k-Nearest Neighbors - Derivations & Proofs

Paul F. Roysdon, Ph.D.

Contents

1 Mathematical Derivations & Proofs

1.1 Introduction

k-Nearest Neighbors (kNN) is a simple, nonparametric method for classification and regression. Given a query point, it finds the k closest training samples under a chosen metric and aggregates their labels (majority vote for classification; average for regression). We present kNN from two complementary views: (i) geometric/algorithmic (distance, neighborhoods) and (ii) first-principles statistical derivations (Bayes rule for classification, local ERM for regression), with all variables and dimensions explicitly declared.

1.2 Data and Notation

Let

$$\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^n, \qquad \mathbf{x}_i \in \mathbb{R}^d \text{ (column vector)},$$
$$y_i \in \begin{cases} \{1,\dots,K\}, & \text{classification,} \\ \mathbb{R}, & \text{regression.} \end{cases}$$

Given a query $\mathbf{x} \in \mathbb{R}^d$, define a metric $d(\cdot,\cdot)$ on $\mathbb{R}^d$ and let $\mathcal{N}_k(\mathbf{x})$ be the index set of the k training points with smallest distances to $\mathbf{x}$ (ties broken by a fixed rule).

1.3 Distance Metrics (with Euclidean Derivation)

A common choice is the Euclidean distance:

$$d(\mathbf{x},\mathbf{z}) = \|\mathbf{x}-\mathbf{z}\|_2 = \sqrt{(\mathbf{x}-\mathbf{z})^\top(\mathbf{x}-\mathbf{z})} = \sqrt{\sum_{j=1}^{d}(x_j - z_j)^2}, \tag{1}$$

where $x_j, z_j \in \mathbb{R}$ are components of $\mathbf{x}, \mathbf{z}$. **Properties:** $d(\mathbf{x},\mathbf{z}) \geq 0$ with equality iff $\mathbf{x} = \mathbf{z}$; d is a scalar.

Derivation of the Euclidean Distance Formula

Let

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_d \end{bmatrix} \quad \text{and} \quad \mathbf{z} = \begin{bmatrix} z_1 \\ z_2 \\ \vdots \\ z_d \end{bmatrix}.$$

Then,

$$\mathbf{x} - \mathbf{z} = \begin{bmatrix} x_1 - z_1 \\ x_2 - z_2 \\ \vdots \\ x_d - z_d \end{bmatrix}.$$

The squared Euclidean norm is given by:

$$\|\mathbf{x} - \mathbf{z}\|_2^2 = (x_1 - z_1)^2 + (x_2 - z_2)^2 + \cdots + (x_d - z_d)^2.$$

Taking the square root yields the Euclidean distance:

$$d(\mathbf{x}, \mathbf{z}) = \sqrt{\sum_{j=1}^{d} (x_j - z_j)^2}.$$

Other useful metrics:

$$\begin{aligned} d_p(\mathbf{x}, \mathbf{z}) &= \Big(\sum_j |x_j - z_j|^p \Big)^{1/p} \quad (p \geq 1), \\ d_{\text{cosine}}(\mathbf{x}, \mathbf{z}) &= 1 - \frac{\mathbf{x}^\top \mathbf{z}}{\|\mathbf{x}\|_2 \|\mathbf{z}\|_2}, \\ d_{\text{Manhattan}}(\mathbf{x}, \mathbf{z}) &= \sqrt{(\mathbf{x} - \mathbf{z})^\top \mathbf{\Sigma}^{-1} (\mathbf{x} - \mathbf{z})}. \end{aligned}$$

Feature scaling (e.g., standardization or whitening for d_M) is often essential.

1.4 Model Formulation: Neighborhoods and Weights

Let $r_k(\mathbf{x})$ be the smallest radius such that the closed ball $B(\mathbf{x}, r_k) = \{\mathbf{z} : d(\mathbf{z}, \mathbf{x}) \leq r_k\}$ contains exactly k training points. Define uniform *neighborhood weights*

$$w_i(\mathbf{x}) = \mathbf{1}\{i \in \mathcal{N}_k(\mathbf{x})\},$$

or more generally *distance weights*

$$w_i(\mathbf{x}) = K\left(\frac{d(\mathbf{x}, \mathbf{x}_i)}{h} \right) \geq 0 \quad (\text{kernel } K, \text{ bandwidth } h > 0), \qquad \text{or} \qquad w_i(\mathbf{x}) \propto \frac{1}{\big(d(\mathbf{x}, \mathbf{x}_i) + \varepsilon\big)^p}.$$

1.5 Classification from First Principles: Bayes Rule ⇒ Majority Vote

For classification, let the labels of the k nearest neighbors be $\{y_i\}_{i \in \mathcal{N}}$. The predicted class $\hat{y}$ is typically determined by a majority vote:

$$\hat{y} = \text{mode}\{y_i : i \in \mathcal{N}\}.$$

Let $\eta_c(\mathbf{x}) = \Pr(Y = c \mid X = \mathbf{x})$ and $\pi_c = \Pr(Y = c)$. The Bayes classifier minimizing 0–1 risk is $h^\star(\mathbf{x}) = \arg\max_c \eta_c(\mathbf{x})$. Estimate posteriors by shrinking neighborhoods: let $k_c(\mathbf{x})$ be the number of neighbors in $\mathcal{N}_k(\mathbf{x})$ with label c, and n_c the number of class-c samples. The kNN density/prior plug-in yields

$$\widehat{\eta}_c(\mathbf{x}) \;\propto\; \widehat{\pi}_c\,\widehat{f}_c(\mathbf{x}) = \frac{n_c}{n} \cdot \frac{k_c(\mathbf{x})}{n_c\,V_d r_k(\mathbf{x})^d} = \frac{k_c(\mathbf{x})}{n\,V_d r_k(\mathbf{x})^d},$$

where V_d is the volume of the d-dimensional unit ball. The common factor cancels across c, so

$$\hat{h}_{k\text{-NN}}(\mathbf{x}) = \arg\max_{c\in\{1,\dots,K\}} k_c(\mathbf{x}) \quad \text{(majority vote among } k \text{ nearest neighbors)}. \tag{2}$$

Prior adjustment (optional). With user-specified priors π_c, take $\hat{h}(\mathbf{x}) = \arg\max_c\ \pi_c\,k_c(\mathbf{x})$.

1.6 Regression from First Principles: Local ERM (Nadaraya–Watson)

For regression, the predicted target value is the average of the neighbors' target values:

$$\hat{y} = \frac{1}{k}\sum_{i\in\mathcal{N}} y_i.$$

For squared loss, the Bayes regressor is $m^\star(\mathbf{x}) = \mathbb{E}[Y \mid X = \mathbf{x}]$. A local-constant estimator minimizes the weighted empirical risk

$$\hat{m}(\mathbf{x}) \in \arg\min_{c\in\mathbb{R}} \sum_{i=1}^{n} w_i(\mathbf{x})\,\big(y_i - c\big)^2, \tag{3}$$

with solution the weighted mean

$$\hat{m}(\mathbf{x}) = \frac{\sum_{i=1}^{n} w_i(\mathbf{x})\,y_i}{\sum_{i=1}^{n} w_i(\mathbf{x})}. \tag{4}$$

kNN regression uses uniform neighborhood weights, giving

$$\hat{m}_{k\text{-NN}}(\mathbf{x}) = \frac{1}{k}\sum_{i\in\mathcal{N}_k(\mathbf{x})} y_i, \tag{5}$$

Nadaraya–Watson with a uniform kernel on $B(\mathbf{x}, r_k)$.

1.7 Choice of k, Bias–Variance, and Consistency

Small $k \Rightarrow$ low bias / high variance; large $k \Rightarrow$ higher bias / low variance. A classical consistency condition is

$$k = k_n \to \infty \quad \text{and} \quad \frac{k_n}{n} \to 0 \qquad (n \to \infty), \tag{6}$$

under which kNN classifiers approach Bayes optimality and kNN regressors converge to $m^\star(\mathbf{x})$ (at almost every $\mathbf{x}$) under mild regularity. In practice, choose k by cross-validation; odd k helps avoid ties in binary classification.

1.8 Computational Aspects

A naive query costs $O(nd)$ per $\mathbf{x}$ (distance to all points and select k smallest, e.g., via a max-heap in $O(n \log k)$). Spatial indexes (kd-/ball-trees) can accelerate queries in moderate d; in high d, approximate methods (LSH, product quantization) are commonly used.

1.9 Algorithm (kNN Classification/Regression)

1. **Input:** $\mathcal{D}$, metric d, neighborhood size k (or kernel/bandwidth / distance-weights).
2. **Distance:** Compute $d_i = d(\mathbf{x}, \mathbf{x}_i)$ for $i = 1, \ldots, n$.
3. **Neighbors:** Let $\mathcal{N}_k(\mathbf{x})$ be the indices of the k smallest d_i.
4. **Prediction:**
   - *Classification:* $\hat{y}(\mathbf{x}) = \arg\max_c \sum_{i \in \mathcal{N}_k(\mathbf{x})} \mathbf{1}\{y_i = c\}$ (or weighted vote with $w_i(\mathbf{x})$).
   - *Regression:* $\hat{y}(\mathbf{x}) = \frac{1}{k} \sum_{i \in \mathcal{N}_k(\mathbf{x})} y_i$ (or the weighted mean Eqn. (4)).

1.10 Variables and Dimensions

- $\mathbf{x}_i \in \mathbb{R}^d$: ith training vector (column); components $x_{ij} \in \mathbb{R}$.
- $y_i \in \{1, \ldots, K\}$ (classification) or $y_i \in \mathbb{R}$ (regression).
- n: number of samples; d: number of features; K: number of classes.
- $\mathbf{x} \in \mathbb{R}^d$: query vector; $d_i = d(\mathbf{x}, \mathbf{x}_i) \in \mathbb{R}_{\geq 0}$.
- $k \in \{1, \ldots, n\}$: neighborhood size; $\mathcal{N}_k(\mathbf{x}) \subset \{1, \ldots, n\}$.
- $w_i(\mathbf{x}) \geq 0$: (optional) neighbor weights; $r_k(\mathbf{x})$: k-NN radius.

1.11 Summary

Geometrically, kNN predicts from the k closest training points under a metric. From first principles, kNN classification arises by plugging kNN density/priors into Bayes' rule, yielding majority vote Eqn. (2); kNN regression is the local-ERM solution under squared loss, i.e., a (weighted) neighborhood average. The metric and feature scaling determine neighborhoods, while k trades bias for variance; $k \to \infty$ and $k/n \to 0$ ensure statistical consistency under mild conditions.