In [1]:

```
import os
os.getcwd()
os.chdir(r"C:\Users\Avirup Gupta Roy\Desktop\breast_cancer\breast_cancer_coimbra")
import pandas as pd
import numpy as np
cancer_data=pd.read_csv(r"breast_cancer_coimbra.csv")
X=cancer_data.iloc[:,0:9].values
y=cancer_data.iloc[:,9].values
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.6, random_state = 0)
```

In [3]:

```
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
lda = LinearDiscriminantAnalysis()
lda.fit(X_train, y_train).transform(X_train)
predicted = lda.predict(X_test) #gives you the predicted label for each sample
from sklearn.metrics import accuracy_score
print("linear discriminant analysis",accuracy_score(y_test,predicted))
```

```
linear discriminant analysis 0.5142857142857142
```

In [4]:

```
from sklearn.ensemble import RandomForestRegressor
forest_reg = RandomForestRegressor(random_state=42)
forest_reg.fit(X_train, y_train)
y_pred = forest_reg.predict(X_test)
print("random forest regressor",accuracy_score(y_test,np.round(y_pred)))
```

```
random forest regressor 0.6142857142857143
```

In [5]:

```
from sklearn.linear_model import LogisticRegression
logreg = LogisticRegression()
logreg.fit(X_train, y_train)
y_pred = logreg.predict(X_test)
print("logistic",accuracy_score(y_test,y_pred))
from sklearn.metrics import confusion_matrix
confusion_matrix = confusion_matrix(y_test, y_pred)
print("confusion matrix")
print(confusion_matrix)
from sklearn.metrics import classification_report
print(classification_report(y_test, y_pred))
from sklearn import model_selection
from sklearn.model_selection import cross_val_score
kfold = model_selection.KFold(n_splits=7, random_state=15)
scoring = 'accuracy'
results = model_selection.cross_val_score(logreg, X_train, y_train, cv=kfold, scoring=scoring)
print("10-fold cross validation average accuracy: %.3f" % (results.mean()))
```

```
logistic 0.5571428571428572
confusion matrix
[[12 23]
 [ 8 27]]
             precision    recall  f1-score   support

          1       0.60      0.34      0.44        35
          2       0.54      0.77      0.64        35

avg / total       0.57      0.56      0.54        70

10-fold cross validation average accuracy: 0.741
```

In [11]:

```
from sklearn import model_selection
from sklearn.model_selection import cross_val_score
from sklearn.ensemble import ExtraTreesClassifier
kfold = model_selection.KFold(n_splits=7, random_state=20)
modelCV = ExtraTreesClassifier()
modelCV.fit(X_train,y_train)
modelCV.score(X_test,y_test)
scoring = 'accuracy'
results = model_selection.cross_val_score(modelCV, X_train, y_train, cv=kfold, scoring=scoring)
print("7-fold cross validation average accuracy: %.3f" % (results.mean()))
```

```
7-fold cross validation average accuracy: 0.803
```

In [13]:

```
feature_labels = np.array(['Age', 'BMI', 'Glucose', 'Insulin', 'HOMA', 'Leptin', 'Adiponectin',
       'Resistin', 'MCP.1', 'Classification'])
importance = modelCV.feature_importances_
feature_indexes_by_importance = importance.argsort()
for index in feature_indexes_by_importance:
    print('{}={:.2f}%'.format(feature_labels[index],(importance[index] *100.0)))
```

```
Leptin=3.19%
Resistin=6.30%
HOMA=6.82%
Insulin=6.84%
Adiponectin=8.35%
BMI=11.69%
MCP.1=15.12%
Age=18.13%
Glucose=23.55%
```

In [14]:

```
from sklearn import ensemble
from sklearn.ensemble import GradientBoostingRegressor
model = ensemble.GradientBoostingRegressor()
model.fit(X_train, y_train)
print('Gradient Boosting R squared": %.4f' % model.score(X_test, y_test))
```

```
Gradient Boosting R squared": 0.0722
```

In [15]:

```
from sklearn.naive_bayes import GaussianNB
model = GaussianNB()
model.fit(X_train, y_train)
print('Naive Bayes R squared": %.4f' % model.score(X_test,y_test))
```

```
Naive Bayes R squared": 0.6000
```

In [16]:

```
from sklearn import tree
model = tree.DecisionTreeClassifier(criterion='gini')
model.fit(X_train, y_train)
predicted= model.predict(X_test)
model.score(X_test,predicted)
```

Out[16]:

```
1.0
```

In [ ]:

```

```