Register No: 21MIS1044

Name: Srinivasan JP

Principal Component Analysis

In [ ]:
```
#imports
import pandas as pd
import numpy as np
from sklearn.datasets import load_breast_cancer
import matplotlib.pyplot as plt
import seaborn as sns
```

Using the Breast cancer dataset which is build in form the sklearn module

In [ ]:
```
cancer = load_breast_cancer(as_frame=True)
df = cancer.frame
print(df.head())
```

```
   mean radius  mean texture  mean perimeter  mean area  mean smoothness  \
0        17.99         10.38          122.80     1001.0          0.11840
1        20.57         17.77          132.90     1326.0          0.08474
2        19.69         21.25          130.00     1203.0          0.10960
3        11.42         20.38           77.58      386.1          0.14250
4        20.29         14.34          135.10     1297.0          0.10030

   mean compactness  mean concavity  mean concave points  mean symmetry  \
0           0.27760          0.3001              0.14710         0.2419
1           0.07864          0.0869              0.07017         0.1812
2           0.15990          0.1974              0.12790         0.2069
3           0.28390          0.2414              0.10520         0.2597
4           0.13280          0.1980              0.10430         0.1809

   mean fractal dimension  ...  worst texture  worst perimeter  worst area  \
0                 0.07871  ...          17.33           184.60      2019.0
1                 0.05667  ...          23.41           158.80      1956.0
2                 0.05999  ...          25.53           152.50      1709.0
3                 0.09744  ...          26.50            98.87       567.7
4                 0.05883  ...          16.67           152.20      1575.0

   worst smoothness  worst compactness  worst concavity  worst concave points  \
0            0.1622             0.6656           0.7119                0.2654
1            0.1238             0.1866           0.2416                0.1860
2            0.1444             0.4245           0.4504                0.2430
3            0.2098             0.8663           0.6869                0.2575
4            0.1374             0.2050           0.4000                0.1625

   worst symmetry  worst fractal dimension  target
0          0.4601                  0.11890       0
1          0.2750                  0.08902       0
2          0.3613                  0.08758       0
3          0.6638                  0.17300       0
4          0.2364                  0.07678       0

[5 rows x 31 columns]
```

In the dataframe the target columns not going to reducted so the target columns are removed in X variable

In [ ]:
```
print('Original Dataframe shape :',df.shape)
X = df[cancer['feature_names']]
print('Inputs Dataframe shape :', X.shape)
# form the dataframes the target value is removed
```

```
Original Dataframe shape : (569, 31)
Inputs Dataframe shape : (569, 30)
```

To Standerdise the data we have to find the mean and standard deveation of the data set

In [ ]:
```
# Mean
X_mean = X.mean()

# Standard deviation
X_std = X.std()

# Standardization
Z = (X - X_mean) / X_std
print(f"Mean:\n{X_mean}",f"Standard Deveation:\n{X_std}",f"Standardizied Z value:\n{Z}",sep="\n\n\n")
```

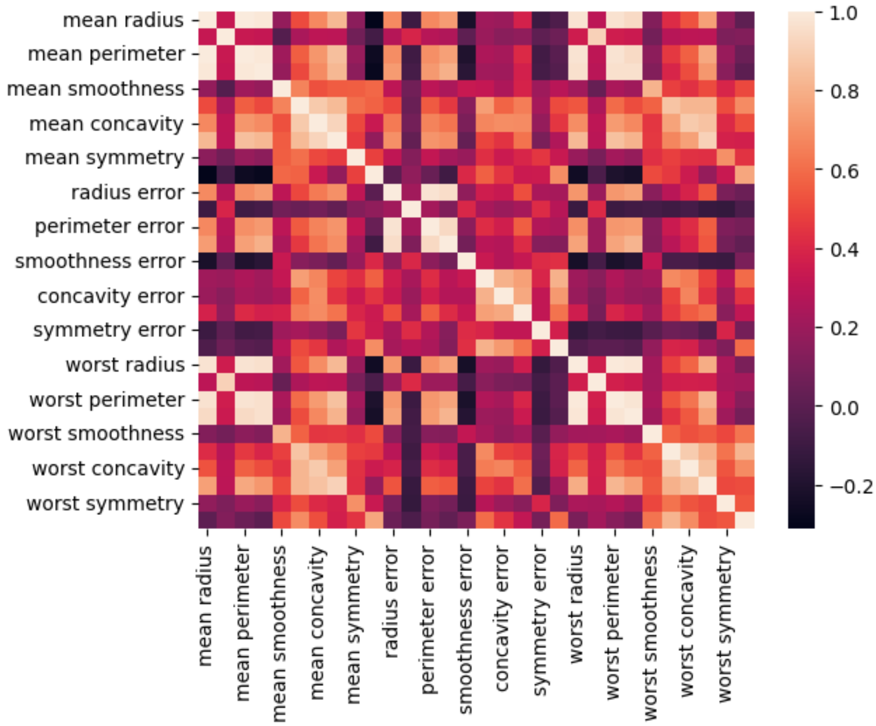
```
Mean:
mean radius                 14.127292
mean texture                19.289649
mean perimeter              91.969033
mean area                  654.889104
mean smoothness              0.096360
mean compactness             0.104341
mean concavity               0.088799
mean concave points          0.048919
mean symmetry                0.181162
mean fractal dimension       0.062798
radius error                 0.405172
texture error                1.216853
perimeter error              2.866059
area error                  40.337079
smoothness error             0.007041
compactness error            0.025478
concavity error              0.031894
concave points error         0.011796
symmetry error               0.020542
fractal dimension error      0.003795
worst radius                16.269190
worst texture               25.677223
worst perimeter            107.261213
worst area                 880.583128
worst smoothness             0.132369
worst compactness            0.254265
worst concavity              0.272188
worst concave points         0.114606
worst symmetry               0.290076
worst fractal dimension      0.083946
dtype: float64


Standard Deveation:
mean radius                  3.524049
mean texture                 4.301036
mean perimeter              24.298981
mean area                  351.914129
mean smoothness              0.014064
mean compactness             0.052813
mean concavity               0.079720
mean concave points          0.038803
mean symmetry                0.027414
mean fractal dimension       0.007060
radius error                 0.277313
texture error                0.551648
perimeter error              2.021855
area error                  45.491006
smoothness error             0.003003
compactness error            0.017908
concavity error              0.030186
concave points error         0.006170
symmetry error               0.008266
fractal dimension error      0.002646
worst radius                 4.833242
worst texture                6.146258
worst perimeter             33.602542
worst area                 569.356993
worst smoothness             0.022832
worst compactness            0.157336
worst concavity              0.208624
worst concave points         0.065732
worst symmetry               0.061867
worst fractal dimension      0.018061
dtype: float64


Standardizied Z value:
     mean radius  mean texture  mean perimeter  mean area  mean smoothness  \
0       1.096100     -2.071512        1.268817   0.983510         1.567087
1       1.828212     -0.353322        1.684473   1.907030        -0.826235
2       1.578499      0.455786        1.565126   1.557513         0.941382
3      -0.768233      0.253509       -0.592166  -0.763792         3.280667
4       1.748758     -1.150804        1.775011   1.824624         0.280125
..           ...           ...             ...        ...              ...
564     2.109139      0.720838        2.058974   2.341795         1.040926
565     1.703356      2.083301        1.614511   1.722326         0.102368
566     0.701667      2.043775        0.672084   0.577445        -0.839745
567     1.836725      2.334403        1.980781   1.733693         1.524426
568    -1.806811      1.220718       -1.812793  -1.346604        -3.109349

     mean compactness  mean concavity  mean concave points  mean symmetry  \
0            3.280628        2.650542             2.530249       2.215566
1           -0.486643       -0.023825             0.547662       0.001391
2            1.052000        1.362280             2.035440       0.938859
```

```
3           3.399917        1.914213             1.450431       2.864862
4           0.538866        1.369806             1.427237      -0.009552
..               ...             ...                  ...            ...
564         0.218868        1.945573             2.318924      -0.312314
565        -0.017817        0.692434             1.262558      -0.217473
566        -0.038646        0.046547             0.105684      -0.808406
567         3.269267        3.294046             2.656528       2.135315
568        -1.149741       -1.113893            -1.260710      -0.819349

     mean fractal dimension  ...  worst radius  worst texture  \
0                  2.253764  ...      1.885031      -1.358098
1                 -0.867889  ...      1.804340      -0.368879
2                 -0.397658  ...      1.510541      -0.023953
3                  4.906602  ...     -0.281217       0.133866
4                 -0.561956  ...      1.297434      -1.465481
..                      ...  ...           ...            ...
564               -0.930209  ...      1.899514       0.117596
565               -1.057681  ...      1.535369       2.045599
566               -0.894800  ...      0.560868       1.373645
567                1.042778  ...      1.959515       2.235958
568               -0.560539  ...     -1.409652       0.763518

     worst perimeter  worst area  worst smoothness  worst compactness  \
0           2.301575    1.999478          1.306537           2.614365
1           1.533776    1.888827         -0.375282          -0.430066
2           1.346291    1.455004          0.526944           1.081980
3          -0.249720   -0.549538          3.391291           3.889975
4           1.337363    1.219651          0.220362          -0.313119
..               ...         ...               ...                ...
564         1.751022    2.013529          0.378033          -0.273077
565         1.420690    1.493644         -0.690623          -0.394473
566         0.578492    0.427529         -0.808876           0.350427
567         2.301575    1.651717          1.429169           3.901415
568        -1.431475   -1.074867         -1.857384          -1.206491

     worst concavity  worst concave points  worst symmetry  \
0           2.107672              2.294058        2.748204
1          -0.146620              1.086129       -0.243675
2           0.854222              1.953282        1.151242
3           1.987839              2.173873        6.040726
4           0.612640              0.728618       -0.867590
..               ...                   ...             ...
564         0.663928              1.627719       -1.358963
565         0.236365              0.733182       -0.531387
566         0.326479              0.413705       -1.103578
567         3.194794              2.287972        1.917396
568        -1.304683             -1.743529       -0.048096

     worst fractal dimension
0                   1.935312
1                   0.280943
2                   0.201214
3                   4.930672
4                  -0.396751
..                       ...
564                -0.708467
565                -0.973122
566                -0.318129
567                 2.217684
568                -0.750546

[569 rows x 30 columns]
```

The covariance matrix helps us visualize how strong the dependency of two features is with each other in the feature space.

In [ ]:
```python
# covariance
c = Z.cov()
```

In [ ]:
```python
# Plot the covariance matrix
sns.heatmap(c)
plt.show()
```



The eigenvalues of the covareance matrix is calculation.

In [ ]:
```
eigenvalues, eigenvectors = np.linalg.eig(c)
print('Eigen values:\n', eigenvalues)
print('Eigen values Shape:', eigenvalues.shape)
print('Eigen Vector Shape:', eigenvectors.shape)
```

```
Eigen values:
 [1.32816077e+01 5.69135461e+00 2.81794898e+00 1.98064047e+00
 1.64873055e+00 1.20735661e+00 6.75220114e-01 4.76617140e-01
 4.16894812e-01 3.50693457e-01 2.93915696e-01 2.61161370e-01
 2.41357496e-01 1.57009724e-01 9.41349650e-02 7.98628010e-02
 5.93990378e-02 5.26187835e-02 4.94775918e-02 1.33044823e-04
 7.48803097e-04 1.58933787e-03 6.90046388e-03 8.17763986e-03
 1.54812714e-02 1.80550070e-02 2.43408378e-02 2.74394025e-02
 3.11594025e-02 2.99728939e-02]
Eigen values Shape: (30,)
Eigen Vector Shape: (30, 30)
```

In [ ]:
```
# Index the eigenvalues in descending order
idx = eigenvalues.argsort()[::-1]

# Sort the eigenvalues in descending order
eigenvalues = eigenvalues[idx]

# sort the corresponding eigenvectors accordingly
eigenvectors = eigenvectors[:,idx]
```

In [ ]:
```
explained_var = np.cumsum(eigenvalues) / np.sum(eigenvalues)
explained_var
```

Out[ ]:
```
array([0.44272026, 0.63243208, 0.72636371, 0.79238506, 0.84734274,
       0.88758796, 0.9100953 , 0.92598254, 0.93987903, 0.95156881,
       0.961366  , 0.97007138, 0.97811663, 0.98335029, 0.98648812,
       0.98915022, 0.99113018, 0.99288414, 0.9945334 , 0.99557204,
       0.99657114, 0.99748579, 0.99829715, 0.99889898, 0.99941502,
       0.99968761, 0.99991763, 0.99997061, 0.99999557, 1.        ])
```

Selecting the components which has more then 0.5

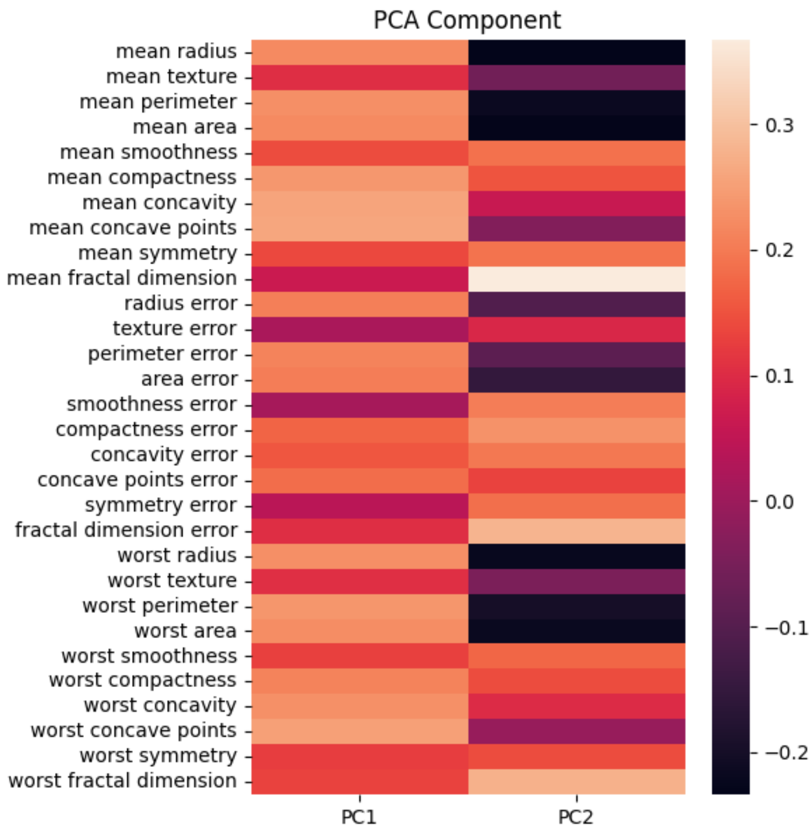
In [ ]:
```
n_components = np.argmax(explained_var >= 0.50) + 1
n_components
```

Out[ ]:
```
2
```

Ploting the detected components

In [ ]:
```
# PCA component or unit matrix
u = eigenvectors[:,:n_components]
pca_component = pd.DataFrame(u,
```

```
                                                          index = cancer['feature_names'],
                                                          columns = ['PC1','PC2']
                                                          )

# plotting heatmap
plt.figure(figsize =(5, 7))
sns.heatmap(pca_component)
plt.title('PCA Component')
plt.show()
```



By Taking the dot product with the standerdize matrix we can get the reducted matrix

In [ ]:
```
# Matrix multiplication or dot Product
Z_pca = Z @ pca_component
# Rename the columns name
Z_pca.rename({'PC1': 'PCA1', 'PC2': 'PCA2'}, axis=1, inplace=True)
# Print the Pricipal Component values
print(Z_pca)
```

```
          PCA1       PCA2
0     9.184755   1.946870
1     2.385703  -3.764859
2     5.728855  -1.074229
3     7.116691  10.266556
4     3.931842  -1.946359
..         ...        ...
564   6.433655  -3.573673
565   3.790048  -3.580897
566   1.255075  -1.900624
567  10.365673   1.670540
568  -5.470430  -0.670047

[569 rows x 2 columns]
```

30 columns converted to 2 columns