In [96]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA
from sklearn.cluster import KMeans
from scipy.spatial.distance import cdist
from scipy.cluster.hierarchy import linkage
import scipy.cluster.hierarchy as sch
from sklearn.cluster import AgglomerativeClustering

import warnings
warnings.filterwarnings('ignore')
```

In [66]:
```
data = pd.read_csv('wine.csv')
data
```

Out[66]:

	Type	Alcohol	Malic	Ash	Alcalinity	Magnesium	Phenols	Flavanoids	Nonflavanoids	Proar
0	1	14.23	1.71	2.43	15.6	127	2.80	3.06	0.28	
1	1	13.20	1.78	2.14	11.2	100	2.65	2.76	0.26	
2	1	13.16	2.36	2.67	18.6	101	2.80	3.24	0.30	
3	1	14.37	1.95	2.50	16.8	113	3.85	3.49	0.24	
4	1	13.24	2.59	2.87	21.0	118	2.80	2.69	0.39	
...	...	...	...	...	...	...	...	...	...	...
173	3	13.71	5.65	2.45	20.5	95	1.68	0.61	0.52	
174	3	13.40	3.91	2.48	23.0	102	1.80	0.75	0.43	
175	3	13.27	4.28	2.26	20.0	120	1.59	0.69	0.43	
176	3	13.17	2.59	2.37	20.0	120	1.65	0.68	0.53	
177	3	14.13	4.10	2.74	24.5	96	2.05	0.76	0.56	

178 rows × 14 columns

In [67]:
```
data.shape
```

Out[67]:
```
(178, 14)
```

In [68]:
```
data.isna().sum()
```

Out[68]:
```
Type              0
Alcohol           0
Malic             0
Ash               0
Alcalinity        0
Magnesium         0
Phenols           0
Flavanoids        0
Nonflavanoids     0
Proanthocyanins   0
Color             0
Hue               0
Dilution          0
Proline           0
dtype: int64
```

In [69]:
```
data.dtypes
```

Out[69]:
```
Type                int64
Alcohol           float64
Malic             float64
Ash               float64
Alcalinity        float64
Magnesium           int64
Phenols           float64
Flavanoids        float64
Nonflavanoids     float64
Proanthocyanins   float64
Color             float64
Hue               float64
Dilution          float64
Proline             int64
dtype: object
```

In [70]:
```
scaler = StandardScaler()
scaler.fit(data)
scaled_data = scaler.transform(data)
```

In [71]:
```
pca = PCA(n_components=10)
pca.fit(scaled_data)
x_pca = pca.transform(scaled_data)
```

In [72]:
```
scaled_data.shape
```

Out[72]:
```
(178, 14)
```

In [73]:
```
x_pca.shape
```

Out[73]:
```
(178, 10)
```

In [74]:
```
var = pca.explained_variance_ratio_
var
pca.components_[0]
```

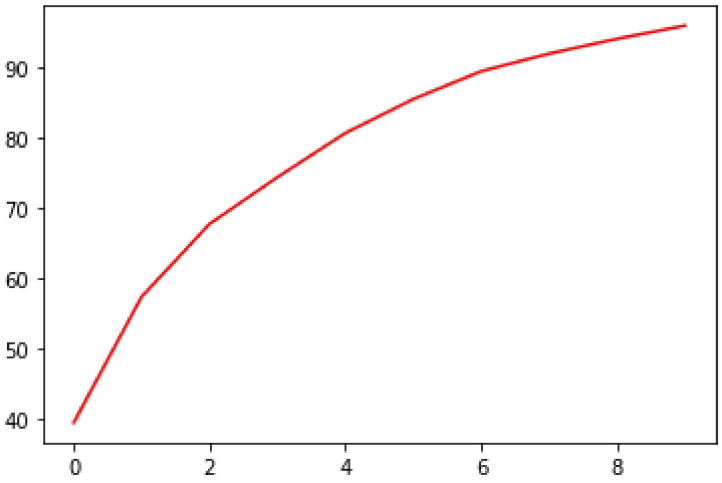
Out[74]:
```
array([ 0.39366953, -0.13632501,  0.22267638, -0.00225793,  0.22429849,
       -0.12463016, -0.35926404, -0.39071171,  0.2670012 , -0.2790625 ,
        0.08931829, -0.27682265, -0.35052618, -0.26951525])
```

In [75]:
```
var1 = np.cumsum(np.round(var,decimals = 4)*100)
var1
```

Out[75]:
```
array([39.54, 57.38, 67.71, 74.34, 80.61, 85.42, 89.38, 91.88, 93.98,
       95.85])
```

In [76]:
```
plt.plot(var1,color="red")
```

Out[76]:
```
[<matplotlib.lines.Line2D at 0x26ed08ef6d0>]
```



In [77]:
```
data.columns
```

Out[77]:
```
Index(['Type', 'Alcohol', 'Malic', 'Ash', 'Alcalinity', 'Magnesium', 'Phenols',
       'Flavanoids', 'Nonflavanoids', 'Proanthocyanins', 'Color', 'Hue',
       'Dilution', 'Proline'],
      dtype='object')
```

In [78]:
```
#X = data.drop(['Alcohol', 'Malic', 'Ash', 'Alcalinity', 'Magnesium', 'Nonflavanc
```

In [79]:
```
X = data.iloc[:,[0,6,7]]
```

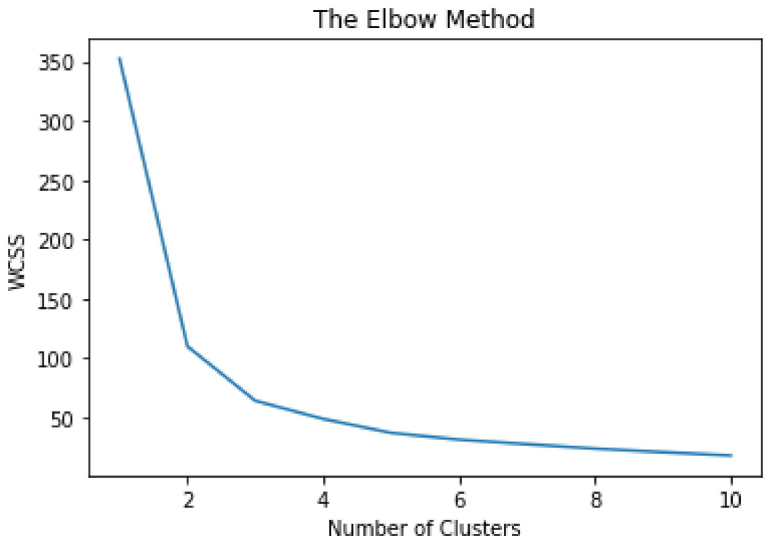
In [80]:
```
X.head()
```

Out[80]:

	Type	Phenols	Flavanoids
0	1	2.80	3.06
1	1	2.65	2.76
2	1	2.80	3.24
3	1	3.85	3.49
4	1	2.80	2.69

In [81]:
```
wcss=[]
for i in range(1,11):
    kmeans=KMeans(n_clusters=i, init='k-means++',random_state=0)
    kmeans.fit(X)
    wcss.append(kmeans.inertia_)

plt.plot(range(1,11),wcss)
plt.title('The Elbow Method')
plt.xlabel('Number of Clusters')
plt.ylabel('WCSS')
plt.show()
```



In [84]:
```
kmeans=KMeans(n_clusters=3,init='k-means++',random_state=0)
y_kmeans=kmeans.fit_predict(X)
y_kmeans
```

Out[84]:
```
array([1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 2, 2, 2, 1, 2, 2,
       1, 2, 2, 2, 0, 1, 2, 1, 1, 2, 2, 2, 2, 1, 2, 2, 2, 2, 2, 2, 2, 2,
       2, 2, 2, 2, 2, 2, 2, 2, 0, 2, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 1,
       1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 1, 1, 2, 1, 1, 1, 1, 2, 2, 2, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0])
```

In [85]:
```
def norm_func(i):
    x = (i-i.min())/(i.max()-i.min())
    #x = (i-i.mean())/i.std()
    return (x)
```

In [86]:
```
df_norm = norm_func(X)
```

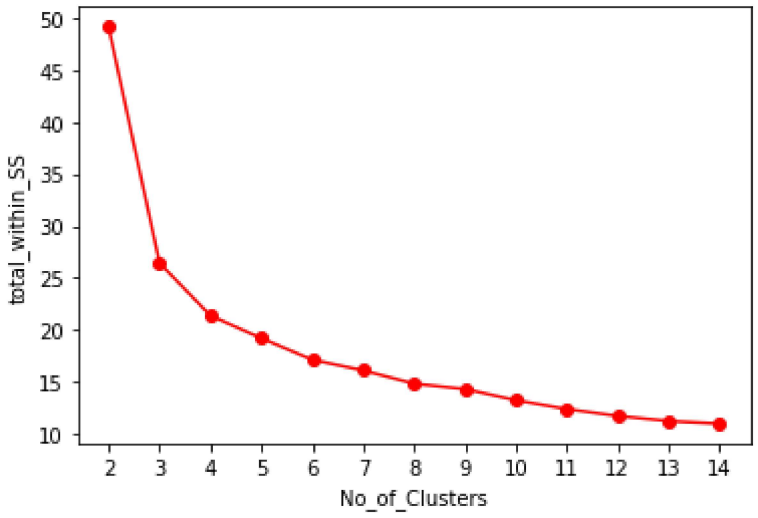
In [87]:
```
df_norm.head()
```

Out[87]:

	Type	Phenols	Flavanoids
0	0.0	0.627586	0.573840
1	0.0	0.575862	0.510549
2	0.0	0.627586	0.611814
3	0.0	0.989655	0.664557
4	0.0	0.627586	0.495781

In [88]:
```
k = list(range(2,15))
k
TWSS = []
for i in k:
    kmeans = KMeans(n_clusters = i)
    kmeans.fit(df_norm)
    WSS = []
    for j in range(i):
        WSS.append(sum(cdist(df_norm.iloc[kmeans.labels_==j,:],kmeans.cluster_cer
    TWSS.append(sum(WSS))
```

In [90]:
```
plt.plot(k,TWSS, 'ro-');plt.xlabel("No_of_Clusters");plt.ylabel("total_within_SS"
plt.show()
```



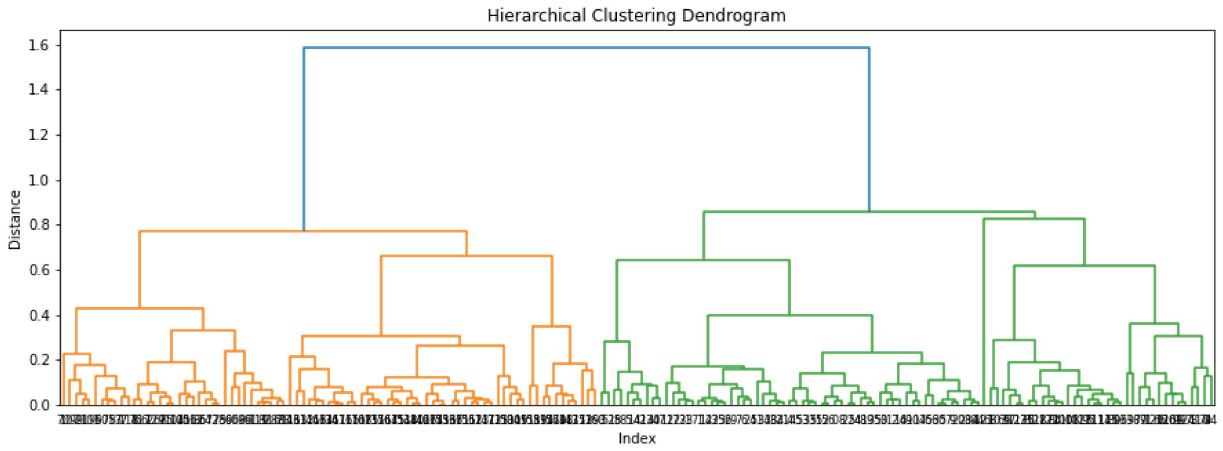
In [91]:
```
model=KMeans(n_clusters=4)
model.fit(df_norm)
```

Out[91]:
```
KMeans(n_clusters=4)
```

In [92]:
```
model.labels_
md=pd.Series(model.labels_)
```

In [93]:
```
z = linkage(df_norm, method="complete",metric="euclidean")
```

In [94]:
```
plt.figure(figsize=(15, 5));plt.title('Hierarchical Clustering Dendrogram');plt.x
sch.dendrogram(
    z,
    leaf_rotation=0.,  # rotates the x axis labels
    leaf_font_size=8.,  # font size for the x axis labels
)
plt.show()
```



In [97]:
```
tiveClustering(n_clusters=4,linkage='complete',affinity = "euclidean").fit(df_norm
```

In [98]:
```
h_complete.labels_
```

Out[98]:
```
array([2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 0, 0, 0, 0, 1, 0, 1,
       1, 0, 1, 0, 0, 1, 0, 1, 1, 0, 0, 0, 0, 1, 1, 1, 0, 0, 1, 0, 0, 0,
       0, 0, 0, 0, 0, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 0, 0, 0, 0, 1, 1,
       1, 1, 0, 1, 1, 1, 0, 0, 0, 0, 1, 3, 1, 1, 1, 1, 1, 1, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0], dtype=int64)
```

In [99]:
```
cluster_labels=pd.Series(h_complete.labels_)
```

In [ ]:
```

```