Using Predictive Analysis To Predict Diagnosis of a Breast Tumor

Identify data sources

The Breast Cancer (https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+%28Diagnostic%29) datasets is available machine learning repository maintained by the University of California, Irvine. The dataset contains **569 samples of malignant and benign tumor cells**.

- The first two columns in the dataset store the unique ID numbers of the samples and the corresponding diagnosis (M=malignant, B=benign), respectively.
- The columns 3-32 contain 30 real-value features that have been computed from digitized images of the cell nuclei, which can be used to build a model to predict whether a tumor is benign or malignant.

Getting Started: Load libraries and set options

In [2]:

```
#load libraries
import numpy as np          # linear algebra
import pandas as pd         # data processing, CSV file I/O (e.g. pd.read_csv)

# Read the file "data.csv".
data = pd.read_csv('C:\data\data.csv', index_col=False,)
```

In [3]:

```
data.head(2)
```

Out[3]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_me
0	842302	M	17.99	10.38	122.8	1001.0	0.118
1	842517	M	20.57	17.77	132.9	1326.0	0.084

2 rows × 32 columns

You can check the number of cases, as well as the number of fields, using the shape method, as shown below.

In [4]:

```
data.shape
```

Out[4]:

```
(569, 32)
```

In [5]:

```
# Id column is redundant and not useful, we want to drop it
data.drop('id', axis =1, inplace=True)
#data.drop('Unnamed: 0', axis=1, inplace=True)
data.head(2)
```

Out[5]:

	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	com
0	M	17.99	10.38	122.8	1001.0	0.11840	
1	M	20.57	17.77	132.9	1326.0	0.08474	

2 rows × 31 columns

In [6]:

```
data.shape
```

Out[6]:

```
(569, 31)
```

In the result displayed, you can see the data has 569 records, each with 32 columns.

The **“info()”** method provides a concise summary of the data; from the output, it provides the type of data in each column, the number of non-null values in each column, and how much memory the data frame is using.

The method **get_dtype_counts()** will return the number of columns of each type in a DataFrame:

In [7]:

```
# Review data types with "info()".
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 569 entries, 0 to 568
Data columns (total 31 columns):
diagnosis                  569 non-null object
radius_mean                569 non-null float64
texture_mean               569 non-null float64
perimeter_mean             569 non-null float64
area_mean                  569 non-null float64
smoothness_mean            569 non-null float64
compactness_mean           569 non-null float64
concavity_mean             569 non-null float64
concave points_mean        569 non-null float64
symmetry_mean              569 non-null float64
fractal_dimension_mean     569 non-null float64
radius_se                  569 non-null float64
texture_se                 569 non-null float64
perimeter_se               569 non-null float64
area_se                    569 non-null float64
smoothness_se              569 non-null float64
compactness_se             569 non-null float64
concavity_se               569 non-null float64
concave points_se          569 non-null float64
symmetry_se                569 non-null float64
fractal_dimension_se       569 non-null float64
radius_worst               569 non-null float64
texture_worst              569 non-null float64
perimeter_worst            569 non-null float64
area_worst                 569 non-null float64
smoothness_worst           569 non-null float64
compactness_worst          569 non-null float64
concavity_worst            569 non-null float64
concave points_worst       569 non-null float64
symmetry_worst             569 non-null float64
fractal_dimension_worst    569 non-null float64
dtypes: float64(30), object(1)
memory usage: 137.9+ KB
```

In [8]:

```
# Review number of columns of each data type in a DataFrame:
data.get_dtype_counts()
```

Out[8]:

```
float64    30
object      1
dtype: int64
```

From the above results, from the 32, variables,column id number 1 is an integer, diagnosis 569 non-null object. and rest are float. More on python variables (https://www.tutorialspoint.com/python/python_variable_types.htm)

In [9]:

```
#check for missing variables
data.isnull().any()
```

Out[9]:

```
diagnosis                  False
radius_mean                False
texture_mean               False
perimeter_mean             False
area_mean                  False
smoothness_mean            False
compactness_mean           False
concavity_mean             False
concave points_mean        False
symmetry_mean              False
fractal_dimension_mean     False
radius_se                  False
texture_se                 False
perimeter_se               False
area_se                    False
smoothness_se              False
compactness_se             False
concavity_se               False
concave points_se          False
symmetry_se                False
fractal_dimension_se       False
radius_worst               False
texture_worst              False
perimeter_worst            False
area_worst                 False
smoothness_worst           False
compactness_worst          False
concavity_worst            False
concave points_worst       False
symmetry_worst             False
fractal_dimension_worst    False
dtype: bool
```

In [10]:

```
print(data.groupby('diagnosis').size())
```

```
diagnosis
B    357
M    212
dtype: int64
```

Let's take a look at the number of Benign and Maglinant cases from the dataset. From the output shown below, majority of the cases are benign (0).

In [12]:

```
#save the cleaner version of dataframe for future analyis
data.to_csv('C:\data\clean-data.csv')
```