# Importing the libraries

import numpy as np

import matplotlib.pyplot as plt

import pandas as pd

from sklearn.metrics import accuracy\_score

# Importing the dataset

df = pd.read\_csv('cancer.csv')

df.replace('?',-99999,inplace=True)

df.drop(['id'],1,inplace=True)

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

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

# Splitting the dataset into the Training set 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 = 0.35, random\_state = 42)

# Feature Scaling

from sklearn.preprocessing import StandardScaler

sc = StandardScaler()

X\_train = sc.fit\_transform(X\_train)

X\_test = sc.transform(X\_test)

#principle component analysis

from sklearn.decomposition import PCA

pca = PCA(n\_components=2)

X\_train = pca.fit\_transform(X\_train)

X\_test = pca.fit\_transform(X\_test)

explained\_variance=pca.explained\_variance\_ratio\_

# Fitting KNN to the Training set

from sklearn.neighbors import KNeighborsClassifier

knn = []

for i in range(1,21):

classifier = KNeighborsClassifier(n\_neighbors=i)

trained\_model=classifier.fit(X\_train,y\_train)

trained\_model.fit(X\_train,y\_train )

# Predicting the Test set results

y\_pred = classifier.predict(X\_test)

# Making the Confusion Matrix

from sklearn.metrics import confusion\_matrix

cm\_KNN = confusion\_matrix(y\_test, y\_pred)

print(cm\_KNN)

print("Accuracy score of train KNN")

print(accuracy\_score(y\_train, trained\_model.predict(X\_train))\*100)

print("Accuracy score of test KNN")

print(accuracy\_score(y\_test, y\_pred)\*100)

knn.append(accuracy\_score(y\_test, y\_pred)\*100)

plt.figure(figsize=(12, 6))

plt.plot(range(1, 21),knn, color='red', linestyle='dashed', marker='o',

markerfacecolor='blue', markersize=10)

plt.title('Accuracy for different K Value')

plt.xlabel('K Value')

plt.ylabel('Accuracy')

# Fitting SVM to the Training set

from sklearn.svm import SVC

classifier = SVC(kernel = 'linear', random\_state = 0)

trained\_model=classifier.fit(X\_train,y\_train)

trained\_model.fit(X\_train,y\_train )

# Predicting the Test set results

y\_pred = classifier.predict(X\_test)

# Making the Confusion Matrix

from sklearn.metrics import confusion\_matrix

cm\_SVM = confusion\_matrix(y\_test, y\_pred)

print(cm\_SVM)

print("Accuracy score of train SVM")

print(accuracy\_score(y\_train, trained\_model.predict(X\_train))\*100)

print("Accuracy score of test SVM")

print(accuracy\_score(y\_test, y\_pred)\*100)