wholesale customers project

Introduction:

The goal of the wholesale customer data analysis to find the relation of milk and grocery. In this data we have different columns such as channel,region,fresh,milk,grocery,frozen,detergents_papaer,delicassen.In this we make a model which find the relation between milk and grocery of a customer which predict delicatessen based on the data.

```
: Project planning –
•   Read client data and check records.
•   Check null values if exist and remove/replace null values if required.
•   Rename data frame column if required.
•   Scale Raw data as per model requirement.
•   Perform descriptive statistics and calculate mean, median etc.
•   Create box plot for numerical column.
•   Group data and create box plot for grouped data if required.
•   Check correlation between variable and draw correlation matrix.
•   Draw histogram of data and check density (KDE) is required.
•   Check type of data for regression or classification.
```

read client data and check records

In [ ]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [158]:

```
data=pd.read_csv("Wholesale customers data.csv")
data.head(5)
```

Out[158]:

	Channel	Region	Fresh	Milk	Grocery	Frozen	Detergents Paper	Delicatessen
0	2	3	12669	9656	7561	214	2674	1338
1	2	3	7057	9810	9568	1762	3293	1776
2	2	3	6353	8808	7684	2405	3516	7844
3	1	3	13265	1196	4221	6404	507	1788
4	2	3	22615	5410	7198	3915	1777	5185

In [4]:

```
data.shape
```

Out[4]:

```
(440, 8)
```

check null values if exist and remove/replace null values if required.

In [5]:

```
data.isnull().sum()  #null values
```

Out[5]:

```
Channel             0
Region              0
Fresh               0
Milk                0
Grocery             0
Frozen              0
Detergents Paper    0
Delicatessen        0
dtype: int64
```

In [28]:

```
data.head(7)
```

Out[28]:

	Channel	Region	Fresh	Milk	Grocery	Frozen	Detergents Paper	Delicatessen
0	2	3	12669	9656	7561	214	2674	1338
1	2	3	7057	9810	9568	1762	3293	1776
2	2	3	6353	8808	7684	2405	3516	7844
3	1	3	13265	1196	4221	6404	507	1788
4	2	3	22615	5410	7198	3915	1777	5185
5	2	3	9413	8259	5126	666	1795	1451
6	2	3	12126	3199	6975	480	3140	545

Rename Data Frame column names if required.

In [146]:

```
data.rename(columns = {'Detergents_Paper':'d_p'},inplace = True)  #rename column
data.head(5)
```

Out[146]:

	Channel	Region	Fresh	Milk	Grocery	Frozen	d_p	Delicatessen
0	2	3	12669	9656	7561	214	2674	1338
1	2	3	7057	9810	9568	1762	3293	1776
2	2	3	6353	8808	7684	2405	3516	7844
3	1	3	13265	1196	4221	6404	507	1788
4	2	3	22615	5410	7198	3915	1777	5185

In [171]:

```
from sklearn.preprocessing import StandardScaler #using StandardScaler
scaler = StandardScaler()
scaled = scaler.fit_transform(data)
df=pd.DataFrame(scaled)
df.head(4)
```

Out[171]:

	0	1	2	3	4	5	6	7
0	1.448652	0.590668	0.052933	0.523568	-0.041115	-0.589367	-0.043569	-0.066339
1	1.448652	0.590668	-0.391302	0.544458	0.170318	-0.270136	0.086407	0.089151
2	1.448652	0.590668	-0.447029	0.408538	-0.028157	-0.137536	0.133232	2.243293
3	-0.690297	0.590668	0.100111	-0.624020	-0.392977	0.687144	-0.498588	0.093411

Scale Raw data as per Model requirement

In [180]:

```
from sklearn.preprocessing import normalize
data_scaled = normalize(data)
data_scaled = pd.DataFrame(data_scaled, columns=data.columns)
data_scaled.head()
```

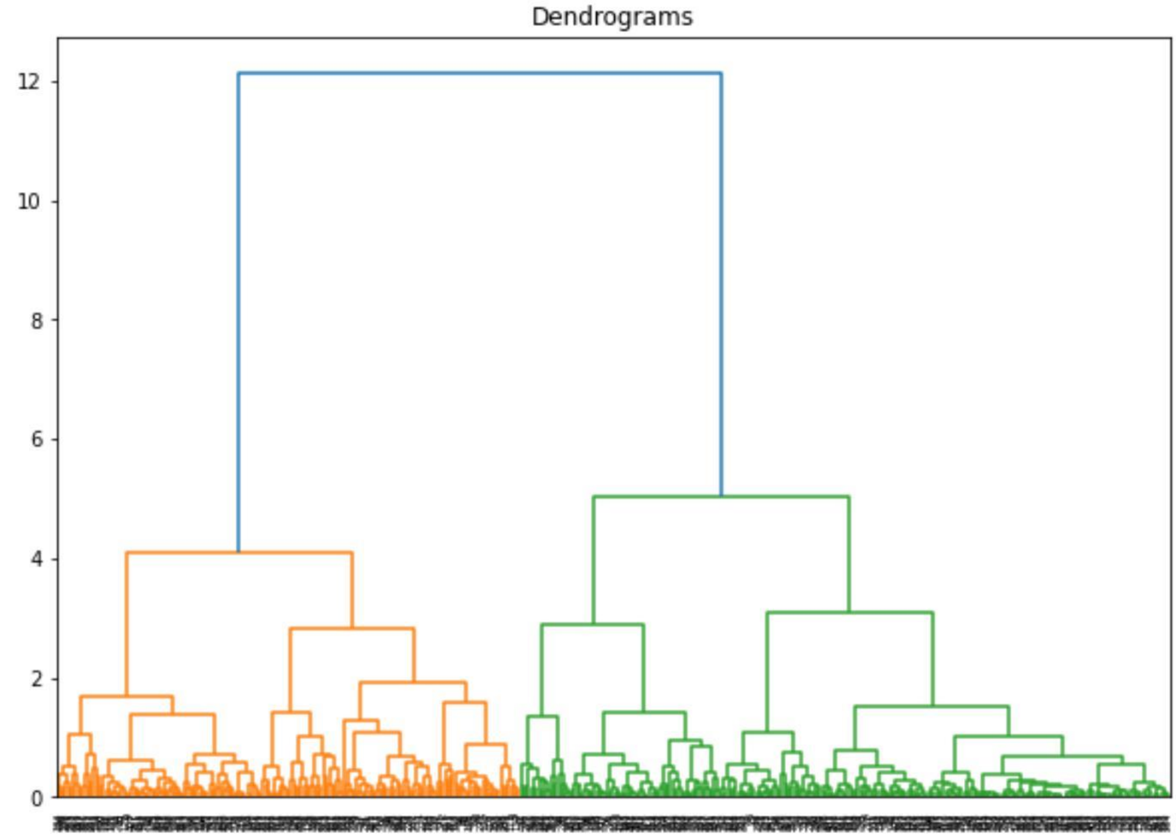
Out[180]:

	Channel	Region	Fresh	Milk	Grocery	Frozen	Detergents Paper	Delicatessen
0	0.000112	0.000168	0.708333	0.539874	0.422741	0.011965	0.149505	0.074809
1	0.000125	0.000188	0.442198	0.614704	0.599540	0.110409	0.206342	0.111286
2	0.000125	0.000187	0.396552	0.549792	0.479632	0.150119	0.219467	0.489619
3	0.000065	0.000194	0.856837	0.077254	0.272650	0.413659	0.032749	0.115494
4	0.000079	0.000119	0.895416	0.214203	0.284997	0.155010	0.070358	0.205294

using dendrograms

In [192]:

```
import scipy.cluster.hierarchy as shc
plt.figure(figsize=(10, 7))
plt.title("Dendrograms")
dend = shc.dendrogram(shc.linkage(data_scaled, method='ward'))
```



```
#Perform descriptive statistic and calculate mean, median
```

In [189]:

```
data.                    #using descriptive statistics
Descriptive()
```

Out[189]:

	Channel	Region	Fresh	Milk	Grocery	Frozen	Deternt
count	440.000000	440.000000	440.000000	440.000000	440.000000	440.000000	
mean	1.322727	2.543182	12000.297727	5796.265909	7951.277273	3071.931818	
std	0.468052	0.774272	12647.328865	7380.377175	9503.162829	4854.673333	
min	1.000000	1.000000	3.000000	55.000000	3.000000	25.000000	
25%	1.000000	2.000000	3127.750000	1533.000000	2153.000000	742.250000	
50%	1.000000	3.000000	8504.000000	3627.000000	4755.500000	1526.000000	
75%	2.000000	3.000000	16933.750000	7190.250000	10655.750000	3554.250000	
max	2.000000	3.000000	112151.000000	73498.000000	92780.000000	60869.000000	4

In [190]:

```
data.mean()    #calculate mean
```

Out[190]:

```
Channel                 1.322727
Region                  2.543182
Fresh               12000.297727
Milk                 5796.265909
Grocery              7951.277273
Frozen               3071.931818
Detergents_Paper     2881.493182
Delicassen           1524.870455
dtype: float64
```

In [191]:

```
data.median()  #calculate median
```

Out[191]:

```
Channel                1.0
Region                 3.0
Fresh               8504.0
Milk                3627.0
Grocery             4755.5
Frozen              1526.0
Detergents_Paper     816.5
Delicassen           965.5
dtype: float64
```

In [55]:

```
data.head(5)
```

Out[55]:

	Channel	Region	Fresh	Milk	Grocery	Frozen	d_p	Delicatessen
0	2	3	12669	9656	7561	214	2674	1338
1	2	3	7057	9810	9568	1762	3293	1776
2	2	3	6353	8808	7684	2405	3516	7844
3	1	3	13265	1196	4221	6404	507	1788
4	2	3	22615	5410	7198	3915	1777	5185

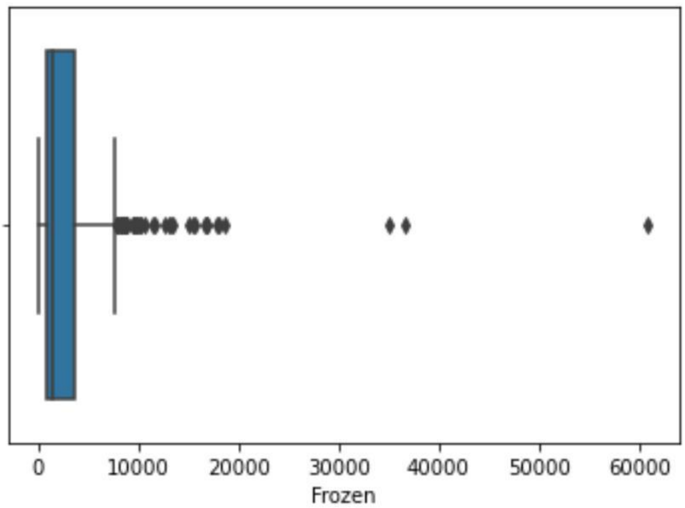
Create BoxPlot for numerical columns.

In [125]:

```
sns.boxplot(x=data['Frozen'])   #create boxplot for numerical columns
```

Out[125]:

```
<AxesSubplot:xlabel='Frozen'>
```



Check Correlation b/w variables and draw correlation matrix

In [82]:

```
data.corr()     #correlation data
```

Out[82]:

	Channel	Region	Fresh	Milk	Grocery	Frozen	d_p	Delicasse
Channel	1.000000	0.062028	-0.169172	0.460720	0.608792	-0.202046	0.636026	0.0560
Region	0.062028	1.000000	0.055287	0.032288	0.007696	-0.021044	-0.001483	0.04521
Fresh	-0.169172	0.055287	1.000000	0.100510	-0.011854	0.345881	-0.101953	0.24469
Milk	0.460720	0.032288	0.100510	1.000000	0.728335	0.123994	0.661816	0.40636
Grocery	0.608792	0.007696	-0.011854	0.728335	1.000000	-0.040193	0.924641	0.20549
Frozen	-0.202046	-0.021044	0.345881	0.123994	-0.040193	1.000000	-0.131525	0.39094
d_p	0.636026	-0.001483	-0.101953	0.661816	0.924641	-0.131525	1.000000	0.06929
Delicatessen	0.056011	0.045212	0.244690	0.406368	0.205497	0.390947	0.069291	1.00000

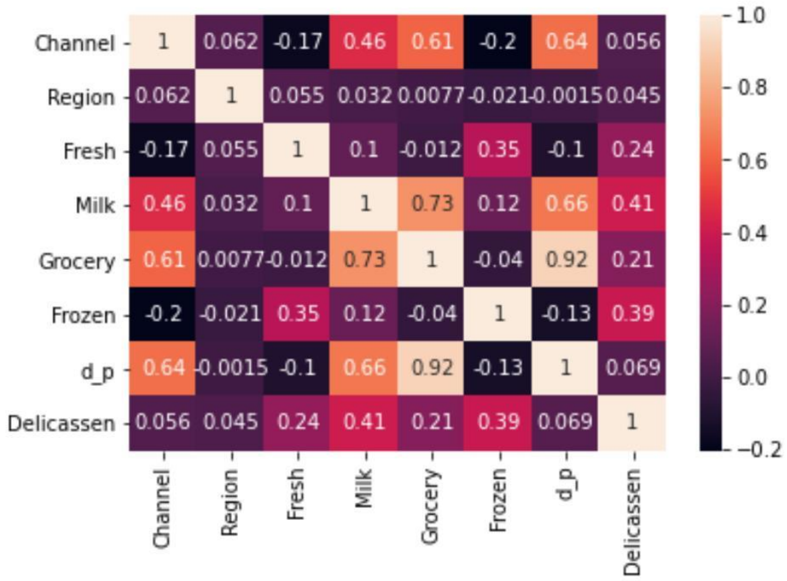
Draw histogram for data and check density(KDE) if required.

In [69]:

```
sns.heatmap(data.corr(),annot=True)     #heapmap
```

Out[69]:

```
<AxesSubplot:>
```



Create model as per requirement and perform classification/regression/clustering

In [*]:

```
from sklearn.cluster import KMeans
SSE = []
for cluster in range(1,20):
    kmeans = KMeans(n_clusters = cluster, init='k-means++')
    kmeans.fit()
    SSE.append(kmeans.inertia_)

frame = pd.DataFrame({'Cluster':range(1,20), 'SSE':SSE})
plt.figure(figsize=(12,6))
plt.plot(frame['Cluster'], frame['SSE'], marker='o')
plt.xlabel('Number of clusters')
plt.ylabel('Inertia')
```

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]: