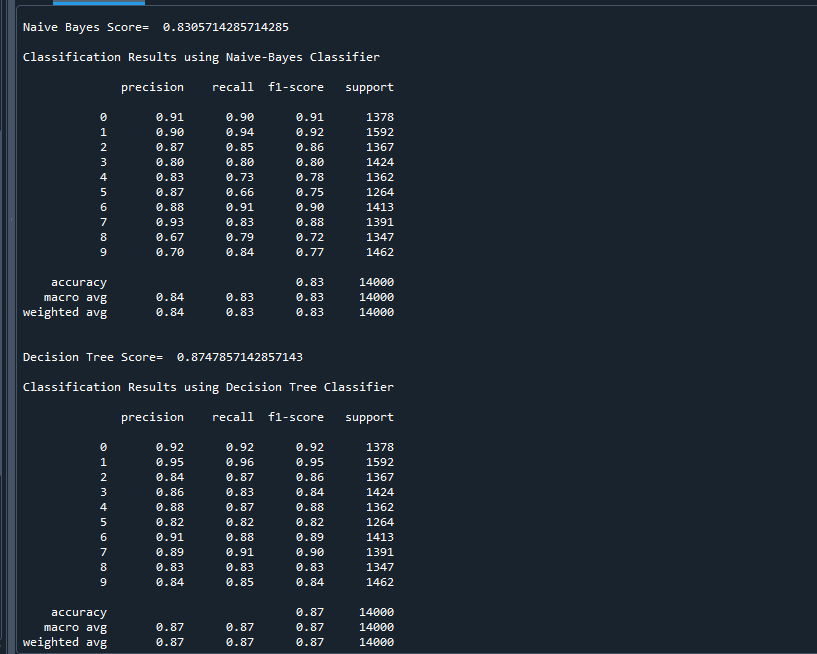
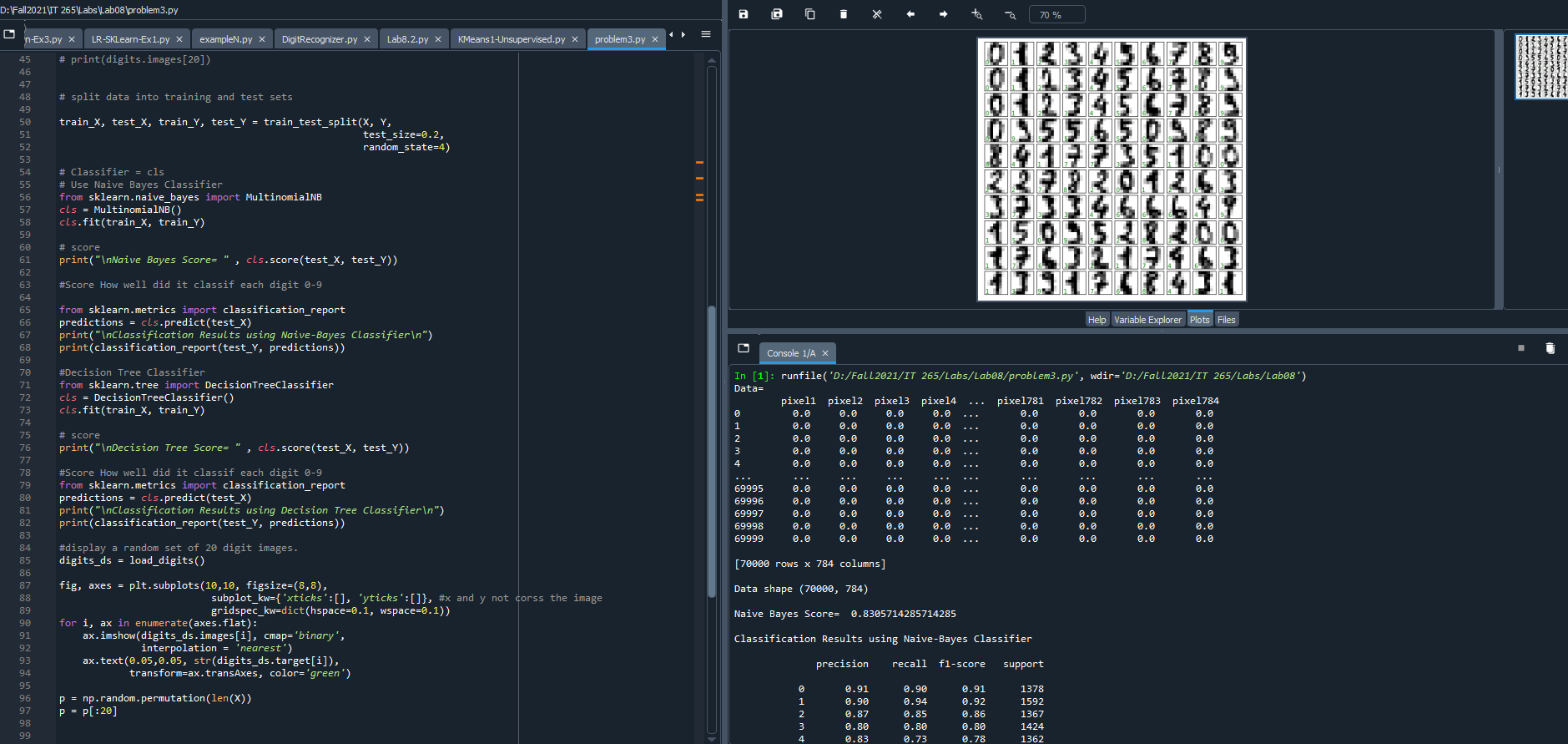
**Student Names: Ghadir Alfadhl & Li Baio**

**Course Name: ITS265** – **Lab08**

**Problem 3:** Import from SKLearn the handwritten digits dataset with 70000 examples. Use

“import fetch\_openml” and the "mnist\_784" dataset. Train the data on two different classifiers – Gaussian Naive Bayes Classifier and a Decision Tree Classifier. Compare the results and explain why one classifier is better at doing this classification than the other. Also, display a random set of 20 digit images.



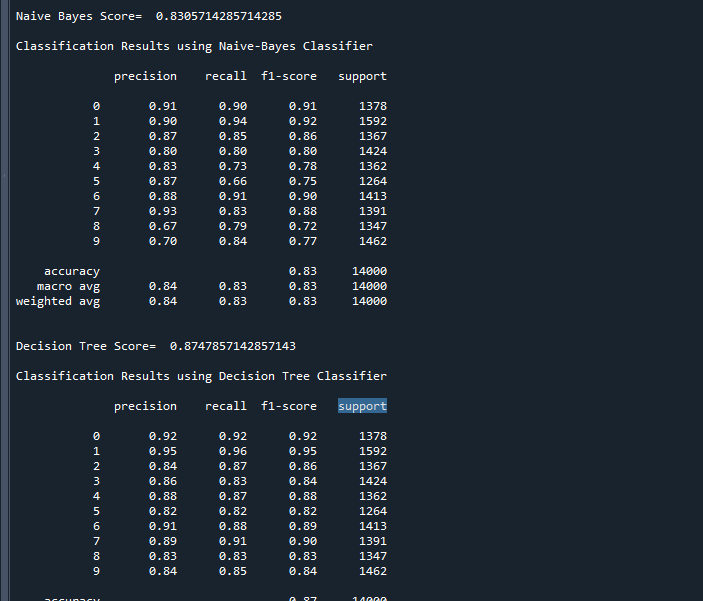


Compare the results and explain why one classifier is better at doing this classification than the other?

Naive Bayes Score= 0.8305714285714285

Decision Tree Score= 0.8747857142857143

The Decision Tree score is the better than Naive Bayes because is it higher and more accrecy. Also, decision trees work better with lots of data compared to Naive Bayes.



**#Code**

import numpy as np

import pandas as pd

from matplotlib import pyplot as plt

from sklearn.datasets import fetch\_openml

from sklearn.linear\_model import LogisticRegression

from sklearn.datasets import load\_digits

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

from sklearn.neural\_network import MLPClassifier

from sklearn import datasets

#mnist = pd.read\_csv("mnist\_784.csv")

#mnist = fetch\_openml(name="mnist\_784")

mnist = fetch\_openml("mnist\_784", version=1)

print("Data=\n", mnist.data)

print("\nData shape",mnist.data.shape)

#print(mnist.target.shape)

#print(mnist)

#print(mnist.target)

#print(mnist.data)

X = mnist['data']

Y = mnist['target']

# digits = datasets.load\_digits()

# print(digits.images[20])

# split data into training and test sets

train\_X, test\_X, train\_Y, test\_Y = train\_test\_split(X, Y,

test\_size=0.2,

random\_state=4)

# Classifier = cls

# Use Naive Bayes Classifier

from sklearn.naive\_bayes import MultinomialNB

cls = MultinomialNB()

cls.fit(train\_X, train\_Y)

# score

print("\nNaive Bayes Score= " , cls.score(test\_X, test\_Y))

#Score How well did it classif each digit 0-9

from sklearn.metrics import classification\_report

predictions = cls.predict(test\_X)

print("\nClassification Results using Naive-Bayes Classifier\n")

print(classification\_report(test\_Y, predictions))

#Decision Tree Classifier

from sklearn.tree import DecisionTreeClassifier

cls = DecisionTreeClassifier()

cls.fit(train\_X, train\_Y)

# score

print("\nDecision Tree Score= " , cls.score(test\_X, test\_Y))

#Score How well did it classif each digit 0-9

from sklearn.metrics import classification\_report

predictions = cls.predict(test\_X)

print("\nClassification Results using Decision Tree Classifier\n")

print(classification\_report(test\_Y, predictions))

#display a random set of 20 digit images.

digits\_ds = load\_digits()

fig, axes = plt.subplots(10,10, figsize=(8,8),

subplot\_kw={'xticks':[], 'yticks':[]}, #x and y not corss the image

gridspec\_kw=dict(hspace=0.1, wspace=0.1))

for i, ax in enumerate(axes.flat):

ax.imshow(digits\_ds.images[i], cmap='binary',

interpolation = 'nearest')

ax.text(0.05,0.05, str(digits\_ds.target[i]),

transform=ax.transAxes, color='green')

p = np.random.permutation(len(X))

p = p[:20]