

**Objective: Classify a breast mass to Benign(Not harmful) and Malignant(harmful) based on 30 feature available.**

**But in reality we cannot visualize more than 3 Dimension So we employ PCA to visualize 30 dimension data into 2 Dimensional data.**

## Sources of the dataset

In [2]:

```
#Breast Cancer Wisconsin (Diagnostic) Data Set downloaded from Kaggle:  
#https://www.kaggle.com/uciml/breast-cancer-wisconsin-data
```

## importing all the Modules

In [3]:

```
import warnings  
warnings.filterwarnings("ignore")  
import pandas as pd  
import numpy as np  
import matplotlib.pyplot as plt
```

## loading the data in dataframe

In [4]:

```
df=pd.read_csv("cancer_diagnostic_data.csv")
```

## displaying the first 5 datapoints of the datasets.

In [5]:

```
df.head()
```

Out[5]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_m
0	842302	M	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.3
1	842517	M	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.0
2	84300903	M	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.1
3	84348301	M	11.42	20.38	77.58	386.1	0.14250	0.28390	0.2
4	84358402	M	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.1

5 rows × 32 columns



**Deleting the the column "id" as it refers to some pointer/name to breast mass image and have no importance in visulization. Moreover its is not feature of breast mass.**

In [6]:

```
dff=df.drop("id",1)
```

## displaying the first 5 datapoints of the datasets after deleting the id column

In [7]:

```
dff.head()
```

Out[7]:

	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mean	points
0	M	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.3001	
1	M	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.0869	
2	M	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.1974	
3	M	11.42	20.38	77.58	386.1	0.14250	0.28390	0.2414	
4	M	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.1980	

5 rows × 31 columns



## Storing the diagnosis column data in dff\_out as this is output data

In [8]:

```
dff_out=dff["diagnosis"] #output dataframe
```

## Finding the shape of output data

In [9]:

```
dff_out.shape
```

Out[9]:

(569,)

## deleting the column "diagnosis" from the feature dataset.

In [10]:

```
dff_final=dff.drop("diagnosis",axis=1)
```

## displaying the 5 datapoints of feature dataset So we can clearly see it have 5 rows and 30 columns.

In [11]:

```
dff_final.head()
```

Out[11]:

	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mean	concave points_mean	symmetry
--	-------------	--------------	----------------	-----------	-----------------	------------------	----------------	---------------------	----------

0	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.3001	0.14710	0.07017	sy
	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mean	points_mean	concave	
1	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.0869	0.07017	0.07017	
2	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.1974	0.12790	0.12790	
3	11.42	20.38	77.58	386.1	0.14250	0.28390	0.2414	0.10520	0.10520	
4	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.1980	0.10430	0.10430	

5 rows × 30 columns

## Displaying the number of datapoints with number of feature.

In [97]:

```
dff_final.shape
```

Out[97]:

```
(569, 30)
```

important thing to observe that we will use sklearn which helps us to write machine learning algorithm. Under sklearn module decomposition is a package. Decomposition package having PCA() function to which converts the 30 dimensional data to 2 dimensional data

In [12]:

```
from sklearn import decomposition
```

In [13]:

```
pca = decomposition.PCA()
print(type(pca))
```

```
<class 'sklearn.decomposition.pca.PCA'>
```

In [16]:

```
pca.n_components = 2
```

In [17]:

```
type(pca.n_components)
```

Out[17]:

```
int
```

## transformation on the data

In [102]:

```
pca_data = pca.fit_transform(dff_final)
```

Clearly we can see the feature is reduced to 2 from 30 that is power of PCA to visualize data 2D coordinate system.

In [103]:

```
pca_data.shape
```

Out[103]:

```
(569, 2)
```

## 2 features with corresponding output value.

In [104]:

```
pca_data = np.vstack((pca_data.T, dff_out)).T
```

In [105]:

```
pca_data
```

Out[105]:

```
array([[1160.142573704138, -293.91754363738903, 'M'],
       [1269.1224431936498, 15.630181843400353, 'M'],
       [995.7938889594554, 39.15674324394633, 'M'],
       ...,
       [314.5017561800185, 47.55352518489632, 'M'],
       [1124.8581153057105, 34.129224970157026, 'M'],
       [-771.5276218767491, -88.64310636345301, 'B']], dtype=object)
```

## Ploting the data using seaborn module

In [107]:

```
import warnings
warnings.filterwarnings("ignore")
import seaborn as sn
pca_df = pd.DataFrame(data=pca_data, columns=("1st_principal", "2nd_principal", "label"))
sn.FacetGrid(pca_df, hue="label", size=6).map(plt.scatter, '1st_principal', '2nd_principal').add_legend()
plt.show()
```

C:\Users\atif\Anaconda3\New folder\lib\site-packages\seaborn\axisgrid.py:230: UserWarning: The `size` paramter has been renamed to `height`; please update your code.  
warnings.warn(msg, UserWarning)



