Machine Learning

K-Nearest Neighbors (KNN)

Supervised Learning – Classification

Class Objectives

By the end of this class, students will be able to:

- Understand what K-Nearest Neighbors (KNN) is
- Explain why KNN is a lazy learning algorithm
- Select an appropriate value of K
- Perform KNN classification numerically step-by-step
- Identify advantages and limitations of KNN

What Is K-Nearest Neighbors (KNN)?

K-Nearest Neighbors (KNN) is a supervised learning algorithm that:

- Stores all training data
- Classifies a new data point based on similarity
- Uses distance measures (most commonly Euclidean distance)

No explicit training phase — prediction happens at runtime.

Why Is KNN Called a Lazy Algorithm?

KNN is a lazy learner because:

- It does not build a model
- It simply memorizes training data
- Computation is deferred until prediction time

Where as, Eager learners like Decision Trees, build a model during training.

Distance Measure Used in KNN

Most commonly used distance: Euclidean Distance

For two points $\mathbf{x} = (x_1, x_2)$ and $\mathbf{a} = (a_1, a_2)$:

$$d(\mathbf{x}, \mathbf{a}) = \sqrt{(x_1 - a_1)^2 + (x_2 - a_2)^2}$$

Squared Euclidean Distance (often used to avoid square roots):

$$d^2(\mathbf{x}, \mathbf{a}) = (x_1 - a_1)^2 + (x_2 - a_2)^2$$

How to Choose the Value of K?

Rule of Thumb

- Start with $k = 1$
- Increase k gradually
- Choose k with minimum error on a validation set

Practical Guidelines

- Use odd values of k (to avoid ties)
- Small $k \rightarrow$ Overfitting
- Large $k \rightarrow$ Underfitting

Overfitting v/s Underfitting

Aspect	Overfitting	Underfitting
Definition	Model learns training data too well, including noise	Model is too simple to capture patterns
Training Error	Very low	High
Test Error	High	High
Bias	Low	High
Variance	High	Low
Generalization	Poor	Poor
Simple Example	Memorizing exam questions instead of understanding concepts	Using only addition to solve all math problems

Bias and Variance

Bias

The error that occurs when a model is too simple to capture the true patterns in the data.

- High bias: The model oversimplifies, misses patterns and underfits the data.
- Low bias: The model captures patterns well and is closer to the true values.

Variance

Variance arises when a model becomes too sensitive to training data and it captures noises in data too. It fails to give prediction on unseen new data.

- High variance: The model is too sensitive to small changes and may overfit.
- Low variance: The model is more stable but might miss some patterns.

Goal: Find the right balance between bias and variance for optimal generalization.

Effect of K on Model Behavior

Small k

- Low bias
- High variance
- Sensitive to noise
- Risk of overfitting

Large k

- High bias
- Low variance
- Smoother decision boundary
- Risk of underfitting

KNN Algorithm – Step by Step

1. Choose the value of k
2. Compute distance between query point and all training points
3. Sort distances in ascending order
4. Select k nearest neighbors
5. Take majority vote (classification)
6. Assign the predicted class

Numerical Example – Problem Statement

Attributes:

- X_1 = Acid Durability
- X_2 = Strength

Target variable:

- Good / Bad

Query (new sample):

$$X_1 = 3, \quad X_2 = 7$$

Goal: Predict whether the tissue is Good or Bad

Step 1: Choose k

Let $k = 3$.

Note: Odd value avoids class ties.

Training Data (Labeled Samples)

Sample	X_1 (Acid)	X_2 (Strength)	Class
A	2	7	Good
B	3	6	Good
C	4	8	Bad
D	5	5	Bad
E	1	4	Good
F	7	7	Bad
G	3	8	Bad

Step 2: Compute Distances

Use Squared Euclidean Distance to avoid square roots:

$$d^2\big((3,7),(a_1,a_2)\big) = (3-a_1)^2 + (7-a_2)^2$$

Sample	Point (a_1, a_2)	d^2 to $(3,7)$	Class
A	(2, 7)	$(3-2)^2 + (7-7)^2 = 1$	Good
B	(3, 6)	$(3-3)^2 + (7-6)^2 = 1$	Good
G	(3, 8)	$(3-3)^2 + (7-8)^2 = 1$	Bad
C	(4, 8)	$(3-4)^2 + (7-8)^2 = 2$	Bad
D	(5, 5)	$(3-5)^2 + (7-5)^2 = 8$	Bad
E	(1, 4)	$(3-1)^2 + (7-4)^2 = 13$	Good
F	(7, 7)	$(3-7)^2 + (7-7)^2 = 16$	Bad

Step 3: Sort Distances (Ascending)

Order of samples by increasing d^2:

1. A (1, Good), B (1, Good), G (1, Bad)
2. C (2, Bad)
3. D (8, Bad)
4. E (13, Good)
5. F (16, Bad)

Step 4: Select k Nearest Neighbors ($k = 3$)

Nearest 3 neighbors: A, B, G

- Neighbor 1 → A (Good)
- Neighbor 2 → B (Good)
- Neighbor 3 → G (Bad)

Step 5: Majority Voting ($k = 3$)

Count class labels among A, B, G:

- Good = 2
- Bad = 1

Majority class = Good

Final Prediction (for $k = 3$)

The new paper tissue with $X_1 = 3$, $X_2 = 7$ is classified as:

GOOD

Comparison with $k = 5$

Select nearest 5 neighbors: A, B, G, C, D

- Good: A, B → 2
- Bad: G, C, D → 3

Result for $k = 5$: **BAD**

This shows how increasing k can change the prediction by smoothing decision boundaries and reducing sensitivity to very-close points.

Applications of KNN

- Classification problems
- Missing value estimation
- Pattern recognition
- Document similarity
- Gene expression analysis
- Image and handwriting recognition

Practice

- Recompute predictions for a different query, e.g., $(X_1, X_2) = (4, 7)$, for $k = 3$ and $k = 5$.