```
from sklearn.datasets import load_breast_cancer
import pandas as pd
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.decomposition import PCA
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
import seaborn as sns


cancer = load_breast_cancer()


df2 = pd.Series(cancer['target'])


df2
```

```
0      0
1      0
2      0
3      0
4      0
      ..
564    0
565    0
566    0
567    0
568    1
Length: 569, dtype: int64
```

```
df = pd.DataFrame(cancer['data'], columns = cancer['feature_names'])


df = df.merge(df2.rename('result'), left_index=True, right_index=True)


df.head()
```

| | mean radius | mean texture | mean perimeter | mean area | mean smoothness | mean compactness | mean concavity | mean concave points | mean symmetry | mean fractal dimension | ... | worst texture | wo perime |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 17.99 | 10.38 | 122.80 | 1001.0 | 0.11840 | 0.27760 | 0.3001 | 0.14710 | 0.2419 | 0.07871 | ... | 17.33 | 18 |
| **1** | 20.57 | 17.77 | 132.90 | 1326.0 | 0.08474 | 0.07864 | 0.0869 | 0.07017 | 0.1812 | 0.05667 | ... | 23.41 | 15 |
| **2** | 19.69 | 21.25 | 130.00 | 1203.0 | 0.10960 | 0.15990 | 0.1974 | 0.12790 | 0.2069 | 0.05999 | ... | 25.53 | 15 |
| **3** | 11.42 | 20.38 | 77.58 | 386.1 | 0.14250 | 0.28390 | 0.2414 | 0.10520 | 0.2597 | 0.09744 | ... | 26.50 | 9 |
| **4** | 20.29 | 14.34 | 135.10 | 1297.0 | 0.10030 | 0.13280 | 0.1980 | 0.10430 | 0.1809 | 0.05883 | ... | 16.67 | 15 |

5 rows × 31 columns

```
df.isnull().sum()
```

```
mean radius               0
mean texture              0
mean perimeter            0
mean area                 0
mean smoothness           0
mean compactness          0
mean concavity            0
mean concave points       0
mean symmetry             0
mean fractal dimension    0
radius error              0
texture error             0
perimeter error           0
area error                0
smoothness error          0
compactness error         0
concavity error           0
concave points error      0
symmetry error            0
```

```
fractal dimension error    0
worst radius               0
worst texture              0
worst perimeter            0
worst area                 0
worst smoothness           0
worst compactness          0
worst concavity            0
worst concave points       0
worst symmetry             0
worst fractal dimension    0
result                     0
dtype: int64
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 569 entries, 0 to 568
Data columns (total 31 columns):
 #   Column                   Non-Null Count  Dtype  
---  ------                   --------------  -----  
 0   mean radius              569 non-null    float64
 1   mean texture             569 non-null    float64
 2   mean perimeter           569 non-null    float64
 3   mean area                569 non-null    float64
 4   mean smoothness          569 non-null    float64
 5   mean compactness         569 non-null    float64
 6   mean concavity           569 non-null    float64
 7   mean concave points      569 non-null    float64
 8   mean symmetry            569 non-null    float64
 9   mean fractal dimension   569 non-null    float64
 10  radius error             569 non-null    float64
 11  texture error            569 non-null    float64
 12  perimeter error          569 non-null    float64
 13  area error               569 non-null    float64
 14  smoothness error         569 non-null    float64
 15  compactness error        569 non-null    float64
 16  concavity error          569 non-null    float64
 17  concave points error     569 non-null    float64
 18  symmetry error           569 non-null    float64
 19  fractal dimension error  569 non-null    float64
 20  worst radius             569 non-null    float64
 21  worst texture            569 non-null    float64
 22  worst perimeter          569 non-null    float64
 23  worst area               569 non-null    float64
 24  worst smoothness         569 non-null    float64
 25  worst compactness        569 non-null    float64
 26  worst concavity          569 non-null    float64
 27  worst concave points     569 non-null    float64
 28  worst symmetry           569 non-null    float64
 29  worst fractal dimension  569 non-null    float64
 30  result                   569 non-null    int64  
dtypes: float64(30), int64(1)
memory usage: 137.9 KB
```

```
X = df.drop(['result'], axis=1)
y = df['result']
```

```
# Standardize the features
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
```

```
X_scaled.shape
```

```
(569, 30)
```

```
# Perform PCA with 5 components
pca = PCA(n_components=5)
X_pca = pca.fit_transform(X_scaled)
```

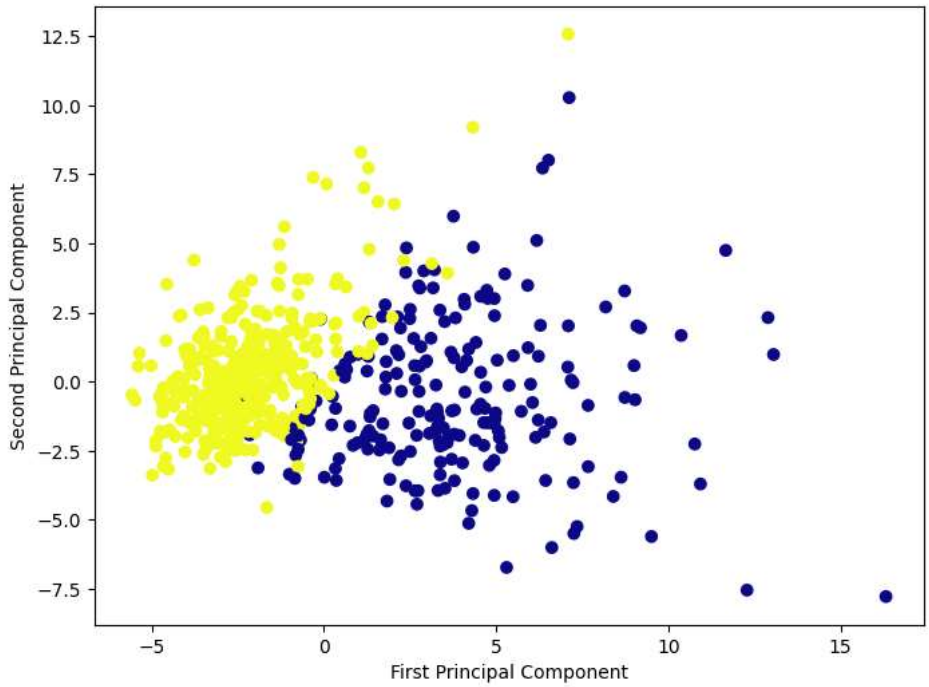
```
X_pca.shape
```

```
(569, 5)
```

```
# Create a scatter plot
plt.figure(figsize =(8, 6))
plt.scatter(X_pca[:, 0], X_pca[:, 1], c = df['result'], cmap ='plasma')
```

```
plt.xlabel('First Principal Component')
plt.ylabel('Second Principal Component')
```

```
Text(0, 0.5, 'Second Principal Component')
```

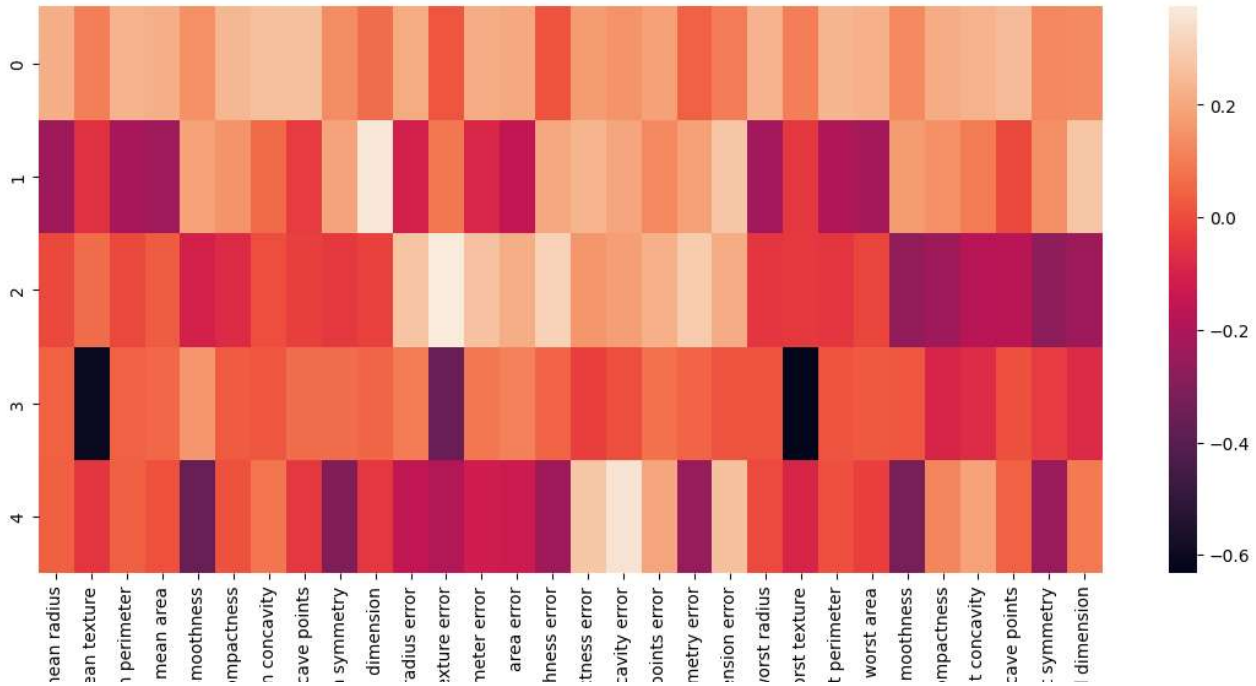



```
# components
pca.components_
```

```
array([[ 0.21890244,  0.10372458,  0.22753729,  0.22099499,  0.14258969,
         0.23928535,  0.25840048,  0.26085376,  0.13816696,  0.06436335,
         0.20597878,  0.01742803,  0.21132592,  0.20286964,  0.01453145,
         0.17039345,  0.15358979,  0.1834174 ,  0.04249842,  0.10256832,
         0.22799663,  0.10446933,  0.23663968,  0.22487053,  0.12795256,
         0.21009588,  0.22876753,  0.25088597,  0.12290456,  0.13178394],
       [-0.23385713, -0.05970609, -0.21518136, -0.23107671,  0.18611302,
         0.15189161,  0.06016536, -0.0347675 ,  0.19034877,  0.36657547,
        -0.10555215,  0.08997968, -0.08945723, -0.15229263,  0.20443045,
         0.2327159 ,  0.19720728,  0.13032156,  0.183848  ,  0.28009203,
        -0.21986638, -0.0454673 , -0.19987843, -0.21935186,  0.17230435,
         0.14359317,  0.09796411, -0.00825724,  0.14188335,  0.27533947],
       [-0.00853124,  0.06454991, -0.00931422,  0.02869953, -0.1042919 ,
        -0.07409158,  0.00273383, -0.02556356, -0.04023993, -0.02257408,
         0.26848138,  0.37463367,  0.26664537,  0.21600653,  0.30883898,
         0.15477971,  0.17646375,  0.22465758,  0.2885843 ,  0.21150375,
        -0.04750699, -0.04229783, -0.04854651, -0.01190231, -0.25979762,
        -0.23607562, -0.17305733, -0.17034408, -0.27131265, -0.2327913 ],
       [ 0.04140896, -0.60305001,  0.0419831 ,  0.0534338 ,  0.15938277,
         0.03179458,  0.01912276,  0.06533596,  0.06712498,  0.04858676,
         0.09794124, -0.35985553,  0.08899241,  0.10820506,  0.04466418,
        -0.02746935,  0.00131687,  0.07406733,  0.04407334,  0.01530476,
         0.01541723, -0.63280788,  0.01380278,  0.02589474,  0.01765221,
        -0.09132842, -0.07395119,  0.006007  , -0.03625068, -0.07705348],
       [ 0.0377863 , -0.04946892,  0.03737462,  0.0103312 , -0.36508859,
         0.01170403,  0.08637554, -0.04386084, -0.30594151, -0.04442442,
        -0.15445647, -0.19165052, -0.12099021, -0.12757439, -0.2320657 ,
         0.27996825,  0.35398202,  0.19554799, -0.25286885,  0.26329758,
        -0.00440665, -0.09288331,  0.0074541 , -0.02739098, -0.32443539,
         0.12180404,  0.18851863,  0.04333215, -0.24455853,  0.09442318]])
```

```
df_comp = pd.DataFrame(pca.components_, columns = cancer['feature_names'])
plt.figure(figsize =(14, 6))
sns.heatmap(df_comp)
```

```
<Axes: >
```




## Before PCA

```
X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.33, random_state=42)
```

```
X_train.shape
```

```
(381, 30)
```

```
from sklearn.linear_model import LinearRegression
```

```
lr = LinearRegression()
```

```
lr.fit(X_train, y_train)
```

```
LinearRegression()
```

```
y_pred = lr.predict(X_test)
```

```
from sklearn.metrics import mean_squared_error, r2_score, accuracy_score
```

```
# Calculate the mean squared error and R-squared value
mse = mean_squared_error(y_test, y_pred)
rmse = mean_squared_error(y_test, y_pred, squared=False)
r2 = r2_score(y_test, y_pred)
accuracy = lr.score(X_test, y_test)
```

```
# Print the results
print("Mean squared error: {:.2f}".format(mse))
print("Root mean squared error: {:.2f}".format(rmse))
print("R-squared value: {:.2f}".format(r2))
print("Accuracy is: {:.2f}".format(accuracy))
```

```
Mean squared error: 0.07
Root mean squared error: 0.27
R-squared value: 0.69
Accuracy is: 0.69
```

## After PCA

```
X_train_2, X_test_2, y_train_2, y_test_2 = train_test_split(X_pca, y, test_size=0.33, random_state=42)
```

```
X_train_2.shape
```

```
(381, 5)
```

```
lr.fit(X_train_2, y_train_2)
```

```
LinearRegression()
```

```
y_pred_2 = lr.predict(X_test_2)
```

```
mse = mean_squared_error(y_test_2, y_pred_2)
rmse = mean_squared_error(y_test_2, y_pred_2, squared=False)
r2 = r2_score(y_test_2, y_pred_2)
accuracy = lr.score(X_test_2, y_test_2)
```

```
print("Mean squared error: {:.2f}".format(mse))
print("Root mean squared error: {:.2f}".format(rmse))
print("R-squared value: {:.2f}".format(r2))
print("Accuracy is: {:.2f}".format(accuracy))
```

```
Mean squared error: 0.07
Root mean squared error: 0.26
R-squared value: 0.70
Accuracy is: 0.70
```