Exercise 1

```python
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, confusion_matrix
from sklearn.neighbors import KNeighborsClassifier
from matplotlib import pyplot as plt

names = ['class', 'Alcohol','Malic Acid','Ash','Acadlinity','Magnisium','Total Phenols',
         'Flavanoids','NonFlavanoid Phenols', 'Proanthocyanins', 'Color Intensity', 'Hue', 'OD280/OD315', 'Proline' ]
df = pd.read_csv('wine.data.csv', names=names)
X = np.array(df.iloc[:, 1:14])
y = np.array(df['class'])
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.20 )


scores=[]
K_range = range(1, 10)

for K in K_range:
    knn = KNeighborsClassifier(n_neighbors=K)
    knn.fit(X_train, y_train)
    y_pred = knn.predict(X_test)
    scores.append(accuracy_score(y_test, y_pred))

plt.plot(K_range, scores)
plt.xlabel("Value of K for KNN")
plt.ylabel('Testing Accuracy')
plt.show()
```

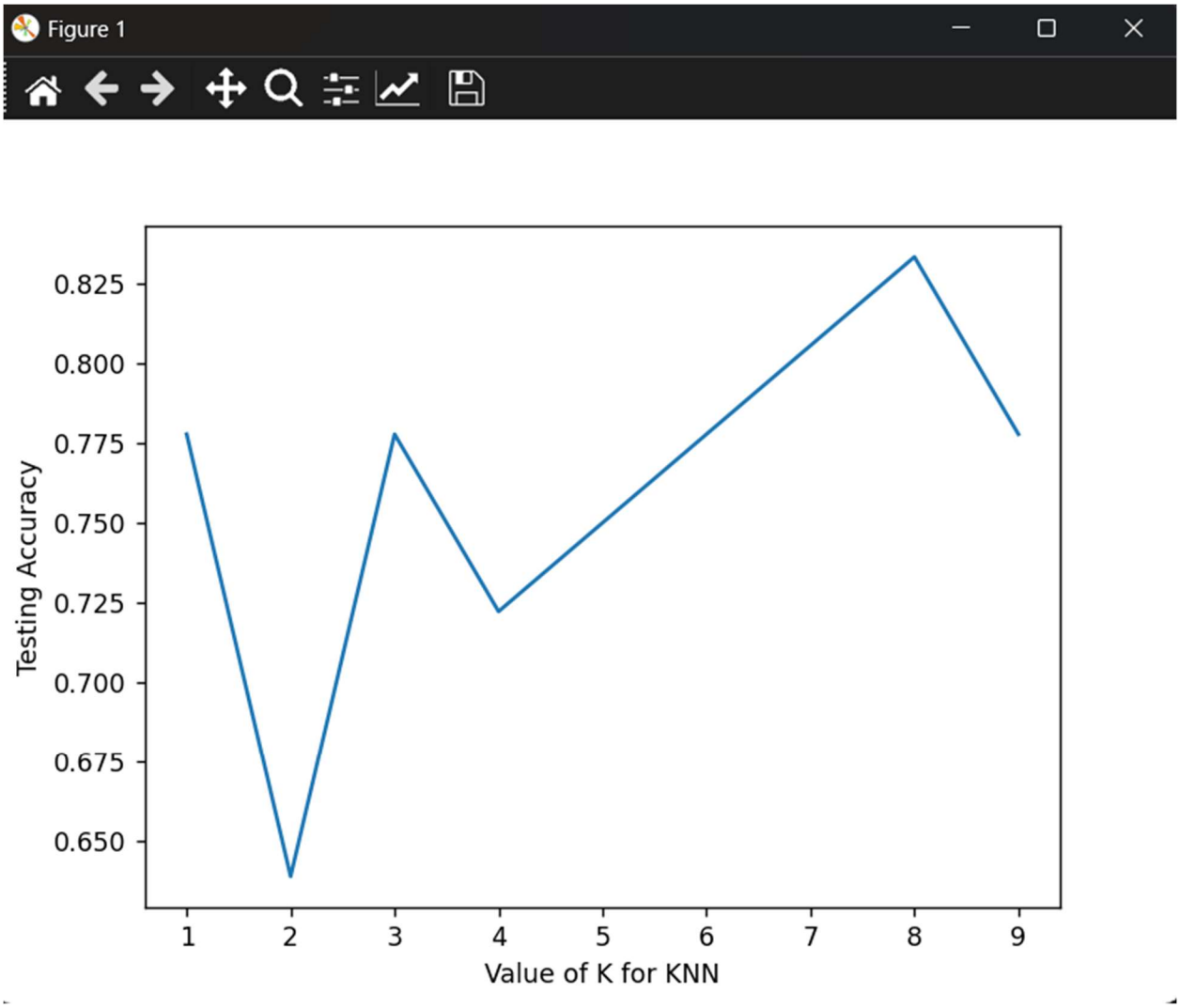
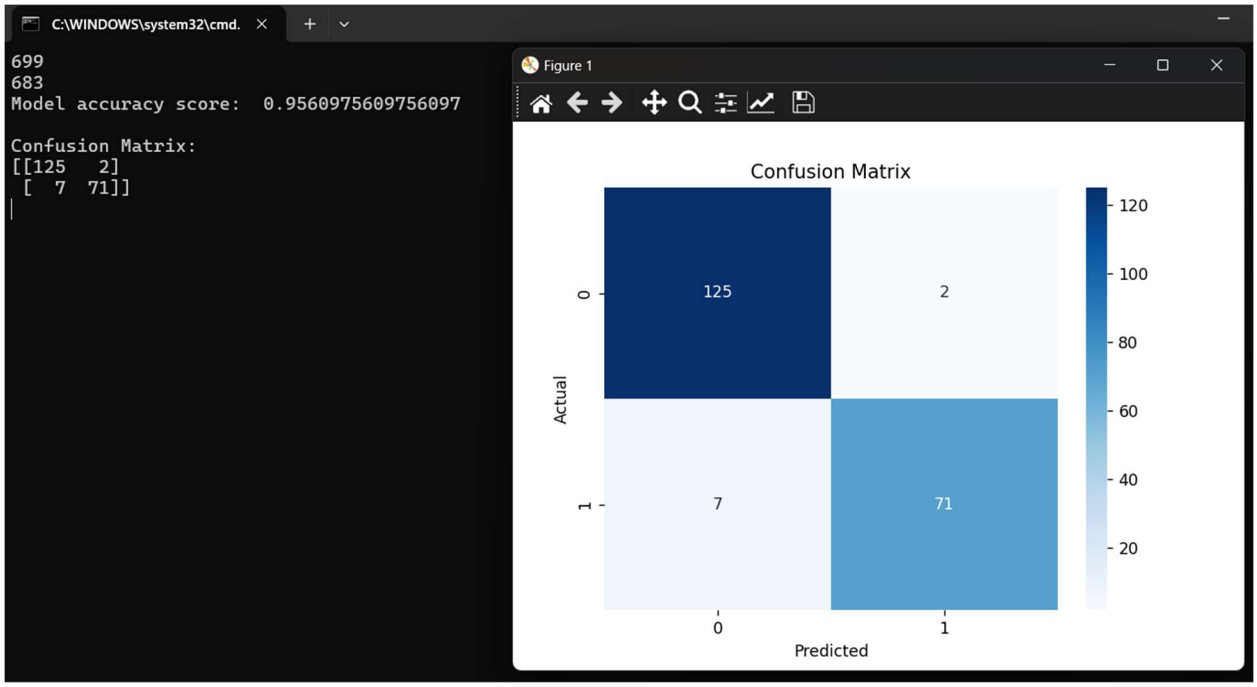


Figure 1
0.825
0.800
0.775
0.750
0.725
0.700
0.675
0.650
Testing Accuracy
1
2
3
4
5
6
7
8
9
Value of K for KNN

Exercise 2

```python
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, confusion_matrix
from sklearn.neighbors import KNeighborsClassifier
import matplotlib.pyplot as plt
import seaborn as sns

names = [ "Sample code number", "Clump Thickness", "Uniformity of Cell Size",
    "Uniformity of Cell Shape", "Marginal Adhesion","Single Epithelial Cell Size",
    "Bare Nuclei", "Bland Chromatin", "Normal Nucleoli","Mitoses","Class"
]
df = pd.read_csv('breast-cancer-wisconsin.data.csv', names=names)

print(len(df))

df.replace('?', pd.NA, inplace=True)
df.dropna(inplace=True)
df.reset_index(drop=True, inplace=True)

print(len(df))

X = np.array(df.loc[:, 'Clump Thickness':'Mitoses'])
y = np.array(df['Class'])

for i in range(len(y)):
    if y[i] == 2:
        y[i] = 0 #  benign
    else:
        y[i] = 1 # mailignant


#use random_state=value to select the same data points in every run
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.30, random_state=42) #set seed to 42 all the time

knn = KNeighborsClassifier(n_neighbors=5)
knn.fit(X_train, y_train)

pred = knn.predict(X_test)

print('Model accuracy score: ', accuracy_score(y_test, pred))

conf_matrix = confusion_matrix(y_test, pred)
print(f'\nConfusion Matrix: \n{conf_matrix}')
```

```python
conf_matrix = confusion_matrix(y_test, pred)
print(f'\nConfusion Matrix: \n{conf_matrix}')

sns.heatmap(conf_matrix, annot=True, fmt='d', cmap='Blues',
xticklabels=knn.classes_, yticklabels=knn.classes_)
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.title('Confusion Matrix')
plt.show()
```


C:\WINDOWS\system32\cmd.
699
683
Model accuracy score: 0.9560975609756097
Confusion Matrix:
[[125 2]
[7 71]]
Figure 1
Confusion Matrix
125
2
7
71
0
1
Actual
Predicted
120
100
80
60
40
20

Exercise 3 (optional)

```python
import pandas as pd
import numpy as np

#intialize df of lockers

myDict = {
    "Locker Num" : np.arange(1, 101),
    "Status" : np.zeros(100)
    }
#0 = closed
#1 = open
#all intiially closed
df = pd.DataFrame(myDict)

def flipBit(x):
    if x == 1:
        return 0
    else:
        return 1


# loop to solve locker problem
for i in range(1, 101):
    for j in range(100):
        if (j + 1) % i == 0:
            current_status = df.loc[j, "Status"]
            df.loc[j, "Status"] = flipBit(current_status)

open_lockers = df[df["Status"] == 1]
print("\nLockers that are open:")
print(open_lockers)
```

```
C:\Users\Aden\anaconda3\py

Lockers that are open:
    Locker Num  Status
0            1     1.0
3            4     1.0
8            9     1.0
15          16     1.0
24          25     1.0
35          36     1.0
48          49     1.0
63          64     1.0
80          81     1.0
99         100     1.0
Press any key to continue . . .
```

I did not get to Exercise 4 (optional)