**Experiment 6**

**Student Name:** Sahil Kaundal **UID:** 21BCS8197

**Branch:** BE CSE (Lateral Entry) **Section/Group:** 616/A

**Semester:** 5th **Date of Performance:** 16/10/2022

**Subject Name:** ML Lab **Subject Code:** 20CSP-317

1. **Aim/Overview of the practical:**

Implement K-Nearest Neighbor on any data set

1. **Task To Be Done:**

Implement K-Nearest Neighbor on any data set using sklearn.

**3. Apparatus / Simulator Used:**

1. Windows 7 or above.
2. Google Collab.

**k-Nearest Neighbors**

The k-Nearest Neighbors algorithm or KNN for short is a very simple technique.

The entire training dataset is stored. When a prediction is required, the k-most similar records to a new record from the training dataset are then located. From these neighbors, a summarized prediction is made.

Similarity between records can be measured many different ways. A problem or data-specific method can be used. Generally, with tabular data, a good starting point is the [Euclidean distance](https://en.wikipedia.org/wiki/Euclidean_distance).

Once the neighbors are discovered, the summary prediction can be made by returning the most common outcome or taking the average. As such, KNN can be used for classification or regression problems.

There is no model to speak of other than holding the entire training dataset. Because no work is done until a prediction is required, KNN is often referred to as a lazy learning method.

**4. Program / Commands:**

#Sahil Kaundal

#21BCS8197

# Import necessary modules

from sklearn.neighbors import KNeighborsClassifier

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

from sklearn.datasets import load\_iris

# Loading data

irisData = load\_iris()

# Create feature and target arrays

X = irisData.data

y = irisData.target

# Split into training and test set

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

X, y, test\_size = 0.2, random\_state=42)

knn = KNeighborsClassifier(n\_neighbors=7)

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

# Predict on dataset which model has not seen before

print(knn.predict(X\_test))

# Import necessary modules

from sklearn.neighbors import KNeighborsClassifier

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

from sklearn.datasets import load\_iris

# Loading data

irisData = load\_iris()

# Create feature and target arrays

X = irisData.data

y = irisData.target

# Split into training and test set

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

X, y, test\_size = 0.2, random\_state=42)

knn = KNeighborsClassifier(n\_neighbors=7)

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

# Calculate the accuracy of the model

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

# Import necessary modules

from sklearn.neighbors import KNeighborsClassifier

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

from sklearn.datasets import load\_iris

import numpy as np

import matplotlib.pyplot as plt

irisData = load\_iris()

# Create feature and target arrays

X = irisData.data

y = irisData.target

# Split into training and test set

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

X, y, test\_size = 0.2, random\_state=42)

neighbors = np.arange(1, 9)

train\_accuracy = np.empty(len(neighbors))

test\_accuracy = np.empty(len(neighbors))

# Loop over K values

for i, k in enumerate(neighbors):

knn = KNeighborsClassifier(n\_neighbors=k)

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

# Compute training and test data accuracy

train\_accuracy[i] = knn.score(X\_train, y\_train)

test\_accuracy[i] = knn.score(X\_test, y\_test)

# Generate plot

plt.plot(neighbors, test\_accuracy, label = 'Testing dataset Accuracy')

plt.plot(neighbors, train\_accuracy, label = 'Training dataset Accuracy')

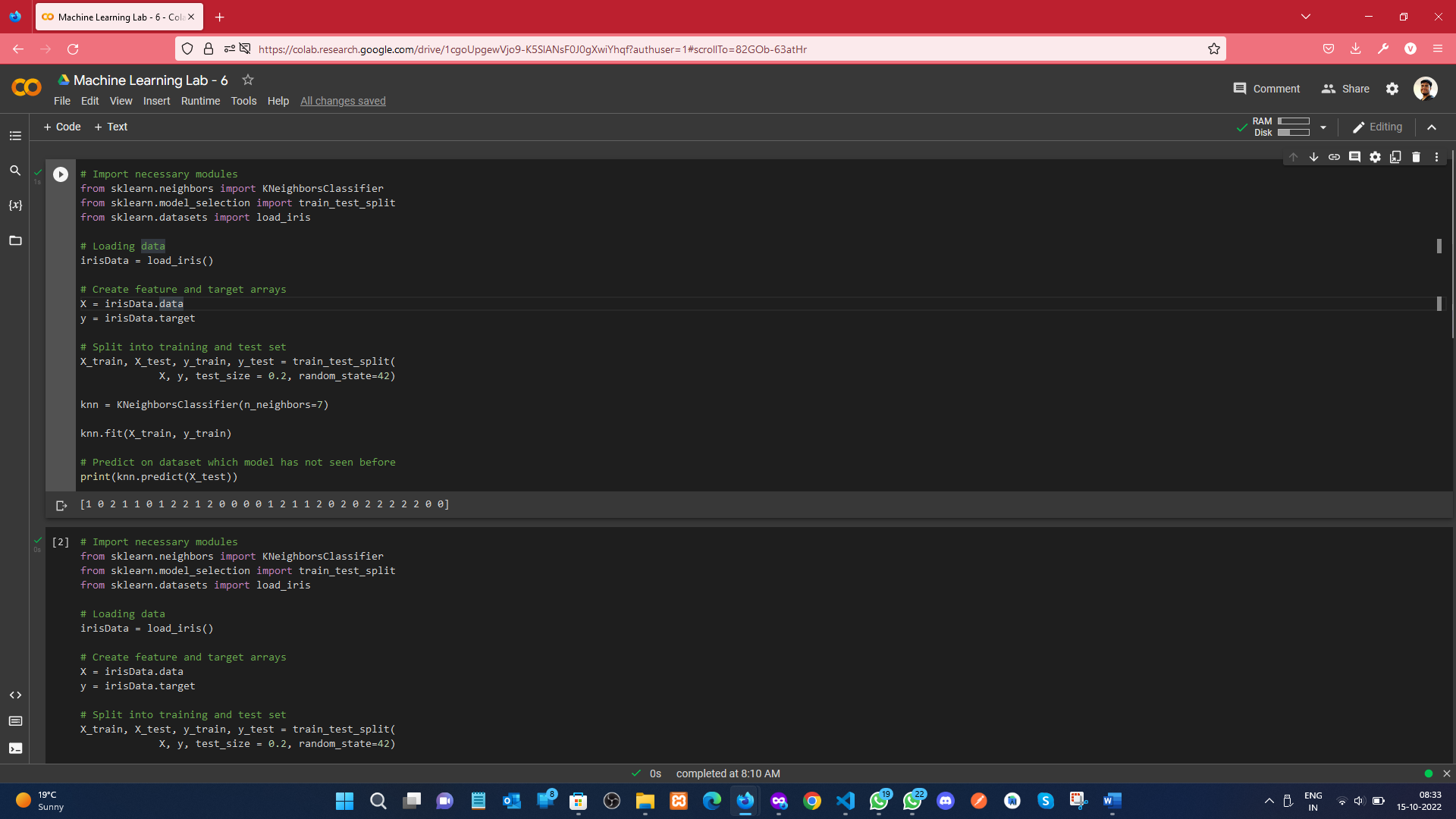
plt.legend()

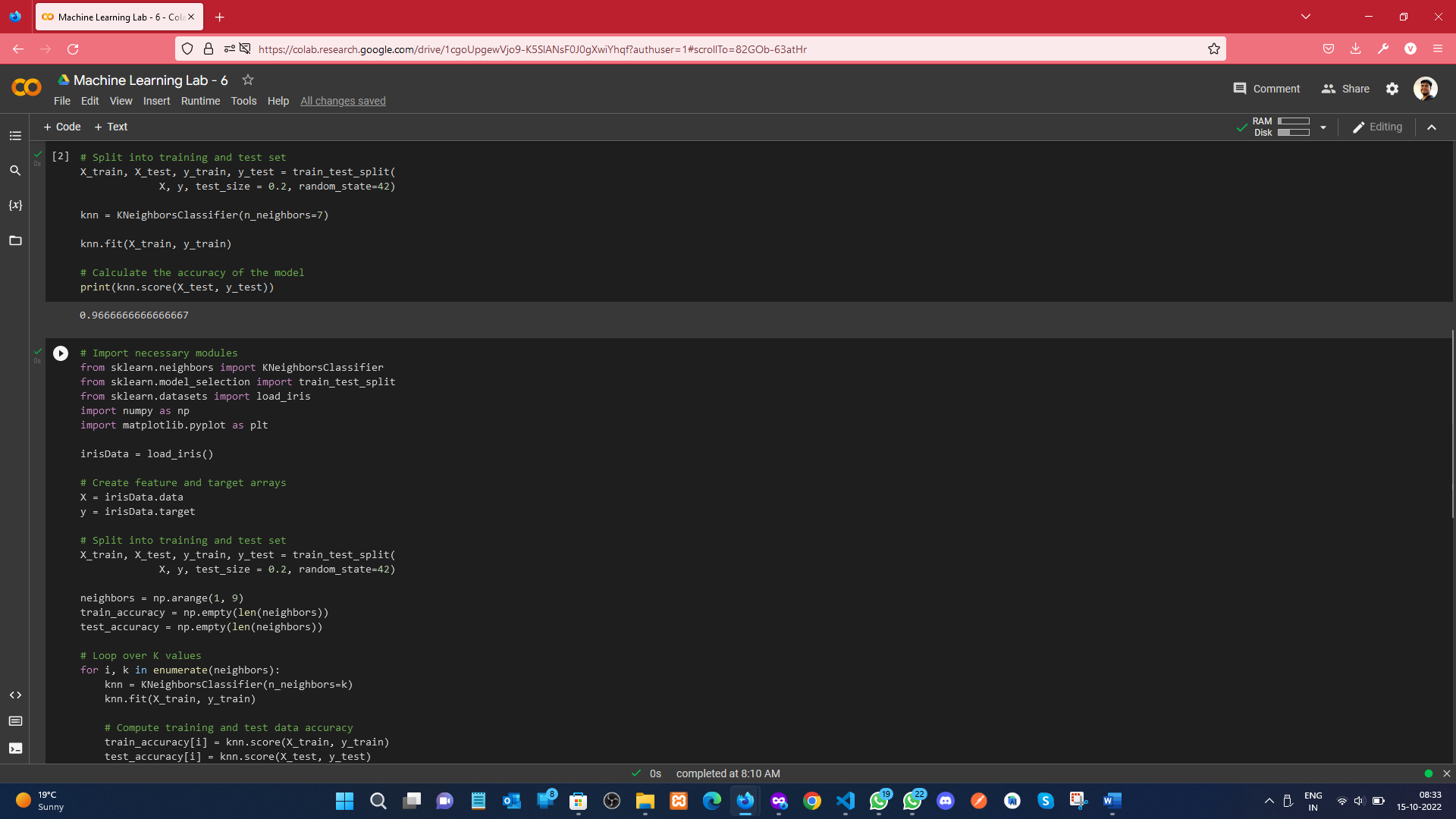
plt.xlabel('n\_neighbors')

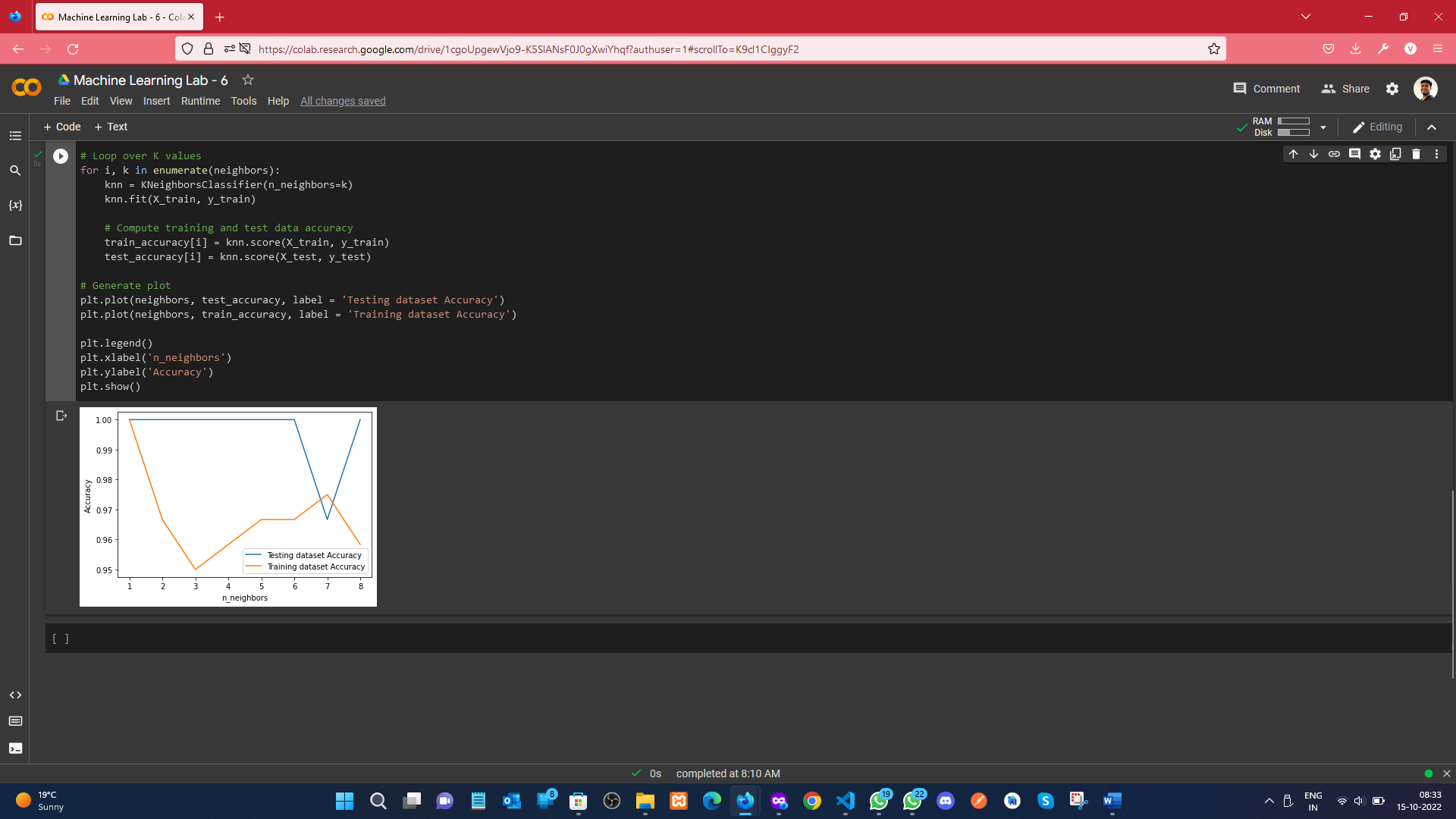
plt.ylabel('Accuracy')

plt.show()

1. **Result/Output/Writing Summary:**







**Learning outcomes (What I have learnt):**

* Understood the concept of KNeighborsClassifier
* Learnt how to load the iris dataset, and splitting the dataset.
* Predicting the test data on KNN.
* Plot the legend graph of Accuracy of the Dataset.
* **Evaluation Grid (To be created as per the SOP and Assessment guidelines by the faculty):**

|  |  |  |  |
| --- | --- | --- | --- |
| Sr. No. | Parameters | Marks Obtained | Maximum Marks |
| 1. |  |  |  |
| 2. |  |  |  |
| 3. |  |  |  |
|  |  |  |  |