Clustering Algorithm

#Hierarchical Clustering

In [2]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
```

In [3]:
```
df = pd.read_csv("Documents/customers data.csv")
```

In [4]:
```
df.head()
```

Out[4]:

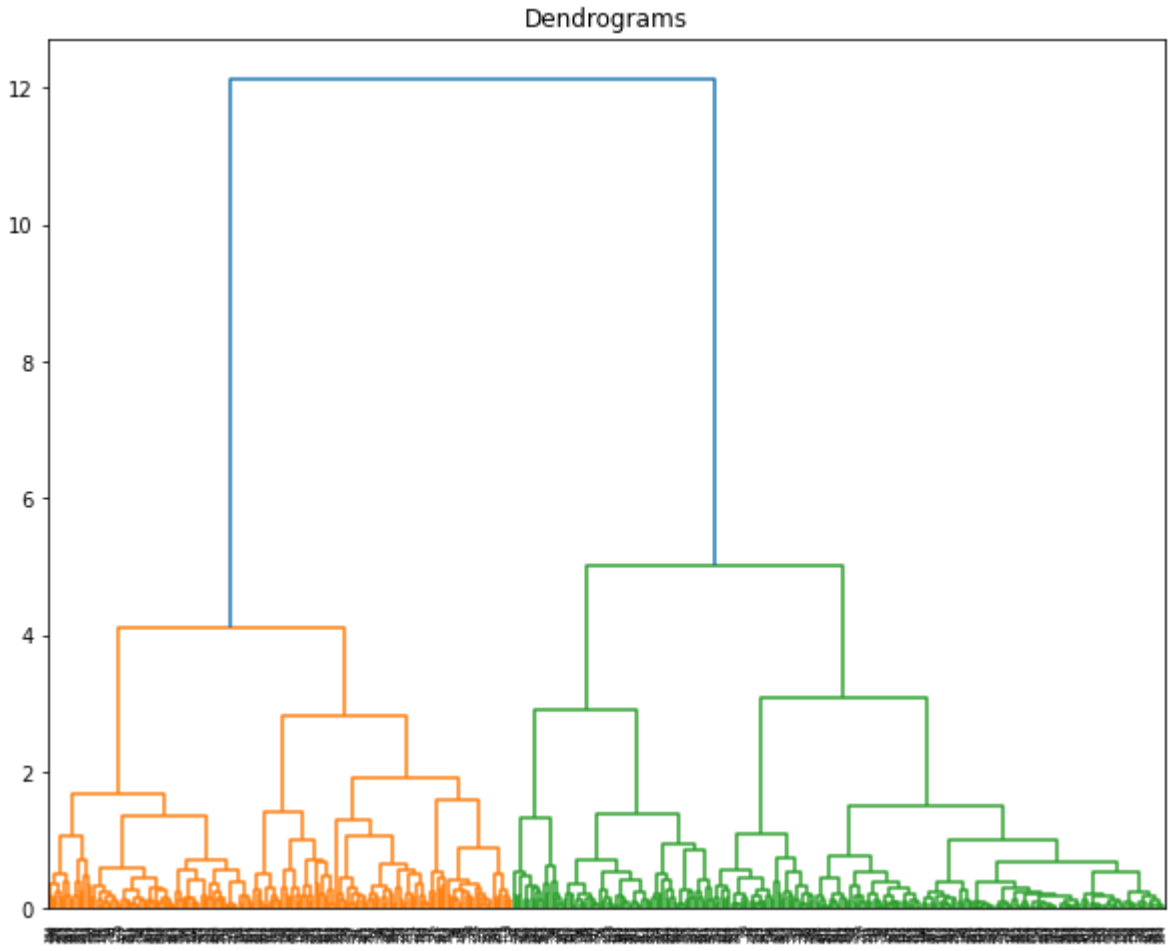
	Channel	Region	Fresh	Milk	Grocery	Frozen	Detergents_Paper	Delicassen
0	2	3	12669	9656	7561	214	2674	1338
1	2	3	7057	9810	9568	1762	3293	1776
2	2	3	6353	8808	7684	2405	3516	7844
3	1	3	13265	1196	4221	6404	507	1788
4	2	3	22615	5410	7198	3915	1777	5185

In [5]:
```
#We need to make clusters, but first we Normalize the data

from sklearn.preprocessing import normalize

data_scaled = pd.DataFrame(normalize(df), columns = df.columns)
data_scaled.head()
```

Out[5]:

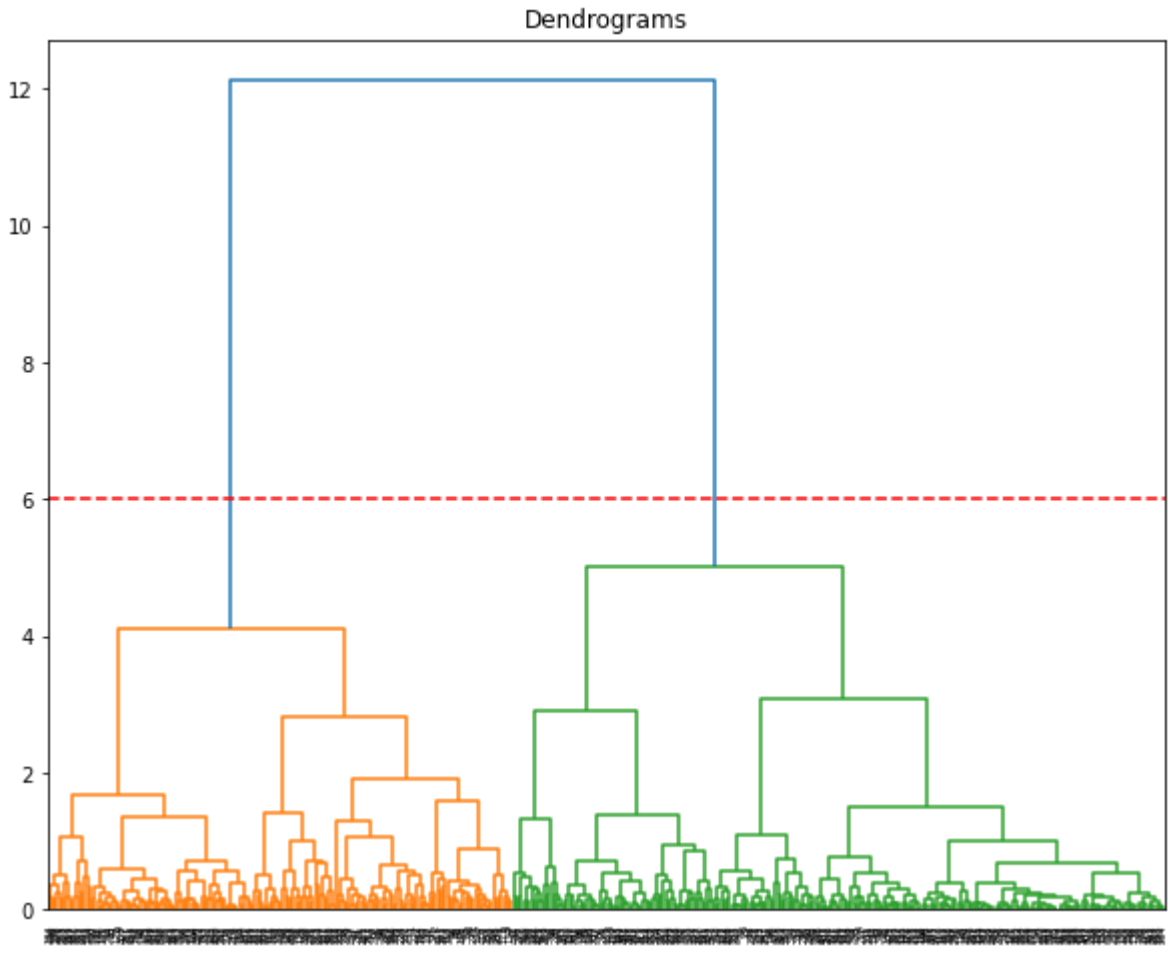
	Channel	Region	Fresh	Milk	Grocery	Frozen	Detergents_Paper	Delicassen
0	0.000112	0.000168	0.708333	0.539874	0.422741	0.011965	0.149505	0.074809
1	0.000125	0.000188	0.442198	0.614704	0.599540	0.110409	0.206342	0.111286
2	0.000125	0.000187	0.396552	0.549792	0.479632	0.150119	0.219467	0.489619
3	0.000065	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

In [6]:
```
#Forming clusters

import scipy.cluster.hierarchy as shc

plt.figure(figsize = (10,8))
plt.title("Dendrograms")
dend = shc.dendrogram(shc.linkage(data_scaled, method = 'ward'))
```



In [8]:
```
#as verticle line with the maximum distance is the blue line and hence we can dec

plt.figure(figsize = (10,8))
plt.title("Dendrograms")
dend = shc.dendrogram(shc.linkage(data_scaled, method = 'ward'))
plt.axhline(y=6, color = 'r', linestyle = '--')
```

Out[8]:
```
<matplotlib.lines.Line2D at 0x237d991e370>
```

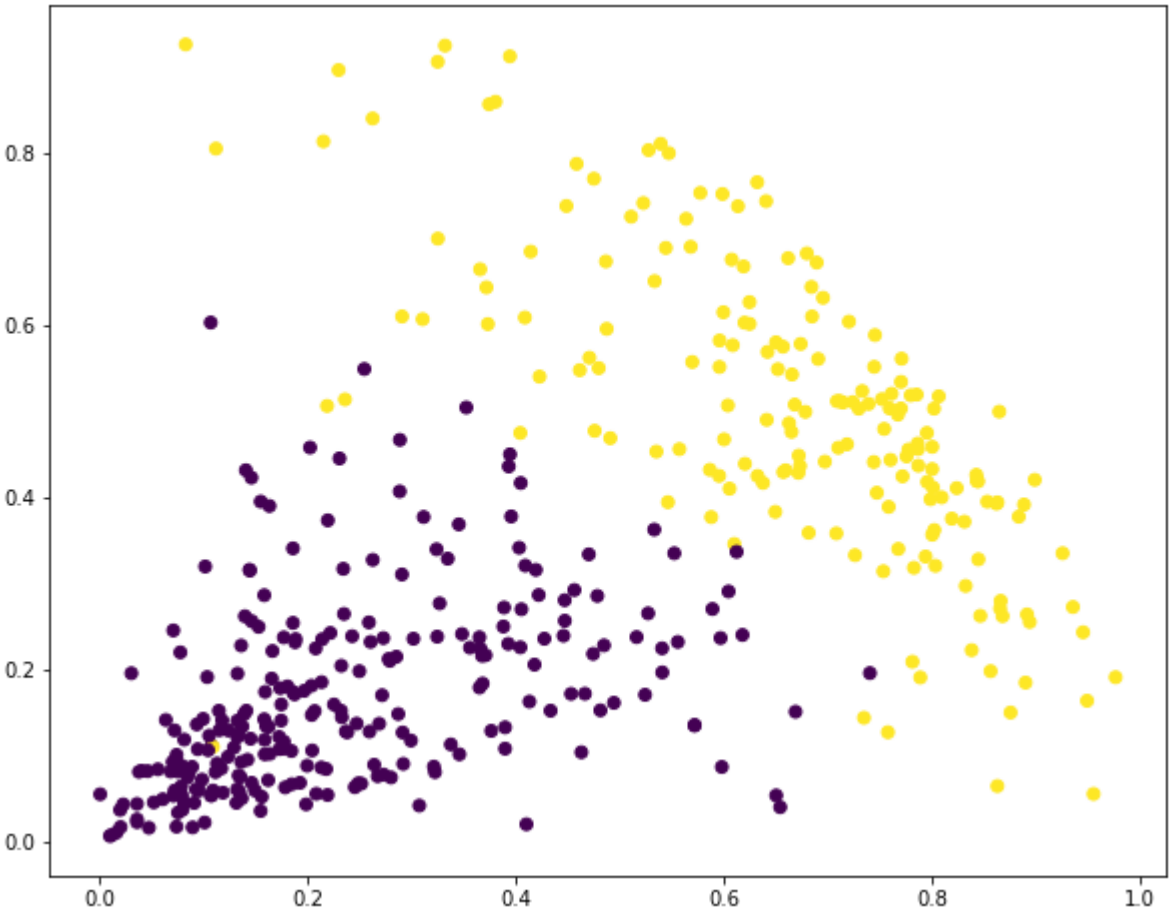


In [9]:
```
from sklearn.cluster import AgglomerativeClustering
cluster = AgglomerativeClustering(n_clusters=2, affinity = 'euclidean', linkage =
cluster.fit_predict(data_scaled)
```

Out[9]:
```
array([1, 1, 1, 0, 0, 1, 0, 1, 1, 1, 1, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0,
       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, 0, 1, 0, 1, 0, 1, 1, 1, 0, 1, 0, 1, 1, 1, 0, 1,
       1, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0, 1, 1, 0, 0, 1, 1, 0,
       0, 0, 0, 0, 1, 0, 1, 1, 1, 0, 0, 0, 1, 1, 1, 0, 0, 0, 1, 1, 1, 1,
       0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0,
       0, 0, 0, 0, 1, 1, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 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, 1, 0, 0, 1, 1, 1, 0, 0, 1, 1, 1, 0, 0, 0, 1, 0, 0, 0, 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, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0,
       0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1,
       1, 1, 1, 0, 1, 0, 0, 1, 1, 0, 1, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 1, 0, 1, 1, 0, 0, 0, 0, 1, 1, 0, 1, 1, 1, 1, 1, 1, 0,
       0, 1, 0, 0, 1, 0, 0, 1, 0, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0,
       0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 0, 1, 1, 0, 1, 1, 0, 1,
       1, 1, 0, 1, 0, 1, 1, 1, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0,
       0, 0, 1, 0, 1, 1, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 1, 1, 0, 0, 0,
       1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 1, 0, 0, 1, 1, 1,
       1, 0, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0, 0, 1, 0, 0, 0, 1, 0, 1],
      dtype=int64)
```

In [30]:
```
plt.figure(figsize = (10,8))
plt.scatter(data_scaled['Grocery'], data_scaled['Milk'], c= cluster.labels_)
```

Out[30]: <matplotlib.collections.PathCollection at 0x237db532dc0>



In [12]:
```
#So we have clearly 2 clusters
```

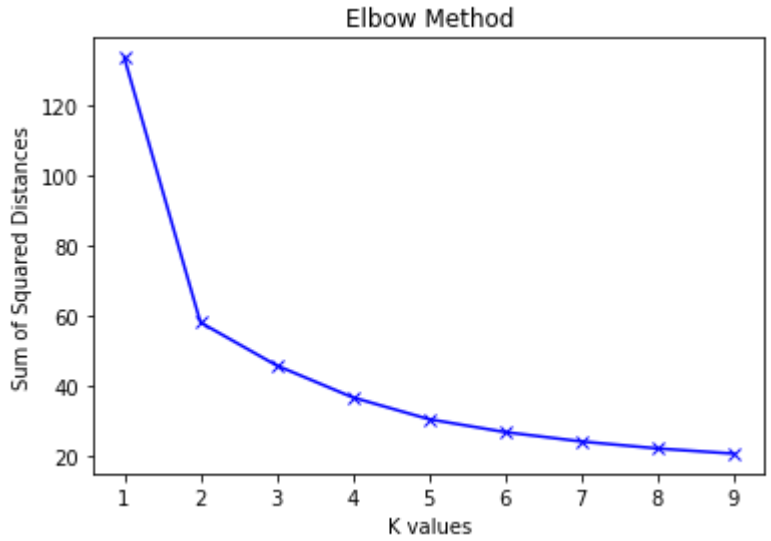
In [16]:
```
# Using K-means clustering

from sklearn import metrics
from sklearn.cluster import KMeans
import matplotlib.pyplot as plt

df
```

Out[16]:

	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
...	...	...	...	...	...	...	...	...
435	1	3	29703	12051	16027	13135	182	2204
436	1	3	39228	1431	764	4510	93	2346
437	2	3	14531	15488	30243	437	14841	1867
438	1	3	10290	1981	2232	1038	168	2125
439	1	3	2787	1698	2510	65	477	52

440 rows × 8 columns

In [22]:
```
#finding the number of clusters we want using elbow method

# Arbitrarily selecting a range of values for K
K = range(1,10)
sum_of_squared_distances = []
# Using Scikit Learn's KMeans Algorithm to find sum of squared distances
for k in K:
    model = KMeans(n_clusters=k).fit(data_scaled)
    sum_of_squared_distances.append(model.inertia_)
plt.plot(K, sum_of_squared_distances, "bx-")
plt.xlabel('K values')
plt.ylabel('Sum of Squared Distances')
plt.title('Elbow Method')
plt.show()
```



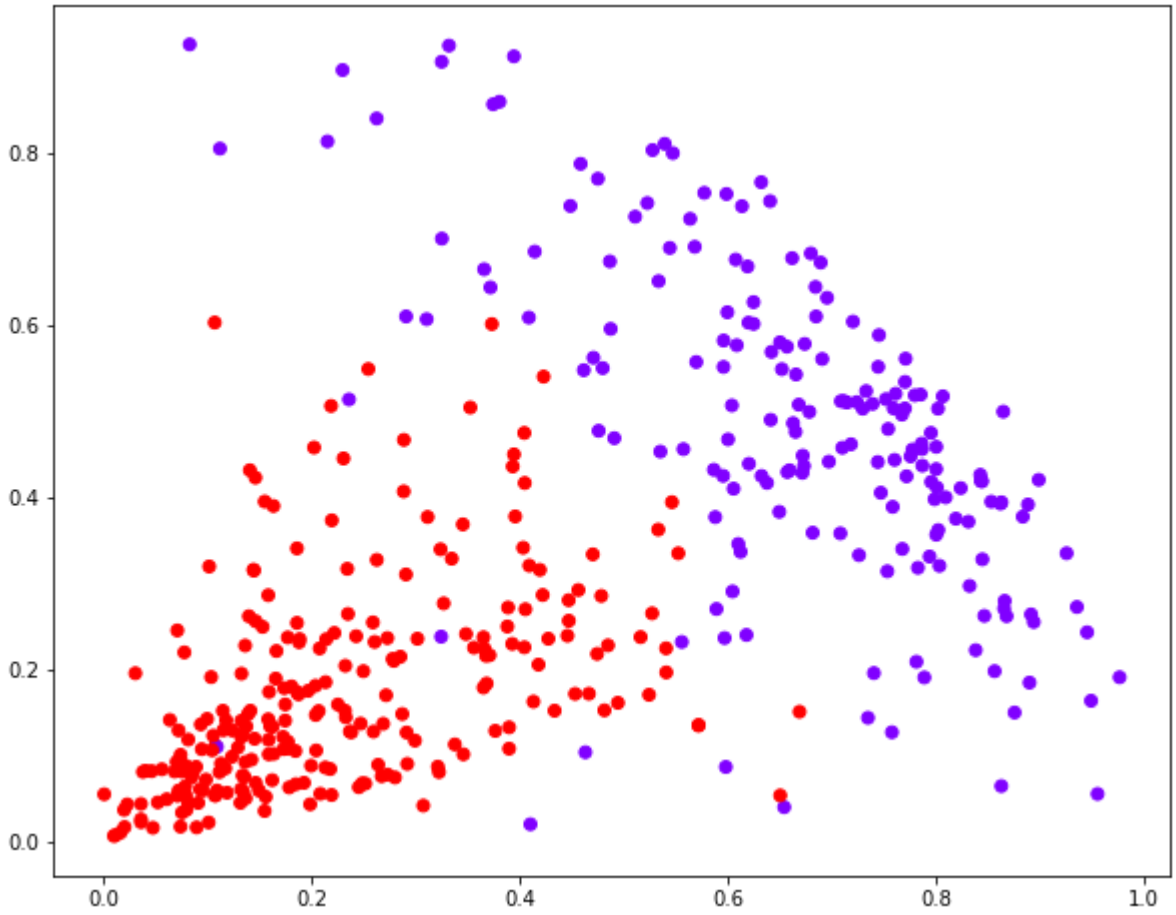
In [23]:
```
#From here we see that 2 clusters must be made

data_kmeans = KMeans(n_clusters=2)
data_kmeans.fit(data_scaled)
labels = data_kmeans.predict(data_scaled)
print(labels)
```

```
[1 0 0 1 1 1 1 0 0 0 0 1 1 1 1 1 0 0 1 0 1 1 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 1 0 1 0 1 0 0 0 1 0 0 0 0 0 1 0 0 1 0 1 1 0 1 1
 0 1 0 0 1 0 1 0 0 1 1 0 0 1 1 1 1 1 0 1 0 0 0 0 0 1 0 0 0 1 1 1 0 0 0 0 1
 0 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 0 1 1 1 1 1 1 1 0 0 1 0 1 1 1 1 1 0 1 1
 1 1 1 0 1 0 1 0 0 1 0 0 0 1 1 0 0 0 0 0 1 1 0 0 0 0 0 0 1 1 1 0 1 1 0 0 0
 1 1 0 0 0 1 1 1 0 1 1 1 0 1 1 0 0 1 0 0 0 1 0 0 0 1 0 1 0 0 0 0 1 0 1 1 0
 1 0 1 1 1 1 0 1 1 0 1 0 1 0 1 1 1 1 1 1 1 1 0 0 1 1 1 1 1 0 1 1 0 1 1 1 1
 1 1 1 1 0 0 0 0 1 0 1 1 0 0 1 0 0 1 1 1 1 0 1 1 1 1 1 1 1 1 1 0 1 0 0 1 1
 1 0 0 0 1 0 0 0 0 0 0 1 1 0 1 1 0 1 1 0 1 0 1 0 0 1 1 1 1 1 1 0 1 1 1 0 1
 0 1 1 1 1 0 0 0 0 0 0 1 0 0 1 1 0 1 0 0 0 1 0 1 0 0 0 1 1 0 1 1 0 1 1 1 1
 1 1 1 1 1 1 0 1 0 0 1 1 1 0 0 1 0 1 1 1 1 0 0 1 1 1 0 1 1 1 1 1 1 1 1 1 1
 0 1 1 1 0 0 0 1 0 0 0 0 1 0 1 1 1 1 1 0 1 0 1 0 1 1 0 1 1 1 0 1 0]
```

In [24]:
```
centroids = data_kmeans.cluster_centers_
```

In [33]:
```
s = metrics.silhouette_score(data_scaled, labels, metric = 'euclidean')
print(f"Silhoutte Coefficient for the Dataset Clusters: {s: .2f}")
plt.figure(figsize = (10,8))

plt.scatter(data_scaled['Grocery'],data_scaled['Milk'], c = labels, cmap = 'rainb
plt.show()
```

```
Silhoutte Coefficient for the Dataset Clusters:  0.50
```



In [ ]:
```
```