend_by_category1 = Spend_by_category.reset_index()
```

In [106…
```
customer_spending = Spend_by_category1.pivot(index='CUSTOMER NUM', columns='PRODUCT
customer_spending.head()
```

Out[106…

CUSTOMER NUM	PRODUCT CATEGORY	APPAREL	ELECTRONICS	FOOD	HEALTH & BEAUTY
10001		4522.14	1677.46	76.22	1292.23
10002		2627.51	2812.79	302.58	NaN
10003		3082.83	5745.41	260.64	NaN
10004		3954.38	2246.19	45.27	NaN
10005		338.18	NaN	NaN	NaN

Total spends and most recent transactions by date

In [110…
```
trans_product['TRANSACTION DATE'].map(lambda x:
```

In [111…
```
recent_trans = trans_product.groupby('CUSTOMER NUM').agg({'TRANSACTION DATE':'max',
```

In [113…
```
recent_trans.rename(columns={'Total Price':'Total Spent', 'TRANSACTION DATE': 'Recen
```

Out[113…

CUSTOMER NUM	Recent TRANSACTION DATE	Total Spent
10001	2015-12-24	7568.05
10002	2015-12-21	5742.88
10003	2015-12-31	9088.88
10004	2015-12-17	6245.84
10005	2015-12-22	338.18
…	…	…
10496	2015-12-08	9242.21
10497	2015-12-24	10950.73
10498	2015-12-23	6494.98
10499	2015-12-18	20.01
10500	2015-12-31	5192.58

500 rows × 2 columns

In [114…
```
recent_trans = recent_trans.reset_index()
```

In [122…
```
customer_kpi = customer_spending.merge(recent_trans, how='inner', left_on = 'CUSTOME
```

In [123…
```
customer_kpi
```

Out[123…

	CUSTOMER NUM	APPAREL	ELECTRONICS	FOOD	HEALTH & BEAUTY	TRANSACTION DATE	Total Price
0	10001	4522.14	1677.46	76.22	1292.23	2015-12-24	7568.05
1	10002	2627.51	2812.79	302.58	NaN	2015-12-21	5742.88
2	10003	3082.83	5745.41	260.64	NaN	2015-12-31	9088.88
3	10004	3954.38	2246.19	45.27	NaN	2015-12-17	6245.84
4	10005	338.18	NaN	NaN	NaN	2015-12-22	338.18
…	…	…	…	…	…	…	…
495	10496	4941.99	4026.33	186.11	87.78	2015-12-08	9242.21
496	10497	3660.36	6612.88	457.40	220.09	2015-12-24	10950.73
497	10498	1919.40	3424.11	819.79	331.68	2015-12-23	6494.98
498	10499	NaN	NaN	NaN	20.01	2015-12-18	20.01
				3	50.09	NaN	2015-12-31	5192.58

500 rows × 7 columns

In [124...
```
customer_kpi = customer_kpi.fillna(0)
customer_kpi.head()
```

Out[124...

	CUSTOMER NUM	APPAREL	ELECTRONICS	FOOD	HEALTH & BEAUTY	TRANSACTION DATE	Total Price
0	10001	4522.14	1677.46	76.22	1292.23	2015-12-24	7568.05
1	10002	2627.51	2812.79	302.58	0.00	2015-12-21	5742.88
2	10003	3082.83	5745.41	260.64	0.00	2015-12-31	9088.88
3	10004	3954.38	2246.19	45.27	0.00	2015-12-17	6245.84
4	10005	338.18	0.00	0.00	0.00	2015-12-22	338.18

In [132...
```
customer_all = customer.merge(customer_kpi, how='inner', left_on='CUSTOMERID', right
```

In [133...
```
customer_all.head()
```

Out[133...

	CUSTOMERID	GENDER	AGE	INCOME	EXPERIENCE SCORE	LOYALTY GROUP	ENROLLMENT DATE	HOUSEHOLD SIZE	M
0	10001	0	64	133498	5	enrolled	2013-03-06	4	
1	10002	0	42	94475	9	notenrolled	NaT	6	
2	10003	0	40	88610	9	enrolled	2010-09-02	5	
3	10004	0	38	84313	8	enrolled	2015-04-06	1	
4	10005	0	30	51498	3	notenrolled	NaT	1	

cross-tabulating the Gender column with Loyalty

Question: Does gender affect loyalty enrollment?

In [134...
```
table = pd.crosstab(customer_all['GENDER'],customer_all['LOYALTY GROUP'])
```

In [135...
```
table
```

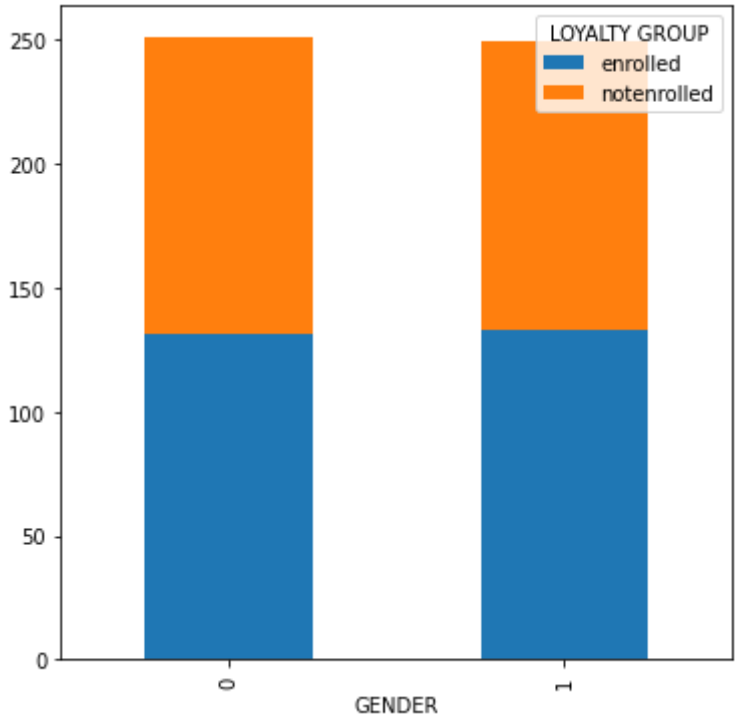
Out[135...

LOYALTY GROUP	enrolled	notenrolled
GENDER		

LOYALTY GROUP	enrolled	notenrolled
GENDER		
1	133	116

In [138…

```
table.plot(kind='bar', stacked=True, figsize = (6,6))
plt.show()
```



Answer:

By visually inspecting the chart, apparently being male or female does not affect the enrollment much. The ratio of enrolled to non-enrolled is the same for both genders (which looks like 1:1).Both genders are likely to join with the same probability.

Question: Does the experience score affect loyalty enrollment?

In [141…

```
table1 = pd.crosstab(customer_all['EXPERIENCE SCORE'],customer_all['LOYALTY GROUP'])
table1
```

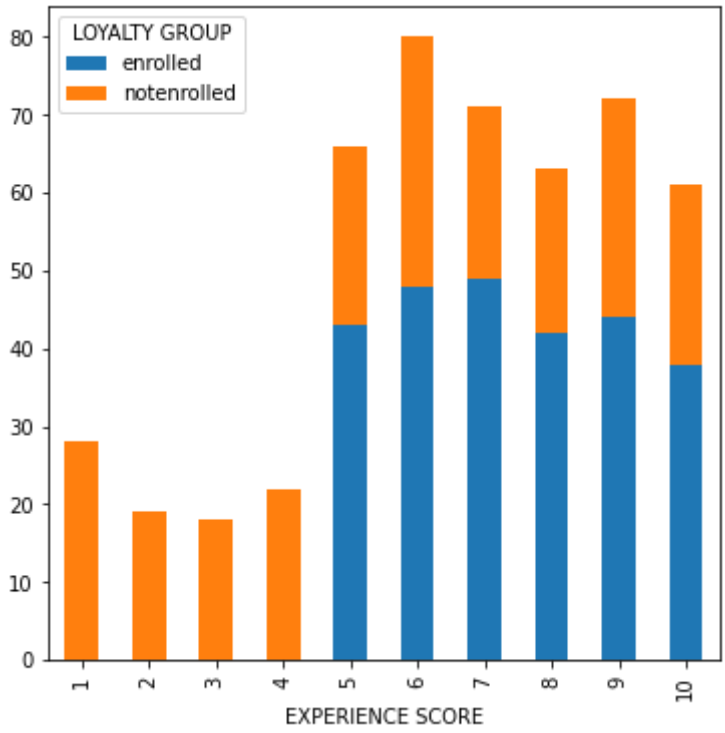
Out[141…

LOYALTY GROUP	enrolled	notenrolled
EXPERIENCE SCORE		
1	0	28
2	0	19
3	0	18
4	0	22

LOYALTY GROUP	enrolled	notenrolled
EXPERIENCE SCORE		
5	43	23
6	48	32
7	49	22
8	42	21
9	44	28
10	38	23

In [142…

```
table1.plot(kind='bar', stacked=True, figsize = (6,6))
plt.show()
```



Answer:

It appears so because customers with experience scores below 5 (1 - 4) did not enroll at all. However, customers with scores 5 or more are likely to enroll. So, knowing a customer's experience score can predict the likelihood of a customer enrolling. If a customer has a score below 5 score, then they are not likely to enroll at all. However, if their experience score is 5 or more, they are likely to join with a probability of 60% - 70%.

Question: Does marital status affect loyalty enrollment?

In [143…

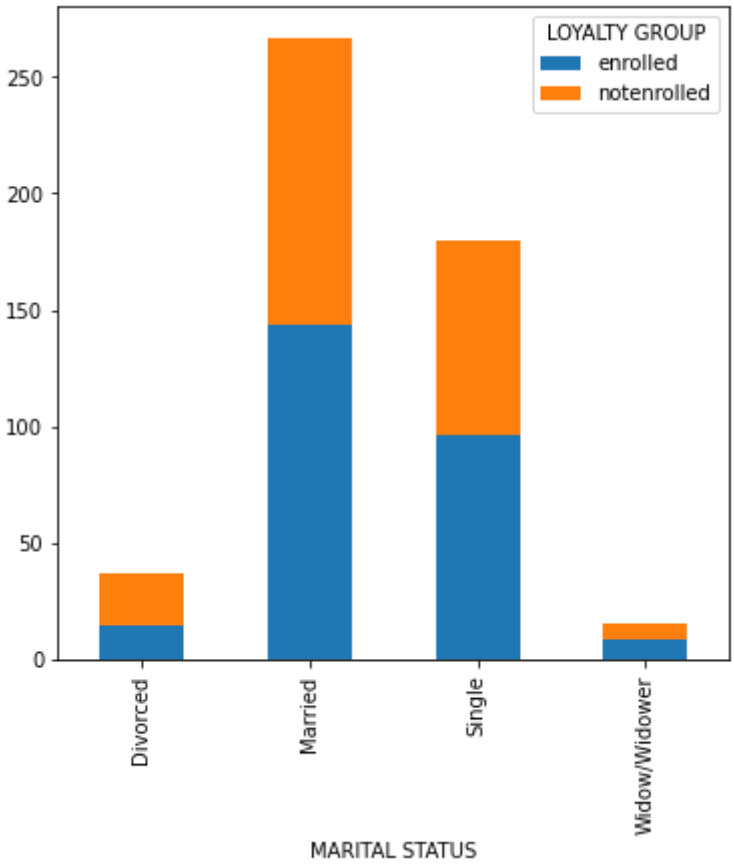
```
table3 = pd.crosstab(customer_all['MARITAL STATUS'],customer_all['LOYALTY GROUP'])
```

Out[143...

LOYALTY GROUP	enrolled	notenrolled
MARITAL STATUS		
Divorced	15	22
Married	144	123
Single	96	84
Widow/Widower	9	7

In [144...

```
table3.plot(kind='bar', stacked=True, figsize = (6,6))
plt.show()
```



Answer:

Apparently, marital status does not have much effect on loyalty enrollment. The ratio of enrolled to non-enrolled appears to be almost the same for all marital statuses, especially for married and singles (who are most of Retailer X's customer base).

Question: Does age affect loyalty enrollment?

In [145...

```
customer_all['age_bins'] = pd.cut(customer_all['AGE'],10)
```

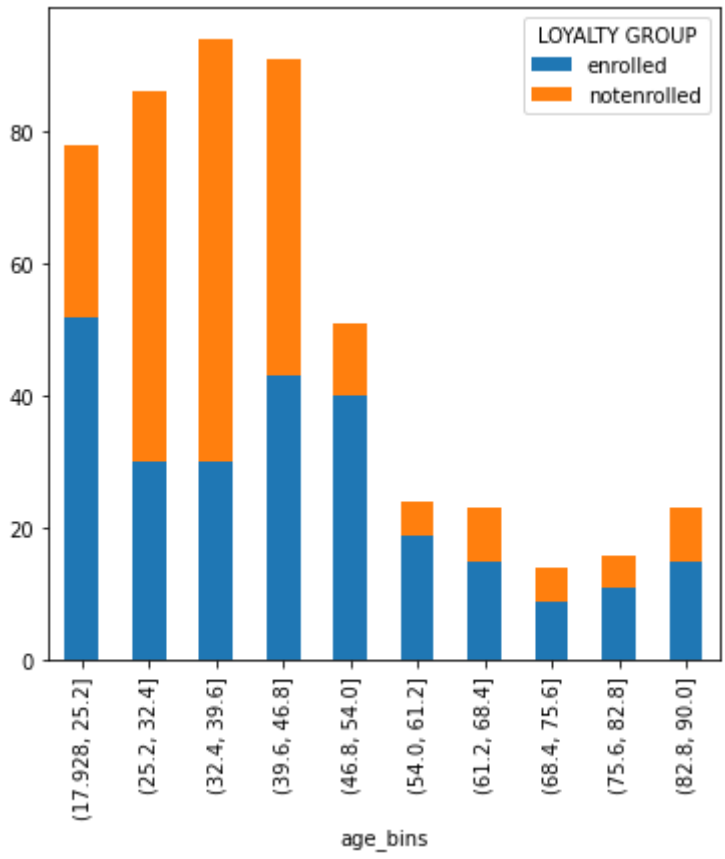
```
ge_bins'],customer_all['LOYALTY GROUP'])
table4
```

Out[146…

LOYALTY GROUP	enrolled	notenrolled
age_bins		
(17.928, 25.2]	52	26
(25.2, 32.4]	30	56
(32.4, 39.6]	30	64
(39.6, 46.8]	43	48
(46.8, 54.0]	40	11
(54.0, 61.2]	19	5
(61.2, 68.4]	15	8
(68.4, 75.6]	9	5
(75.6, 82.8]	11	5
(82.8, 90.0]	15	8

In [147…

```
table4.plot(kind='bar', stacked=True, figsize = (6,6))
plt.show()
```



Answer:

Based on the above chart, middle aged customers are less likely to join loyalty programs, and younger and elderly people are more likely to join. This piece of information reveals that there is some sort of significant relationship between age and loyalty enrollment

Question: Does total spent affect loyalty enrollment?

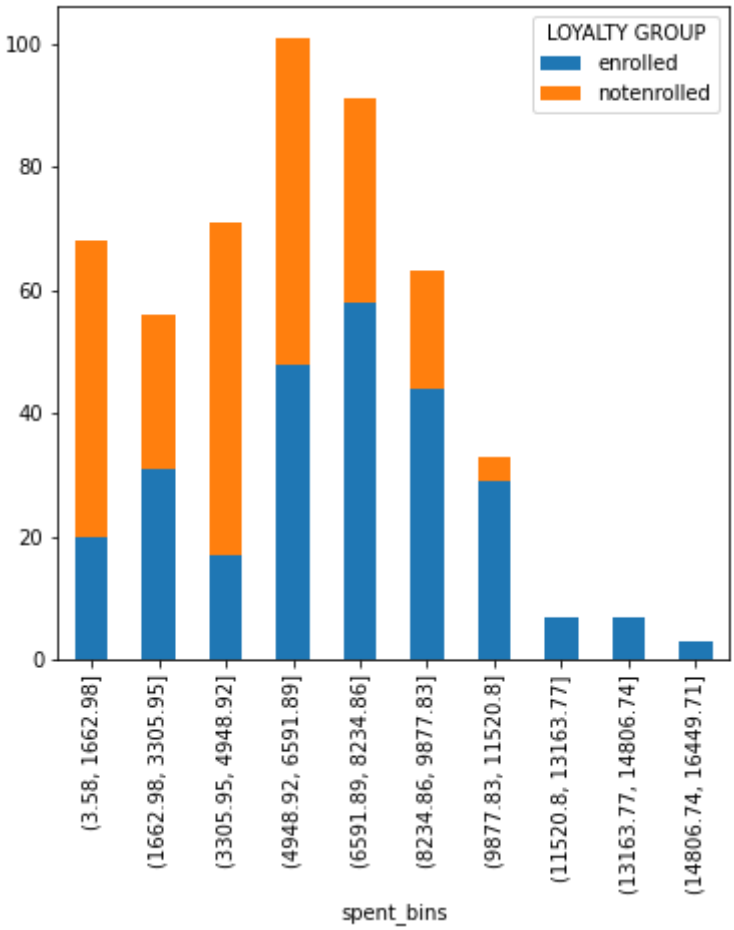
In [148…
```
customer_all['spent_bins'] = pd.cut(customer_all['Total Price'],10)
```

In [149…
```
table5 = pd.crosstab(customer_all['spent_bins'],customer_all['LOYALTY GROUP'])
table5
```

Out[149…

LOYALTY GROUP	enrolled	notenrolled
spent_bins		
(3.58, 1662.98]	20	48
(1662.98, 3305.95]	31	25
(3305.95, 4948.92]	17	54
(4948.92, 6591.89]	48	53
(6591.89, 8234.86]	58	33
(8234.86, 9877.83]	44	19
(9877.83, 11520.8]	29	4
(11520.8, 13163.77]	7	0
(13163.77, 14806.74]	7	0
(14806.74, 16449.71]	3	0

In [150…
```
table5.plot(kind='bar', stacked=True, figsize = (6,6))
plt.show()
```



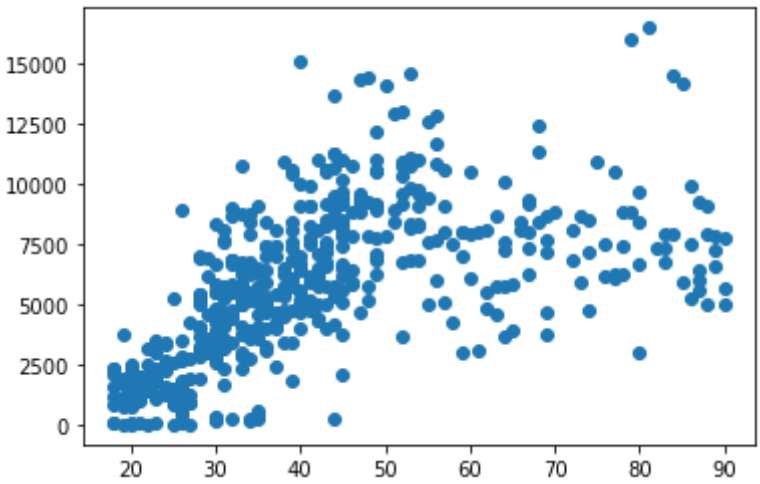
Answer:

As an overall pattern, the graph shows that as the total spend of customer increase, so does their chances of enrollment as well.

Question: Based on this graph, does age influence total spending?

In [155…

```
plt.scatter(customer_all['AGE'],customer_all['Total Price'])
plt.show()
```



Answer:

To some degree, yes. If you look at the figure, you find that there is some positive correlation between both variables, that is, the total spend of a customer increases as their age increases.

Check correlation coefficient by Pearson Coefficient

In [157…
```
from scipy.stats import pearsonr
```

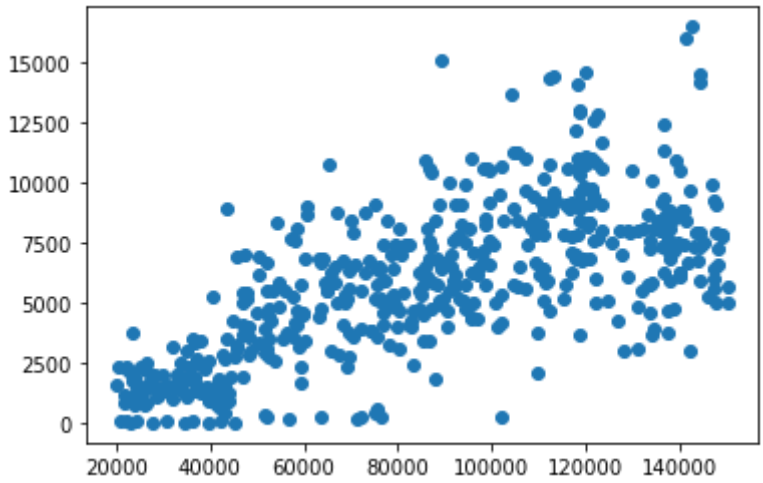
In [158…
```
pearsonr(customer_all['AGE'],customer_all['Total Price'])
```

Out[158…
```
(0.5794158742016526, 3.576098614466672e-46)
```

The correlation coefficient is 0.579, which implies a moderately strong correlation between both factors.

Question: Based on this graph, does income influence total spending?

In [159…
```
plt.scatter(customer_all['INCOME'],customer_all['Total Price'])
plt.show()
```



In [160…
```
pearsonr(customer_all['INCOME'],customer_all['Total Price'])
```

Out[160…
```
(0.6894401416952896, 9.242752867744121e-72)
```

Income has a higher correlation coefficient [illegible] cates that it has a strong

relationship with customer spending. This finding is logical: Anyone would expect that customer spending is heavily dependent on their income.

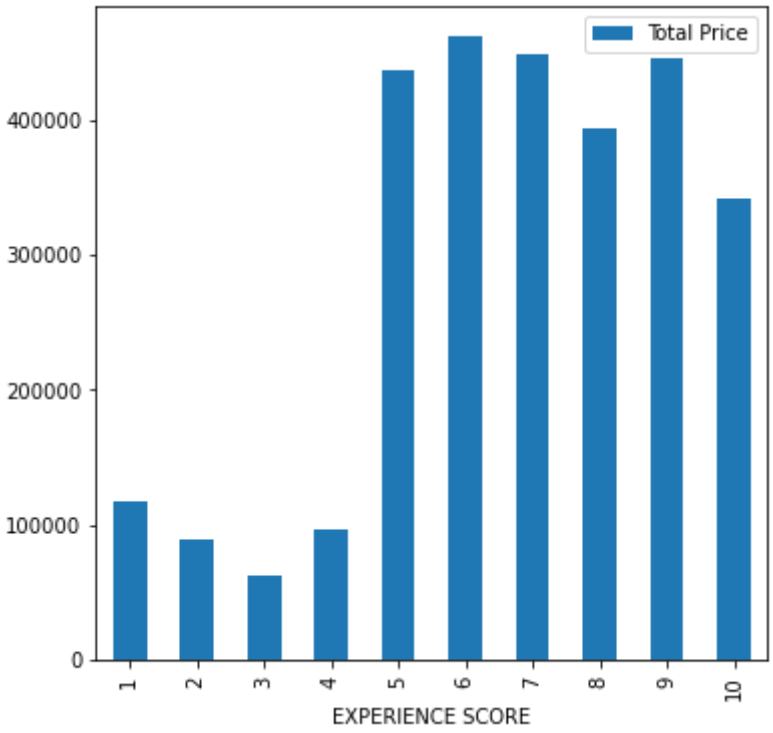
Examine the relationship between the experience score (which is a categorical feature) and total spend (which is continuous)

In [162…
```
Spent = customer_all.groupby('EXPERIENCE SCORE').agg({'Total Price':'sum'})
Spent
```

Out[162…

EXPERIENCE SCORE	Total Price
1	116921.80
2	88723.87
3	63088.87
4	96944.54
5	436216.73
6	461005.15
7	448824.73
8	393248.62
9	444632.33
10	341907.37

In [164…
```
Spent.plot(kind='bar', stacked=True, figsize = (6,6))
plt.show()
```



Obviously, customers with experience scores 1 - 4 have a relatively lower average spend than customers with higher experience scores (5 - 10). This indicates some sort of relationship between the two variables.