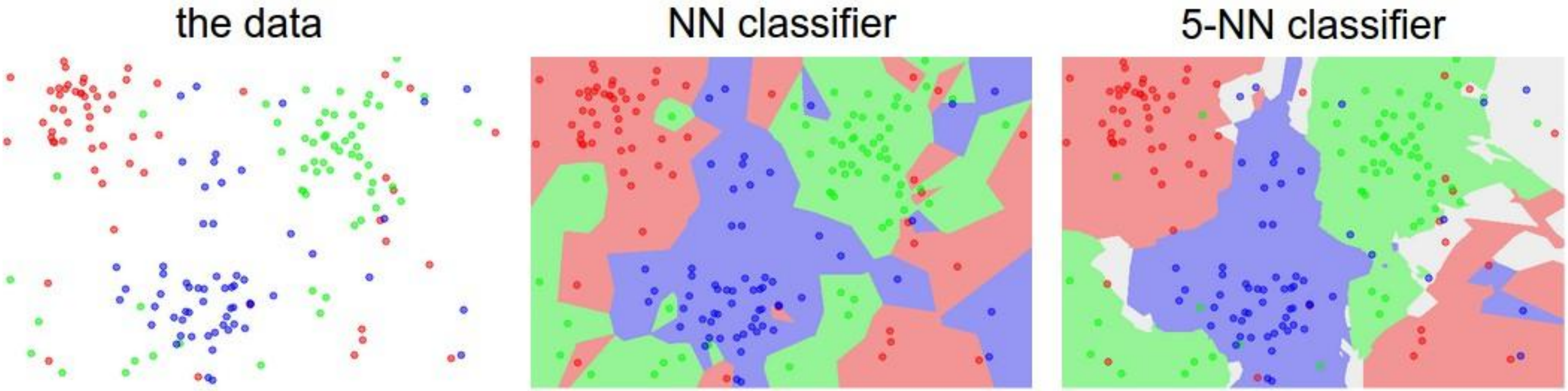
K-Nearest Neighbors

Hrachya Asatryan

K-Nearest Neighbors (KNN)

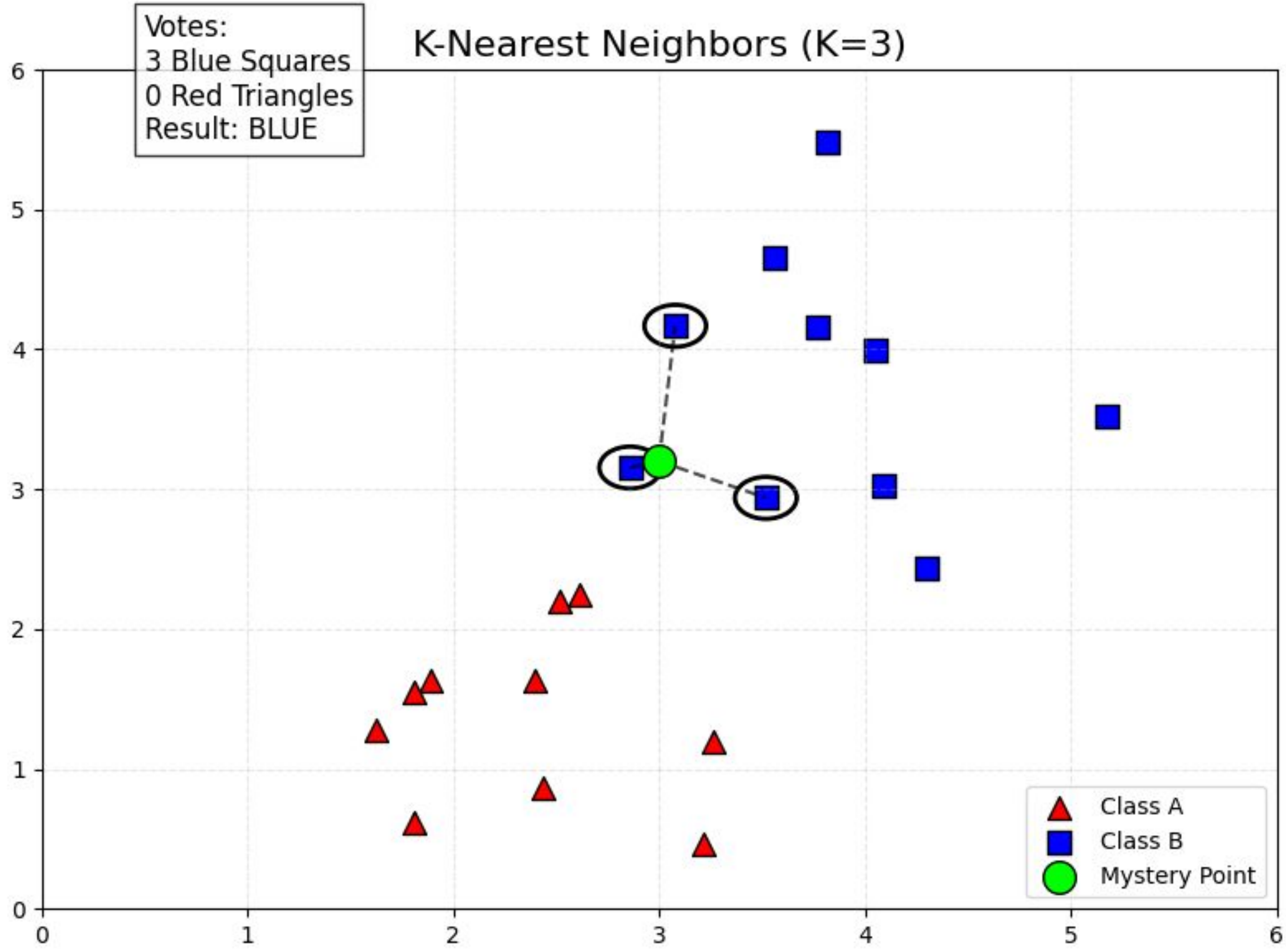
- **The Intuition:** "Tell me who your friends are, and I'll tell you who you are."
- **The Logic:**
  - We assume similar things exist in close proximity.
  - If a point is sitting in a sea of "Blue" points, it is probably "Blue."
- **Type of Learning:**
  - **Supervised:** Requires labeled data.
  - **Instance-Based (Lazy):** No training phase! It just memorizes the data.

K-NN decision regions



The Algorithm (Step-by-Step)

- **Input:** Dataset (x_k, y_k) and a new observation x.
- **Step 1:** Choose a number K (e.g., 3).
- **Step 2:** Choose a **Distance Metric** (How do we measure "Close"?).
- **Step 3:** Find the K points in the data closest to x.
- **Step 4 (Classification):**
  - **Vote:** Count the labels of the K neighbors.
  - **Decision:** The majority wins. (Tie-breaker: Pick randomly or weight by distance).


K-Nearest Neighbors (K=3)
Votes:
3 Blue Squares
0 Red Triangles
Result: BLUE
Class A
Class B
Mystery Point
0
1
2
3
4
5
6
0
1
2
3
4
5
6

Mathematical definition

Our aim is to find the maximum of

$$P(Y = 0|X = x) \text{ and } P(Y = 1|X = x).$$

In this case, we approximate the Probabilities in the following way: first we take a natural number k and define $NN_k(x)$ = The set of k Nearest Points x_i from x. Then we take

$$P(Y = 0|X = x) \approx \#\{y_i = 0 \text{ and } x_i \in NN_k(x)\} \,/\, k\,,$$

and, similarly,

$$P(Y = 1|X = x) \approx \#\{y_i = 1 \text{ and } x_i \in NN_k(x)\} \,/\, k\,,$$

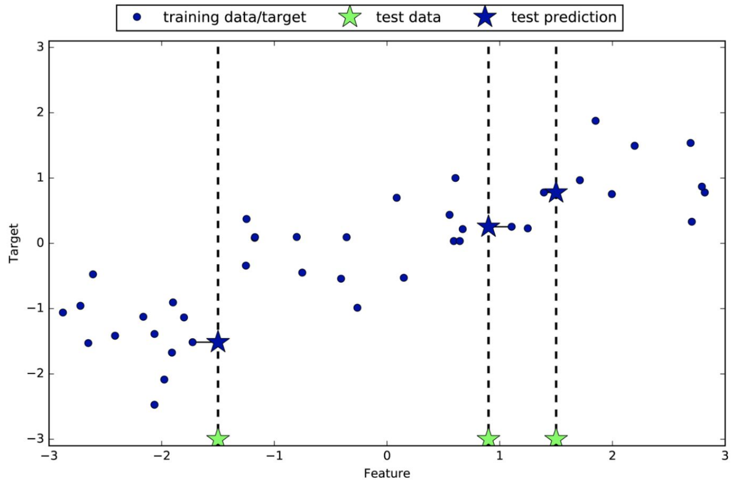
Mathematical definition

Now, to compare the Probabilities of labels, it is enough to compare the numerators

$$\#\{y_i = 0 \text{ and } x_i \in NN_k(x)\} \text{ and } \#\{y_i = 1 \text{ and } x_i \in NN_k(x)\}$$

And we will predict that label, for which the above number is maximal. This is the idea of the k-NN. Of course, we need to define a distance to calculate the Nearest Points to x.

Visualization



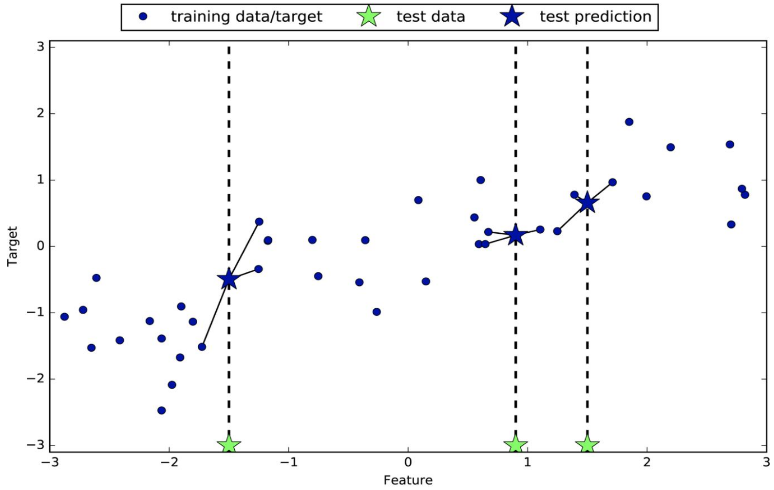


figure 2 . predictions make by three- nearest-neighbors regression on the wave dataset

Figure taken from:

https://medium.com/analytics-vidhya/k-neighbors-regression-analysis-in-python-61532d56d8e4

Measuring Similarity: Distance Metrics

Euclidean Distance (L2 Norm):

- The "Ruler" distance. Straight line.
- *NumPy:* `np.linalg.norm(a - b)`

Manhattan Distance (L1 Norm):

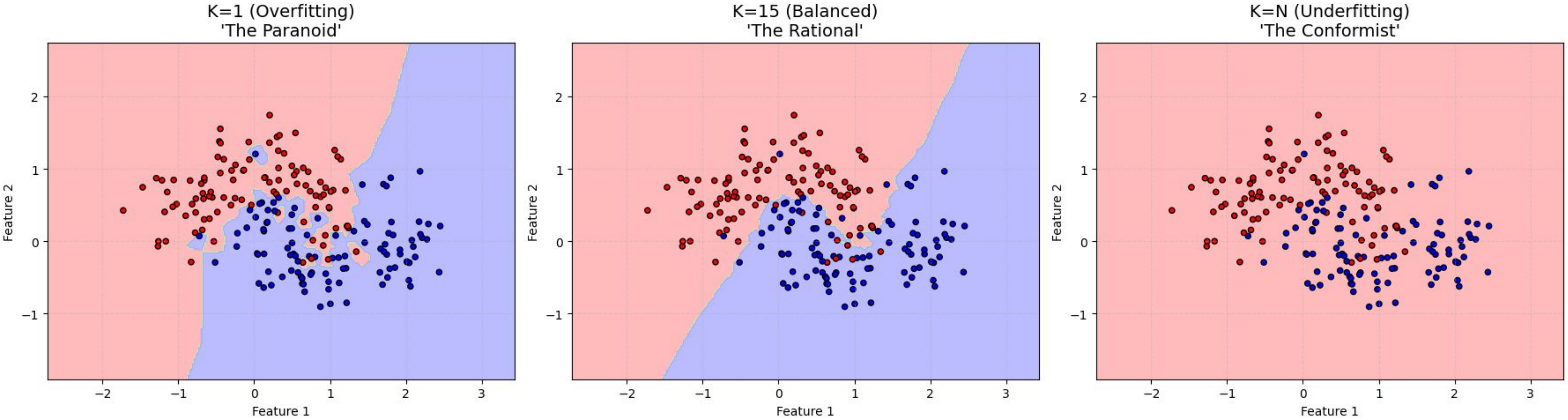
- The "Taxi" distance. Grid movement only.
- *Use case:* High dimensional sparse data.

NOTE: There are many different distance metrics for each specific type of data.

The "K" in KNN (Hyperparameters)

- **What is a Hyperparameter?** A setting *we* choose before the model runs (unlike weights, which the model learns).
- **Case 1: K=1 (The Over-Reactor)**
  - The decision boundary is jagged and chaotic.
  - Every outlier forces the boundary to curve around it.
  - Captures Noise.
- **Case 2: K=100 (The Conformist)**
  - The boundary is smooth and simple.
  - Subtle patterns are ignored.
  - Misses the point.

K-NN decision regions



KNN is not just for Classification

- **KNN Regression:**
  - Instead of voting (Red vs Blue), we **Average**.
  - If neighbors have values `[10,12,14]`, prediction is `12`.
- **The Effect of K on Regression:**
  - Small K: The prediction line looks shaky (follows every bump).
  - Large K: The prediction line looks flat (averages everything out).

KNN Regression example

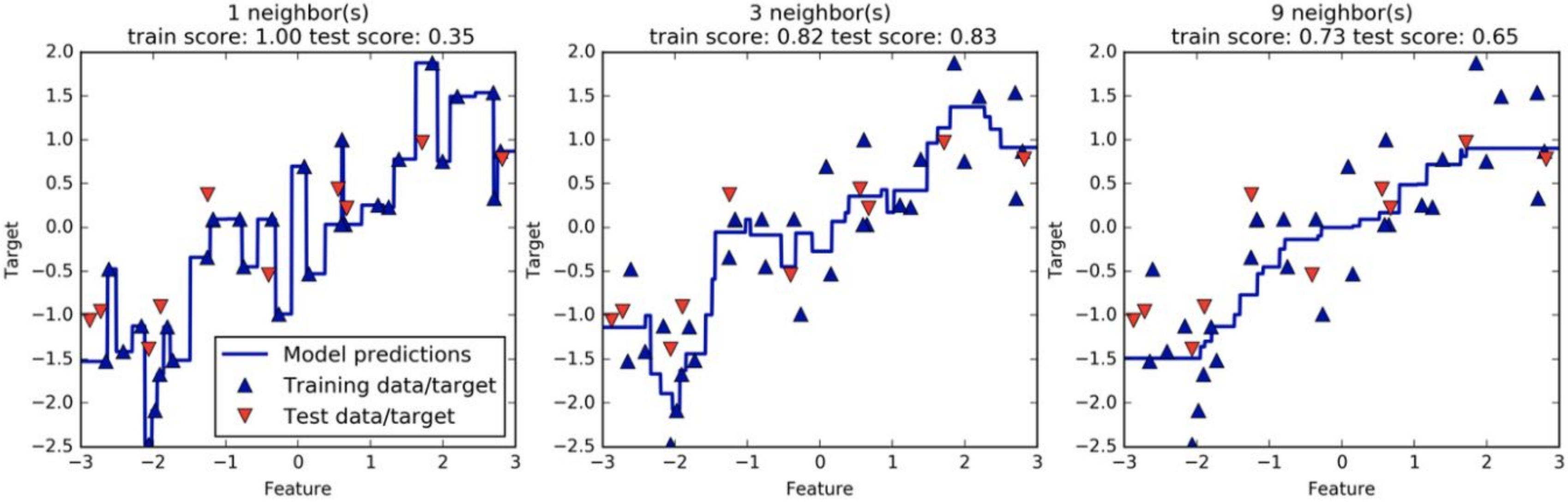


Figure taken from:
https://medium.com/analytics-vidhya/k-neighbors-regression-analysis-in-python-61532d56d8e4

The Pros and Cons

Pros:

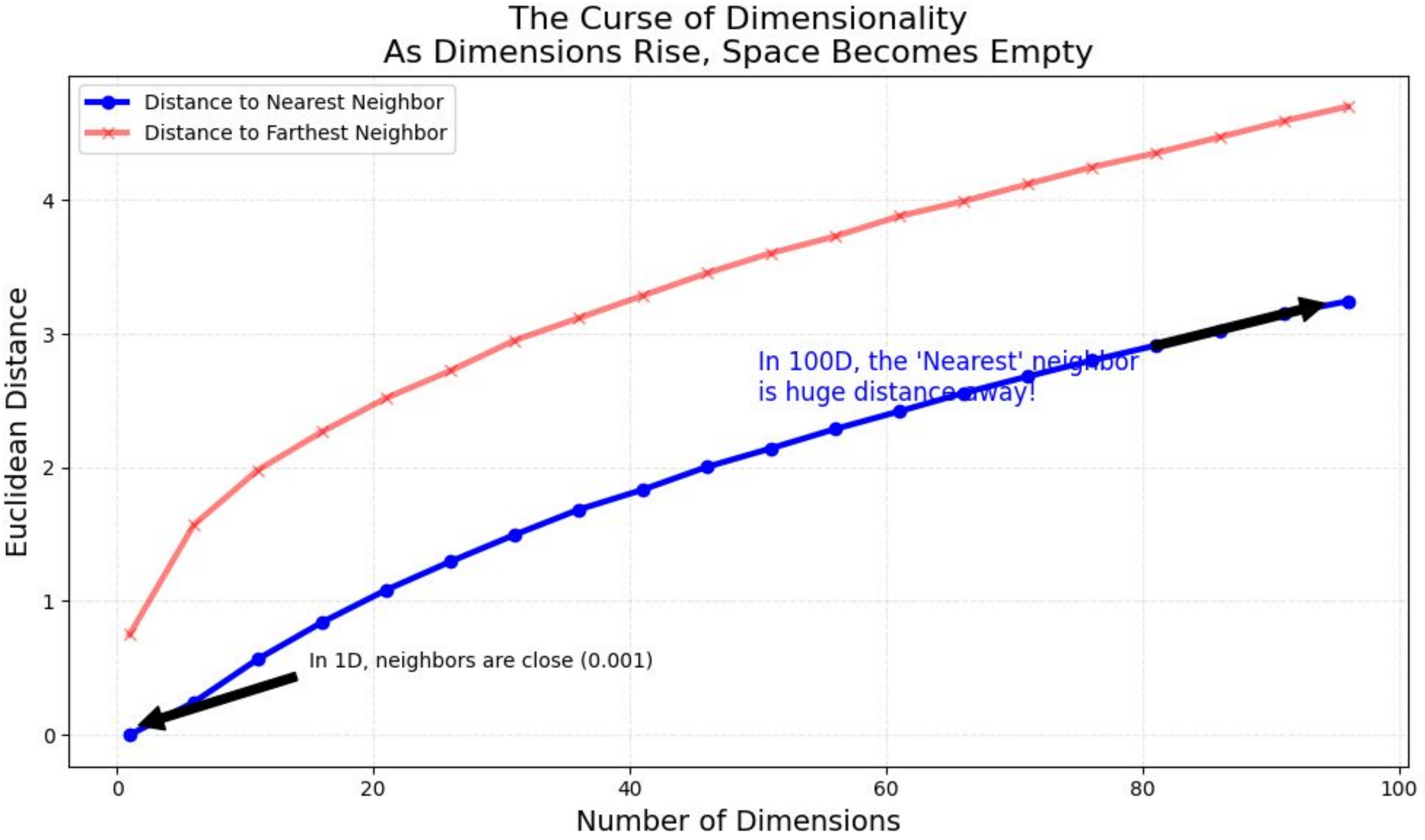
- Simple to understand.
- No training time (O(1) training).
- Non-parametric (doesn't assume data is a straight line).

Cons:

- **Slow Inference:** To predict *one* point, you must calculate distance to *every* point. (O(N) prediction).
- Memory Intensive: Must keep all data in RAM.
- **The Curse of Dimensionality.**

The Curse of Dimensionality

- **The Problem:** In high dimensions (e.g., 1000 columns), *all* points are far away from each other.
- **The Geometry:**
  - In 2D, a circle fills most of a square.
  - In 100D, the "volume" is all in the corners.
  - "Nearest" neighbor becomes meaningless because everyone is equally distant.
- **Impact:** KNN breaks down with too many features (like raw pixels).


The Curse of Dimensionality
As Dimensions Rise, Space Becomes Empty
Distance to Nearest Neighbor
Distance to Farthest Neighbor
In 100D, the 'Nearest' neighbor
is huge distance away!
In 1D, neighbors are close (0.001)
Euclidean Distance
Number of Dimensions
0
1
2
3
4
0
20
40
60
80
100

Thank you!