**Classification in Machine Learning**

**Using K-Nearest Neighbour**

**Theory:**

The intuition behind the KNN algorithm is one of the simplest of all the supervised machine learning algorithms. It simply calculates the distance of a new data point to all other training data points. The distance can be of any type e.g Euclidean or Manhattan etc. It then selects the K-nearest data points, where K can be any integer. Finally it assigns the data point to the class to which the majority of the K data points belong.

**Implementing KNN Algorithm with Scikit-Learn**

**Importing Libraries**

In [3]: import numpy as np

import matplotlib.pyplot as plt

import pandas as pd

**Importing Dataset**

In [4]: url = "D:/As a Trainer/Freelance Training/ML-NIVT/Classificatio

n/fruits.csv"

dataset = pd.read\_csv(url)

**Displaying dataset**

In [5]: dataset.head()

Out[5]:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **Sl** | **Name** | **Category** | **Supplier** | **Brand** | **Price Group** |
| 0 | 1 | Apple | A | A.Co. | Nutts | P1 |
| 1 | 2 | Apple | A | Fruits\_Lover | Fruits#1 | P2 |
| 2 | 3 | Lemon | B | Fruits\_Lover | Fruits#1 | P2 |
| 3 | 4 | Watermelon | B | A.Co. | Fruits#1 | P1 |
| 4 | 5 | Apple | B | A.Co. | Costa  Rica | P2 |

**Preprocessing**

In [6]: X = dataset.iloc[:, 6:].values

y = dataset.iloc[:, 4].values

#print(X)

#print(y)

**Train Test Split**

In [8]: from sklearn.model\_selection import train\_test\_split

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

ize=0.20)

**Feature Scaling**

In [9]: from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()

scaler.fit(X\_train)

X\_train = scaler.transform(X\_train)

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

**Training and Predictions**

In [10]: from sklearn.neighbors import KNeighborsClassifier

classifier = KNeighborsClassifier(n\_neighbors=5)

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

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

**Evaluating the Algorithm**

In [11]: from sklearn.metrics import classification\_report, confusion\_mat

rix

print(confusion\_matrix(y\_test, y\_pred))

print(y\_pred)

[[0 1 1 0]

[1 1 2 0]

[0 0 1 1]

[1 2 1 0]]

['Fruits#1' 'Fruits#1' 'Del Monte' 'Fruits#1' 'Del Monte' 'Co

sta Rica'

'Fruits#1' 'Del Monte' 'Costa Rica' 'Fruits#1' 'Nutts' 'Del

Monte']