Import neccessery libraries

In [36]:

```
import pandas as pd
import numpy as np
from matplotlib import pyplot as plt
from sklearn.cluster import KMeans
from sklearn.preprocessing import StandardScaler
from sklearn import preprocessing
from sklearn.decomposition import PCA
import seaborn as sns
import scipy.cluster.hierarchy as sch
from sklearn.cluster import AgglomerativeClustering
from scipy.cluster.hierarchy import linkage

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

Problem

Perform Principal component analysis and perform clustering using first 3 principal component scores (both heirarchial and k mean clustering(scree plot or elbow curve) and obtain optimum number of clusters and check whether we have obtained same number of clusters with the original data

Import data

In [2]:

```
wine_data = pd.read_csv('wine.csv')
wine_data
```

Out[2]:

	Type	Alcohol	Malic	Ash	Alcalinity	Magnesium	Phenols	Flavanoids	Nonflavanoids	Proanthocyanins	Color	Hue	Dilution	Proline
0	1	14.23	1.71	2.43	15.6	127	2.80	3.06	0.28	2.29	5.64	1.04	3.92	1065
1	1	13.20	1.78	2.14	11.2	100	2.65	2.76	0.26	1.28	4.38	1.05	3.40	1050
2	1	13.16	2.36	2.67	18.6	101	2.80	3.24	0.30	2.81	5.68	1.03	3.17	1185

	Type	Alcohol	Malic	Ash	Alcalinity	Magnesium	Phenols	Flavanoids	Nonflavanoids	Proanthocyanins	Color	Hue	Dilution	Proline
3	1	14.37	1.95	2.50	16.8	113	3.85	3.49	0.24	2.18	7.80	0.86	3.45	1480
4	1	13.24	2.59	2.87	21.0	118	2.80	2.69	0.39	1.82	4.32	1.04	2.93	735
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
173	3	13.71	5.65	2.45	20.5	95	1.68	0.61	0.52	1.06	7.70	0.64	1.74	740
174	3	13.40	3.91	2.48	23.0	102	1.80	0.75	0.43	1.41	7.30	0.70	1.56	750
175	3	13.27	4.28	2.26	20.0	120	1.59	0.69	0.43	1.35	10.20	0.59	1.56	835
176	3	13.17	2.59	2.37	20.0	120	1.65	0.68	0.53	1.46	9.30	0.60	1.62	840
177	3	14.13	4.10	2.74	24.5	96	2.05	0.76	0.56	1.35	9.20	0.61	1.60	560

Data understanding

In [3]:

```
wine_data.shape
```

Out[3]:

```
(178, 14)
```

In [4]:

```
wine_data.isna().sum()
```

Out[4]:

```
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 [5]:

```
wine_data.dtypes
```

Out[5]:

```
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 [6]:

```
wine_data.iloc[:,1:]
```

Out[6]:

	Alcohol	Malic	Ash	Alcalinity	Magnesium	Phenols	Flavanoids	Nonflavanoids	Proanthocyanins	Color	Hue	Dilution	Proline
0	14.23	1.71	2.43	15.6	127	2.80	3.06	0.28	2.29	5.64	1.04	3.92	1065
1	13.20	1.78	2.14	11.2	100	2.65	2.76	0.26	1.28	4.38	1.05	3.40	1050
2	13.16	2.36	2.67	18.6	101	2.80	3.24	0.30	2.81	5.68	1.03	3.17	1185
3	14.37	1.95	2.50	16.8	113	3.85	3.49	0.24	2.18	7.80	0.86	3.45	1480
4	13.24	2.59	2.87	21.0	118	2.80	2.69	0.39	1.82	4.32	1.04	2.93	735
...	...	...	...	...	...	...	...	...	...	...	...	...	...
173	13.71	5.65	2.45	20.5	95	1.68	0.61	0.52	1.06	7.70	0.64	1.74	740
174	13.40	3.91	2.48	23.0	102	1.80	0.75	0.43	1.41	7.30	0.70	1.56	750
175	13.27	4.28	2.26	20.0	120	1.59	0.69	0.43	1.35	10.20	0.59	1.56	835
176	13.17	2.59	2.37	20.0	120	1.65	0.68	0.53	1.46	9.30	0.60	1.62	840
177	14.13	4.10	2.74	24.5	96	2.05	0.76	0.56	1.35	9.20	0.61	1.60	560

178 rows × 13 columns

In [7]:

```
wine_data.describe()
```

Out[7]:

	Type	Alcohol	Malic	Ash	Alcalinity	Magnesium	Phenols	Flavanoids	Nonflavanoids	Proanthocyanins	Color
count	178.000000	178.000000	178.000000	178.000000	178.000000	178.000000	178.000000	178.000000	178.000000	178.000000	178.000000
mean	1.938202	13.000618	2.336348	2.366517	19.494944	99.741573	2.295112	2.029270	0.361854	1.590899	5.058090
std	0.775035	0.811827	1.117146	0.274344	3.339564	14.282484	0.625851	0.998859	0.124453	0.572359	2.318286
min	1.000000	11.030000	0.740000	1.360000	10.600000	70.000000	0.980000	0.340000	0.130000	0.410000	1.280000
25%	1.000000	12.362500	1.602500	2.210000	17.200000	88.000000	1.742500	1.205000	0.270000	1.250000	3.220000
50%	2.000000	13.050000	1.865000	2.360000	19.500000	98.000000	2.355000	2.135000	0.340000	1.555000	4.690000
75%	3.000000	13.677500	3.082500	2.557500	21.500000	107.000000	2.800000	2.875000	0.437500	1.950000	6.200000
max	3.000000	14.830000	5.800000	3.230000	30.000000	162.000000	3.880000	5.080000	0.660000	3.580000	13.000000

Finding correlation between the variables in the data

In [8]:

```
corr_matrix = wine_data.corr()
```

In [9]:

```
corr_matrix.style.background_gradient(cmap='coolwarm')
```

Out[9]:

	Type	Alcohol	Malic	Ash	Alcalinity	Magnesium	Phenols	Flavanoids	Nonflavanoids	Proanthocyanins	Color
Type	1.000000	-0.328222	0.437776	-0.049643	0.517859	-0.209179	-0.719163	-0.847498	0.489109	-0.499130	0.265668
Alcohol	-0.328222	1.000000	0.094397	0.211545	-0.310235	0.270798	0.289101	0.236815	-0.155929	0.136698	0.546364
Malic	0.437776	0.094397	1.000000	0.164045	0.288500	-0.054575	-0.335167	-0.411007	0.292977	-0.220746	0.248985
Ash	-0.049643	0.211545	0.164045	1.000000	0.443367	0.286587	0.128980	0.115077	0.186230	0.009652	0.258887
Alcalinity	0.517859	-0.310235	0.288500	0.443367	1.000000	-0.083333	-0.321113	-0.351370	0.361922	-0.197327	0.018732

	Type	Alcohol	Malic	Ash	Alcalinity	Magnesium	Phenols	Flavanoids	Nonflavanoids	Proanthocyanins	Color
Magnesium	-0.209179	0.270798	-0.054575	0.286587	-0.083333	1.000000	0.214401	0.195784	-0.256294	0.236441	0.199950
Phenols	-0.719163	0.289101	-0.335167	0.128980	-0.321113	0.214401	1.000000	0.864564	-0.449935	0.612413	-0.055136
Flavanoids	-0.847498	0.236815	-0.411007	0.115077	-0.351370	0.195784	0.864564	1.000000	-0.537900	0.652692	-0.172379
Nonflavanoids	0.489109	-0.155929	0.292977	0.186230	0.361922	-0.256294	-0.449935	-0.537900	1.000000	-0.365845	0.139057
Proanthocyanins	-0.499130	0.136698	-0.220746	0.009652	-0.197327	0.236441	0.612413	0.652692	-0.365845	1.000000	-0.025250
Color	0.265668	0.546364	0.248985	0.258887	0.018732	0.199950	-0.055136	-0.172379	0.139057	-0.025250	1.000000
Hue	-0.617369	-0.071747	-0.561296	-0.074667	-0.273955	0.055398	0.433681	0.543479	-0.262640	0.295544	-0.521813
Dilution	-0.788230	0.072343	-0.368710	0.003911	-0.276769	0.066004	0.699949	0.787194	-0.503270	0.519067	-0.428815

There are some quite correlation between variables. For example the correlation between flavanoids and dilution is pretty high (78%). Thus we can remove that variable from our dataset. However this method is long and tedious. Hence we PCA method for Dimensionality Reduction

Dimensionality Reduction with PCA

In [10]:

```
# normalizing the data
wine_norm = StandardScaler().fit_transform(wine_data)
```

In [11]:

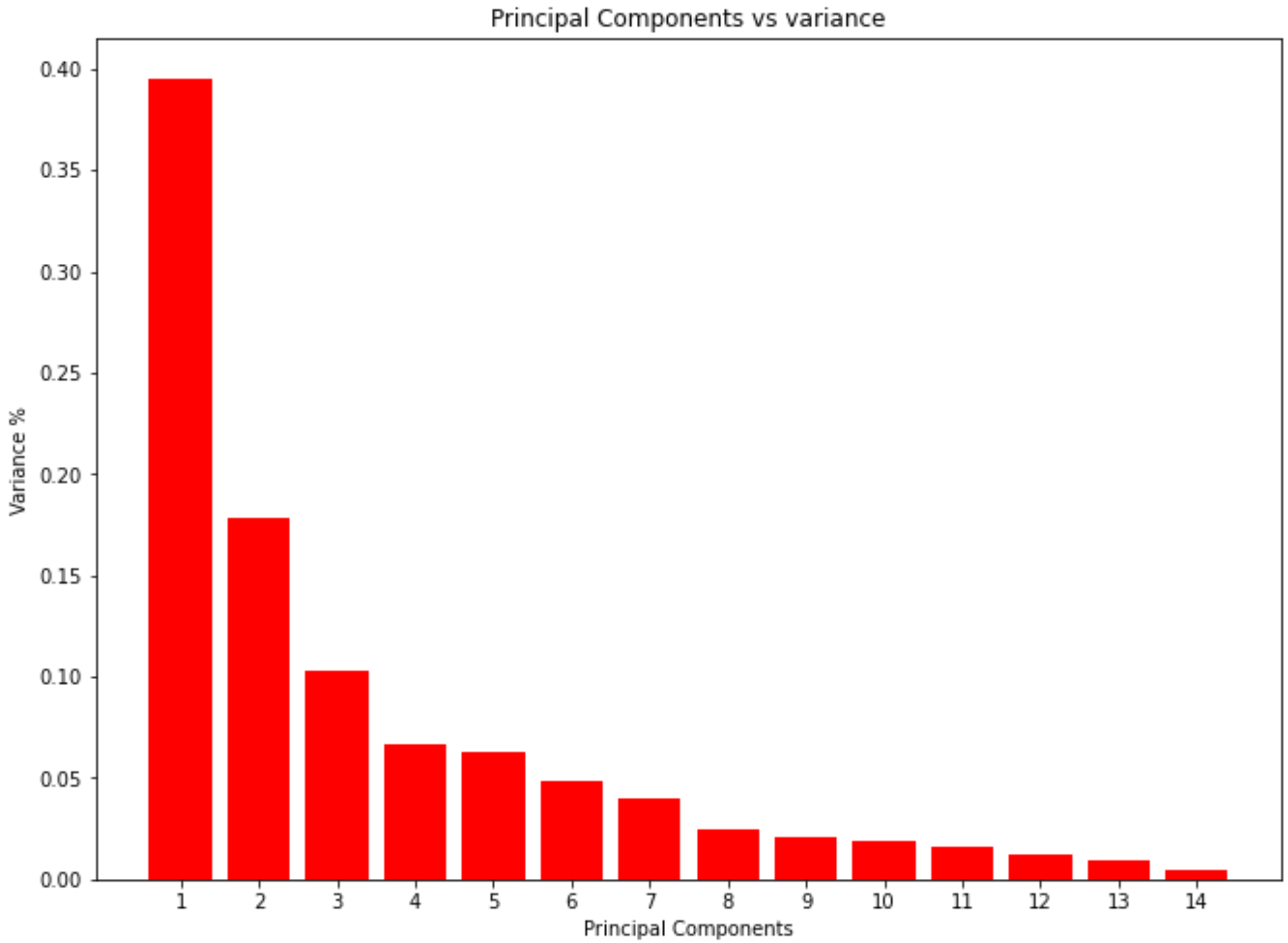
```
pca = PCA(n_components=14)
```

In [12]:

```
principal_components = pca.fit_transform(wine_norm)
```

In [18]:

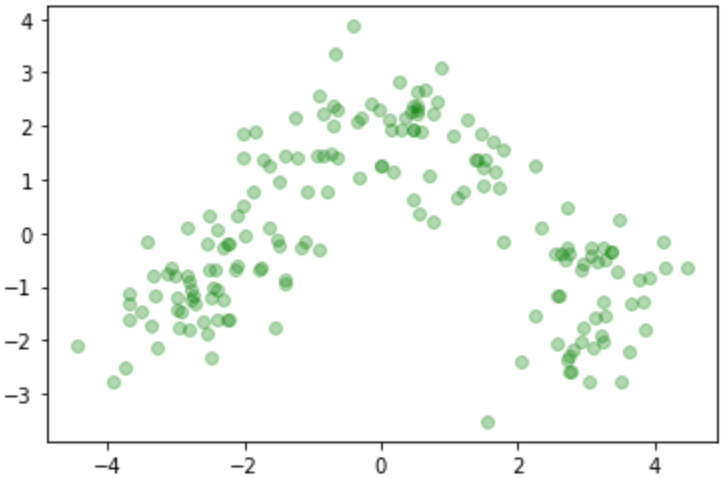
```
PC = range(1, pca.n_components_+1)
plt.figure(figsize=(11,8))
plt.bar(PC, pca.explained_variance_ratio_, color='red')
plt.xlabel('Principal Components')
plt.ylabel('Variance %')
plt.title('Principal Components vs variance')
plt.xticks(PC)
plt.show()
```



In [23]:
```
PCA_components = pd.DataFrame(principal_components)
```

In [25]:
```
plt.scatter(PCA_components[0],PCA_components[1],alpha=.3,color='green')
```

Out[25]:
```
<matplotlib.collections.PathCollection at 0x234246b7610>
```

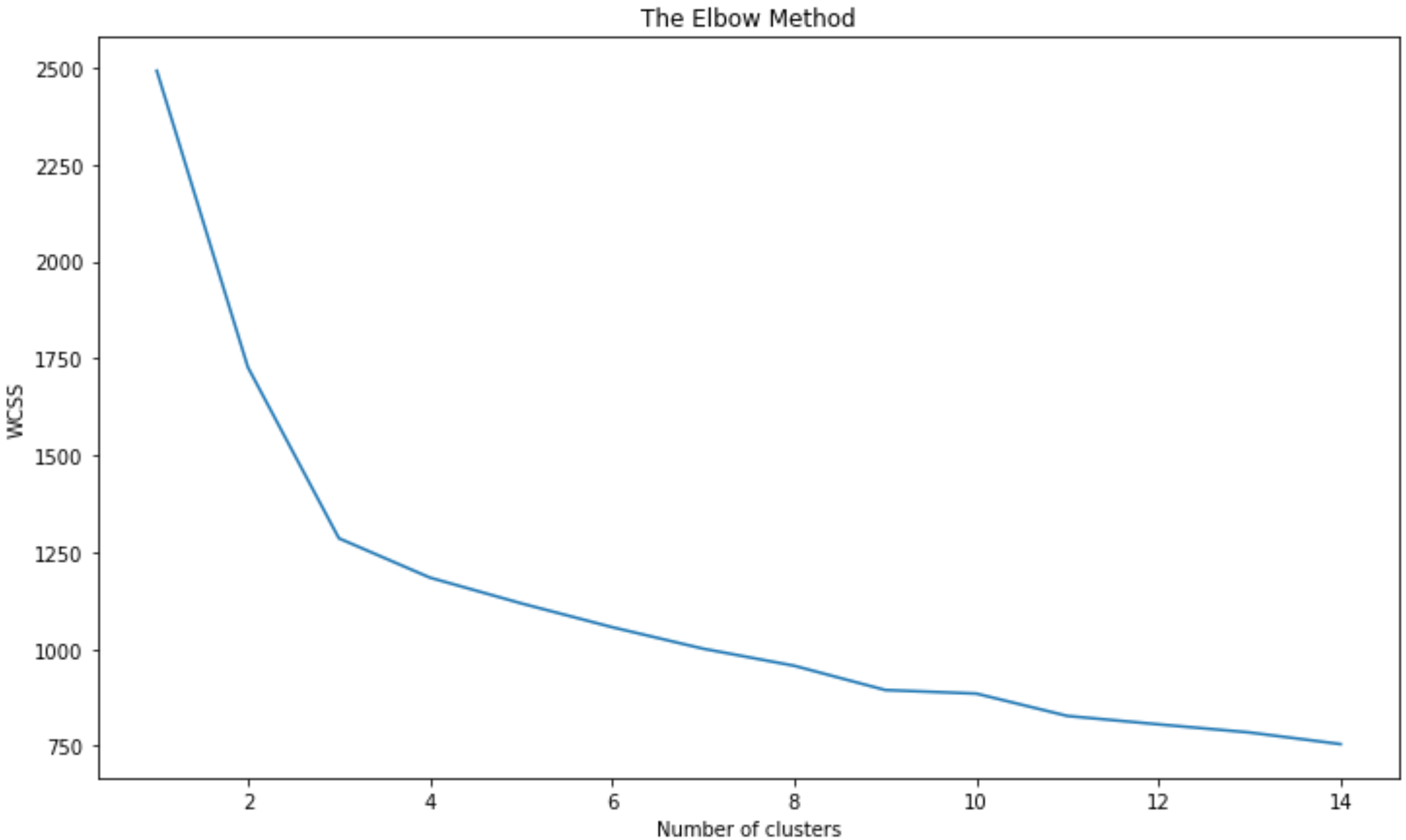


As shown in the bar graph, the most of varianve is put in the first 2 components. Since there is not much variance present from 3rd component, lets just the first 2 componets in our analysis. The scatter plot given an indication that there may be 3 clusters present

Finding out the optimal number of clusters

In [37]:

```
plt.figure(figsize=(12, 7))
wcss = []
for i in range(1, 15):
    kmeans = KMeans(n_clusters = i, init = 'k-means++', random_state = 42)
    kmeans.fit(wine_norm)
    wcss.append(kmeans.inertia_)
plt.plot(range(1, 15), wcss)
plt.title('The Elbow Method')
plt.xlabel('Number of clusters')
plt.ylabel('WCSS')
plt.show()
```



The scree plot levels off at k=3 and let's use it to determine the clusters

K-mean clustering

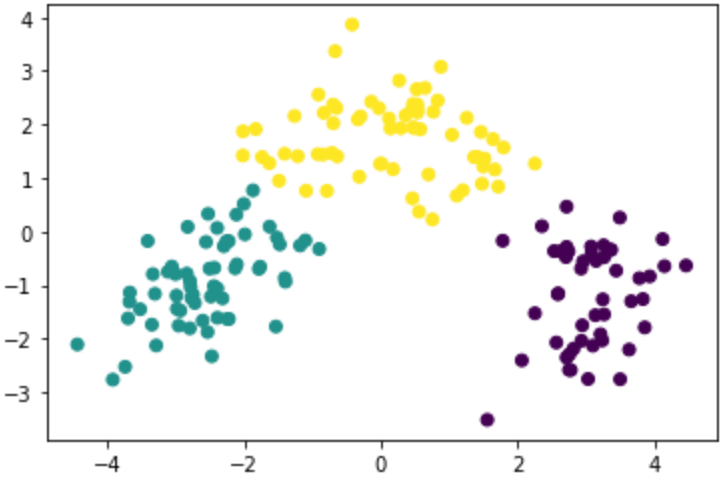
In [40]:
```
for i in range(1, 15):
    kmeans = KMeans(n_clusters = i, init = 'k-means++', random_state = 42)
    kmeans.fit(PCA_components.iloc[:,:3])
    wcss.append(kmeans.inertia_)
```

In [41]:
```
k_model = KMeans(n_clusters=3)
k_model.fit(PCA_components.iloc[:,:2])
```

Out[41]:
```
KMeans(n_clusters=3)
```

In [43]:
```
labels = k_model.predict(PCA_components.iloc[:,:2])
```

In [44]:
```
plt.scatter(PCA_components[0], PCA_components[1], c=labels)
plt.show()
```



In [46]:

```
k_new_data=pd.DataFrame(principal_components[:,0:2])
model_k = KMeans(n_clusters=3)
model_k.fit(k_new_data)
```

Out[46]:

```
KMeans(n_clusters=3)
```

In [47]:

```
model_k.labels_
```

Out[47]:

```
array([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, 0, 0, 0, 0, 0, 0, 2, 2, 2, 2, 2, 2, 2,
       2, 2, 2, 2, 2, 2, 2, 0, 2, 2, 2, 2, 2, 2, 2, 2, 2, 1, 2, 2, 2, 2,
       2, 2, 2, 2, 2, 2, 2, 0, 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, 2, 2, 2, 2, 2, 2, 2, 2, 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])
```

In [49]:

```
model_data = pd.Series(model_k.labels_)
wine_data['cluster']=model_data
```

In [50]:

```
k_new_data.head()
```

Out[50]:

	0	1
0	-3.522934	-1.453098
1	-2.528858	0.330019
2	-2.785029	-1.036936
3	-3.922588	-2.768210
4	-1.407511	-0.867773

In [51]:

```
wine_data.groupby(wine_data.cluster).mean()
```

Out[51]:

cluster	Type	Alcohol	Malic	Ash	Alcalinity	Magnesium	Phenols	Flavanoids	Nonflavanoids	Proanthocyanins	Color	Hue	Dilu
0	1.048387	13.676774	1.997903	2.466290	17.462903	107.967742	2.847581	3.003226	0.292097	1.922097	5.453548	1.065484	3.16
1	2.979592	13.151633	3.344490	2.434694	21.438776	99.020408	1.678163	0.797959	0.450816	1.163061	7.343265	0.685918	1.69
2	2.000000	12.264478	1.912239	2.224328	19.953731	92.656716	2.235075	2.028507	0.361343	1.597313	3.020896	1.056060	2.77

Hierarchical clustering

In [53]:

```
model_2 = AgglomerativeClustering(n_clusters=3, affinity='euclidean', linkage='ward')
h_clusters = model_2.fit(PCA_components.iloc[:,:2])
```

In [54]:

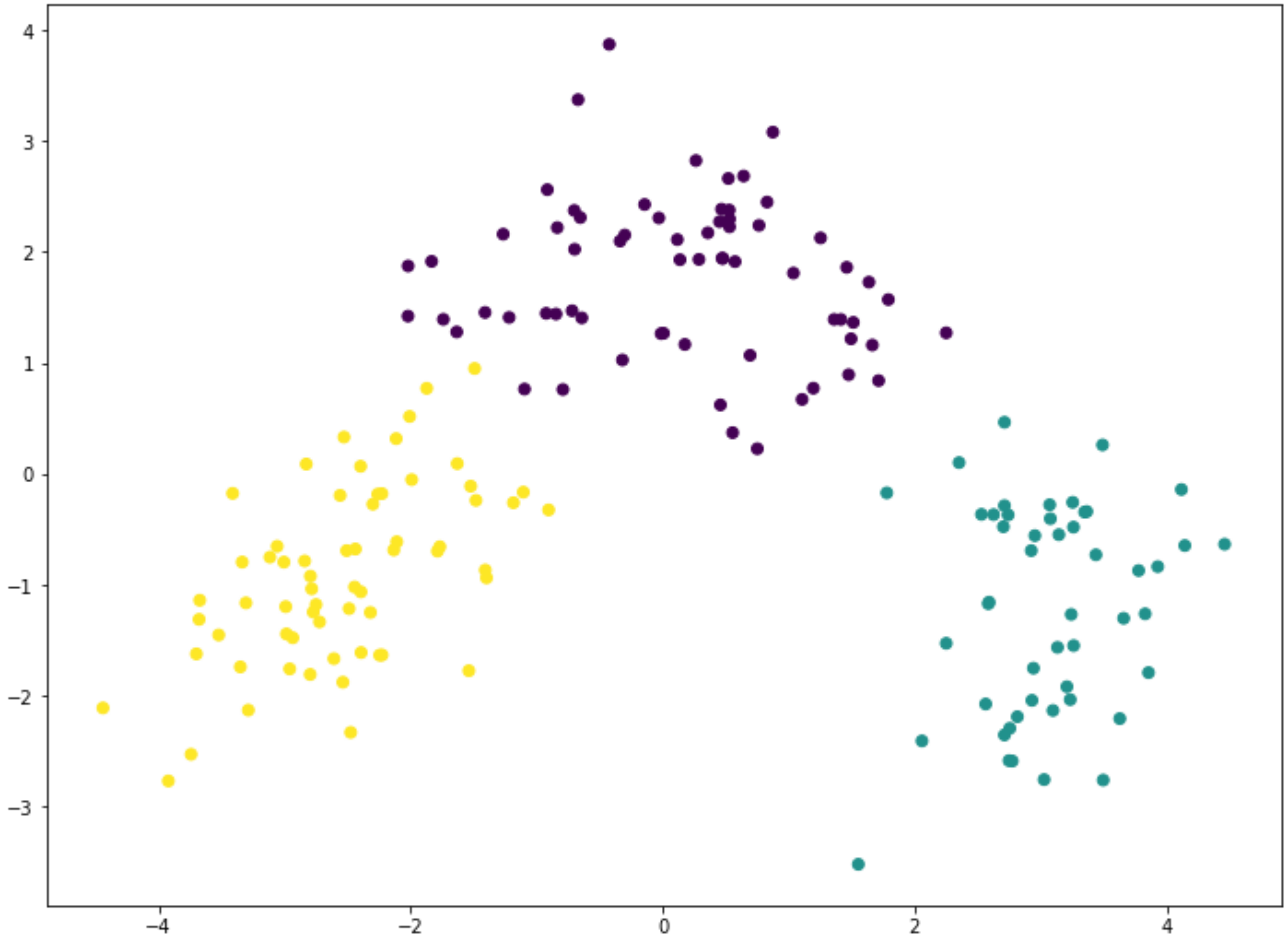
```
label_2 = model_2.labels_
```

In [55]:

```
x = PCA_components.iloc[:,:1]
y = PCA_components.iloc[:,1:2]
```

In [58]:

```
plt.figure(figsize=(12, 9))
plt.scatter(x, y, c=label_2)
plt.show()
```



In [60]:

```
h_new_data_2=pd.DataFrame(principal_components[:,0:2])
h_new_data_2.head()
```

Out[60]:

	0	1
0	-3.522934	-1.453098
1	-2.528858	0.330019

	0	1
2	-2.785029	-1.036936
3	-3.922588	-2.768210

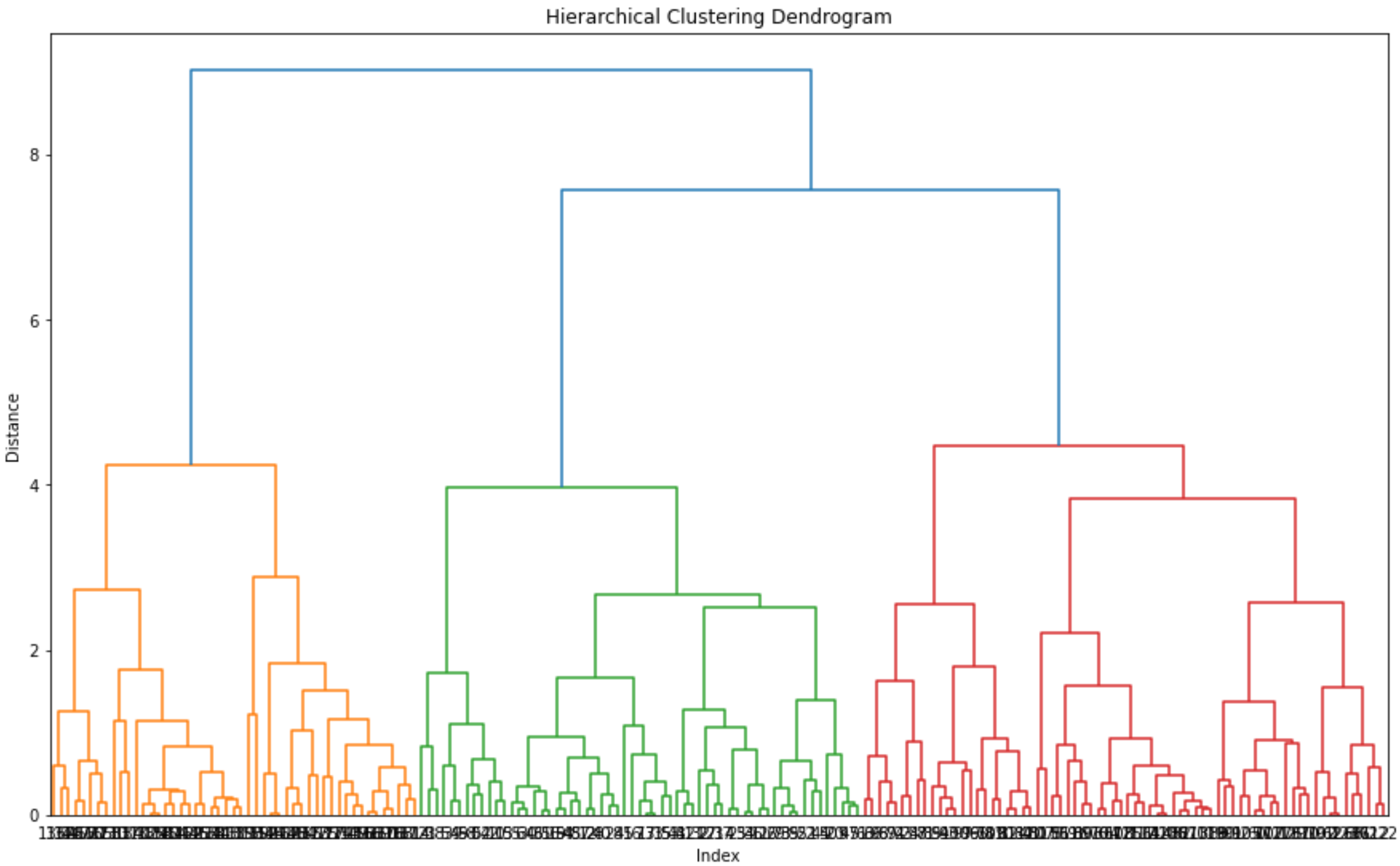
Hierarchical Clustering Dendrogram

In [61]:

```
hcf = linkage(h_new_data_2,method="complete",metric="euclidean")
```

In [65]:

```
plt.figure(figsize=(15, 9));plt.title('Hierarchical Clustering Dendrogram');plt.xlabel('Index');plt.ylabel('Distance
sch.dendrogram( hcf, leaf_rotation=0., leaf_font_size=10.,)
plt.show()
```



In [67]:

```
h_complete = AgglomerativeClustering(n_clusters=5,linkage='complete',affinity = "euclidean").fit(h_new_data_2)
h_complete.labels_
```

Out[67]:

```
array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 3, 3, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 3, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 3, 1, 3,
       3, 1, 1, 3, 1, 3, 1, 0, 3, 1, 1, 1, 3, 1, 1, 3, 1, 4, 3, 3, 1, 1,
```

```
       1, 1, 1, 1, 1, 3, 3, 0, 1, 3, 3, 3, 3, 1, 1, 1, 1, 1, 1, 1, 1, 3,
       3, 1, 1, 1, 1, 1, 1, 1, 1, 1, 3, 0, 1, 1, 3, 1, 1, 1, 1, 1, 4, 4,
       4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 2, 4, 4, 4, 2, 2, 2, 2, 2, 2,
       4, 2, 2, 4, 2, 2, 4, 2, 4, 4, 2, 4, 2, 2, 2, 2, 4, 4, 2, 2, 2, 2,
       2, 2], dtype=int64)
```

In [68]:

```python
cluster_label = pd.Series(h_complete.labels_)
```

In [72]:

```python
wine_data['cluster']=cluster_label
```

In [74]:

```python
wine_data.head()
```

Out[74]:

	Type	Alcohol	Malic	Ash	Alcalinity	Magnesium	Phenols	Flavanoids	Nonflavanoids	Proanthocyanins	Color	Hue	Dilution	Proline	cluster
0	1	14.23	1.71	2.43	15.6	127	2.80	3.06	0.28	2.29	5.64	1.04	3.92	1065	0
1	1	13.20	1.78	2.14	11.2	100	2.65	2.76	0.26	1.28	4.38	1.05	3.40	1050	0
2	1	13.16	2.36	2.67	18.6	101	2.80	3.24	0.30	2.81	5.68	1.03	3.17	1185	0
3	1	14.37	1.95	2.50	16.8	113	3.85	3.49	0.24	2.18	7.80	0.86	3.45	1480	0
4	1	13.24	2.59	2.87	21.0	118	2.80	2.69	0.39	1.82	4.32	1.04	2.93	735	0

Using PCA we reduced the variables to only 2 from 13 and use clustering classification, we can safely assume that there exists 3 cluster in the wine data sets

In [ ]:

```python

```