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

```
df=pd.read_csv("/content/Wholesale customers data.csv")
df.head()
```

	Channel	Region	Fresh	Milk	Grocery	Frozen	Detergents_Paper	Delicassen
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

```
df.shape
```

```
(440, 8)
```

```
df.describe()
```

	Channel	Region	Fresh	Milk	Grocery	Frozen	Detergents_Paper	Delicassen
count	440.000000	440.000000	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	2881.493182	1524.870455
std	0.468052	0.774272	12647.328865	7380.377175	9503.162829	4854.673333	4767.854448	2820.105937
min	1.000000	1.000000	3.000000	55.000000	3.000000	25.000000	3.000000	3.000000
25%	1.000000	2.000000	3127.750000	1533.000000	2153.000000	742.250000	256.750000	408.250000
50%	1.000000	3.000000	8504.000000	3627.000000	4755.500000	1526.000000	816.500000	965.500000
75%	2.000000	3.000000	16933.750000	7190.250000	10655.750000	3554.250000	3922.000000	1820.250000
max	2.000000	3.000000	112151.000000	73498.000000	92780.000000	60869.000000	40827.000000	47943.000000

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 440 entries, 0 to 439
Data columns (total 8 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   Channel           440 non-null    int64
 1   Region            440 non-null    int64
 2   Fresh             440 non-null    int64
 3   Milk              440 non-null    int64
 4   Grocery           440 non-null    int64
 5   Frozen            440 non-null    int64
 6   Detergents_Paper  440 non-null    int64
 7   Delicassen        440 non-null    int64
dtypes: int64(8)
memory usage: 27.6 KB
```

```
df.isnull().sum()
```

```
Channel             0
Region              0
Fresh               0
Milk                0
Grocery             0
Frozen              0
Detergents_Paper    0
Delicassen          0
dtype: int64
```

```
df.corr()
```

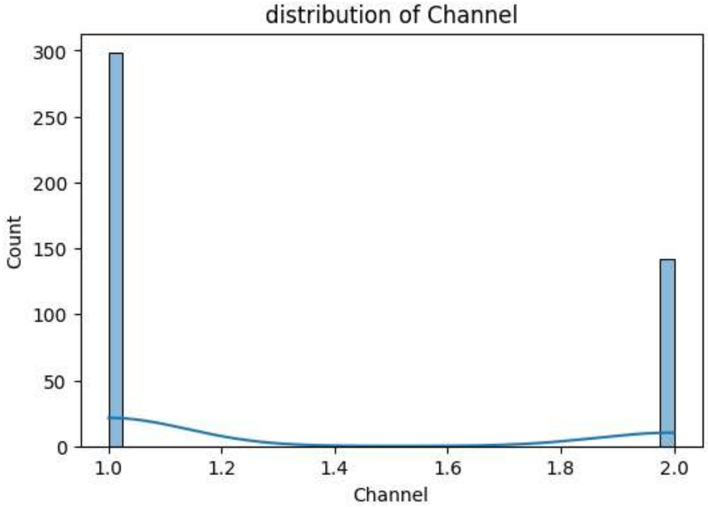
	Channel	Region	Fresh	Milk	Grocery	Frozen	Detergents_Paper	Delicassen
Channel	1.000000	0.062028	-0.169172	0.460720	0.608792	-0.202046	0.636026	0.056011
Region	0.062028	1.000000	0.055287	0.032288	0.007696	-0.021044	-0.001483	0.045212
Fresh	-0.169172	0.055287	1.000000	0.100510	-0.011854	0.345881	-0.101953	0.244690
Milk	0.460720	0.032288	0.100510	1.000000	0.728335	0.123994	0.661816	0.406368
Grocery	0.608792	0.007696	-0.011854	0.728335	1.000000	-0.040193	0.924641	0.205497
Frozen	-0.202046	-0.021044	0.345881	0.123994	-0.040193	1.000000	-0.131525	0.390947
Detergents_Paper	0.636026	-0.001483	-0.101953	0.661816	0.924641	-0.131525	1.000000	0.069291
Delicassen	0.056011	0.045212	0.244690	0.406368	0.205497	0.390947	0.069291	1.000000

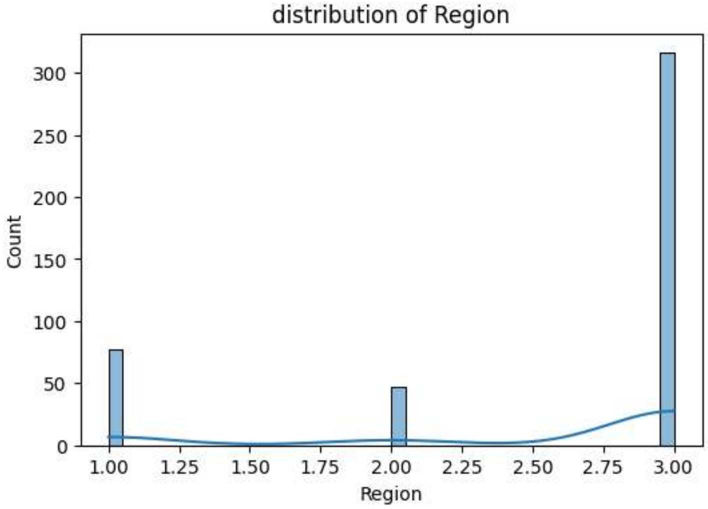
```
df.columns
```

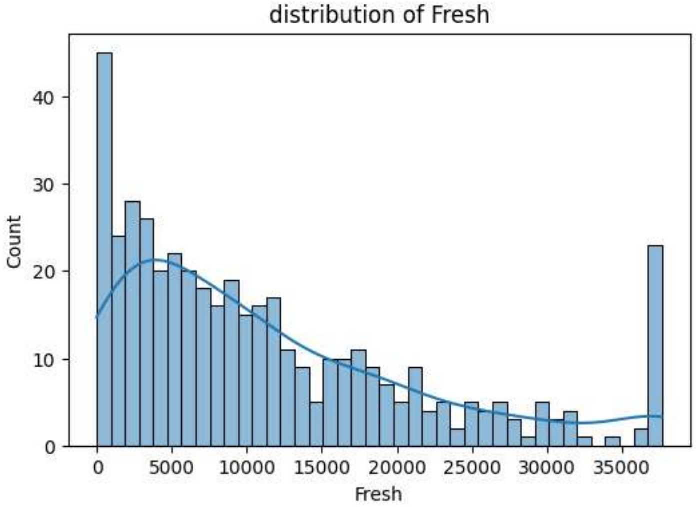
```
Index(['Channel', 'Region', 'Fresh', 'Milk', 'Grocery', 'Frozen',
       'Detergents_Paper', 'Delicassen'],
      dtype='object')
```

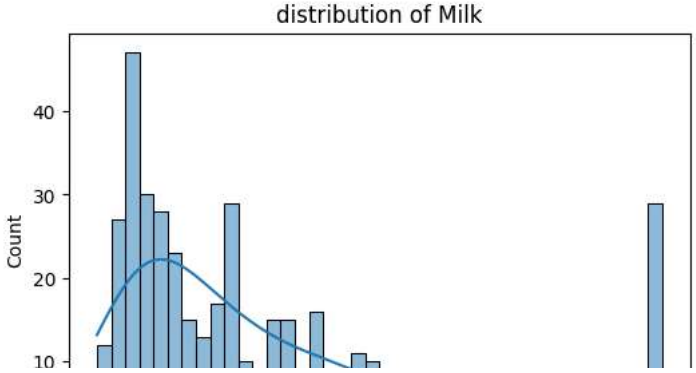
Data distribution

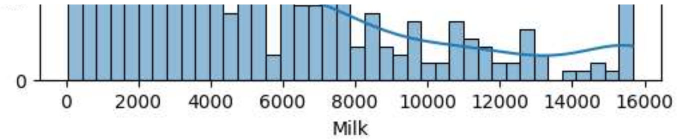
```
for column in df.columns:
  plt.figure(figsize=(6,4))
  sns.histplot(df[column],bins=40, kde=True)
  plt.title(f'distribution of {column}')
  plt.show()
```

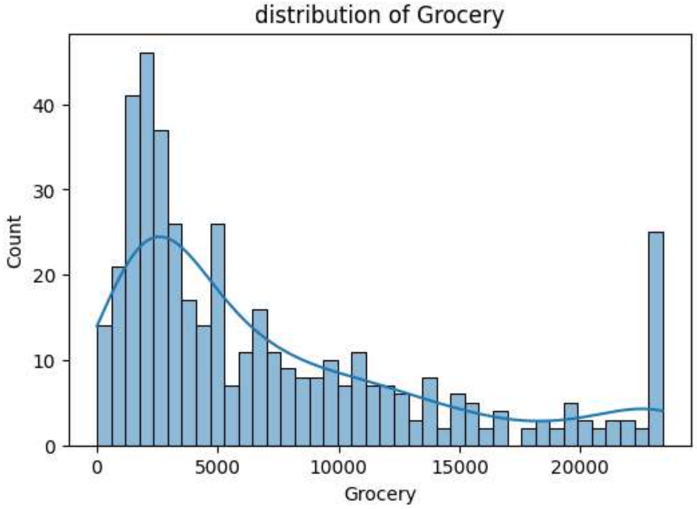

distribution of Channel
Count
300
250
200
150
100
50
0
1.0
1.2
1.4
1.6
1.8
2.0
Channel

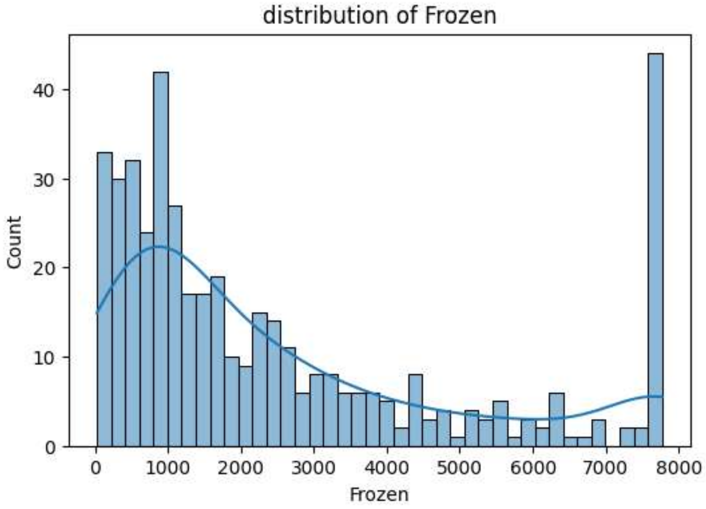

distribution of Region
Count
300
250
200
150
100
50
0
1.00
1.25
1.50
1.75
2.00
2.25
2.50
2.75
3.00
Region

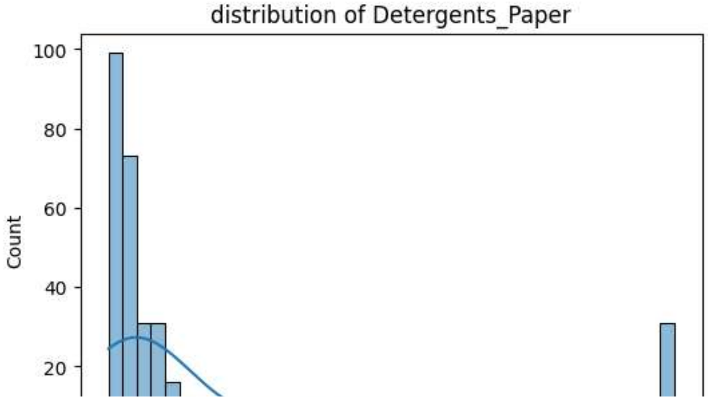

distribution of Fresh
Count
40
30
20
10
0
0
5000
10000
15000
20000
25000
30000
35000
Fresh


distribution of Milk
Count
40
30
20
10



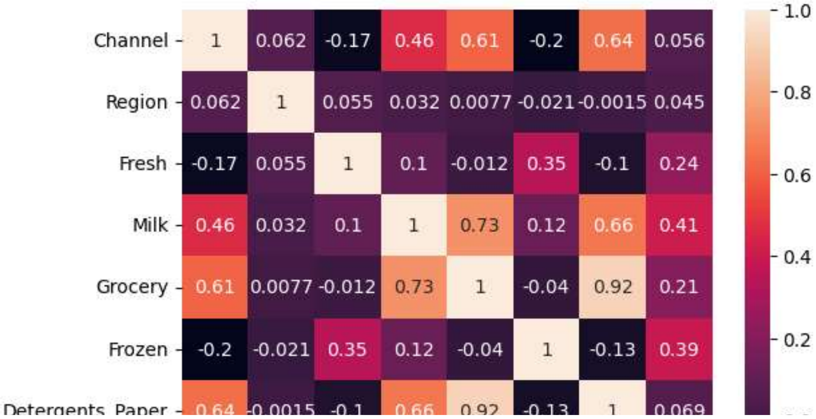

distribution of Grocery


distribution of Frozen

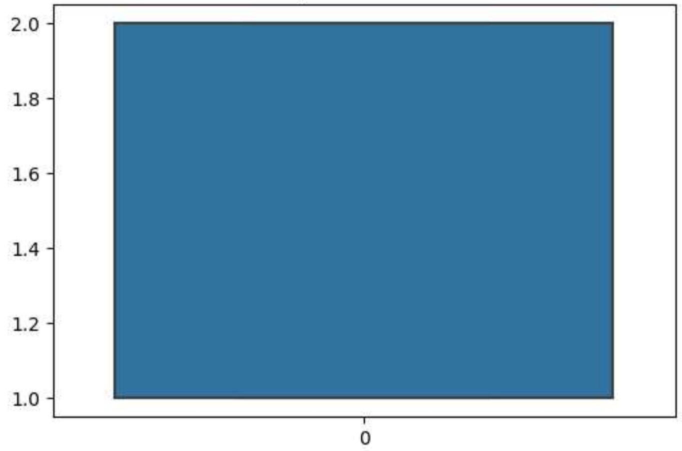

distribution of Detergents_Paper

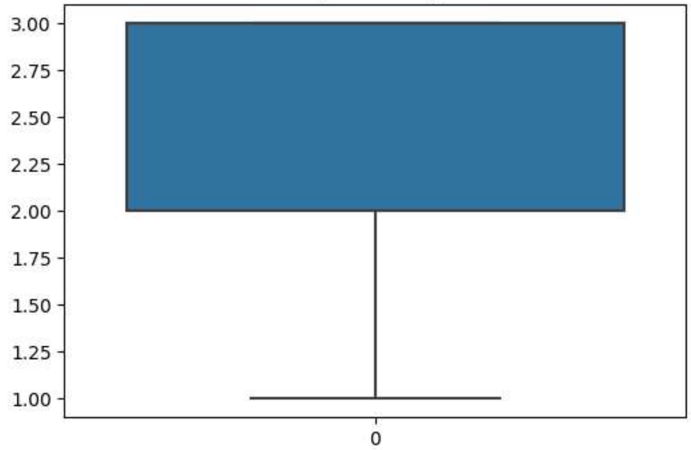
```
sns.heatmap(df.corr(), annot=True)
```

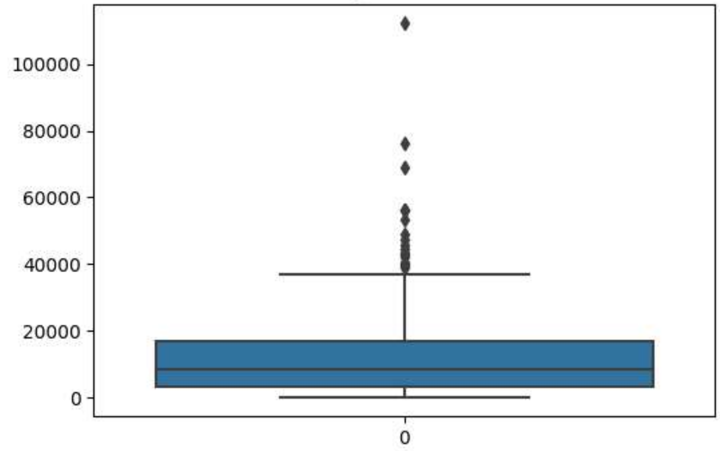
```
<Axes: >
```

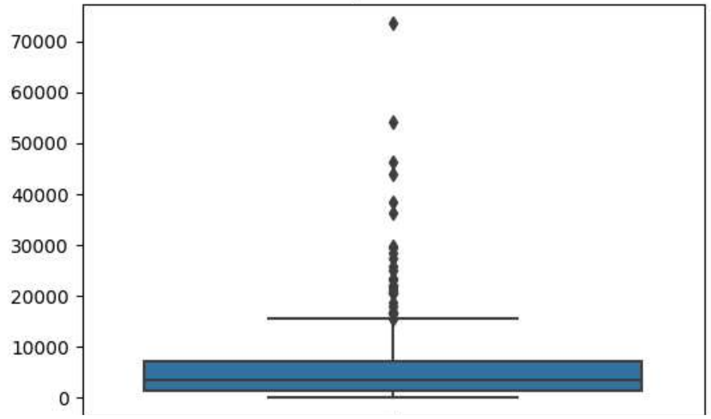


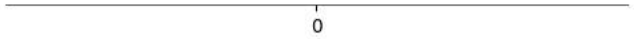
```
for column in df.columns:
  plt.figure(figsize=(6,4))
  sns.boxplot(df[column])
  plt.title(f'boxplot of {column}')
  plt.show()
```

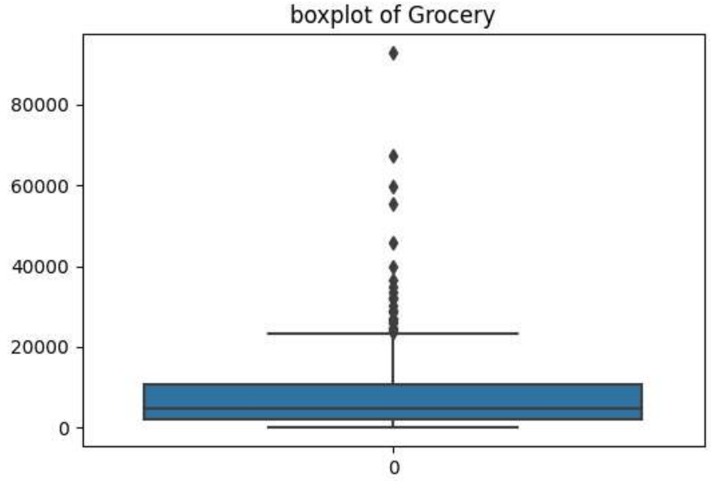

boxplot of Channel

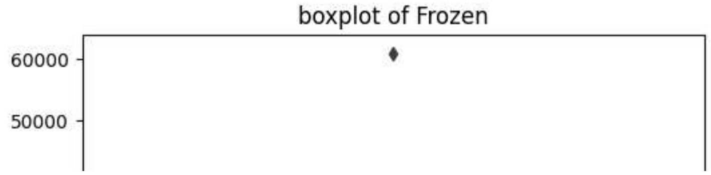

boxplot of Region


boxplot of Fresh


boxplot of Milk







```
def handle_outliers(dataframe,column):
  Q1=dataframe[column].quantile(0.25)
  Q3=dataframe[column].quantile(0.75)
  IQR=Q3-Q1
  lower_limit=Q1- 1.5*IQR
  upper_limit=Q3+1.5*IQR
  dataframe[column]=dataframe[column].apply(lambda x:upper_limit if x > upper_limit else lower_limit if x < lower_limit else x)

for column in df.columns:
  handle_outliers(df,column)
```

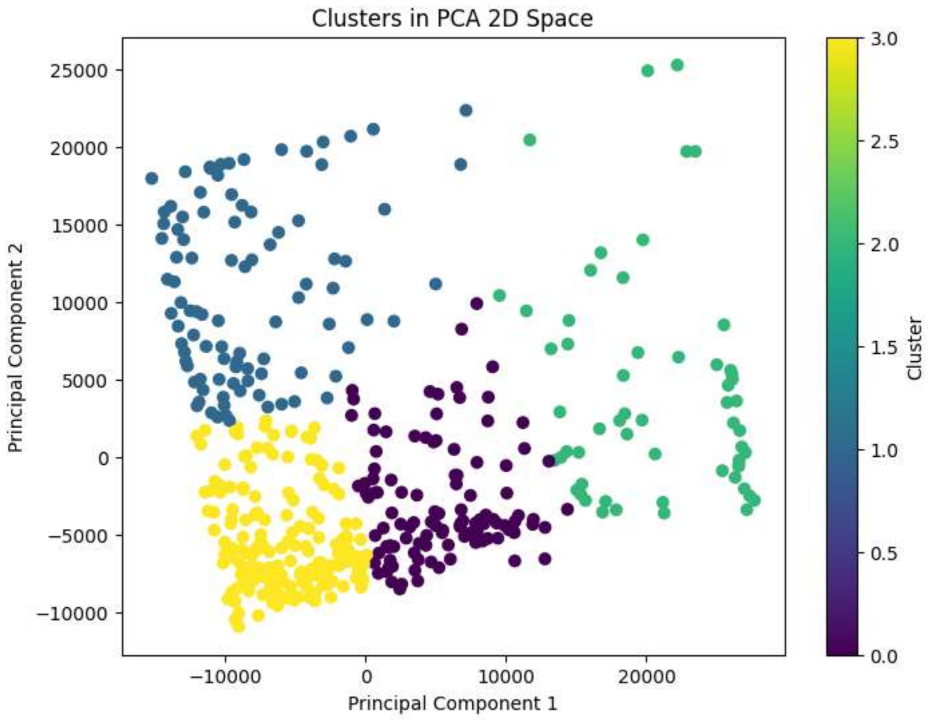
```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
df_scaled = pd.DataFrame(scaler.fit_transform(df), columns=df.columns)
from sklearn.cluster import KMeans
import matplotlib.pyplot as plt
wcss = []
max_clusters = 15
for i in range(1, max_clusters+1):
    kmeans = KMeans(n_clusters=i, init='k-means++', random_state=42)
    kmeans.fit(df)
    wcss.append(kmeans.inertia_)
plt.plot(range(1, max_clusters+1), wcss)
plt.title('The Elbow Method')
plt.xlabel('Number of clusters')
plt.ylabel('WCSS')
plt.grid(True)
plt.show()
```

```
from sklearn.cluster import KMeans
kmeans = KMeans(n_clusters=4, init='k-means++', random_state=42)
kmeans.fit(df)
cluster_labels = kmeans.labels_
df['Cluster'] = cluster_labels
print(df['Cluster'].unique())
```

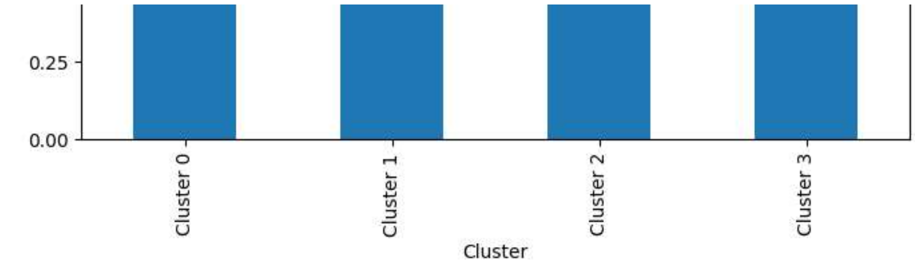
```
[0 1 3 2]
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarning: The default value of `n_init` will change from 1
  warnings.warn(
```

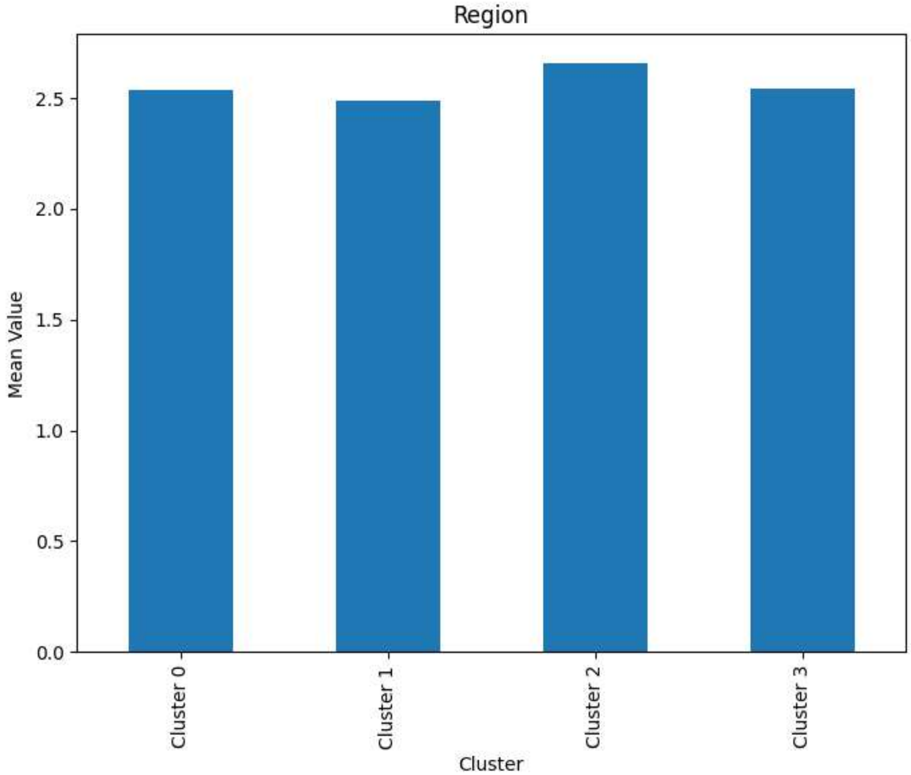
```
from sklearn.decomposition import PCA
import matplotlib.pyplot as plt
pca = PCA(n_components=2)
principalComponents = pca.fit_transform(df.drop('Cluster', axis=1))
PCA_components = pd.DataFrame(principalComponents, columns=['Principal Component 1', 'Principal Component 2'])
```

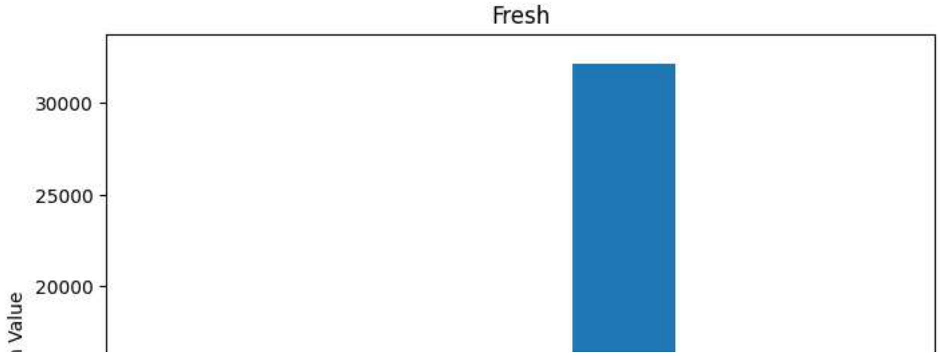
```
PCA_components['Cluster'] = df['Cluster']
plt.figure(figsize=(8,6))
plt.scatter(PCA_components['Principal Component 1'],
PCA_components['Principal Component 2'], c=PCA_components['Cluster'])
plt.title('Clusters in PCA 2D Space')
plt.xlabel('Principal Component 1')
plt.ylabel('Principal Component 2')
plt.colorbar(label='Cluster')
plt.show()
```

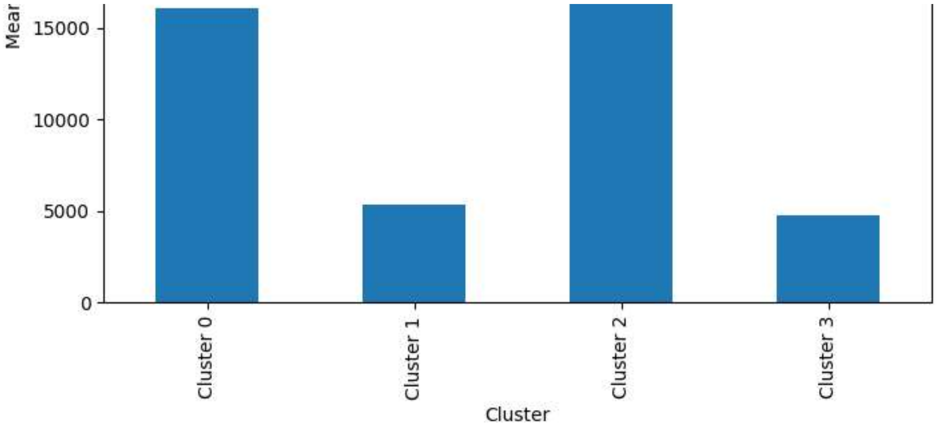


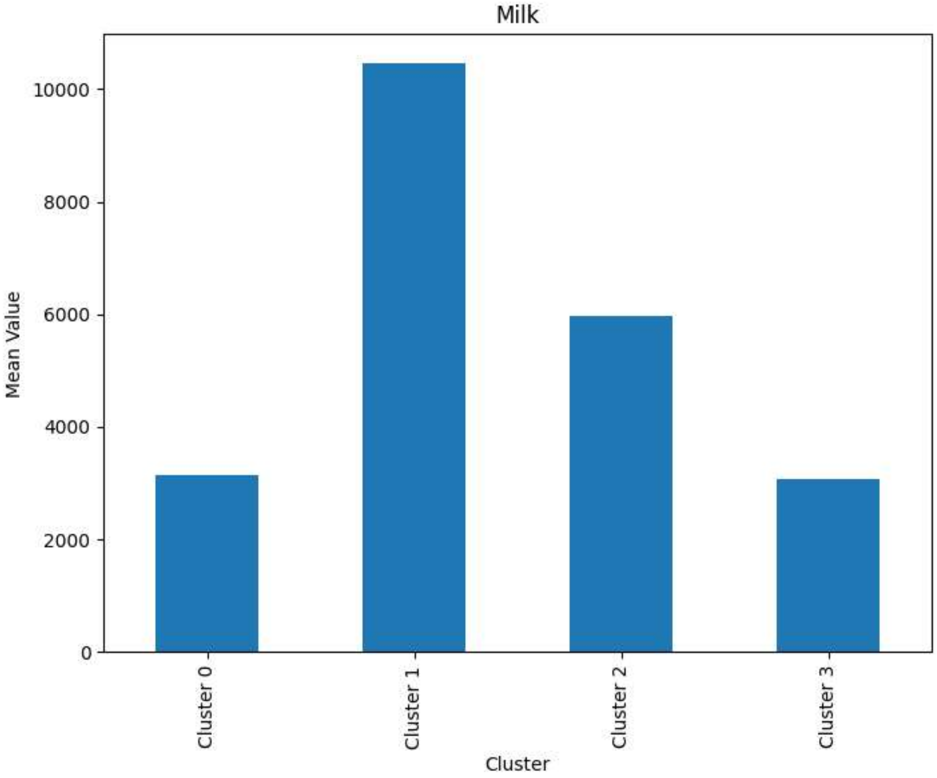
```
cluster_means = df.groupby('Cluster').mean()
cluster_means = cluster_means.transpose()
for feature in cluster_means.index:
    cluster_means.loc[feature].plot(kind='bar', figsize=(8,6))
    plt.title(feature)
    plt.ylabel('Mean Value')
    plt.xticks(ticks=range(4), labels=['Cluster 0', 'Cluster 1', 'Cluster 2', 'Cluster 3'])
    plt.show()
```

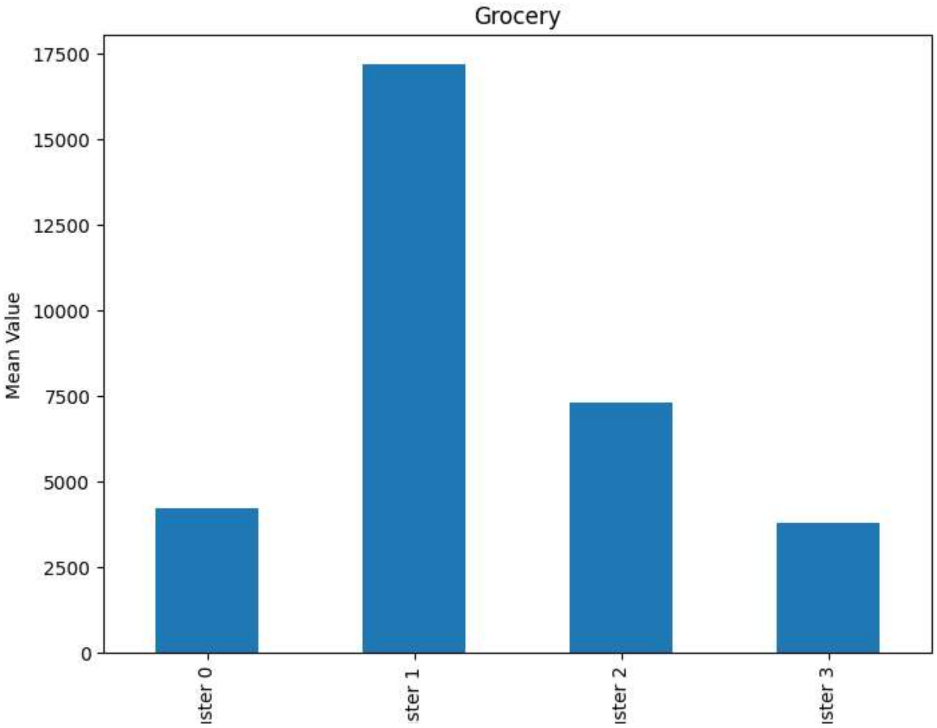

0.25
0.00
Cluster 0
Cluster 1
Cluster 2
Cluster 3
Cluster


Region
2.5
2.0
1.5
1.0
0.5
0.0
Mean Value
Cluster 0
Cluster 1
Cluster 2
Cluster 3
Cluster


Fresh
30000
25000
20000
Value


Mear
15000
10000
5000
0
Cluster 0
Cluster 1
Cluster 2
Cluster 3
Cluster


Milk
Mean Value
10000
8000
6000
4000
2000
0
Cluster 0
Cluster 1
Cluster 2
Cluster 3
Cluster


Grocery
Mean Value
17500
15000
12500
10000
7500
5000
2500
0
ıster 0
ıster 1
ıster 2
ıster 3