**KNN classifier**

The K-NN algorithm compares a new data entry to the values in a given data set (with different classes or categories).

Based on its closeness or similarities in a given range (K) of neighbors, the algorithm assigns the new data to a class or category in the data set (training data).

**Algorithm**

Step #1 - Assign a value to K.

Step #2 - Calculate the distance between the new data entry and all other existing data entries (you'll learn how to do this shortly). Arrange them in ascending order.

Step #3 - Find the K nearest neighbors to the new entry based on the calculated distances.

Step #4 - Assign the new data entry to the majority class in the nearest neighbors.

#KNN classifier

import pandas as pd

import numpy as np

df = pd.read\_csv('/content/sample\_data/Pokemon.csv')

df.head()

df.drop(columns=['#','Name',  'Type 1', 'Type 2'],inplace=True)

df

df.Legendary.value\_counts()

# Import label encoder

from sklearn import preprocessing

label\_encoder = preprocessing.LabelEncoder()

df['Legendary']= label\_encoder.fit\_transform(df['Legendary'])

df['Legendary'].unique()

df

df.Legendary.value\_counts()

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

pokemon\_features = df.drop("Legendary",axis=1)

target = df["Legendary"]

X\_train,X\_test,Y\_train,Y\_test = train\_test\_split(pokemon\_features,target,test\_size=0.20,random\_state=0)

from sklearn.metrics import accuracy\_score

from sklearn.neighbors import KNeighborsClassifier

knn = KNeighborsClassifier(n\_neighbors=7)

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

Y\_pred\_knn=knn.predict(X\_test)

score\_knn = round(accuracy\_score(Y\_pred\_knn,Y\_test)\*100,2)

print("The accuracy score achieved using KNN is: "+str(score\_knn)+" %")

