**CHANDIGARH UNIVERSITY**

**UNIVERSITY INSTITUTE OF ENGINEERING**

**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING**



|  |  |
| --- | --- |
| **Submitted By: Kanishk Soni Submitted To: Er. Sudhanshu Sharma** | |
| **Subject Name** | **Machine Learning Lab** |
| **Subject Code** | **20CSP-317** |
| **Branch** | **BE-CSE** |
| **Semester** | **5th** |

**EXPERIMENT-6**

**Student’s Name: Kanishk Soni UID: 20BCS9398**

**Section/Group: 707\_WM\_B Subject Code: 20CSP-317**

**Subject Name: ML Lab Date of performance:2/11/2022**

**Branch: BE CSE Semester:5th**

**Aim:** Implement KNN (K Nearest Neighbour).

**Objective:** To do KNN on data set.

**Software/Hardware Requirements:** Windows 7 & above version.

**Tools to be used:**

1. Anaconda Jupyter Notebook,
2. numpy, matplotlib, sklearn.

**Introduction to KNN:**

KNN is a simple, supervised machine learning (ML) algorithm that can be used for classification or regression tasks - and is also frequently used in missing value imputation. It is based on the idea that the observations closest to a given data point are the most "similar" observations in a data set, and we can therefore classify unforeseen points based on the values of the closest existing points. By choosing K, the user can select the number of nearby observations to use in the algorithm.

**Code:**

--

# 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

--

# 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=6)

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

# Predict on dataset which model has not seen before

print(knn.predict(x\_test))

--

# Split into training and test set

neighbors = np.arange(1, 9)

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

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

--

# calculate the accuracy of the model

print(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()

**Output:**





