**First Step:-**

**import all the required libraries**

import numpy as np

from sklearn import preprocessing , neighbors , svm , model\_selection

import pandas as pd

**Second Step:-**

**pre-proccessing**

**----------------------------------------------------------------------------------**

access the dataset by DataFrame or in form of a numpy array

DataFrame:-

csv file

df.read\_csv("FILE NAME OR URL")

array:-

np.array("name")

**----------------------------------------------------------------------------------**

**adjusting all NaN values**

**adjusting all categorical values like ?**

**reshaping arrays**

**and all other work on dataset**

df= pd.read\_csv('[https://archive.ics.uci.edu/ml/machine-learning-databases/breast-cancer-wisconsin/breast-cancer-wisconsin.data')**#READING**](https://archive.ics.uci.edu/ml/machine-learning-databases/breast-cancer-wisconsin/breast-cancer-wisconsin.data')#READING) **DATA**

df.replace('?',-99999,inplace=True)**#ADJUSTING VALUES**

col\_names=['id','clump\_thickness','cell\_size','cell shape','adhesion','epithelial','nuclei','brandchlomatin','nucleoli','mitoses','class']

df.columns=col\_names

print(df.head())

df.drop(['id'],1,inplace=True)**#ADJUSTING VALUES**

**----------------------------------------------------------------------------------**

**Third Step:-**

making X and Y arrays mean train and test data

spliting data into train and test part

X=np.array(df.drop(['class'],1))

y=np.array(df['class'])

X\_train,X\_test,y\_train,y\_test = model\_selection.train\_test\_split(X,y,test\_size=0.2)

**----------------------------------------------------------------------------------**

**Fourth Step:-**

create a model for it

then load train data into it

then check accuracy of model with the help of test data

after all this try to improve the accuracy of model to its maximum by making necessary changes in it

clf=neighbors.KNeighborsClassifier()

clf.fit(X\_train,y\_train)

confidence=clf.score(X\_test,y\_test)

print("Confidence is :- ", int(confidence\*100),'%')

example\_measures=np.array([4,.2,1,1,1,2,3,2,1])

example\_measures.shape=(1,-1)

prediction=clf.predict(example\_measures)

print('Cancer state : ','Benign' if prediction==2 else 'Malignant')

**----------------------------------------------------------------------------------**

**Fifth Step:-**

USE FOR REAL DATA