In [1]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

In [2]:

```
from sklearn.datasets import load_breast_cancer
```

In [3]:

```
cancer_data = load_breast_cancer()
```

In [4]:

```
cancer_data.keys()
```

Out[4]:

```
dict_keys(['data', 'target', 'target_names', 'DESCR', 'feature_nam
es', 'filename'])
```

In [5]:

```
print(cancer_data['DESCR'])
```

.. _breast_cancer_dataset:

Breast cancer wisconsin (diagnostic) dataset

Data Set Characteristics:

    :Number of Instances: 569

    :Number of Attributes: 30 numeric, predictive attributes and t
he class

    :Attribute Information:
        - radius (mean of distances from center to points on the p
erimeter)
        - texture (standard deviation of gray-scale values)
        - perimeter
        - area
        - smoothness (local variation in radius lengths)
        - compactness (perimeter^2 / area - 1.0)
        - concavity (severity of concave portions of the contour)
        - concave points (number of concave portions of the contou
r)
        - symmetry
        - fractal dimension ("coastline approximation" - 1)

        The mean, standard error, and "worst" or largest (mean of
the three
        largest values) of these features were computed for each i
mage,
        resulting in 30 features. For instance, field 3 is Mean R
adius, field
        13 is Radius SE, field 23 is Worst Radius.

        - class:
                - WDBC-Malignant
                - WDBC-Benign

    :Summary Statistics:

    ===================================== ====== ======
                                           Min   Max
    ===================================== ====== ======
    radius (mean):   6.981 28.11
    texture (mean):   9.71   39.28
    perimeter (mean):   43.79 188.5
    area (mean):   143.5 2501.0
    smoothness (mean):   0.053 0.163
    compactness (mean):   0.019 0.345
    concavity (mean):   0.0   0.427
    concave points (mean):   0.0   0.201
    symmetry (mean):   0.106 0.304
    fractal dimension (mean):   0.05   0.097
    radius (standard error):   0.112 2.873
    texture (standard error):   0.36   4.885
    perimeter (standard error):   0.757 21.98
    area (standard error):   6.802 542.2
    smoothness (standard error):   0.002 0.031
    compactness (standard error):   0.002 0.135

    concavity (standard error):   0.0   0.396
    concave points (standard error):   0.0   0.053
    symmetry (standard error):   0.008 0.079
    fractal dimension (standard error):   0.001 0.03
    radius (worst):   7.93   36.04
    texture (worst):   12.02 49.54
    perimeter (worst):   50.41 251.2
    area (worst):   185.2 4254.0
    smoothness (worst):   0.071 0.223
    compactness (worst):   0.027 1.058
    concavity (worst):   0.0   1.252
    concave points (worst):   0.0   0.291
    symmetry (worst):   0.156 0.664
    fractal dimension (worst):   0.055 0.208
    ==================================== ====== ======

    :Missing Attribute Values: None

    :Class Distribution: 212 - Malignant, 357 - Benign

    :Creator: Dr. William H. Wolberg, W. Nick Street, Olvi L. Man
gasarian

    :Donor: Nick Street

    :Date: November, 1995

This is a copy of UCI ML Breast Cancer Wisconsin (Diagnostic) data
sets.
https://goo.gl/U2Uwz2

Features are computed from a digitized image of a fine needle
aspirate (FNA) of a breast mass. They describe
characteristics of the cell nuclei present in the image.

Separating plane described above was obtained using
Multisurface Method-Tree (MSM-T) [K. P. Bennett, "Decision Tree
Construction Via Linear Programming." Proceedings of the 4th
Midwest Artificial Intelligence and Cognitive Science Society,
pp. 97-101, 1992], a classification method which uses linear
programming to construct a decision tree. Relevant features
were selected using an exhaustive search in the space of 1-4
features and 1-3 separating planes.

The actual linear program used to obtain the separating plane
in the 3-dimensional space is that described in:
[K. P. Bennett and O. L. Mangasarian: "Robust Linear
Programming Discrimination of Two Linearly Inseparable Sets",
Optimization Methods and Software 1, 1992, 23-34].

This database is also available through the UW CS ftp server:

ftp ftp.cs.wisc.edu
cd math-prog/cpo-dataset/machine-learn/WDBC/

.. topic:: References

   - W.N. Street, W.H. Wolberg and O.L. Mangasarian. Nuclear featu
re extraction
     for breast tumor diagnosis. IS&T/SPIE 1993 International Symp

```
osium on
     Electronic Imaging: Science and Technology, volume 1905, page
s 861-870,
     San Jose, CA, 1993.
   - O.L. Mangasarian, W.N. Street and W.H. Wolberg. Breast cancer
diagnosis and
     prognosis via linear programming. Operations Research, 43(4),
pages 570-577,
     July-August 1995.
   - W.H. Wolberg, W.N. Street, and O.L. Mangasarian. Machine lear
ning techniques
     to diagnose breast cancer from fine-needle aspirates. Cancer
Letters 77 (1994)
     163-171.
```

In [6]:

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

In [7]:

```
df.head()
```

Out[7]:

	mean radius	mean texture	mean perimeter	mean area	mean smoothness	mean compactness	mean concavity	mean concave points	sym
0	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.3001	0.14710	(
1	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.0869	0.07017	(
2	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.1974	0.12790	(
3	11.42	20.38	77.58	386.1	0.14250	0.28390	0.2414	0.10520	(
4	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.1980	0.10430	(

5 rows × 30 columns

In [8]:

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

MinMax Scaler

In [9]:

```
from sklearn.preprocessing import MinMaxScaler
```

In [10]:

```
min_max_scaler = MinMaxScaler()
```

In [11]:

```
min_max_scaler.fit(df)
```

Out[11]:

```
MinMaxScaler(copy=True, feature_range=(0, 1))
```

In [12]:

```
minmax_scaled_data=min_max_scaler.transform(df)
```

In [13]:

```
minmax_scaled_data[:1,:]
```

Out[13]:

```
array([[0.52103744, 0.0226581 , 0.54598853, 0.36373277, 0.5937528
2,
        0.7920373 , 0.70313964, 0.73111332, 0.68636364, 0.6055181
1,
        0.35614702, 0.12046941, 0.3690336 , 0.27381126, 0.1592956
5,
        0.35139844, 0.13568182, 0.30062512, 0.31164518, 0.1830424
4,
        0.62077552, 0.14152452, 0.66831017, 0.45069799, 0.6011358
4,
        0.61929156, 0.56861022, 0.91202749, 0.59846245, 0.4188639
6]])
```

Standard Normalization

In [14]:

```
from sklearn.preprocessing import StandardScaler
```

In [15]:

```
scaler = StandardScaler()
```

In [16]:

```
scaler.fit(df)
```

Out[16]:

```
StandardScaler(copy=True, with_mean=True, with_std=True)
```

In [17]:

```
df.head()
```

Out[17]:

	mean radius	mean texture	mean perimeter	mean area	mean smoothness	mean compactness	mean concavity	mean concave points	sym
0	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.3001	0.14710	(
1	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.0869	0.07017	(
2	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.1974	0.12790	(
3	11.42	20.38	77.58	386.1	0.14250	0.28390	0.2414	0.10520	(
4	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.1980	0.10430	(

5 rows × 30 columns

In [18]:

```
scaled_data=scaler.transform(df)
```

In [19]:

```
scaled_data[:1,:]
```

Out[19]:

```
array([[ 1.09706398, -2.07333501,  1.26993369,  0.9843749 ,  1.568
46633,
         3.28351467,  2.65287398,  2.53247522,  2.21751501,  2.255
74689,
         2.48973393, -0.56526506,  2.83303087,  2.48757756, -0.214
00165,
         1.31686157,  0.72402616,  0.66081994,  1.14875667,  0.907
08308,
         1.88668963, -1.35929347,  2.30360062,  2.00123749,  1.307
68627,
         2.61666502,  2.10952635,  2.29607613,  2.75062224,  1.937
01461]])
```

In [20]:

```
from sklearn.decomposition import PCA
```

In [21]:

```
pca=PCA(n_components = 2)
```

In [22]:

```
pca= pca.fit(scaled_data)
```

In [23]:

```
x_pca=pca.transform(scaled_data)
```

In [24]:

```
scaled_data.shape
```

Out[24]:

```
(569, 30)
```

In [25]:

```
x_pca.shape
```

Out[25]:

```
(569, 2)
```

In [ ]:

```

```

In [26]:

```
pca2=PCA(n_components = 2)
pca2= pca2.fit(minmax_scaled_data)
x_pca2=pca2.transform(minmax_scaled_data)
```

In [27]:

```
minmax_scaled_data.shape
```

Out[27]:

```
(569, 30)
```

In [28]:

```
x_pca2.shape
```

Out[28]:

```
(569, 2)
```

In [ ]:

```

```