Objective: To understand the importance of scaling on PCA

In [33]:

```
from sklearn.decomposition import PCA
from sklearn import preprocessing
from sklearn import metrics
from scipy import linalg as LA
import numpy as np
import matplotlib.pyplot as plt
from sklearn.datasets import load_wine
```

Task 0: Write the function to compute the pca using Eigenvector approach

In [34]:

```
from numpy.linalg import svd
def pca(X):
    cov = np.cov(X, rowvar = False)
    evals , P = LA.eigh(cov)
    idx = np.argsort(evals)[::-1]
    P = P[:,idx]
    evals = evals[idx]
    T = np.dot(X, P)
    Sigma=LA.norm(T,axis=0)
    return T, Sigma, P #Score, Variace, Loadings
```

In [35]:

```
features, target = load_wine(return_X_y=True)
X=features
y=target
```

Three different ways of scaling

- Scaling by removing the mean and divising by the standard deviation

```
#standard_scaling=preprocessing.StandardScaler()
#X_standard=standard_scaling.fit_transform(X)
```

- Scaling to min and maximum values of each feature

```
#minmax_scaling=preprocessing.MinMaxScaler()
#X_minmax=minmax_scaling.fit_transform(X)
```

- Scaling by diving by the maximumabsolute values of each features

```
#max_abs_scaler=preprocessing.MaxAbsScaler()
#X_maxabs=max_abs_scaler.fit_transform(X)
```

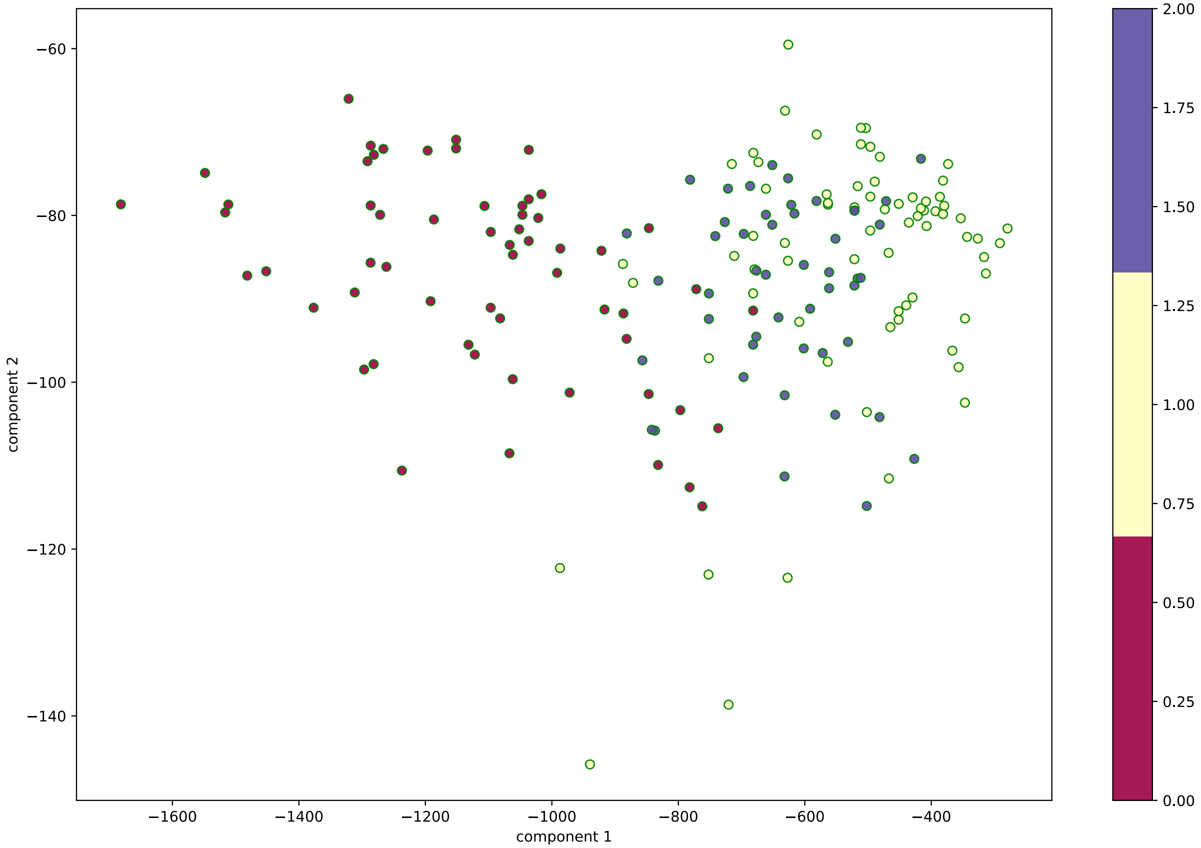
Task 1: Create the scores plot without any scaling

In [36]:

```
T,S,P=pca(X)
plt.figure(figsize=(15,10))
plt.scatter(T[:, 0], T[:, 1],
            c=y, edgecolor='green', alpha=0.9,
            cmap=plt.cm.get_cmap('Spectral', 3))
plt.xlabel('component 1')
plt.ylabel('component 2')
plt.colorbar()
```

Out[36]:

```
<matplotlib.colorbar.Colorbar at 0x2ac16b025e0>
```



Task 2: Create the scores plot without any standard scaling.

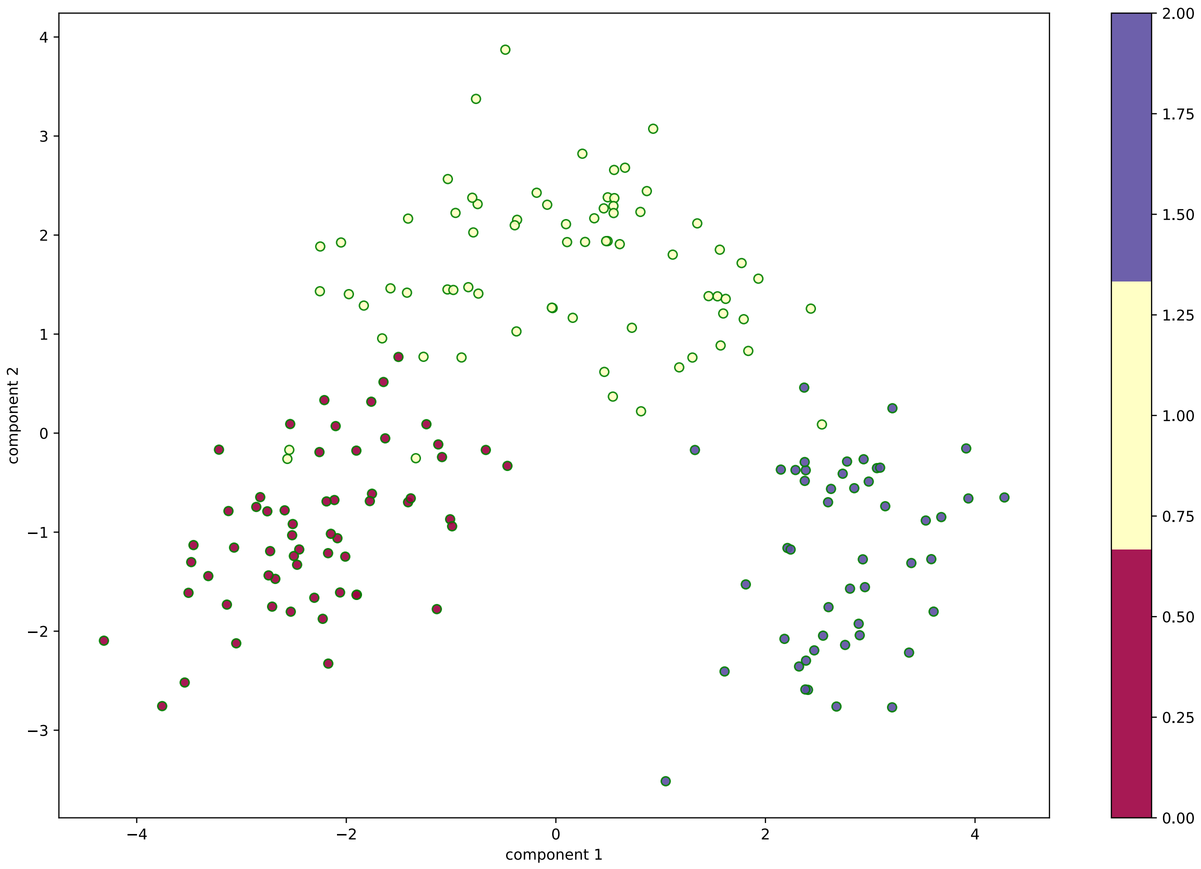
In [40]:
```
#I assume we are suppose to create plot WITH standard scaling

standard_scaling=preprocessing.StandardScaler()
X_standard=standard_scaling.fit_transform(X)

T,S,P=pca(X_standard)
plt.figure(figsize=(15,10))
plt.scatter(T[:, 0], T[:, 1],
            c=y, edgecolor='green', alpha=0.9,
            cmap=plt.cm.get_cmap('Spectral', 3))
plt.xlabel('component 1')
plt.ylabel('component 2')
plt.colorbar()
```

Out[40]:

```
<matplotlib.colorbar.Colorbar at 0x2ac145df940>
```



Task 3: Create the scores plot without any min max scaling

In [41]:

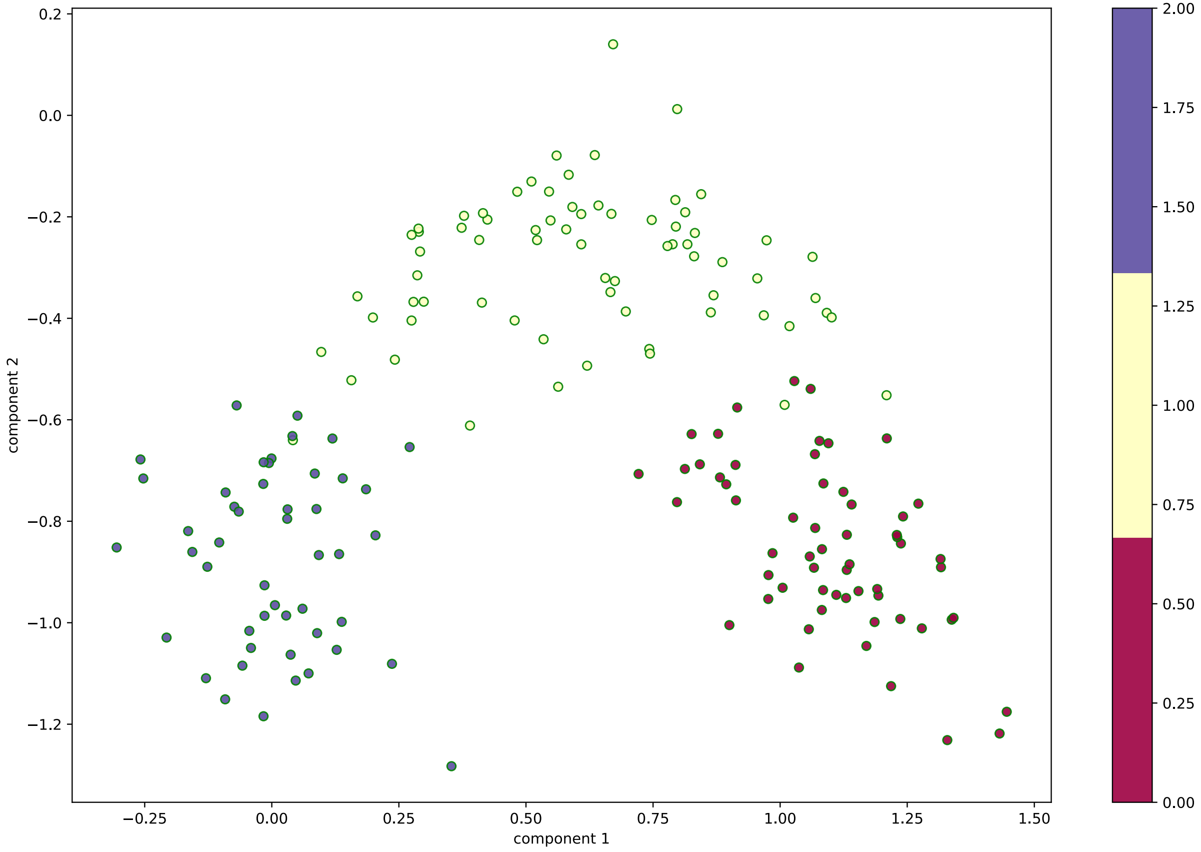
```
#I assume we are suppse to make plot WITH min max scaling
#min max scalinng
minmax_scaling=preprocessing.MinMaxScaler()
X_minmax=minmax_scaling.fit_transform(X)



T,S,P=pca(X_minmax)
plt.figure(figsize=(15,10))
plt.scatter(T[:, 0], T[:, 1],
            c=y, edgecolor='green', alpha=0.9,
            cmap=plt.cm.get_cmap('Spectral', 3))
plt.xlabel('component 1')
plt.ylabel('component 2')
plt.colorbar()
```

Out[41]:

```
<matplotlib.colorbar.Colorbar at 0x2ac169e03d0>
```



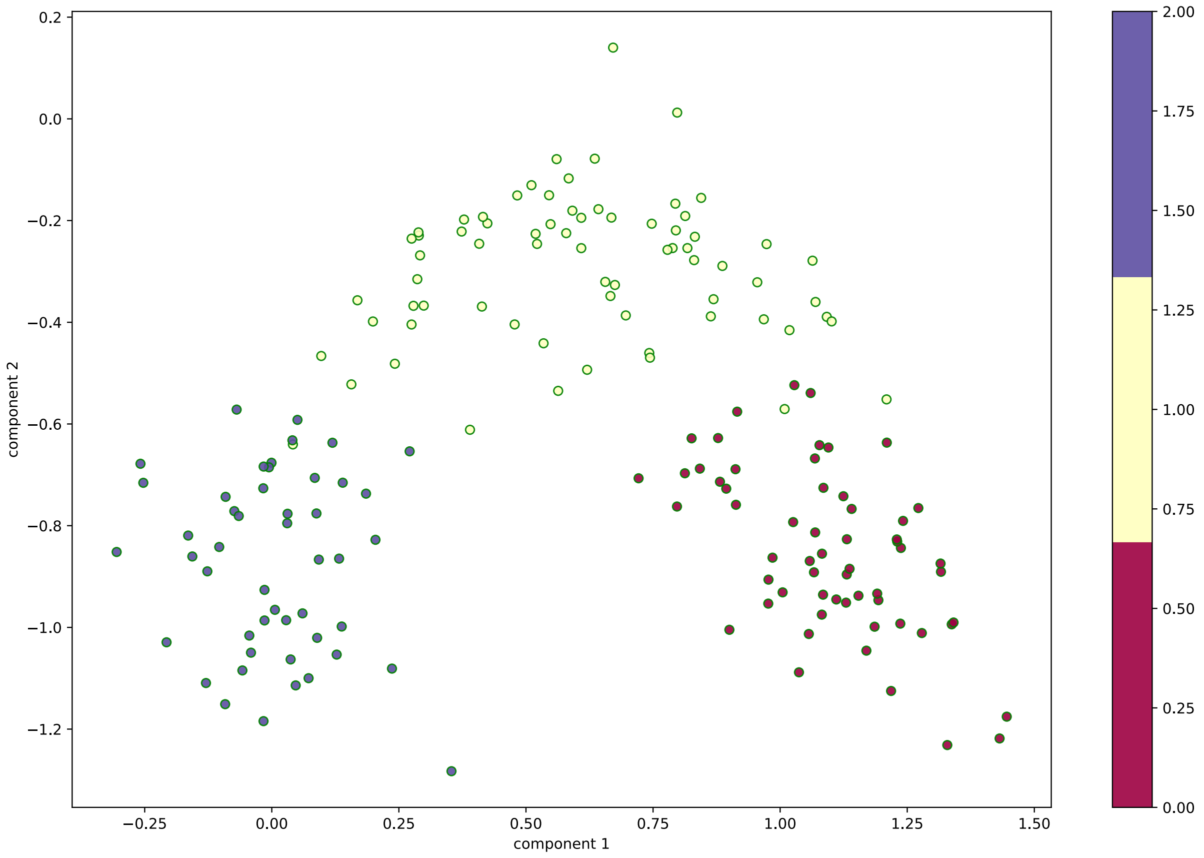
Task 4: Create the scores plot without any max abs scaling scaling

In [42]:

```
#I assume we are suppose to create plot WITH max abs scaling
max_abs_scaler=preprocessing.MaxAbsScaler()
X_maxabs=max_abs_scaler.fit_transform(X_minmax)

T,S,P=pca(X_maxabs)
plt.figure(figsize=(15,10))
plt.scatter(T[:, 0], T[:, 1],
            c=y, edgecolor='green', alpha=0.9,
            cmap=plt.cm.get_cmap('Spectral', 3))
plt.xlabel('component 1')
plt.ylabel('component 2')
plt.colorbar()
```

Out[42]:
```
<matplotlib.colorbar.Colorbar at 0x2ac176c7940>
```



In [ ]: