# Machine Learning & Deep Learning

Week-9

**Instructor:** *Najam Aziz*

# Supervised Learning Algorithm

*K-Nearest Neighbor (KNN)*

# Example-01. Classification

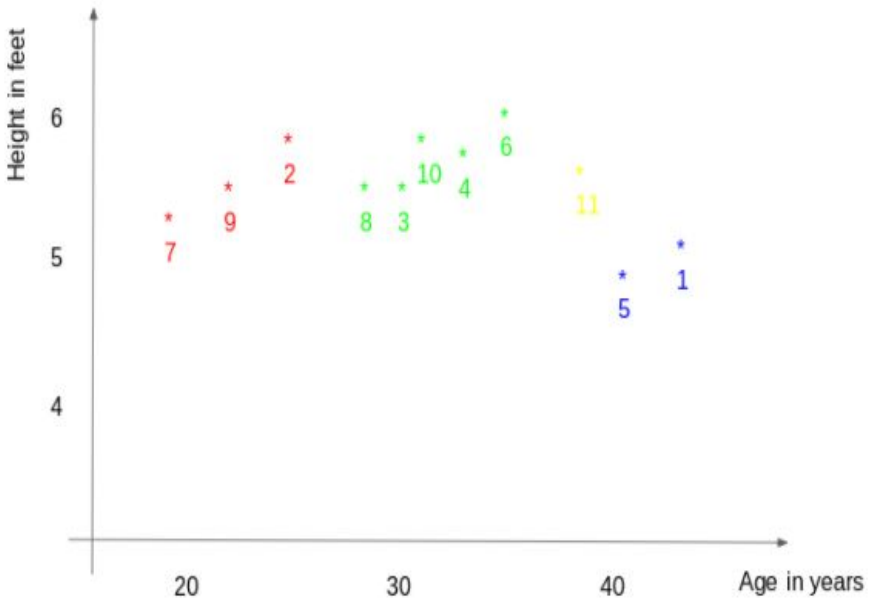
| Name | Age | Gender | Sport |
|---|---|---|---|
| A | 32 | M = 0 | Football |
| B | 40 | M | Neither |
| C | 16 | F = 1 | Cricket |
| D | 34 | F | Cricket |
| E | 55 | M | Neither |
| F | 40 | M | Cricket |
| G | 20 | F | Neither |
| H | 15 | M | Cricket |
| I | 55 | F | Football |
| J | 15 | M | Football |
| **Z** | **5** | **F** | **?** |

| Distance |
|---|
| 27.02 |
| 35.01 |
| 11 |
| 9.00 |
| 50.01 |
| 35.01 |
| 15 |
| 10.00 |
| 50 |
| 10.05 |

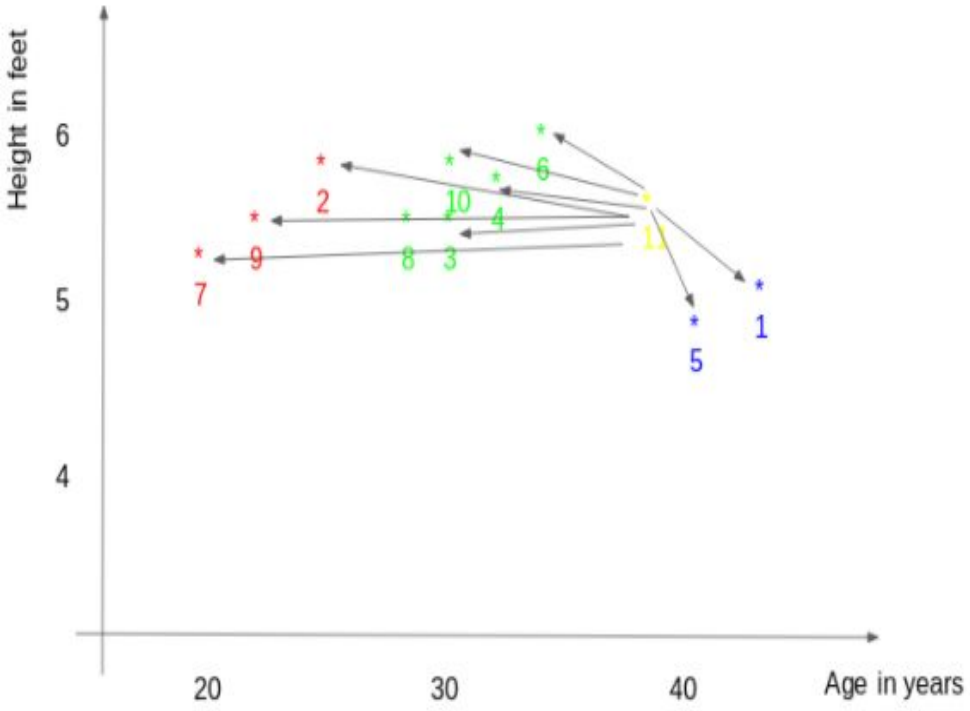
If K = 3, K = 5

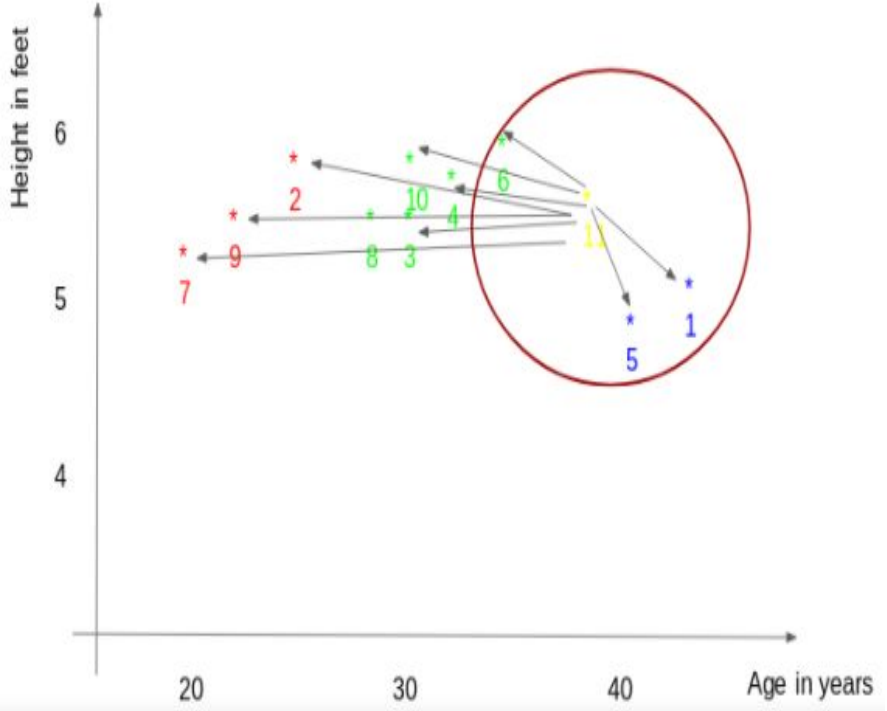
# Example-02. Regression

| ID | Height | Age | Weight |
|---|---|---|---|
| 1 | 5 | 45 | 77 |
| 2 | 5.11 | 26 | 47 |
| 3 | 5.6 | 30 | 55 |
| 4 | 5.9 | 34 | 59 |
| 5 | 4.8 | 40 | 72 |
| 6 | 5.8 | 36 | 60 |
| 7 | 5.3 | 19 | 40 |
| 8 | 5.8 | 28 | 60 |
| 9 | 5.5 | 23 | 45 |
| 10 | 5.6 | 32 | 58 |
| 11 | 5.5 | 38 | ? |



# Example02 -Regression



# KNN- Assumption

- The KNN algorithm assumes that similar things exist in close proximity. In other words, similar things are near to each other.
- KNN captures the idea of similarity (sometimes called distance, proximity, or closeness) with some mathematics — ***calculating the distance between points on a graph.***
- Multiple ways of calculating distance b/w two points, and one way might be preferable depending on the problem we are solving. However, the straight-line distance (also called the Euclidean distance) is a popular and familiar choice.

# How does KNN Work?

- As the name (K Nearest Neighbor) suggests it considers K Nearest Neighbors (Data points that have minimum distance in feature space from our new data point) to predict the class or continuous value for the new Datapoint.
- And K is the number of such data points we consider in our implementation of the algorithm.
- Therefore, distance metric and K value are two important considerations while using the KNN algorithm.

# How does KNN Work?

Euclidean distance is the most popular distance metric. You can also use Hamming distance, Manhattan distance, Minkowski distance as per your need. For predicting class/ continuous value for a new data point, it considers all the data points in the training dataset. Finds new data point's 'K' Nearest Neighbors (Data points) from feature space and their class labels or continuous values.

Then:

- **For classification:** A class label assigned to the majority of K Nearest Neighbors from the training dataset is considered as a predicted class for the new data point.

- **For regression:** Mean(in case of normal distribution) or median(in case of skewed distribution, bcz then it is a better measure of central tendency) of continuous values assigned to K Nearest Neighbors from training dataset is a predicted continuous value for our new data point

# Distance Formula (Euclidean Distance)

**Distance between two points**

Given the two points (x1, y1) and (x2, y2), the distance d between these points is given by the formula:

$$d = \sqrt{[(x2 - x1)^2 + (y2 - y1)^2]}$$

**Note: General use case of Distance**

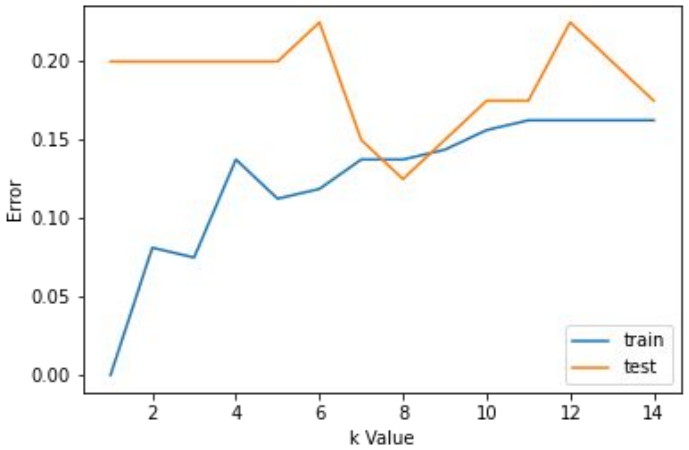
**Continuous Variable** : Euclidean & Manhattan Distance

**Categorical Variable:** Hamming Distance

# *How to choose the value for K?*

To select the K that's right for your data, we run the KNN algorithm several times with different values of K and choose the K that reduces the number of errors we encounter while maintaining the algorithm's ability to accurately make predictions when it's given data it hasn't seen before.

1. Using error curves
2. Domain Knowledge
3. Binary classification -K = odd



# Overfitting & Underfitting

- Overfitting imply that the model is well on the training data but has poor performance when new data is coming.
- Underfitting refers to a model that is not good on the training data and also cannot be generalized to predict new data.

**Note:** It’s very important to have the right k-value when analyzing the dataset to avoid overfitting and underfitting of the dataset.

# Use Case: what Kind of problem it can solve?

KNN is simple, easy-to-implement supervised machine learning algorithm that can be used to solve both

- Classification Problems
- Regression Problems
- Search Problem

# KNN Pros & Cons

**Pros:**

- The algorithm is simple and easy to implement.
- There’s no need to build a model, tune several parameters, or make additional assumptions.
- The algorithm is versatile. It can be used for classification, regression, and search

**Cons:**

- The algorithm gets significantly slower as the number of examples and/or predictors/independent variables increase.
- as the volume of data increases it makes it an impractical choice in environments where predictions need to be made rapidly.
- useful in solving problems that have solutions that depend on identifying similar objects. An example of this is using the KNN algorithm in recommender systems, an application of KNN-search.
- major drawback of becoming significantly slows as the size of that data in use grows.

# Required Data Preparation

1. **Data Scaling:** To locate the data point in multidimensional feature space, it would be helpful if all features are on the same scale. Hence normalization or standardization of data will help.

2. **Dimensionality Reduction:** KNN may not work well if there are too many features. Hence dimensionality reduction techniques like feature selection, principal component analysis can be implemented.

3. **Missing value treatment:** If out of M features one feature data is missing for a particular example in the training set then we cannot locate or calculate distance from that point. Therefore deleting that row or imputation is required.

# KNN application -Recommender Systems

- Recommending products on Amazon, articles on Medium, movies on Netflix, or videos on YouTube. Although, we can be certain they all use more efficient means of making recommendations due to the enormous volume of data they process.
  However, we could replicate one of these recommender systems on a smaller scale using KNN

- **Task:** Given a movies data set, what are the 5 most similar movies to a movie query?