Importing Modules and Preprocessing

```
!pip install mat4py
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
import seaborn as sns
from mat4py import loadmat
%matplotlib inline
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/pub
Collecting mat4py
  Downloading mat4py-0.5.0-py2.py3-none-any.whl (13 kB)
Installing collected packages: mat4py
Successfully installed mat4py-0.5.0
```

```
from google.colab import drive
drive.mount('/content/gdrive')
```

```
Mounted at /content/gdrive
```

```
%cd 'gdrive/MyDrive/ML Lab Datasets'
```

```
/content/gdrive/MyDrive/ML Lab Datasets
```

```
data = loadmat('Matlab_cancer.mat')
```

```
data.keys()
```

```
dict_keys(['t', 'x'])
```

```
data_x = data['x']
data_y = data['t']
data_x = np.transpose(data_x)
np.transpose(data_y)
data_y = data_y[0]
data_x = np.array(data_x)
data_y = np.array(data_y)
```

```
data_x.shape
```

```
(216, 100)
```

```
data_y.shape
```

```
(216,)
```

```
df = pd.DataFrame(data_x)
```

```
def PCA(X , num_components):

    #Step-1
    X_meaned = X - np.mean(X , axis = 0)

    #Step-2
    cov_mat = np.cov(X_meaned , rowvar = False)

    #Step-3
    eigen_values , eigen_vectors = np.linalg.eigh(cov_mat)

    #Step-4
    sorted_index = np.argsort(eigen_values)[::-1]
    sorted_eigenvalue = eigen_values[sorted_index]
    sorted_eigenvectors = eigen_vectors[:,sorted_index]

    #Step-5
    eigenvector_subset = sorted_eigenvectors[:,0:num_components]

    #Step-6
    X_reduced = np.dot(eigenvector_subset.transpose() , X_meaned.transpose() ).transpose()

    return X_reduced,eigenvector_subset
```

Removing One Eigenvector

```
x = df
target = data_y
mat_reduced,eg = PCA(x,99)
principal_df = pd.DataFrame(mat_reduced)
principal_df = pd.concat([principal_df , pd.DataFrame(target)] , axis = 1)
```

```
principal_df.head()
```

	0	1	2	3	4	5	6	7	
0	-0.673201	-0.889922	0.212169	-0.529382	-0.459633	-0.335541	0.115979	-0.088246	0.0
1	-1.472090	-0.887787	0.803924	-0.017798	-0.471674	-0.315940	0.252919	-0.175758	0.1
2	-4.143399	0.630552	-0.028757	0.353891	-0.197041	-0.449780	0.044882	0.288183	0.0
3	-2.062684	-0.662332	0.191229	-0.611367	-0.207995	0.115587	0.421927	0.062953	0.3

```
df1 = pd.DataFrame(data = np.dot(mat_reduced, eg.transpose()) + np.mean(data_x , axis = 0))
```

5 rows × 100 columns

```
df1.head()
```

	0	1	2	3	4	5	6	7	8
0	0.462904	0.406727	0.306230	0.488106	0.175152	0.356669	0.439481	0.246682	1.373504
1	0.370543	0.306492	0.239608	0.476348	0.176013	0.352937	0.476563	0.259575	1.338724
2	0.461538	0.279014	0.206549	0.744336	0.152196	0.247182	0.304567	0.170737	1.047340
3	0.375159	0.287289	0.241179	0.426815	0.205807	0.209151	0.283525	0.152258	1.213896
4	0.113859	0.108382	0.086326	0.219959	0.050740	0.116557	0.188167	0.069400	0.706046

5 rows × 100 columns

```
df.head()
```

	0	1	2	3	4	5	6	7	8
0	0.462901	0.406736	0.306215	0.488105	0.175169	0.356551	0.439552	0.246779	1.373469
1	0.370565	0.306431	0.239715	0.476358	0.175900	0.353762	0.476071	0.258894	1.338965
2	0.461550	0.278980	0.206608	0.744342	0.152133	0.247645	0.304292	0.170356	1.047475
3	0.375149	0.287317	0.241130	0.426810	0.205859	0.208770	0.283752	0.152572	1.213785
4	0.113849	0.108411	0.086275	0.219954	0.050794	0.116163	0.188402	0.069726	0.705930

5 rows × 100 columns

```
error = np.sum(np.mean(df - df1)**2)/100
```

```
error
```

```
3.8518425065196046e-34
```

Remove 10 EigenVectors

```
x = df
target = data_y
mat_reduced,eg = PCA(x,90)
principal_df = pd.DataFrame(mat_reduced)
principal_df = pd.concat([principal_df , pd.DataFrame(target)] , axis = 1)
```

```
principal_df.head()
```

	0	1	2	3	4	5	6	7	
0	-0.673201	-0.889922	0.212169	-0.529382	-0.459633	-0.335541	0.115979	-0.088246	0.04
1	-1.472090	-0.887787	0.803924	-0.017798	-0.471674	-0.315940	0.252919	-0.175758	0.12
2	-4.143399	0.630552	-0.028757	0.353891	-0.197041	-0.449780	0.044882	0.288183	0.08
3	-2.062684	-0.662332	0.191229	-0.611367	-0.207995	0.115587	0.421927	0.062953	0.30
4	-7.110172	1.066008	2.404473	0.110741	-0.281457	0.269950	0.308334	-0.147788	0.19

5 rows × 91 columns

```
df1 = pd.DataFrame(data = np.dot(mat_reduced, eg.transpose()) + np.mean(data_x , axis = 0))
```

```
df1.head()
```

	0	1	2	3	4	5	6	7	8

```
df.head()
```

	0	1	2	3	4	5	6	7	8
0	0.462901	0.406736	0.306215	0.488105	0.175169	0.356551	0.439552	0.246779	1.373469
1	0.370565	0.306431	0.239715	0.476358	0.175900	0.353762	0.476071	0.258894	1.338965
2	0.461550	0.278980	0.206608	0.744342	0.152133	0.247645	0.304292	0.170356	1.047475
3	0.375149	0.287317	0.241130	0.426810	0.205859	0.208770	0.283752	0.152572	1.213785
4	0.113849	0.108411	0.086275	0.219954	0.050794	0.116163	0.188402	0.069726	0.705930

5 rows × 100 columns

```
error = np.sum(np.mean(df - df1)**2)/100
```

```
error
```

```
6.0932575675323265e-34
```

✓ 0s completed at 3:23 PM