


```
#print the features are there in the breast cancer dataset
features=breast.feature_names
print(features)

['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']

#label field missing
features_labels=np.append(features,'label')

#Embedding the column names to the dataframe
breast_dataset.columns=features_labels
breast_dataset.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	worst perimeter	worst area
0	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.3001	0.14710	0.2419	0.07871	...	17.33	184.60	2019.0
1	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.0869	0.07017	0.1812	0.05667	...	23.41	158.80	1956.0
2	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.1974	0.12790	0.2069	0.05999	...	25.53	152.50	1709.0
3	11.42	20.38	77.58	386.1	0.14250	0.28390	0.2414	0.10520	0.2597	0.09744	...	26.50	98.87	567.7
4	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.1980	0.10430	0.1809	0.05883	...	16.67	152.20	1575.0

5 rows × 31 columns

```
#Replace the target values
breast_dataset['label'].replace(0,'Benign',inplace=True)
breast_dataset['label'].replace(1,'Malignant',inplace=True)
breast_dataset.tail()
```



	mean radius	mean texture	mean perimeter	mean area	mean smoothness	mean compactness	mean concavity	mean concave points	mean symmetry	mean fractal dimension	...	worst texture	worst perimeter	worst area
564	21.56	22.39	142.00	1479.0	0.11100	0.11590	0.24390	0.13890	0.1726	0.05623	...	26.40	166.10	2027.0
565	20.13	28.25	131.20	1261.0	0.09780	0.10340	0.14400	0.09791	0.1752	0.05533	...	38.25	155.00	1731.0
566	16.60	28.08	108.30	858.1	0.08455	0.10230	0.09251	0.05302	0.1590	0.05648	...	34.12	126.70	1124.0
567	20.60	29.33	140.10	1265.0	0.11780	0.27700	0.35140	0.15200	0.2397	0.07016	...	39.42	184.60	1821.0
568	7.76	24.54	47.92	181.0	0.05263	0.04362	0.00000	0.00000	0.1587	0.05884	...	30.37	59.16	268.6

5 rows × 31 columns

```
from sklearn.preprocessing import StandardScaler
x = breast_dataset.loc[:,features].values
x =StandardScaler().fit_transform(x) # normalizing the features
print(x.shape)

(569, 30)

#check
np.mean(x),np.std(x)

(-6.118909323768877e-16, 1.0)
```

```
#Convert the normalized features into tabular format
feat_cols = ['feature'+str(i) for i in range(x.shape[1])]
normalised_breast=pd.DataFrame(x,columns=feat_cols)
print(normalised_breast)
```

```

    feature0 feature1 feature2 feature3 feature4 feature5 feature6 \
0  1.097064 -2.073335 1.269934 0.984375 1.568466 3.283515 2.652874
1  1.829821 -0.353632 1.685955 1.908708 -0.826962 -0.487072 -0.023846
2  1.579888 0.456187 1.566503 1.558884 0.942210 1.052926 1.363478
3  -0.768909 0.253732 -0.592687 -0.764464 3.283553 3.402909 1.915897
4  1.750297 -1.151816 1.776573 1.826229 0.280372 0.539340 1.371011
..    ...    ...    ...    ...    ...    ...    ...
564 2.110995 0.721473 2.060786 2.343856 1.041842 0.219060 1.947285
565 1.704854 2.085134 1.615931 1.723842 0.102458 -0.017833 0.693043
566 0.702284 2.045574 0.672676 0.577953 -0.840484 -0.038680 0.046588
567 1.838341 2.336457 1.982524 1.735218 1.525767 3.272144 3.296944
568 -1.808401 1.221792 -1.814389 -1.347789 -3.112085 -1.150752 -1.114873

```

```

    feature7 feature8 feature9 ... feature20 feature21 feature22 \
0  2.532475 2.217515 2.255747 ... 1.886690 -1.359293 2.303601
1  0.548144 0.001392 -0.868652 ... 1.805927 -0.369203 1.535126
2  2.037231 0.939685 -0.398008 ... 1.511870 -0.023974 1.347475
3  1.451707 2.867383 4.910919 ... -0.281464 0.133984 -0.249939
4  1.428493 -0.009560 -0.562450 ... 1.298575 -1.466770 1.338539
..    ...    ...    ...    ...    ...    ...    ...
564 2.320965 -0.312589 -0.931027 ... 1.901185 0.117700 1.752563
565 1.263669 -0.217664 -1.058611 ... 1.536720 2.047399 1.421940
566 0.105777 -0.809117 -0.895587 ... 0.561361 1.374854 0.579001
567 2.658866 2.137194 1.043695 ... 1.961239 2.237926 2.303601
568 -1.261820 -0.820070 -0.561032 ... -1.410893 0.764190 -1.432735

```

```

    feature23 feature24 feature25 feature26 feature27 feature28 \
0  2.001237 1.307686 2.616665 2.109526 2.296076 2.750622
1  1.890489 -0.375612 -0.430444 -0.146749 1.087084 -0.243890
2  1.456285 0.527407 1.082932 0.854974 1.955000 1.152255
3  -0.550021 3.394275 3.893397 1.989588 2.175786 6.046041
4  1.220724 0.220556 -0.313395 0.613179 0.729259 -0.868353
..    ...    ...    ...    ...    ...    ...
564 2.015301 0.378365 -0.273318 0.664512 1.629151 -1.360158
565 1.494959 -0.691230 -0.394820 0.236573 0.733827 -0.531855
566 0.427906 -0.809587 0.350735 0.326767 0.414069 -1.104549
567 1.653171 1.430427 3.904848 3.197605 2.289985 1.919083
568 -1.075813 -1.859019 -1.207552 -1.305831 -1.745063 -0.048138

```

```

feature29
0  1.937015
1  0.281190
2  0.201391
3  4.935010
4  -0.397100
..    ...
564 -0.709091
565 -0.973978
566 -0.318409
567 2.219635
568 -0.751207

```

[569 rows x 30 columns]

normalised_breast.tail()

	feature0	feature1	feature2	feature3	feature4	feature5	feature6	feature7	feature8	feature9	...	feature20	feature21	feat
564	2.110995	0.721473	2.060786	2.343856	1.041842	0.219060	1.947285	2.320965	-0.312589	-0.931027	...	1.901185	0.117700	1.7
565	1.704854	2.085134	1.615931	1.723842	0.102458	-0.017833	0.693043	1.263669	-0.217664	-1.058611	...	1.536720	2.047399	1.4
566	0.702284	2.045574	0.672676	0.577953	-0.840484	-0.038680	0.046588	0.105777	-0.809117	-0.895587	...	0.561361	1.374854	0.5
567	1.838341	2.336457	1.982524	1.735218	1.525767	3.272144	3.296944	2.658866	2.137194	1.043695	...	1.961239	2.237926	2.3
568	-1.808401	1.221792	-1.814389	-1.347789	-3.112085	-1.150752	-1.114873	-1.261820	-0.820070	-0.561032	...	-1.410893	0.764190	-1.4

5 rows x 30 columns

```

#thirty dimensional
from sklearn.decomposition import PCA
pca_breast=PCA(n_components=2)
principalComponents_breast=pca_breast.fit_transform(x)
principal_breast_Df=pd.DataFrame(data=principalComponents_breast
                                ,columns = ['principal component 1','principal component 2'])
principal_breast_Df.tail()

```

	principal component 1	principal component 2
564	6.439315	-3.576817
565	3.793382	-3.584048
566	1.256179	-1.902297
567	10.374794	1.672010
568	-5.475243	-0.670637

```
print('Explained variation per principal component: {}'.format(pca_breast.explained_variance_ratio_))
```

```
Explained variation per principal component: [0.44272026 0.18971182]
```

```
#plot the pca
import matplotlib.pyplot as plt
plt.figure()
plt.figure(figsize=(10,10))
plt.xticks(fontsize=12)
plt.yticks(fontsize=14)
plt.xlabel('Principal Component - 1',fontsize=20)
plt.ylabel('Principal Component - 2',fontsize=20)
plt.title("Principal Component Analysis of Breast Cancer Dataset",fontsize=20)
targets=['Benign','Malignant']
colors=['r','g']
for target, color in zip(targets,colors):
    indicesToKeep = breast_dataset['label']== target
    plt.scatter(principal_breast_Df.loc[indicesToKeep,'principal component 1'],
                principal_breast_Df.loc[indicesToKeep,'principal component 2'],c=color,s=5 )
plt.legend(targets,prop={'size':15})
```

<Figure size 640x480 with 0 Axes>

