Implementation of PCA

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
```

In [3]:
```
from sklearn.decomposition import PCA
from sklearn.preprocessing import StandardScaler
```

In [5]:
```
from sklearn.datasets import load_breast_cancer
data=load_breast_cancer()
```

In [7]:
```
data.keys()
```

Out[7]:
```
dict_keys(['data', 'target', 'frame', 'target_names', 'DESCR', 'feature_names', 'filename', 'data_module'])
```

In [9]:
```
print(data['target_names'])
```

```
['malignant' 'benign']
```

In [11]:
```
print(data['feature_names'])
```

```
['mean radius' 'mean texture' 'mean perimeter' 'mean area'
 'mean smoothness' 'mean compactness' 'mean concavity'
 'mean concave points' 'mean symmetry' 'mean fractal dimension'
 'radius error' 'texture error' 'perimeter error' 'area error'
 'smoothness error' 'compactness error' 'concavity error'
 'concave points error' 'symmetry error' 'fractal dimension error'
 'worst radius' 'worst texture' 'worst perimeter' 'worst area'
 'worst smoothness' 'worst compactness' 'worst concavity'
 'worst concave points' 'worst symmetry' 'worst fractal dimension']
```

In [13]:
```
df1=pd.DataFrame(data['data'],columns=data['feature_names'])
```

In [15]:
```
scaling=StandardScaler()
```

In [17]:
```
scaling.fit(df1)
Scaled_data=scaling.transform(df1)
```

In [19]:
```
principal=PCA(n_components=3)
principal.fit(Scaled_data)
x=principal.transform(Scaled_data)
```

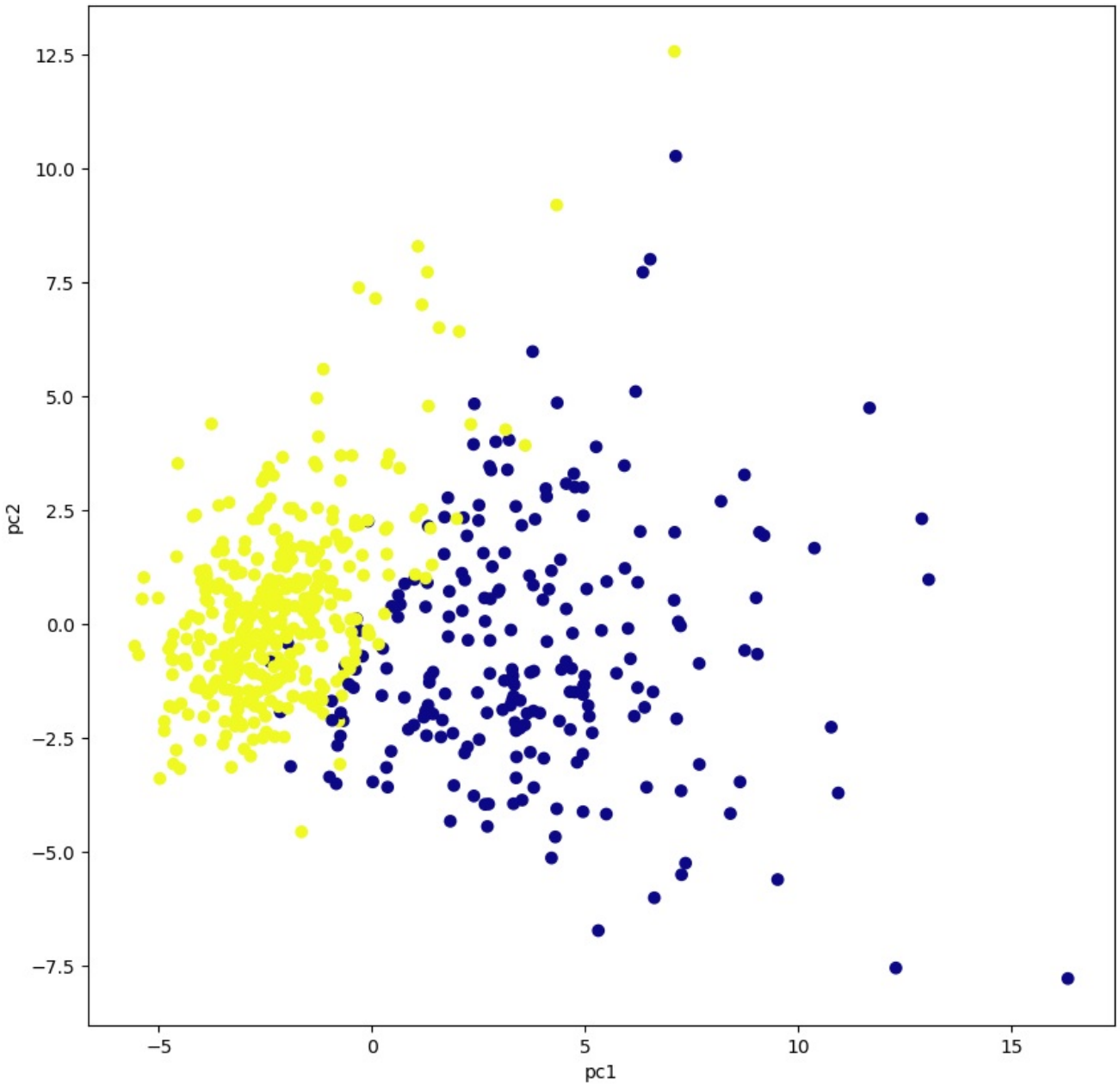
In [21]:
```
print(x.shape)
```

```
(569, 3)
```

In [23]:
```
principal.components_
plt.figure(figsize=(10,10))
plt.scatter(x[:,0],x[:,1],c=data['target'],cmap='plasma')
plt.xlabel('pc1')
plt.ylabel('pc2')
```

Out[23]:
```
Text(0, 0.5, 'pc2')
```

In [25]:
```
plt.show()
```



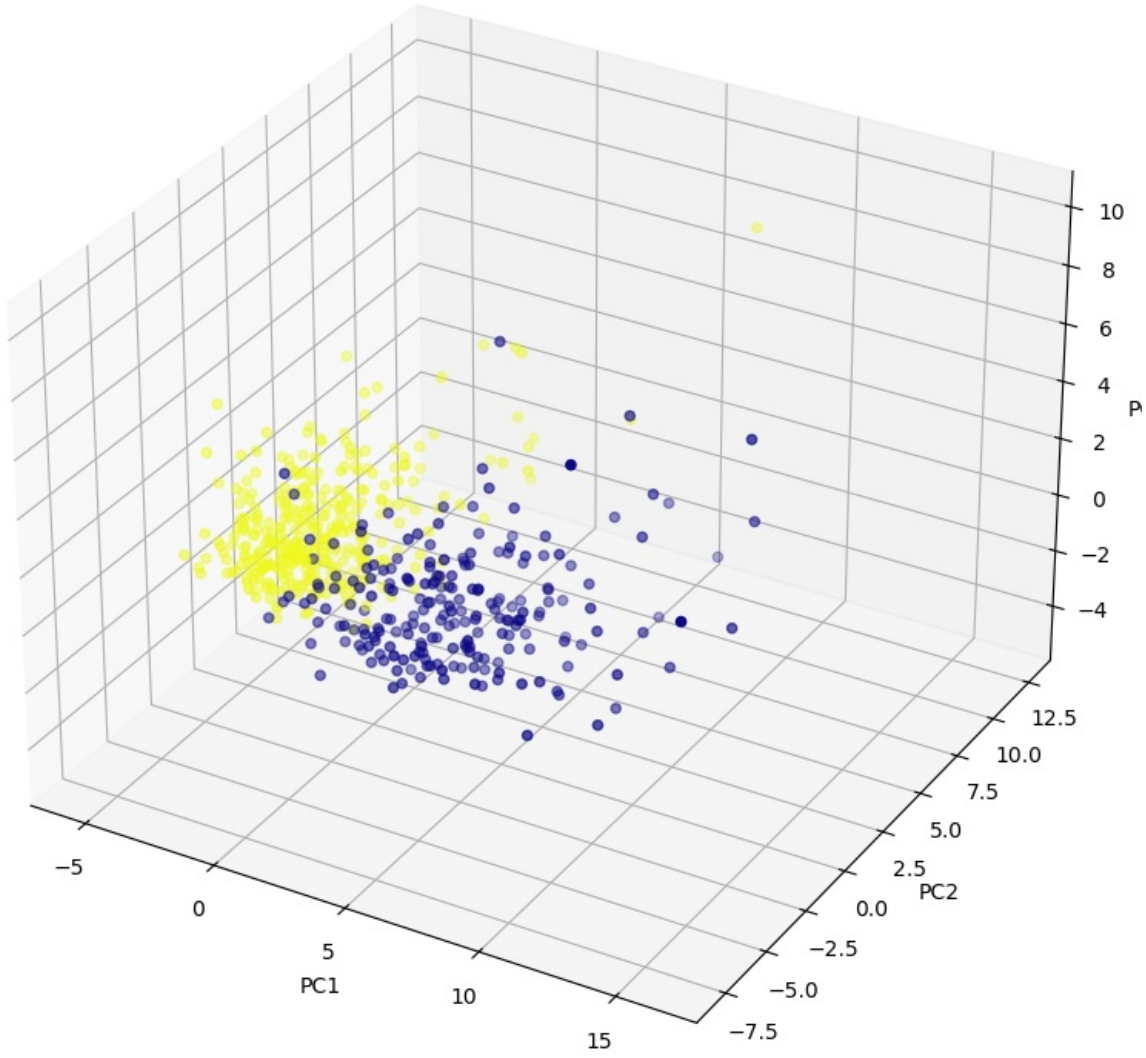
In [27]:
```
from mpl_toolkits.mplot3d import Axes3D
fig = plt.figure(figsize=(10,10))
```

In [30]:
```
axis = fig.add_subplot(111, projection='3d')
```

In [32]:
```
# x[:,0]is pc1,x[:,1] is pc2 while x[:,2] is pc3
axis.scatter(x[:,0],x[:,1],x[:,2], c=data['target'],cmap='plasma')
axis.set_xlabel("PC1", fontsize=10)
axis.set_ylabel("PC2", fontsize=10)
axis.set_zlabel("PC3", fontsize=10)
```

Out[32]:
```
Text(0.09332506951644345, 0.012503188505254661, 'PC3')
```

In [36]:
```
plt.show()
```



In [39]:
```
print(principal.explained_variance_ratio_)
```

```
[0.44272026 0.18971182 0.09393163]
```

Conclusion

1) PCA reduces many features into a few important ones while keeping most of the information.

2) It helps in making data easier to understand, visualize, and use for machine learning.