Introduction to Euclidean Distance and KNN

Calculating Nearest Neighbors

Emp_ID	Age	Salary
75	25	85000
22	29	75000
78	28	78000
56	38	47000
7	22	53000
72	25	74000
60	29	60000
70	36	64000
50	39	58000

Which 3 employees are closest to employee with ID 7?

Based on **Age**:

ID: 75,78,72

Based on **Salary**:

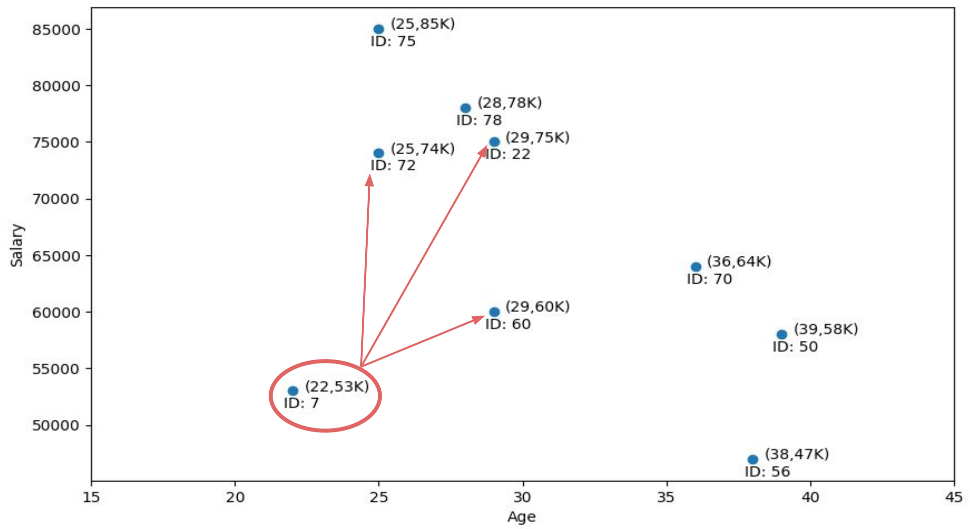
ID: 56,50,60

Based on Both **Age** and **Salary**:

?

Visualizing Distance Measures

Emp_ID	Age	Salary
75	25	85000
22	29	75000
78	28	78000
56	38	47000
7	22	53000
72	25	74000
60	29	60000
70	36	64000
50	39	58000



Visually, the nearest three points are with ID 22,60 and 72.

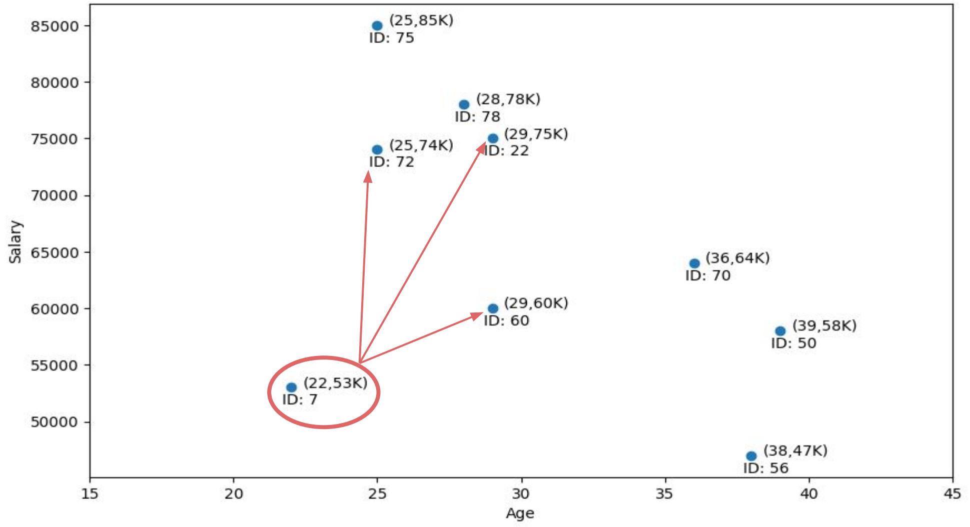
But how can we quantify it?

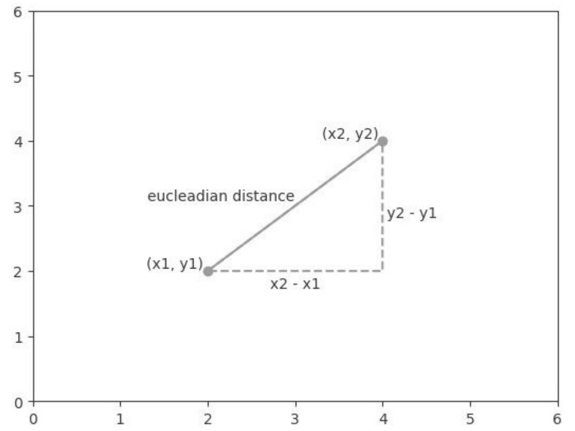
Common Distance Metrics

Some common metrics to quantify distance in machine learning:

- **Euclidean Distance:**
  The straight-line distance between two points in space, calculated as the square root of the sum of the squared differences between corresponding coordinates.
- **Manhattan Distance:**
  The sum of the absolute differences of the coordinates, useful in grid-like paths or environments. Also known as the L1 distance.
- **Minkowski Distance:**
  A generalization of Euclidean and Manhattan distances, defined as the p-th root of the sum of the absolute differences raised to the power of p. It becomes the Euclidean distance when p = 2 and the Manhattan distance when p = 1.
- **Cosine Similarity:**
  Measures the cosine of the angle between two non-zero vectors, indicating their orientation rather than magnitude. Commonly used in text analysis to assess similarity between documents.

Understanding Euclidean Distance



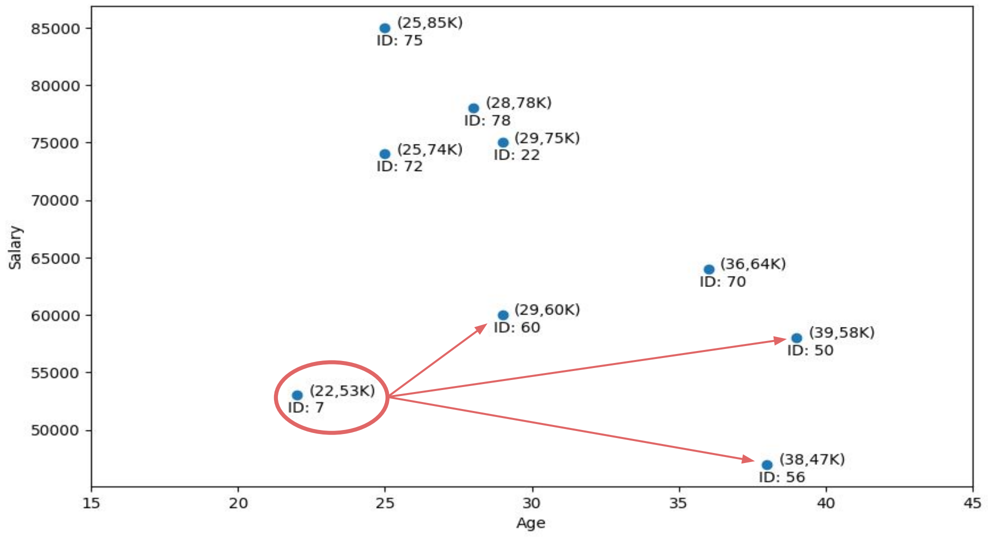


According to Pythagorean Theorem, the square of the length of the hypotenuse is equal to the sum of the squares of the lengths of the other two sides. The Euclidean distance formula is directly derived from the Pythagorean Theorem. In a two-dimensional Cartesian coordinate system, the Euclidean distance between two points (x1,y1) and (x2,y2) is the square root of the hypotenuse.

$$distance = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

Calculating Euclidean Distance

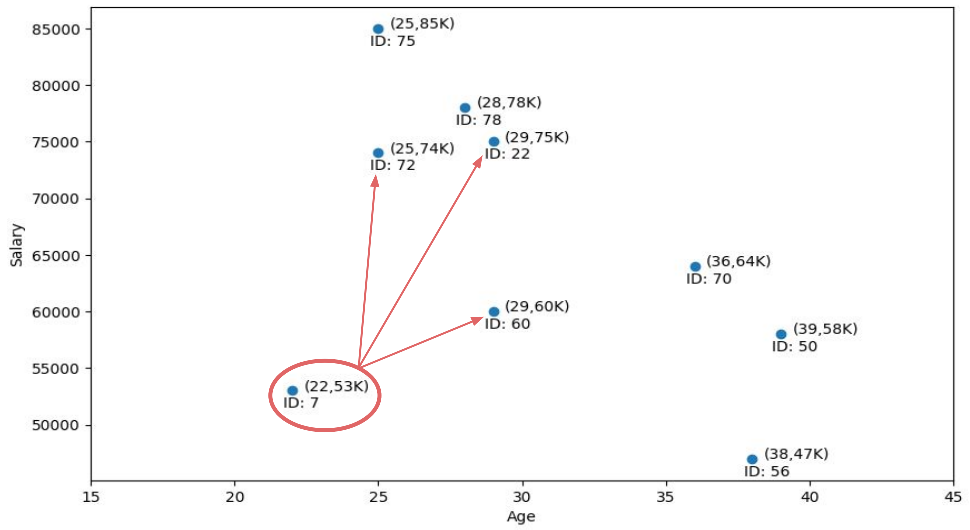
Emp_ID	Age	Salary	Distance
7	22	53000	0
50	39	58000	5000
56	38	47000	6000
60	29	60000	7000
70	36	64000	11000
72	25	74000	21000
22	29	75000	22000
78	28	78000	25000
75	25	85000	32000



Since Salary has higher magnitude than age, it is dominating the distance measure.

Normalized Euclidean Distance

Emp_ID	Age	Salary	Distance
7	0	0.16	0
60	0.41	0.34	0.45
72	0.18	0.71	0.58
22	0.41	0.74	0.71
78	0.35	0.82	0.75
75	0.18	1	0.86
70	0.82	0.45	0.87
56	0.94	0	0.95
50	1	0.29	1.01



When data is normalized, both Age and Salary have same magnitude; therefore, they are treated equally in distance calculation.

Applications of Euclidean Distance

K-Nearest Neighbors (K-NN):
Euclidean distance is often used as a similarity measure in the K-NN algorithm. It helps identify the k-nearest neighbors to a given data point based on the distance in the feature space.

K-Means Clustering:
Euclidean distance is used to assign data points to clusters in the K-means clustering algorithm. The algorithm minimizes the sum of squared Euclidean distances between data points and their cluster centroids.

Dimensionality Reduction:
Euclidean distance is sometimes used as a measure of dissimilarity in dimensionality reduction techniques like Multidimensional Scaling (MDS) or t-Distributed Stochastic Neighbor Embedding (t-SNE).

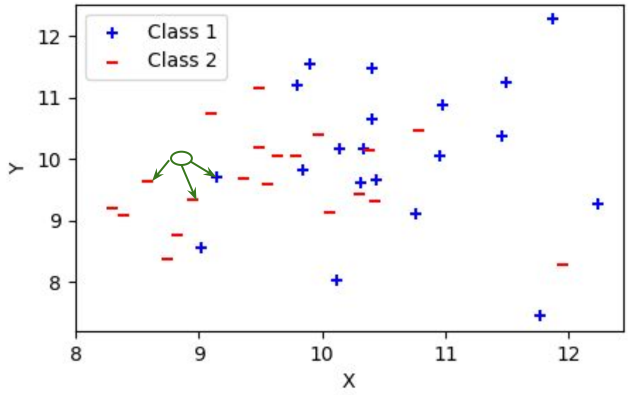
Introduction to K-Nearest Neighbors (K-NN)

What is KNN:

- K-NN is a simple, yet powerful algorithm used for both classification and regression tasks in machine learning. It is based on the concept that similar items are usually close to each other. The prediction in KNN is made by looking at its **'k'** closest neighbor data points.
- It is non-parametric, meaning it does not learn any parameter during training.
- It is an instance-based learning. Unlike model based learnings that learns a generalized model during the training phase, instance-based learning algorithms wait until the prediction phase to generalize the data. This means that the learning process is deferred until a query is made to the system, at which point the algorithm uses the training instances to respond.

KNN Classification

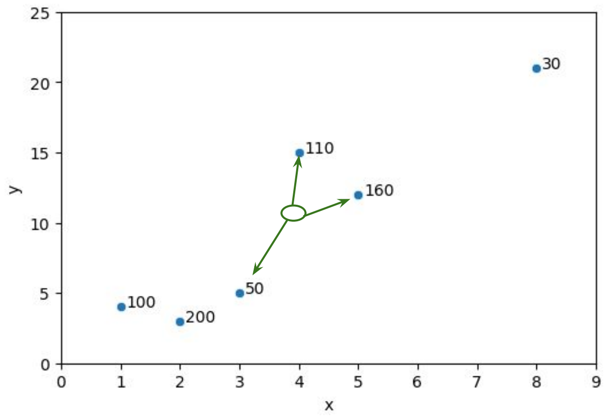
- In KNN classification, the output is a class membership. The data point (needing classification) gets its class from the most common class among its k nearest points, based on a majority vote.
- First, the distance from the query data point to all training samples are calculated to find the k nearest points.
- Once the k nearest points are found, the classification is done by a majority vote among these neighbors. Each neighbor votes for their class, and the class with the highest number of votes is the prediction of the KNN algorithm for the query data point.



With K = 3, the predicted class is (-)

KNN Regression

- KNN regression is an extension of the KNN algorithm applied to regression problems, where the goal is to predict a continuous output variable.
- First, the distance from the query data point to all training samples are calculated to find the k nearest points.
- Unlike classification, where the outcome is determined by a majority vote, KNN regression predicts the value of the query point by averaging the values of the k nearest neighbors.

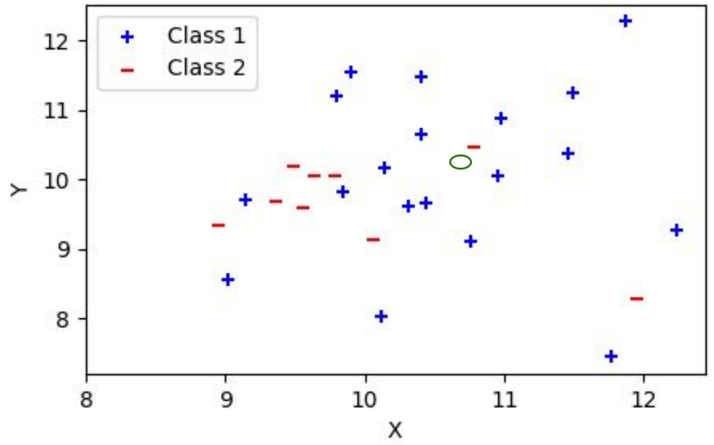


With K = 3, the predicted value is 106.7

Choosing the Right Value of K

- **Too Small k:** Using K=1 in KNN makes the model pay too much attention to the small details or outliers in the training data, rather than the overall trend.

- **Too Large k:** Setting K very high, for example equal to the total number of data points, makes the model too simple and causes it to underperform.



Selecting the optimal K involves multiple strategies. Cross-validation aids in determining the k value with the highest accuracy by partitioning the dataset and evaluating model performance across different subsets. The Square Root Rule offers a heuristic suggesting an initial k value approximately equal to the square root of the dataset size, providing a starting point for experimentation.

Limitations of KNN

Limitations of KNN include:

- KNN can be quite slow with large datasets because it requires computing the distance between each query instance and all training examples to determine the nearest neighbors.
- KNN works effectively when the number of features is small. As the number of features increases, the performance of KNN tends to degrade. This is due to the increased difficulty in calculating meaningful distances and identifying true nearest neighbors among a larger number of dimensions.
- Since KNN requires storing the entire training dataset for prediction, it can be memory-intensive, especially as dataset sizes grow.
- KNN can struggle with binary or categorical variables since it is not always clear how to calculate distances between such variables in a way that is meaningful for the KNN algorithm.