LAB 2

Task 2 : Apply algorithm on breast cancer wisconsin dataset - One Hot Encoding of features: and Train test

Division 50%-50%

In [1]:
```
from sklearn import datasets
from sklearn import preprocessing
from sklearn.metrics import confusion_matrix,precision_score,recall_score,ac
curacy_score
from sklearn.naive_bayes import GaussianNB,MultinomialNB
from sklearn.model_selection import train_test_split

breast_cancer = datasets.load_breast_cancer()
```

In [2]:
```
#print("Features: ", breast_cancer.feature_names)
#print("\nLabels: ", breast_cancer.target_names)
breast_cancer.data.shape

print('\nData : \n',breast_cancer.data[0:3,:])
print('\nTarget : \n',breast_cancer.target)
```

```
Data :
 [[1.799e+01 1.038e+01 1.228e+02 1.001e+03 1.184e-01 2.776e-01 3.001e-01
  1.471e-01 2.419e-01 7.871e-02 1.095e+00 9.053e-01 8.589e+00 1.534e+02
  6.399e-03 4.904e-02 5.373e-02 1.587e-02 3.003e-02 6.193e-03 2.538e+01
  1.733e+01 1.846e+02 2.019e+03 1.622e-01 6.656e-01 7.119e-01 2.654e-01
  4.601e-01 1.189e-01]
 [2.057e+01 1.777e+01 1.329e+02 1.326e+03 8.474e-02 7.864e-02 8.690e-02
  7.017e-02 1.812e-01 5.667e-02 5.435e-01 7.339e-01 3.398e+00 7.408e+01
  5.225e-03 1.308e-02 1.860e-02 1.340e-02 1.389e-02 3.532e-03 2.499e+01
  2.341e+01 1.588e+02 1.956e+03 1.238e-01 1.866e-01 2.416e-01 1.860e-01
  2.750e-01 8.902e-02]
 [1.969e+01 2.125e+01 1.300e+02 1.203e+03 1.096e-01 1.599e-01 1.974e-01
  1.279e-01 2.069e-01 5.999e-02 7.456e-01 7.869e-01 4.585e+00 9.403e+01
  6.150e-03 4.006e-02 3.832e-02 2.058e-02 2.250e-02 4.571e-03 2.357e+01
  2.553e+01 1.525e+02 1.709e+03 1.444e-01 4.245e-01 4.504e-01 2.430e-01
  3.613e-01 8.758e-02]]

Target :
 [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 1 0 0 0 0 0 0 0 0 1 0 1 1 1 1 1 0 0 1 0 0 1 1 1 1 0 1 0 0 1 1 1 1 0 1 0 0
 1 0 1 0 0 1 1 1 0 0 1 0 0 0 1 1 1 0 1 1 0 0 1 1 1 0 0 1 1 1 1 0 1 1 0 1 1
 1 1 1 1 1 1 0 0 0 1 0 0 1 1 1 0 0 1 0 1 0 0 1 0 0 1 1 0 1 1 0 1 1 1 1 0 1
 1 1 1 1 1 1 1 1 0 1 1 1 1 0 0 1 0 1 1 0 0 1 1 0 0 1 1 1 1 0 1 1 0 0 0 1 0
 1 0 1 1 1 0 1 1 0 0 1 0 0 0 0 1 0 0 0 1 0 1 0 1 1 0 1 0 0 0 0 1 1 0 0 1 1
 1 0 1 1 1 1 1 0 0 1 1 0 1 1 0 0 1 0 1 1 1 1 0 1 1 1 1 1 0 1 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 1 1 1 1 1 1 0 1 0 1 1 0 1 1 0 1 0 0 1 1 1 1 1 1 1 1 1 1 1 1
 1 0 1 1 0 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 0 1 0 1 1 1 1 0 0 0 1 1
 1 1 0 1 0 1 0 1 1 1 0 1 1 1 1 1 1 1 0 0 0 1 1 1 1 1 1 1 1 1 1 1 0 0 1 0 0
 0 1 0 0 1 1 1 1 1 0 1 1 1 1 1 0 1 1 1 0 1 1 0 0 1 1 1 1 1 1 0 1 1 1 1 1 1
 1 0 1 1 1 1 1 0 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 0 0 1 0 1 1 1 1 1 0 1 1
 0 1 0 1 1 0 1 0 1 1 1 1 1 1 1 1 0 0 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 0 1
 1 1 1 1 1 1 0 1 0 1 1 0 1 1 1 1 1 1 0 0 1 0 1 0 1 1 1 1 1 1 0 1 1 0 1 0 1 0 0
 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 0 1 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 0 0 0 0 0 0 1]
```

In [3]:
```
#split data set into train and test sets

x_train, x_test, y_train, y_test = train_test_split(breast_cancer.data,
                         breast_cancer.target, test_size = 0.5, random_state
= 129)
print(y_test)
```

```
[1 1 0 1 1 0 1 1 0 0 1 0 1 1 0 0 1 0 1 0 1 0 1 0 0 1 1 1 1 0 0 0 1 1 1 0 1
 1 0 0 1 1 0 1 0 1 1 0 0 0 1 1 1 1 1 1 0 1 1 1 1 1 1 0 0 1 1 0 1 1 1 1 1 1
 1 0 1 1 0 1 1 1 1 0 1 1 1 1 1 0 1 1 0 1 0 1 1 1 1 0 1 0 0 0 0 0 1 1 1 1 0
 0 0 1 1 0 0 1 0 1 1 1 1 0 0 0 0 0 0 1 0 0 1 1 0 0 1 0 1 1 0 1 0 0 0 1 1 1
 0 0 1 1 1 0 1 1 1 1 0 0 1 0 0 1 1 0 1 0 1 1 0 1 0 1 1 1 0 0 1 0 1 1 0 0 1
 0 1 0 0 1 1 1 0 1 1 1 1 0 0 1 1 0 0 1 0 1 1 0 1 1 0 1 1 1 1 1 1 1 0 1 1 1
 1 0 0 0 0 0 0 1 0 0 0 0 1 1 1 1 0 0 1 0 1 1 0 1 1 1 1 1 1 0 1 1 1 1 1 0 0
 0 1 0 1 1 0 1 1 1 0 1 1 0 1 1 1 0 1 1 0 1 1 0 0 0 1]
```

In [4]:
```
#Create a Classifier
mnb=MultinomialNB()

# Train the model using the training sets
mnb.fit(x_train,y_train)

#Predict the response for test dataset
y_pred = mnb.predict(x_test)
#print(y_pred)
```

In [5]:
```
# Model Accuracy, how often is the classifier correct?
print("Accuracy:",accuracy_score(y_test, y_pred))

print("\nConfusion Matrix :")
confusion_matrix(y_test, y_pred)
```

```
Accuracy: 0.887719298245614

Confusion Matrix :
```

Out[5]:
```
array([[ 87,  28],
       [  4, 166]])
```

In [6]:
```
precision = precision_score(y_test, y_pred)
print('\nprecision: {}'.format(precision))

recall = recall_score(y_test, y_pred)
print('\nrecall: {}'.format(recall))
```

```
precision: 0.8556701030927835

recall: 0.9764705882352941
```