Hierarchical Clustering

```
1 # Importing the libraries
2 import numpy as np
3 import matplotlib.pyplot as plt
4 import pandas as pd
```

Importing the dataset

```
1 from google.colab import files
2 upload = files.upload()
```

> Choose Files No file chosen   Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.
> Saving Wholesale customers data.csv to Wholesale customers data.csv

```
1 dataset = pd.read_csv('Wholesale customers data.csv')
2 #X = dataset.iloc[:, [3, 4]].values
3 #X = dataset.iloc[:, [3, 4]].values
4 from sklearn.preprocessing import normalize
5 data_scaled = normalize(dataset)
6 data_scaled = pd.DataFrame(data_scaled, columns=dataset.columns)
7 data_scaled.head()
8 X=data_scaled
9 X.head()
```

	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

Using the dendrogram to find the optimal number of clusters

```
1 import scipy.cluster.hierarchy as sch
2 dendrogram = sch.dendrogram(sch.linkage(X, method = 'complete'))
3 plt.title('Dendrogram')
4 plt.xlabel('Customers')
5 plt.ylabel('Euclidean distances')
6 plt.show()
```
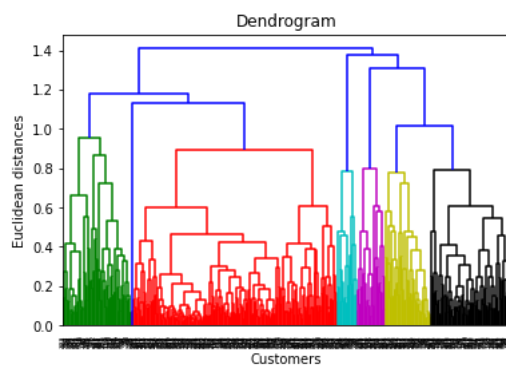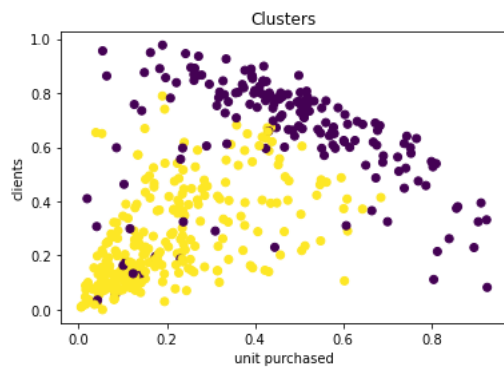

Training the Hierarchical Clustering model using complete link on the dataset

```
1 from sklearn.cluster import AgglomerativeClustering
2 hc = AgglomerativeClustering(n_clusters = 2, affinity = 'euclidean', linkage = 'complete')
3 y_hc = hc.fit_predict(X)
```

Visualising the clusters

```
1 #plt.figure(figsize=(7, 6))
2 plt.scatter(X['Milk'], X['Grocery'], c=hc.labels_)
3 plt.title('Clusters')
4 plt.xlabel('unit purchased')
5 plt.ylabel('clients')
6 plt.show()
```