Lab Program - 3

Ques) **Consider the health care dataset that consists of several imaging details from patients that had a biopsy to test for breast cancer. The variable diagnosis classifies the biopsied tissue as M = malignant or B = benign. Describe and preprocess the dataset. Use KNN supervised learning model to predict Diagnosis using texture\_mean and radius\_mean. Analyze the model using different k values and display the performance of the model.**

# performing linear algebra

import numpy as np

# data processing

import pandas as pd

# visualisation

import matplotlib.pyplot as plt

import seaborn as sns

df = pd.read\_excel("C:/Users/HP/OneDrive/Desktop/DataScience/Customer\_Churn.xlsx")

df.info()

# Data Understanding

df.describe()

#dealing missing values

df.isnull().sum()

#Replacing missing values with mean

df['Seconds of Use'] = df['Seconds of Use'].fillna(df['Seconds of Use'].mean())

df['Frequency of SMS'] = df['Frequency of SMS'].fillna(df['Frequency of SMS'].mean())

#defining X and y variable

X = np.array(df.iloc[0:, 1:])

X

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

X.shape

y.shape

# using training and test set

from sklearn.model\_selection import train\_test\_split

X\_train, X\_test, y\_train, y\_test = train\_test\_split(

X, y, test\_size = 0.20, random\_state = 42)

from sklearn.neighbors import KNeighborsClassifier

knn = KNeighborsClassifier(n\_neighbors = 6)

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

knn.score(X\_test, y\_test)

#Prediction on training data

y\_train\_predict=knn.predict(X\_train)

#confusion matrix

import seaborn as sns

from sklearn.metrics import confusion\_matrix

from sklearn.metrics import precision\_score, recall\_score, f1\_score

classnames=['0','1']

confusionmatrix=confusion\_matrix(y\_train,y\_train\_predict)

sns.heatmap(confusionmatrix,annot=True,yticklabels=classnames,xticklabels=classnames,fmt='g',cmap="Greens");

from sklearn import metrics

print ("Accuracy:",metrics.accuracy\_score(y\_train,y\_train\_predict))

print('Precision Score: %.3f' % precision\_score(y\_train,y\_train\_predict))

print('Recall Score: %.3f' % recall\_score(y\_train,y\_train\_predict))

print('F1 Score: %.3f' % f1\_score(y\_train,y\_train\_predict))

#Prediction on test data

y\_train\_predict=knn.predict(X\_test)

#confusion matrix

import seaborn as sns

from sklearn.metrics import confusion\_matrix

classnames=['0','1']

confusionmatrix=confusion\_matrix(y\_test,y\_train\_predict)

sns.heatmap(confusionmatrix,annot=True,yticklabels=classnames,xticklabels=classnames,fmt='g',cmap="Greens");

from sklearn import metrics

print ("Accuracy:",metrics.accuracy\_score(y\_test,y\_train\_predict))

print('Precision Score: %.3f' % precision\_score(y\_test,y\_train\_predict))

print('Recall Score: %.3f' % recall\_score(y\_test,y\_train\_predict))

print('F1 Score: %.3f' % f1\_score(y\_test,y\_train\_predict))

# predict for a new data

result = knn.predict([[0,38,0,4370.0,71,5.0,17,3,1,1,30,197.64,1]])

if result == 1:

print ("1")

else:

print ("0")