20bsit154-ass-4

March 30, 2023

[1]:
```
# Assignment - IV
# ID : 20BSIT154
```

[3]:
```
#1.Read the Wholesale_customers_data.csv file.
import pandas as pd

df = pd.read_csv("D:/sem6/dma/practical/Wholesale_customers_data.csv")
print(df)
```

```
     Channel  Region  Fresh   Milk  Grocery  Frozen  Detergents_Paper  \
0          2       3  12669   9656     7561     214              2674
1          2       3   7057   9810     9568    1762              3293
2          2       3   6353   8808     7684    2405              3516
3          1       3  13265   1196     4221    6404               507
4          2       3  22615   5410     7198    3915              1777
..       ...     ...    ...    ...      ...     ...               ...
435        1       3  29703  12051    16027   13135               182
436        1       3  39228   1431      764    4510                93
437        2       3  14531  15488    30243     437             14841
438        1       3  10290   1981     2232    1038               168
439        1       3   2787   1698     2510      65               477

     Delicassen
0          1338
1          1776
2          7844
3          1788
4          5185
..          ...
435        2204
436        2346
437        1867
438        2125
439          52

[440 rows x 8 columns]
```

[4]:
```
# Q-2.Perform all data pre-processing techniques.
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
```

[5]:
```
df.isnull().sum()
```

[5]:
```
Channel             0
Region              0
Fresh               0
Milk                0
Grocery             0
Frozen              0
Detergents_Paper    0
Delicassen          0
dtype: int64
```

[6]:
```
#Q-3 Perform one-hot encoding and normalization.
df2 = pd.get_dummies(df, columns = ['Channel','Region'])
df2
```

[6]:

	Fresh	Milk	Grocery	Frozen	Detergents_Paper	Delicassen	Channel_1	\
0	12669	9656	7561	214	2674	1338	0	
1	7057	9810	9568	1762	3293	1776	0	
2	6353	8808	7684	2405	3516	7844	0	
3	13265	1196	4221	6404	507	1788	1	
4	22615	5410	7198	3915	1777	5185	0	
..	…	…	…	…	…	…	…	
435	29703	12051	16027	13135	182	2204	1	
436	39228	1431	764	4510	93	2346	1	
437	14531	15488	30243	437	14841	1867	0	
438	10290	1981	2232	1038	168	2125	1	
439	2787	1698	2510	65	477	52	1	

```
     Channel_2  Region_1  Region_2  Region_3
0            1         0         0         1
1            1         0         0         1
2            1         0         0         1
3            0         0         0         1
4            1         0         0         1
..         …       …       …       …
435          0         0         0         1
436          0         0         0         1
437          1         0         0         1
438          0         0         0         1
439          0         0         0         1

[440 rows x 11 columns]
```

[7]:
```
from sklearn import preprocessing
df3 = preprocessing.normalize(df2)
df3
```

[7]:
```
array([[7.08332707e-01, 5.39873757e-01, 4.22740832e-01, …,
        0.00000000e+00, 0.00000000e+00, 5.59107039e-05],
       [4.42198263e-01, 6.14703834e-01, 5.99539886e-01, …,
        0.00000000e+00, 0.00000000e+00, 6.26609413e-05],
       [3.96551689e-01, 5.49791796e-01, 4.79632171e-01, …,
        0.00000000e+00, 0.00000000e+00, 6.24195954e-05],
       …,
       [3.64461534e-01, 3.88464679e-01, 7.58544504e-01, …,
        0.00000000e+00, 0.00000000e+00, 2.50816554e-05],
       [9.37737421e-01, 1.80530401e-01, 2.03404269e-01, …,
        0.00000000e+00, 0.00000000e+00, 9.11309447e-05],
       [6.72295989e-01, 4.09601216e-01, 6.05476473e-01, …,
        0.00000000e+00, 0.00000000e+00, 2.41225687e-04]])
```

[9]:
```
#Q-4.Apply k-means clustering algorithm.
from sklearn.cluster import KMeans
kmeans = KMeans(3)
kmeans.fit(df3)
```

[9]:
```
KMeans(n_clusters=3)
```

[10]:
```
pred = kmeans.predict(df3)
pred
```

[10]:
```
array([0, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0, 0, 1, 1, 0, 1, 0, 2,
       0, 1, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 1, 1, 0, 1, 1, 0, 0, 0, 1, 1,
       1, 1, 1, 1, 1, 1, 2, 1, 0, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 2, 1,
```

```
1, 0, 2, 0, 2, 1, 2, 0, 1, 0, 2, 1, 0, 1, 0, 1, 1, 0, 0, 1, 1, 0,
2, 0, 0, 2, 1, 2, 1, 1, 1, 2, 2, 2, 1, 1, 1, 0, 0, 0, 1, 1, 1, 1,
0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2, 0, 1, 0, 0, 0, 0, 1, 0, 2, 0,
0, 0, 0, 0, 1, 1, 0, 1, 0, 0, 0, 2, 0, 1, 0, 0, 2, 0, 0, 1, 0, 1,
0, 1, 1, 0, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1,
0, 0, 0, 2, 0, 0, 1, 2, 1, 0, 2, 1, 1, 1, 0, 0, 0, 1, 0, 0, 2, 1,
0, 0, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 1, 0, 1, 1, 1, 1, 0, 1, 0,
0, 1, 2, 2, 0, 0, 0, 2, 1, 2, 2, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0,
0, 0, 1, 1, 0, 0, 0, 2, 2, 1, 2, 0, 1, 0, 0, 2, 0, 0, 0, 2, 0, 1,
1, 1, 1, 0, 1, 0, 2, 1, 1, 0, 1, 2, 0, 2, 0, 0, 1, 0, 0, 0, 0, 0,
0, 0, 0, 0, 1, 0, 2, 1, 0, 0, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 0,
0, 1, 2, 0, 1, 0, 0, 1, 0, 1, 0, 1, 1, 0, 0, 0, 0, 2, 2, 1, 0, 2,
0, 1, 0, 1, 2, 0, 0, 2, 2, 2, 1, 1, 1, 1, 2, 1, 1, 0, 0, 1, 2, 1,
1, 1, 0, 1, 0, 1, 1, 1, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 1, 0, 1, 1, 0, 0, 0, 2, 1, 0, 1, 0, 0, 0, 2, 1, 1, 0, 0, 0,
1, 2, 0, 0, 2, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 1, 2, 2, 1, 1, 1,
1, 0, 1, 0, 0, 0, 0, 2, 1, 0, 1, 0, 1, 2, 0, 1, 0, 0, 0, 1, 0, 1])
```

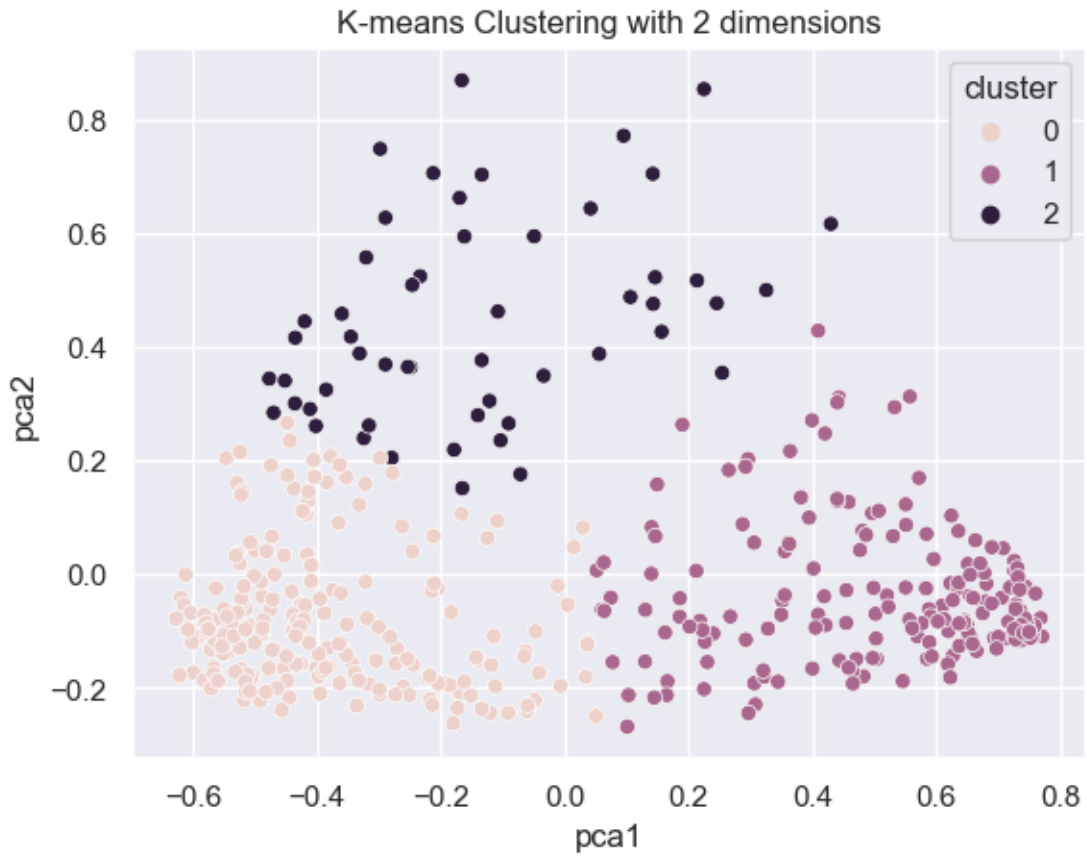
[11]:
```
frame = pd.DataFrame(df3)
frame['cluster'] = pred
frame['cluster'].value_counts()
```

[11]:
```
0    210
1    177
2     53
Name: cluster, dtype: int64
```

[16]:
```
from sklearn.preprocessing import OneHotEncoder
from sklearn.decomposition import PCA
import seaborn as sns
import statsmodels.api as sm
import random as rd
import matplotlib.pyplot as plt
sns.set()
OneHotEncoder().fit_transform(df3)

reduced_data = PCA(n_components=2).fit_transform(df3)
results = pd.DataFrame(reduced_data,columns=['pca1','pca2'])

sns.scatterplot(x="pca1", y="pca2", hue=frame['cluster'], data=results)
plt.title('K-means Clustering with 2 dimensions')
plt.show()
```



[19]:
```
df_scaled = pd.DataFrame(df3, columns=df2.columns)
df_scaled.head()
```

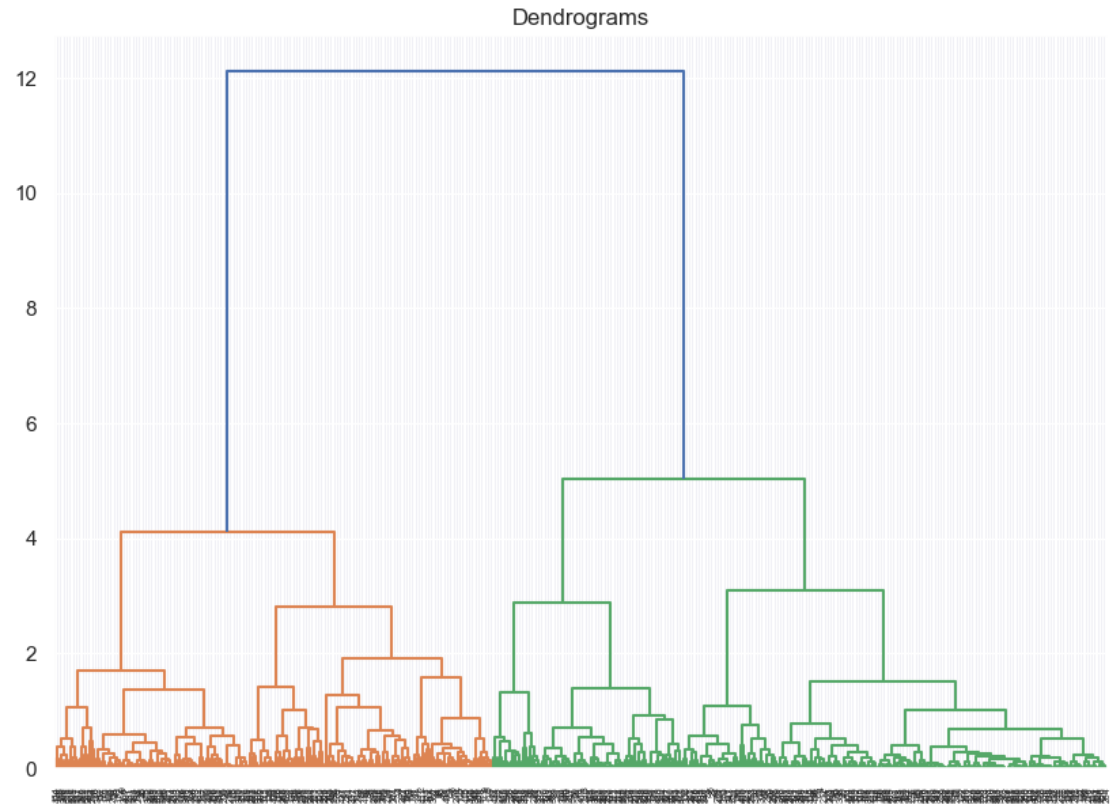
[19]:

	Fresh	Milk	Grocery	Frozen	Detergents_Paper	Delicassen	\
0	0.708333	0.539874	0.422741	0.011965	0.149505	0.074809	
1	0.442198	0.614704	0.599540	0.110409	0.206342	0.111286	
2	0.396552	0.549792	0.479632	0.150119	0.219467	0.489619	
3	0.856837	0.077254	0.272650	0.413659	0.032749	0.115494	
4	0.895416	0.214203	0.284997	0.155010	0.070358	0.205294	

	Channel_1	Channel_2	Region_1	Region_2	Region_3
0	0.000000	0.000056	0.0	0.0	0.000056
1	0.000000	0.000063	0.0	0.0	0.000063
2	0.000000	0.000062	0.0	0.0	0.000062
3	0.000065	0.000000	0.0	0.0	0.000065
4	0.000000	0.000040	0.0	0.0	0.000040

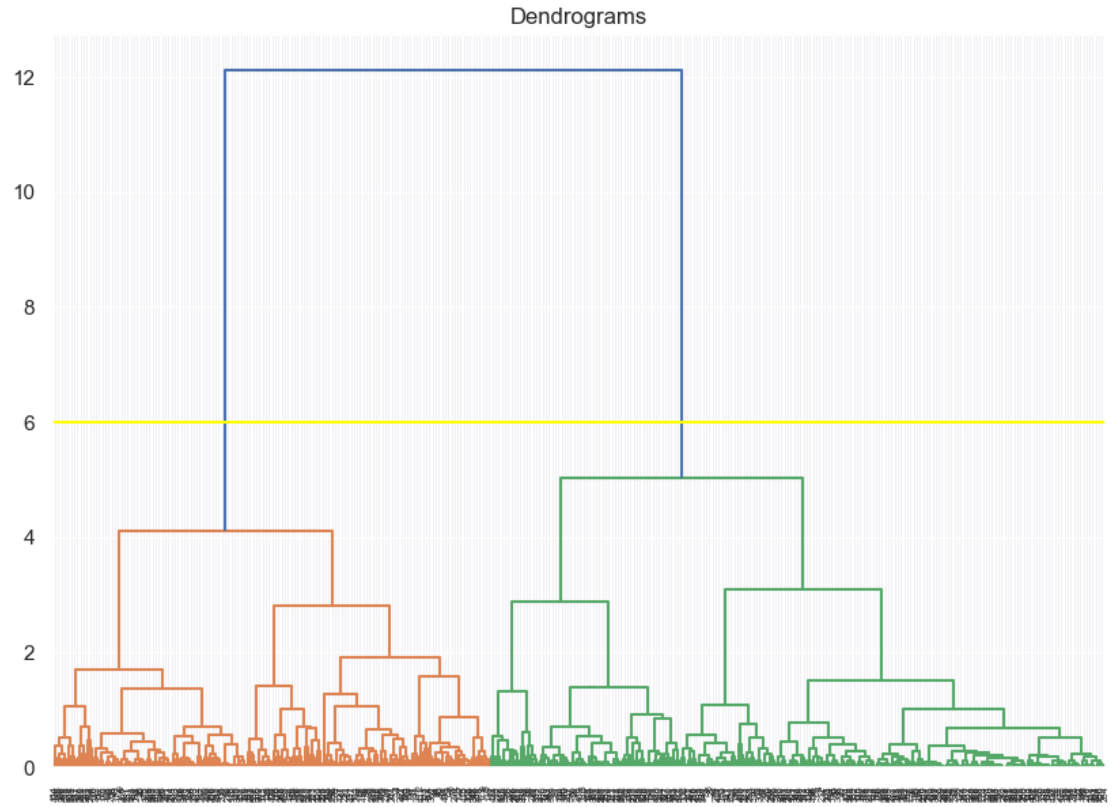
[20]:
```
import scipy.cluster.hierarchy as shc
plt.figure(figsize=(10, 7))
plt.title("Dendrograms")
```

```
dend = shc.dendrogram(shc.linkage(df_scaled, method='ward'))
```



[21]:
```
plt.figure(figsize=(10, 7))
plt.title("Dendrograms")
dend = shc.dendrogram(shc.linkage(df_scaled, method='ward'))
plt.axhline(y=6, color='yellow', linestyle='-')
```

[21]: <matplotlib.lines.Line2D at 0x17958223700>



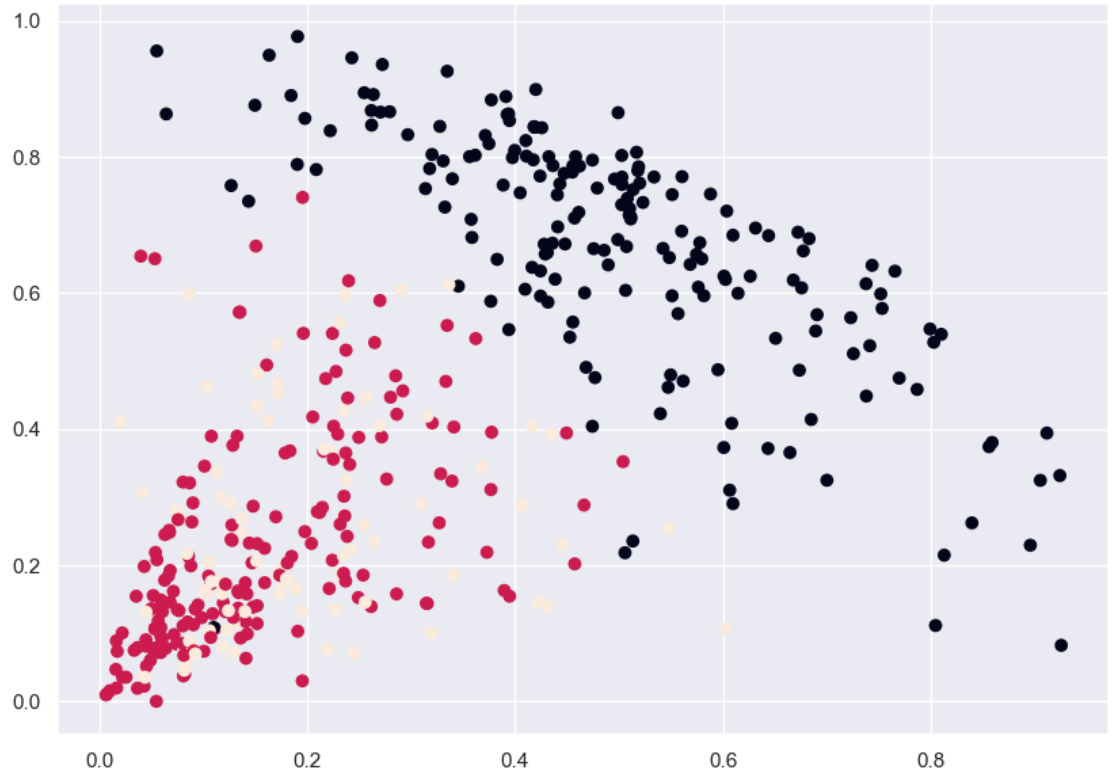
[22]:
```
from sklearn.cluster import AgglomerativeClustering
cluster = AgglomerativeClustering(n_clusters=3, affinity='euclidean',␣
 ↪linkage='ward')
cluster.fit_predict(df_scaled)
```

[22]:
```
array([0, 0, 0, 2, 1, 0, 1, 0, 0, 0, 0, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 2,
       1, 0, 1, 1, 1, 1, 0, 1, 1, 0, 1, 1, 0, 0, 1, 0, 0, 1, 1, 1, 0, 0,
       0, 0, 0, 0, 0, 0, 2, 0, 1, 0, 1, 0, 0, 0, 1, 0, 1, 0, 0, 0, 2, 0,
       0, 1, 0, 1, 2, 1, 2, 2, 0, 1, 2, 0, 1, 0, 1, 0, 0, 1, 1, 0, 0, 1,
       2, 1, 1, 2, 0, 2, 0, 0, 0, 2, 2, 2, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0,
       1, 0, 2, 1, 1, 1, 1, 0, 1, 2, 1, 2, 1, 0, 1, 1, 2, 1, 0, 1, 2, 1,
       1, 1, 1, 1, 0, 0, 1, 0, 1, 1, 1, 2, 1, 0, 1, 1, 2, 1, 1, 0, 1, 0,
       1, 0, 0, 1, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 2, 0, 0, 0, 0, 0, 0,
       1, 1, 1, 0, 1, 1, 0, 0, 0, 2, 2, 0, 0, 0, 2, 1, 1, 0, 1, 2, 2, 0,
       2, 1, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 1, 0, 2,
       1, 0, 2, 2, 1, 1, 1, 2, 0, 2, 2, 0, 1, 0, 1, 0, 1, 1, 1, 1, 2, 1,
       1, 1, 0, 0, 1, 1, 1, 2, 2, 0, 2, 1, 0, 1, 1, 2, 1, 1, 1, 2, 1, 0,
       0, 0, 0, 1, 0, 1, 2, 0, 0, 1, 0, 0, 1, 2, 2, 1, 0, 1, 1, 2, 1, 1,
       1, 2, 1, 1, 0, 2, 0, 0, 1, 1, 1, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1,
       1, 0, 2, 1, 0, 2, 1, 0, 1, 0, 1, 0, 0, 2, 1, 1, 1, 2, 2, 0, 2, 2,
       2, 0, 1, 0, 2, 1, 1, 2, 2, 2, 0, 0, 0, 0, 2, 0, 0, 1, 0, 0, 2, 0,
```

```
       0, 0, 2, 0, 1, 0, 0, 0, 1, 1, 0, 1, 2, 0, 1, 2, 1, 1, 1, 2, 0, 1,
       1, 1, 0, 1, 0, 0, 1, 2, 1, 2, 0, 1, 0, 1, 1, 1, 2, 0, 0, 2, 1, 1,
       0, 2, 2, 1, 2, 2, 1, 1, 1, 1, 2, 0, 1, 1, 1, 0, 0, 2, 2, 0, 0, 0,
       0, 1, 0, 1, 1, 1, 1, 2, 0, 2, 0, 1, 0, 2, 1, 0, 1, 2, 1, 0, 1, 0],
      dtype=int64)
```

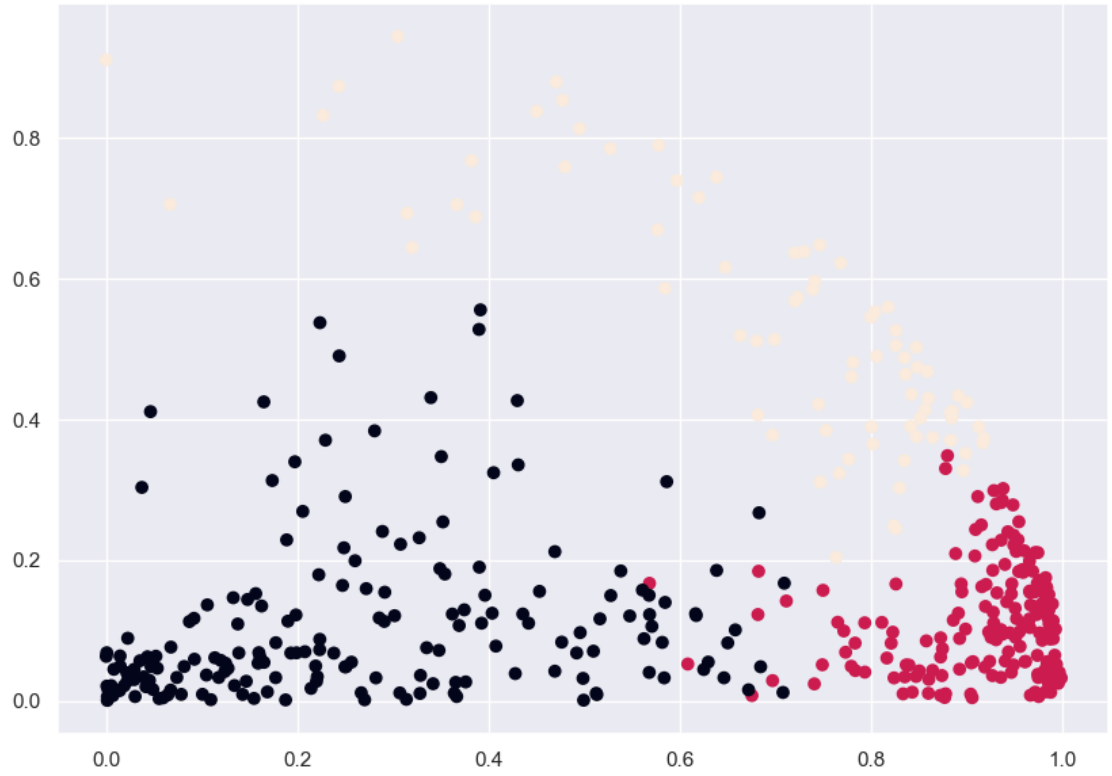
[23]:
```
plt.figure(figsize=(10, 7))
plt.scatter(df_scaled['Milk'], df_scaled['Grocery'], c=cluster.labels_)
```

[23]: <matplotlib.collections.PathCollection at 0x17957d17b80>



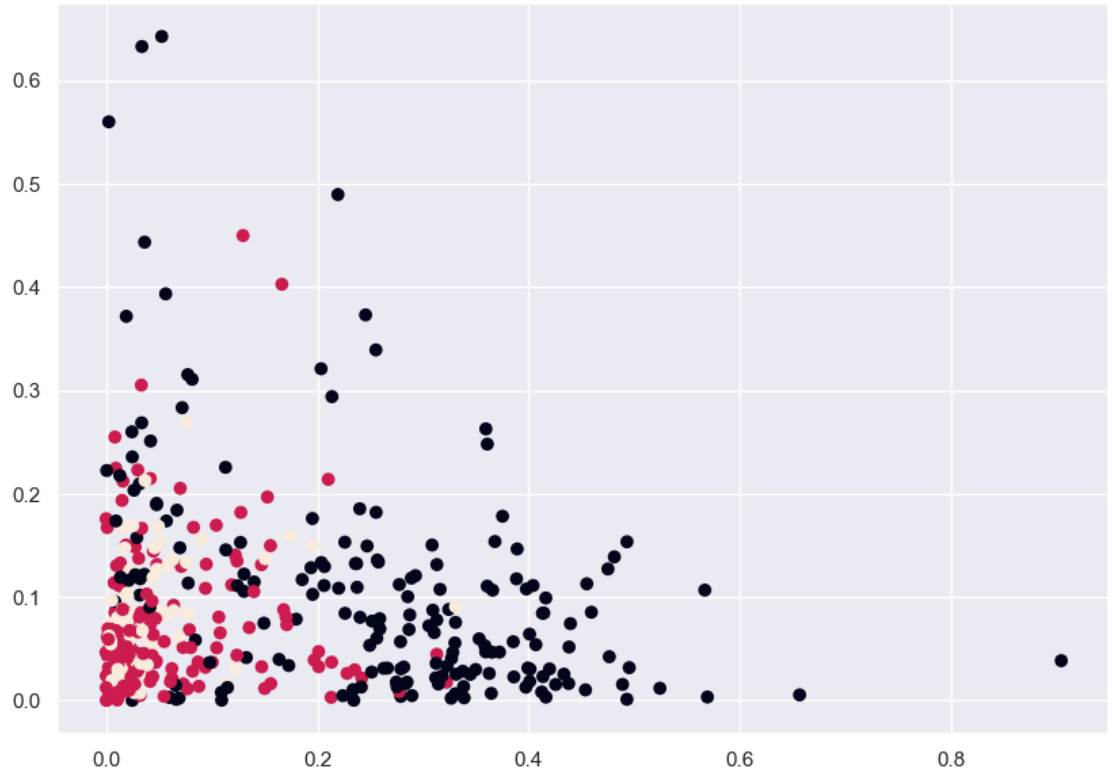
[24]:
```
plt.figure(figsize=(10, 7))
plt.scatter(df_scaled['Fresh'], df_scaled['Frozen'], c=cluster.labels_)
```

[24]: <matplotlib.collections.PathCollection at 0x17957dd6580>



[25]:
```
plt.figure(figsize=(10, 7))
plt.scatter(df_scaled['Detergents_Paper'], df_scaled['Delicassen'], c=cluster.
 ↪labels_)
```

[25]: <matplotlib.collections.PathCollection at 0x17958293190>


0.6
0.5
0.4
0.3
0.2
0.1
0.0
0.0
0.2
0.4
0.6
0.8

[ ]: