# updated assignment

March 8, 2022

Project Title:Vehicle Sales Analysis

Authors: Mostafa Abdelazim, Zeina Kishk, and Hania Raslan

Abstract: This dataset describes the vehicle market in prospective of sales depending on several factors: pricing,timing, location(city,and country), and describes how the product reached the consumer in addition to the process of shipping. We obtained the dataset from kaggle.

[3]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import statsmodels.api as sm
from statsmodels.graphics.mosaicplot import mosaic
```

[2]:
```
data = pd.read_csv("dataset.csv", encoding= 'unicode_escape')
print(data.head())
print(data.tail())
```

```
   ORDERNUMBER  QUANTITYORDERED  PRICEEACH  ORDERLINENUMBER    SALES   STATUS  \
0        10107               30      95.70                2  2871.00  Shipped
1        10121               34      81.35                5  2765.90  Shipped
2        10134               41      94.74                2  3884.34  Shipped
3        10145               45      83.26                6  3746.70  Shipped
4        10159               49     100.00               14  5205.27  Shipped

   QTR_ID  MONTH_ID  YEAR_ID  PRODUCTLINE  …             PHONE  \
0       1         2     2003  Motorcycles  …        2125557818
1       2         5     2003  Motorcycles  …        26.47.1555
2       3         7     2003  Motorcycles  …  +33 1 46 62 7555
3       3         8     2003  Motorcycles  …        6265557265
4       4        10     2003  Motorcycles  …        6505551386

                  ADDRESSLINE1           CITY STATE POSTALCODE COUNTRY  \
0        897 Long Airport Avenue            NYC    NY      10022     USA
1             59 rue de l'Abbaye          Reims   NaN      51100  France
2  27 rue du Colonel Pierre Avia          Paris   NaN      75508  France
3             78934 Hillside Dr.       Pasadena    CA      90003     USA
4                7734 Strong St.  San Francisco    CA        NaN     USA
```

```
  TERRITORY CONTACTLASTNAME CONTACTFIRSTNAME DEALSIZE
0       NaN              Yu             Kwai    Small
1      EMEA         Henriot             Paul    Small
2      EMEA        Da Cunha           Daniel   Medium
3       NaN           Young            Julie   Medium
4       NaN           Brown            Julie   Medium

[5 rows x 23 columns]
      ORDERNUMBER  QUANTITYORDERED  PRICEEACH  ORDERLINENUMBER    SALES  \
2818        10350               20     100.00               15  2244.40
2819        10373               29     100.00                1  3978.51
2820        10386               43     100.00                4  5417.57
2821        10397               34      62.24                1  2116.16
2822        10414               47      65.52                9  3079.44

        STATUS  QTR_ID  MONTH_ID  YEAR_ID PRODUCTLINE  …           PHONE  \
2818   Shipped       4        12     2004       Ships  …  (91) 555 94 44
2819   Shipped       1         1     2005       Ships  …      981-443655
2820  Resolved       1         3     2005       Ships  …  (91) 555 94 44
2821   Shipped       1         3     2005       Ships  …      61.77.6555
2822   On Hold       2         5     2005       Ships  …      6175559555

                ADDRESSLINE1      CITY STATE POSTALCODE  COUNTRY TERRITORY  \
2818     C/ Moralzarzal, 86    Madrid   NaN      28034    Spain      EMEA
2819            Torikatu 38      Oulu   NaN      90110  Finland      EMEA
2820     C/ Moralzarzal, 86    Madrid   NaN      28034    Spain      EMEA
2821  1 rue Alsace-Lorraine  Toulouse   NaN      31000   France      EMEA
2822     8616 Spinnaker Dr.    Boston    MA      51003      USA       NaN

     CONTACTLASTNAME CONTACTFIRSTNAME DEALSIZE
2818          Freyre            Diego    Small
2819       Koskitalo           Pirkko   Medium
2820          Freyre            Diego   Medium
2821          Roulet          Annette    Small
2822         Yoshido             Juri   Medium

[5 rows x 23 columns]
```

[3]:
```
##Mean
print(data['PRICEEACH'].mean())
print(data['SALES'].mean())
print(data['QUANTITYORDERED'].mean())
print(data['ORDERLINENUMBER'].mean())
print(data['MSRP'].mean())
```

```
83.65854410201929
3553.88907190932
35.09280906836698
```

```
6.466170740347148
100.71555083244775
```

[4]:
```
#Median
print(data['PRICEEACH'].median())
print(data['SALES'].median())
print(data['QUANTITYORDERED'].median())
print(data['ORDERLINENUMBER'].median())
print(data['MSRP'].median())
```

```
95.7
3184.8
35.0
6.0
99.0
```

[5]:
```
#Variance and Std
print(data['ORDERLINENUMBER'].var())
print(data['ORDERLINENUMBER'].std())
print(data['MSRP'].var())
print(data['MSRP'].std())
print(data['PRICEEACH'].var())
print(data['PRICEEACH'].std())
print(data['QUANTITYORDERED'].var())
print(data['QUANTITYORDERED'].std())
print(data['SALES'].var())
print(data['SALES'].std())
```

```
17.85773185886007
4.225840964690942
1615.0682449746778
40.187911677203104
407.0014334217844
20.174276527840703
94.89570659960627
9.74144273706961
3392467.0677432865
1841.8651057401805
```

[6]:
```
#Descibe (Quantiles)
data.describe()
```

[6]:
```
        ORDERNUMBER  QUANTITYORDERED    PRICEEACH  ORDERLINENUMBER  \
count   2823.000000      2823.000000  2823.000000      2823.000000
mean   10258.725115        35.092809    83.658544         6.466171
std       92.085478         9.741443    20.174277         4.225841
min    10100.000000         6.000000    26.880000         1.000000
25%    10180.000000        27.000000    68.860000         3.000000
50%    10262.000000        35.000000    95.700000         6.000000
```

```
75%     10333.500000         43.000000    100.000000          9.000000
max     10425.000000         97.000000    100.000000         18.000000

               SALES        QTR_ID      MONTH_ID      YEAR_ID          MSRP
count    2823.000000   2823.000000   2823.000000   2823.00000   2823.000000
mean     3553.889072      2.717676      7.092455   2003.81509    100.715551
std      1841.865106      1.203878      3.656633      0.69967     40.187912
min       482.130000      1.000000      1.000000   2003.00000     33.000000
25%      2203.430000      2.000000      4.000000   2003.00000     68.000000
50%      3184.800000      3.000000      8.000000   2004.00000     99.000000
75%      4508.000000      4.000000     11.000000   2004.00000    124.000000
max     14082.800000      4.000000     12.000000   2005.00000    214.000000
```

[7]:
```python
#Discrete Values frequency
print(data['PRODUCTLINE'].value_counts(normalize=True))
print(data['MONTH_ID'].value_counts(normalize=True))
print(data['STATUS'].value_counts(normalize=True))
print(data['CITY'].value_counts(normalize=True))
print(data['COUNTRY'].value_counts(normalize=True))
```

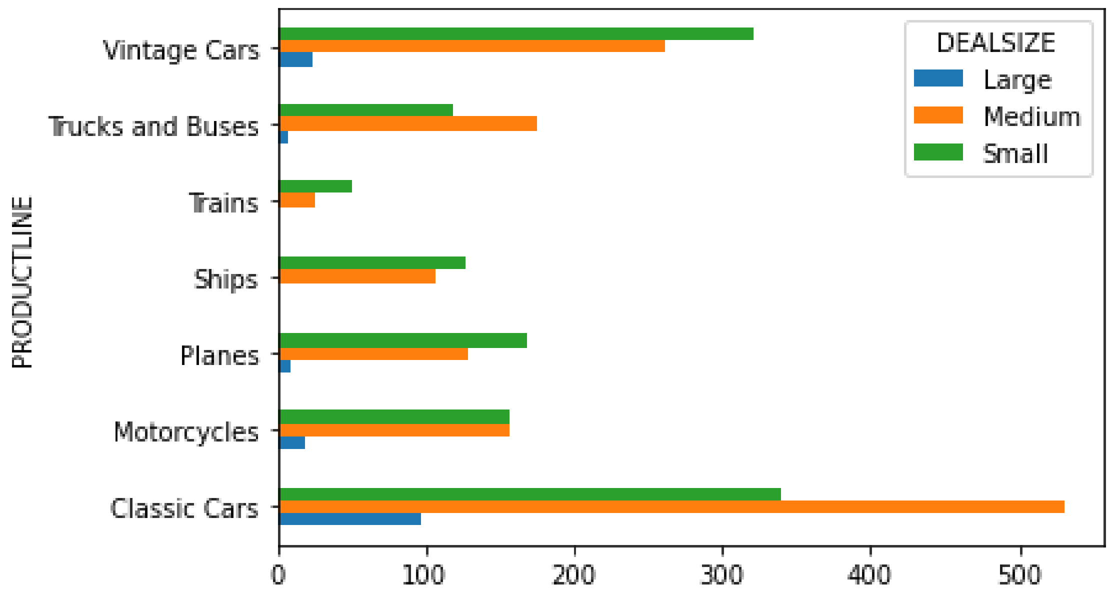
```
Classic Cars        0.342543
Vintage Cars        0.215019
Motorcycles         0.117251
Planes              0.108395
Trucks and Buses    0.106624
Ships               0.082891
Trains              0.027276
Name: PRODUCTLINE, dtype: float64
11    0.211477
10    0.112292
5     0.089267
1     0.081119
2     0.079348
3     0.075097
8     0.067659
12    0.063762
4     0.063053
9     0.060574
7     0.049947
6     0.046405
Name: MONTH_ID, dtype: float64
Shipped       0.927028
Cancelled     0.021254
Resolved      0.016649
On Hold       0.015586
In Process    0.014524
Disputed      0.004959
Name: STATUS, dtype: float64
```

```
Madrid          0.107687
San Rafael      0.063762
NYC             0.053843
Singapore       0.027984
Paris           0.024796
                  …
Graz            0.005313
Los Angeles     0.004959
Munich          0.004959
Burbank         0.004605
Charleroi       0.002834
Name: CITY, Length: 73, dtype: float64
USA             0.355650
Spain           0.121148
France          0.111229
Australia       0.065533
UK              0.051010
Italy           0.040028
Finland         0.032589
Norway          0.030110
Singapore       0.027984
Canada          0.024796
Denmark         0.022317
Germany         0.021962
Sweden          0.020191
Austria         0.019483
Japan           0.018420
Belgium         0.011690
Switzerland     0.010981
Philippines     0.009210
Ireland         0.005668
Name: COUNTRY, dtype: float64
```

[8]:
```
#Grouping by a variable
grouped = data.groupby('PRODUCTLINE')
```

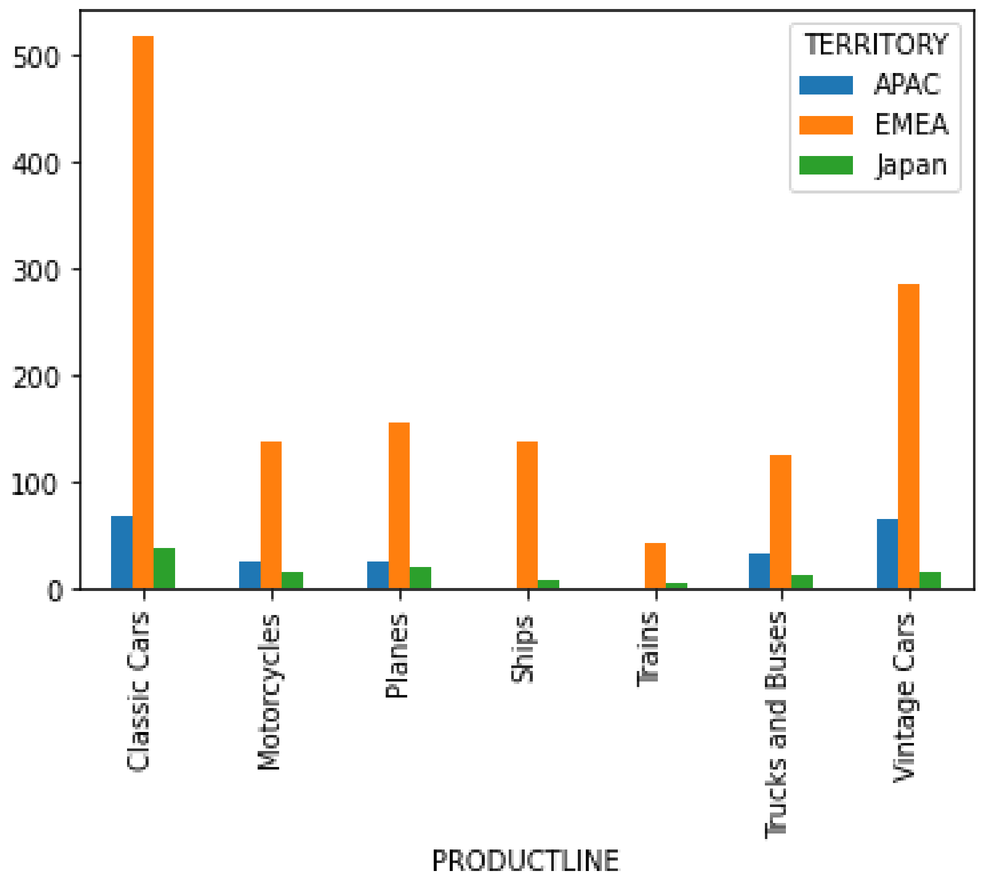
[9]:
```
#Barplot 1
grouped['DEALSIZE'].value_counts().unstack().plot(kind="barh")
```

[9]:
```
<AxesSubplot:ylabel='PRODUCTLINE'>
```



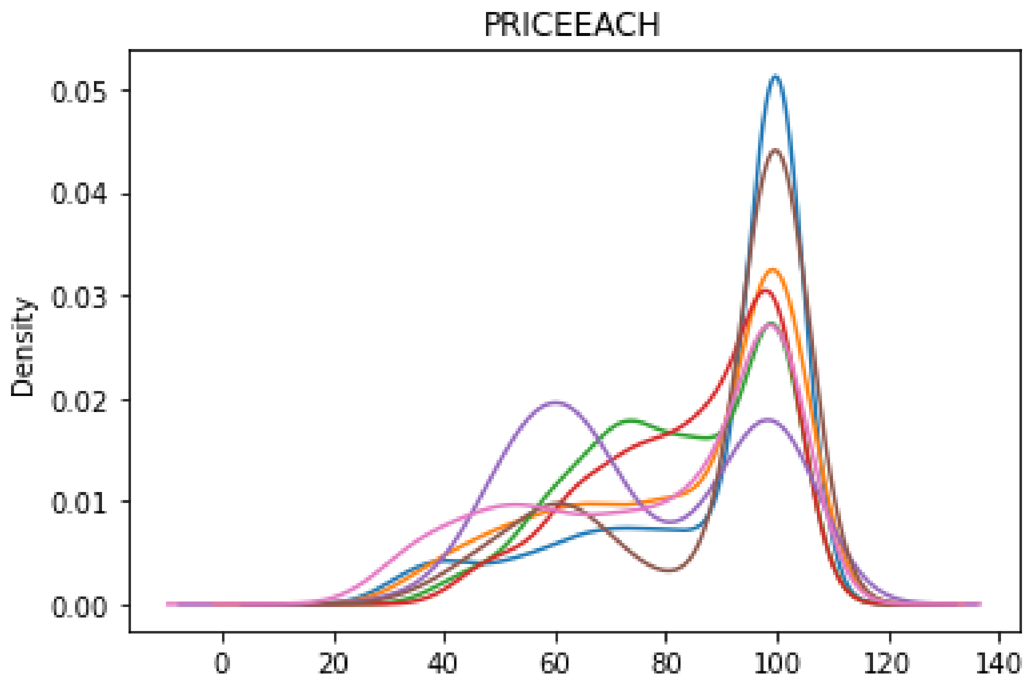
```
[10]: #Barplot 2
      grouped['TERRITORY'].value_counts().unstack().plot(kind="bar")
```

```
[10]: <AxesSubplot:xlabel='PRODUCTLINE'>
```



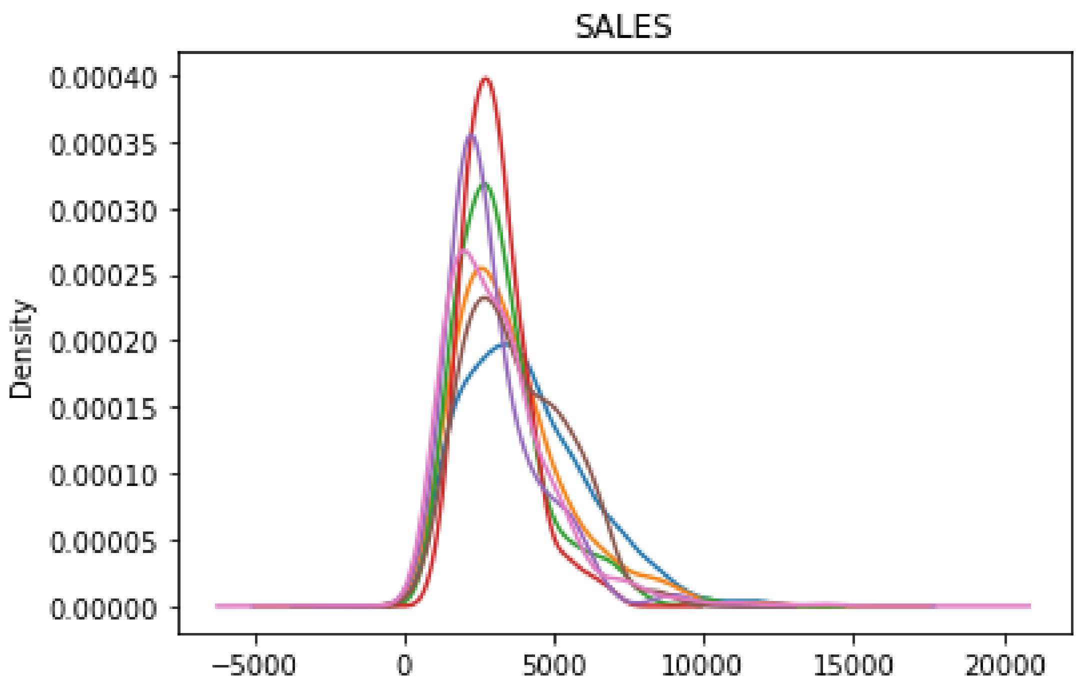
```
[11]: #Density Plot 1
      grouped['PRICEEACH'].plot(kind='density', title='PRICEEACH')
```

```
[11]: PRODUCTLINE
      Classic Cars        AxesSubplot(0.125,0.125;0.775x0.755)
      Motorcycles         AxesSubplot(0.125,0.125;0.775x0.755)
      Planes              AxesSubplot(0.125,0.125;0.775x0.755)
      Ships               AxesSubplot(0.125,0.125;0.775x0.755)
      Trains              AxesSubplot(0.125,0.125;0.775x0.755)
      Trucks and Buses    AxesSubplot(0.125,0.125;0.775x0.755)
      Vintage Cars        AxesSubplot(0.125,0.125;0.775x0.755)
      Name: PRICEEACH, dtype: object
```



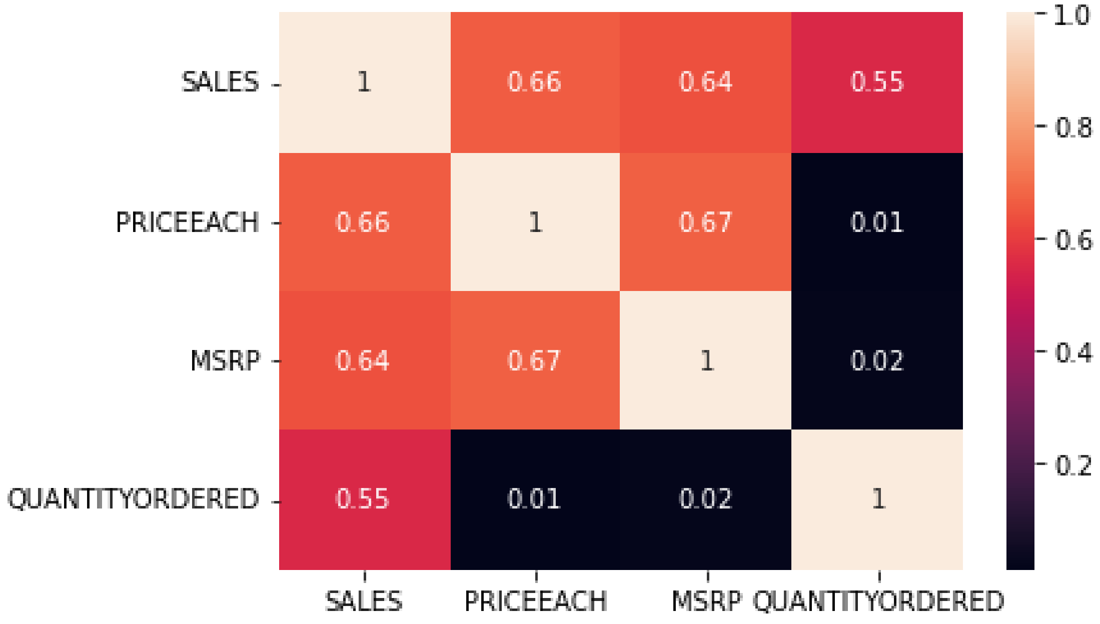
[12]:
```
#Density Plot 2
grouped['SALES'].plot(kind='density', title='SALES')
```

[12]:
```
PRODUCTLINE
Classic Cars        AxesSubplot(0.125,0.125;0.775x0.755)
Motorcycles         AxesSubplot(0.125,0.125;0.775x0.755)
Planes              AxesSubplot(0.125,0.125;0.775x0.755)
Ships               AxesSubplot(0.125,0.125;0.775x0.755)
Trains              AxesSubplot(0.125,0.125;0.775x0.755)
Trucks and Buses    AxesSubplot(0.125,0.125;0.775x0.755)
Vintage Cars        AxesSubplot(0.125,0.125;0.775x0.755)
Name: SALES, dtype: object
```



```
[13]: #Heat Map 1
      cormat=data[['SALES','PRICEEACH','MSRP','QUANTITYORDERED']].corr().round(2)
      sns.heatmap(cormat, annot=True)
```

```
[13]: <AxesSubplot:>
```



```
[14]: # CORRELATION
      print(data.corr())
      # CORRELATION BETWEEN SALES AND PRICE EACH
      print(data[['SALES','PRICEEACH']].corr(method='pearson'))
      # CORRELATION BETWEEN SALES AND MSRP
      print(data[['MSRP','SALES']].corr(method='pearson'))
      # CORRELATION BETWEEN MSRP AND PRICE EACH
      print(data[['MSRP','PRICEEACH']].corr(method='pearson'))
      # CORRELATION BETWEEN QUANTITYORDERED AND PRICE EACH
      print(data[['QUANTITYORDERED','PRICEEACH']].corr(method='pearson'))
      # CORRELATION BETWEEN SALES AND QUANTITYORDERED
      print(data[['QUANTITYORDERED','SALES']].corr(method='pearson'))
```

```
                 ORDERNUMBER  QUANTITYORDERED  PRICEEACH  ORDERLINENUMBER  \
ORDERNUMBER         1.000000         0.065543  -0.002935        -0.055550
QUANTITYORDERED     0.065543         1.000000   0.005564        -0.018397
PRICEEACH          -0.002935         0.005564   1.000000        -0.020965
ORDERLINENUMBER    -0.055550        -0.018397  -0.020965         1.000000
SALES               0.039919         0.551426   0.657841        -0.058400
QTR_ID             -0.051383        -0.035323   0.008712         0.040716
MONTH_ID           -0.039723        -0.039048   0.005152         0.034016
YEAR_ID             0.904596         0.069535  -0.005938        -0.057367
MSRP               -0.010280         0.017881   0.670625        -0.021067

                    SALES    QTR_ID  MONTH_ID   YEAR_ID      MSRP
ORDERNUMBER      0.039919 -0.051383 -0.039723  0.904596 -0.010280
QUANTITYORDERED  0.551426 -0.035323 -0.039048  0.069535  0.017881
PRICEEACH        0.657841  0.008712  0.005152 -0.005938  0.670625
ORDERLINENUMBER -0.058400  0.040716  0.034016 -0.057367 -0.021067
SALES            1.000000 -0.006796 -0.009605  0.035647  0.635239
QTR_ID          -0.006796  1.000000  0.979300 -0.433052  0.010234
MONTH_ID        -0.009605  0.979300  1.000000 -0.430163  0.008170
YEAR_ID          0.035647 -0.433052 -0.430163  1.000000 -0.014310
MSRP             0.635239  0.010234  0.008170 -0.014310  1.000000
              SALES  PRICEEACH
SALES      1.000000   0.657841
PRICEEACH  0.657841   1.000000
           MSRP     SALES
MSRP   1.000000  0.635239
SALES  0.635239  1.000000
               MSRP  PRICEEACH
MSRP       1.000000   0.670625
PRICEEACH  0.670625   1.000000
                 QUANTITYORDERED  PRICEEACH
QUANTITYORDERED         1.000000   0.005564
```

```
PRICEEACH              0.005564   1.000000
                 QUANTITYORDERED     SALES
QUANTITYORDERED        1.000000  0.551426
SALES                  0.551426  1.000000
```

For the correlation between SALES and PRICEEACH, we can see that they have a positive correlation. For the correlation between SALES and MSRP, we can see that they also have a positive correlation which is very similar to the one between SALES and PRICEEACH which may be considered moderate (not strong but not weak). For the correrlation between MSRP and PRICEEACH, it is also a positive moderate correlation. For the correlation between QUANTITYORDERED and PRICEEACH, we can see that they have a very weak positive correlation. Lastly, for the correlation between QUANTITYORDERED and SALES, the correlation is also moderate and positive.

[15]:
```
#Cross Tables (contengiency table)
tab = pd.crosstab(data['SALES'], data['PRICEEACH'])
print(tab)
tab1 = pd.crosstab(data['SALES'], data['MSRP'])
print(tab1)
tab2 = pd.crosstab(data['PRODUCTLINE'], data['SALES'])
print(tab2)
tab3 = pd.crosstab(data['SALES'], data['QUANTITYORDERED'])
print(tab3)
tab4 = pd.crosstab(data['MSRP'], data['QUANTITYORDERED'])
print(tab4)
```

```
PRICEEACH  26.88   27.22   28.29   28.88   29.21   29.54   29.70   29.87   \
SALES
482.13         0       0       0       0       0       0       0       0
541.14         0       0       0       0       0       0       0       0
553.95         0       0       0       0       0       0       0       0
577.60         0       0       0       1       0       0       0       0
640.05         0       0       0       0       0       0       0       0
…            …     …     …     …     …     …     …     …
11886.60       0       0       0       0       0       0       0       0
11887.80       0       0       0       0       0       0       0       0
12001.00       0       0       0       0       0       0       0       0
12536.50       0       0       0       0       0       0       0       0
14082.80       0       0       0       0       0       0       0       0

PRICEEACH  30.06   30.20   …  99.55   99.57   99.58   99.66   99.67   \
SALES                      …
482.13         0       0   …      0       0       0       0       0
541.14         0       0   …      0       0       0       0       0
553.95         0       0   …      0       0       0       0       0
577.60         0       0   …      0       0       0       0       0
640.05         0       0   …      0       0       0       0       0
…            …     …  …    …    …     …     …     …
11886.60       0       0   …      0       0       0       0       0
11887.80       0       0   …      0       0       0       0       0
```

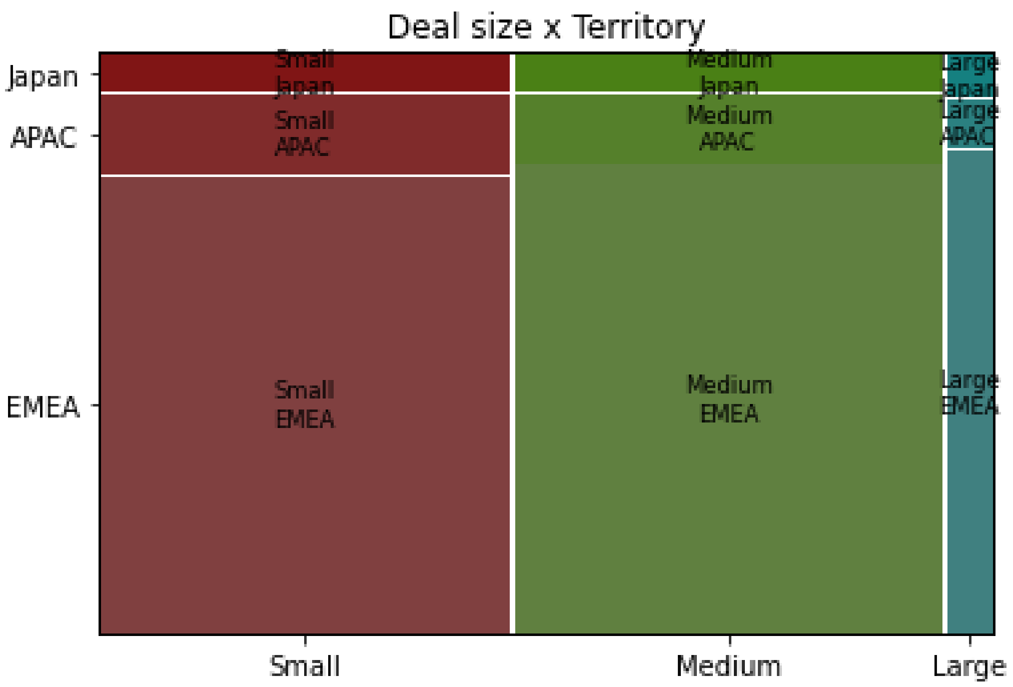
```
12001.00        0        0  …        0        0        0        0        0
12536.50        0        0  …        0        0        0        0        0
14082.80        0        0  …        0        0        0        0        0

PRICEEACH  99.69   99.72   99.82   99.91   100.00
SALES
482.13          0       0       0       0       0
541.14          0       0       0       0       0
553.95          0       0       0       0       0
577.60          0       0       0       0       0
640.05          0       0       0       0       0
…             …      …      …      …      …
11886.60        0       0       0       0       1
11887.80        0       0       0       0       1
12001.00        0       0       0       0       1
12536.50        0       0       0       0       1
14082.80        0       0       0       0       1

[2763 rows x 1016 columns]
MSRP      33   35   37   40   41   43   44   49   50   53   …  157  163  \
SALES                                                      …
482.13      0    0    0    0    0    0    0    0    0    0  …    0    0
541.14      0    0    0    0    0    0    0    0    0    0  …    0    0
553.95      0    0    0    0    1    0    0    0    0    0  …    0    0
577.60      1    0    0    0    0    0    0    0    0    0  …    0    0
640.05      0    0    1    0    0    0    0    0    0    0  …    0    0
…         …  …  …  …  …  …  …  …  …  …  …  …  …  …
11886.60    0    0    0    0    0    0    0    0    0    0  …    0    0
11887.80    0    0    0    0    0    0    0    0    0    0  …    0    0
12001.00    0    0    0    0    0    0    0    0    0    0  …    0    0
12536.50    0    0    0    0    0    0    0    0    0    0  …    0    0
14082.80    0    0    0    0    0    0    0    0    0    0  …    0    0

MSRP      168  169  170  173  193  194  207  214
SALES
482.13      0    0    0    0    0    0    0    0
541.14      0    0    0    0    0    0    0    0
553.95      0    0    0    0    0    0    0    0
577.60      0    0    0    0    0    0    0    0
640.05      0    0    0    0    0    0    0    0
…         …  …  …  …  …  …  …  …
11886.60    0    0    0    0    1    0    0    0
11887.80    0    1    0    0    0    0    0    0
12001.00    0    0    0    0    0    0    0    1
12536.50    0    0    0    0    0    0    0    0
14082.80    0    0    1    0    0    0    0    0

[2763 rows x 80 columns]
```

```
SALES             482.13    541.14    553.95    577.60    640.05    651.80    \
PRODUCTLINE
Classic Cars             0         0         0         0         1         0
Motorcycles              0         0         0         0         0         1
Planes                   0         0         0         0         0         0
Ships                    0         0         0         0         0         0
Trains                   0         0         0         0         0         0
Trucks and Buses         1         0         0         0         0         0
Vintage Cars             0         1         1         1         0         0

SALES             652.35    683.80    694.60    703.60    …  10993.50  \
PRODUCTLINE                                             …
Classic Cars             0         0         0         0  …         1
Motorcycles              0         0         0         0  …         0
Planes                   0         0         0         0  …         0
Ships                    0         0         0         0  …         0
Trains                   0         0         0         0  …         0
Trucks and Buses         0         0         0         0  …         0
Vintage Cars             1         1         1         1  …         0

SALES             11279.20  11336.70  11623.70  11739.70  11886.60  11887.80  \
PRODUCTLINE
Classic Cars             1         0         1         1         0         1
Motorcycles              0         0         0         0         1         0
Planes                   0         0         0         0         0         0
Ships                    0         0         0         0         0         0
Trains                   0         0         0         0         0         0
Trucks and Buses         0         0         0         0         0         0
Vintage Cars             0         1         0         0         0         0

SALES             12001.00  12536.50  14082.80
PRODUCTLINE
Classic Cars             1         0         0
Motorcycles              0         0         0
Planes                   0         0         0
Ships                    0         0         0
Trains                   0         0         0
Trucks and Buses         0         0         0
Vintage Cars             0         1         1

[7 rows x 2763 columns]
QUANTITYORDERED  6   10  11  12  13  15  16  18  19  20  …  61  62  64  65  \
SALES                                                    …
482.13            0   0   1   0   0   0   0   0   0   0  …   0   0   0   0
541.14            1   0   0   0   0   0   0   0   0   0  …   0   0   0   0
553.95            0   0   0   0   0   1   0   0   0   0  …   0   0   0   0
577.60            0   0   0   0   0   0   0   0   0   1  …   0   0   0   0
640.05            0   0   0   0   0   1   0   0   0   0  …   0   0   0   0
```

```
…               ..  ..  ..  ..  ..  ..  ..  ..  ..  ..  …  ..  ..  ..  ..
11886.60         0   0   0   0   0   0   0   0   0   0  …   0   0   0   0
11887.80         0   0   0   0   0   0   0   0   0   0  …   0   0   0   0
12001.00         0   0   0   0   0   0   0   0   0   0  …   0   0   0   0
12536.50         0   0   0   0   0   0   0   0   0   0  …   0   0   0   0
14082.80         0   0   0   0   0   0   0   0   0   0  …   0   0   0   0

QUANTITYORDERED  66  70  76  77  85  97
SALES
482.13           0   0   0   0   0   0
541.14           0   0   0   0   0   0
553.95           0   0   0   0   0   0
577.60           0   0   0   0   0   0
640.05           0   0   0   0   0   0
…               ..  ..  ..  ..  ..  ..
11886.60         1   0   0   0   0   0
11887.80         0   0   0   0   0   0
12001.00         0   0   0   0   0   0
12536.50         0   0   0   0   0   0
14082.80         0   0   1   0   0   0

[2763 rows x 58 columns]
QUANTITYORDERED  6   10  11  12  13  15  16  18  19  20  …  61  62  64  65  \
MSRP                                                        …
33               0   0   0   0   0   0   0   0   0   5  …   1   0   0   0
35               0   0   0   0   0   0   0   0   0   1  …   0   0   0   0
37               0   0   0   0   0   1   0   0   0   2  …   0   0   0   0
40               0   0   0   0   0   0   0   0   0   1  …   0   0   0   0
41               0   0   0   0   0   2   0   0   0   0  …   0   0   0   0
…               ..  ..  ..  ..  ..  ..  ..  ..  ..  ..  …  ..  ..  ..  ..
173              0   0   0   0   0   0   0   0   0   2  …   0   0   0   0
193              0   0   0   0   0   0   0   0   0   2  …   0   0   0   0
194              0   0   0   1   0   0   0   0   0   1  …   0   0   0   0
207              0   0   0   0   0   0   0   0   0   1  …   0   0   0   0
214              0   0   0   0   0   0   0   0   0   0  …   0   0   0   0

QUANTITYORDERED  66  70  76  77  85  97
MSRP
33               0   0   0   0   0   0
35               0   0   0   0   0   0
37               0   0   0   0   0   0
40               0   0   0   0   0   0
41               0   0   0   0   0   0
…               ..  ..  ..  ..  ..  ..
173              0   0   0   0   0   0
193              1   0   0   0   0   0
194              0   0   0   0   0   0
207              0   0   0   0   0   0
```

```
214                0    0    0    0    0    0

[80 rows x 58 columns]
```

```
[16]: #Mosaic Graph 1
mosaic(data, ['DEALSIZE', 'TERRITORY'], title=' Deal size x Territory ')
plt.show()
```


Deal size x Territory
Japan
APAC
EMEA
Small Japan
Small APAC
Small EMEA
Medium Japan
Medium APAC
Medium EMEA
Large Japan
Large APAC
Large EMEA
Small
Medium
Large