KNN - K Nearest Neighbour

KNN - KNN which stands for K Nearest Neighbour is supervised machine learning algorithm used for classification and regression tasks. It works by finding the k closest data points (neighbors) to a new, unseen data point in the feature space & predict the output based on those neighbors

- For classification, KNN assigns the class most common among the k nearest neighbors

- For regression, KNN predicts the value by averaging the values of the k-nearest neighbor

KNN relies on distance metric (like Euclidean distance) to measure how close neighbors are:

Eg

Person	Weather	Feeling	Want to go gym?
A	Sunny	Energytic	Yes
B	Rainy	Lazy	No
C	Cloudy	Normal	Yes

Person B also say he also want to go gym → this is K-Nearest Neighbor

~~4 steps~~

~~[illegible]~~

~~[illegible]~~

~~[illegible]~~

Training data points and labels:

→ # Input ← Output

Point #	Weight (g)	Size (cm)	Label (0 = Apple, 1 = Orange)
1	180	7	0
2	200	7.5	0
3	250	8	0
4	300	8.5	1
5	330	9	1
6	360	9.5	1

Step 1: Calculating Euclidean distance from new point P to each training point

Formula:

$$\text{distance} = \sqrt{(x - xi)^2 + (y - yi)^2}$$

P = (x, y) :- This is our new point

Q = (xi, yi) :- Training point

1. New Point P = (200, 10), Training point Q = (180, 7):

$$d_1 = \sqrt{(200 - 180)^2 + (10 - 7)^2} = \sqrt{20^2 + 3^2} = \sqrt{400 + 9} = 20.2$$

2. $d_2 = \sqrt{(200 - 200)^2 + (10 - 7.5)^2} = 2.5$

3. $d_3 = \sqrt{(200 - 250)^2 + (10 - 8)^2} = 50.04$

4. $d_4 = \sqrt{(200 - 300)^2 + (10 - 8.5)^2} = 100.01$

5. $d_5 = \sqrt{(200 - 330)^2 + (10 - 9)^2} = 130.00$

6. $d_6 = \sqrt{(200 - 360)^2 + (10 - 9.5)^2} = 160.00$

KNN formula

$$Distance = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2 + \dots + (x_n - y_n)^2}$$

Date
Page

Step 2 : List distances with labels

Point	Distance	Label
1	20.22	0
2	2.5	0
3	50.04	0
4	100.01	1
5	130.00	1
6	160.00	1

Step 3 : Sort by distance (ascending)

→ [(2.5, 0), (20.22, 0), (50.04, 0), (100.01, 1), (130.00, 1), (160.00, 1)]

Step 4 : Pick the 3 smallest distances and their labels

→ [(2.5, 0), (20.22, 0), (50.04, 0)]

All labels are 0 (Apple)

Step 5 : Final prediction

→ Since all 3 nearest neighbors are labeled Apple, KNN predicts your new input [200, 10] as Apple.

- If your data has more than 2 features (more columns)?
  General Euclidean distance for n-dimensional points
  - New input Point is : $P = (x_1, x_2, x_3 \dots, x_n)$
  - Training Point is : $Q = (x_{i1}, x_{i2}, x_{i3} \dots x_{in})$

$$d = \sqrt{(x_1 - x_{i1})^2 + (x_2 - x_{i2})^2 + \dots + (x_n - x_{in})^2}$$

or

$$d = \sqrt{\sum_{j=1}^{n} (x_j - x_{ij})^2}$$

$(x_i - y_i)^2$

KNN - Code

```
from sklearn.neighbors import KNeighborsClassifier

# input contains weight in (grams) and size in (cm)
X = [ [180, 7],
      [200, 7.5],
      [250, 8],
      [300, 8.5],
      [330, 9],
      [360, 9.5]
]

# 0 = Apple and 1 = Orange
y = [0, 0, 0, 1, 1, 1]

# creating and object and passing parameter this should
  be odd numbers only
model = KNeighborsClassifier(n_neighbors = 3)

# train the model
model.fit(X, y)

# user input
weight = float(input('Enter the weight in grams'))
size = float(input('Enter the size in cm'))
# Testing the model
prediction = model.predict([[weight, size]])[0]

if prediction == 1:
  print('This is likely an Apple')
else:
  print('This is likely an Orange')
```

(This is used because it will tell us that how many nearest neighbors the algorithm will look at when making a prediction for a new data point)