CREATE A CHATBOT IN PYTHON

# NAME: Abhinand R

# REG NO: 720421104001

# INTRODUCTION :

**K-Nearest Neighbours (KNN)** is a popular machine learning algorithm used for classification and regression tasks. It is a **lazy learning**, non-parametric algorithm that uses data with several classes to predict the classification of the new sample point. KNN is non-parametric since it doesn’t make any assumptions on the data being studied.

During the training phase, the KNN algorithm stores the entire training dataset as a reference. When implementing an algorithm, you will always need a data set. So, you start by loading the training and the test data. Then, you choose the nearest data points (the value of K). K can be any integer.

The working of KNN Algorithm in Machine Learning can be summarized in three steps:

1. Load the data
2. Choose the nearest data points (the value of K)
3. Do the following, for each test data –
   * Calculate the distance between test data and each row of training data
   * Sort the calculated distances in ascending order based on distance values
   * Get top K rows from sorted array
   * Get the most frequent class of these rows
   * Return this class as output.

# PROCESS:

# 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))

OUTPUT:

[1 0 2 1 1 0 1 2 2 1 2 0 0 0 0 1 2 1 1 2 0 2 0 2 2 2 2 2 0 0]

# 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))

OUTPUT:

0.9666666666666667

# 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)

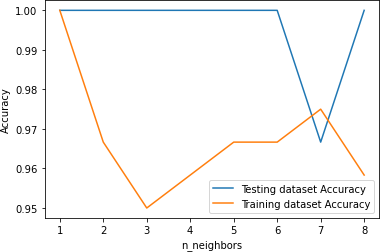
# Generate plot

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()

OUTPUT:



**CONCLUSION:**

In this article, we covered the workings of the KNN algorithm and its implementation in Python. It’s one of the most basic yet effective machine- learning models. For KNN implementation in R, you can go through this tutorial: [kNN Algorithm using R](https://www.analyticsvidhya.com/blog/2015/08/learning-concept-knn-algorithms-programming/). You can also go for our free course – [K-](https://courses.analyticsvidhya.com/courses/K-Nearest-Neighbors-KNN-Algorithm?utm_source=blog&utm_medium=knn_for_regression) [Nearest Neighbors (KNN) Algorithm in Python and R](https://courses.analyticsvidhya.com/courses/K-Nearest-Neighbors-KNN-Algorithm?utm_source=blog&utm_medium=knn_for_regression), to further your foundations of KNN.

In this article, we used the KNN model directly from the *scikit-learn* library. You can also implement KNN from scratch (I recommend this!), which is covered in this article: [KNN simplified](https://www.analyticsvidhya.com/blog/2018/03/introduction-k-neighbours-algorithm-clustering/).