

Lecture 11: Unsupervised Learning

Yasuyuki Sawada, Yaolang Zhong

University of Tokyo

`yaolang.zhong@e.u-tokyo.ac.jp`

January 11, 2026

Lecture Overview

1. Recap: Classification

- ▶ Logistic Regression
- ▶ K-Nearest Neighbors (KNN)

2. Unsupervised Learning

- ▶ Task 1: Clustering (K-Means)
- ▶ Task 2: Dimensionality Reduction (PCA)
- ▶ Task 3: Density Estimation (KDE)

Recap: Classification and Logistic Regression

- ▶ Setup: The response variable Y is categorical, taking values in $\mathcal{C} = \{1, 2, \dots, K\}$.
- ▶ Goal: Estimate conditional class probabilities $\hat{p}_c(x) = \Pr(Y = c \mid X = x)$, and assign observations to the class with highest probability.
- ▶ Loss Function: Misclassification rate $\frac{1}{n} \sum_{i=1}^n \mathbf{1}(y_i \neq \hat{y}_i)$.
- ▶ Example (Logistic Regression): For binary classification ($Y \in \{0, 1\}$),

$$\Pr(Y = 1 \mid X) = \frac{e^{\beta_0 + \beta_1 X}}{1 + e^{\beta_0 + \beta_1 X}},$$

where β is estimated by maximum likelihood.

K - Nearest Neighbors (KNN)

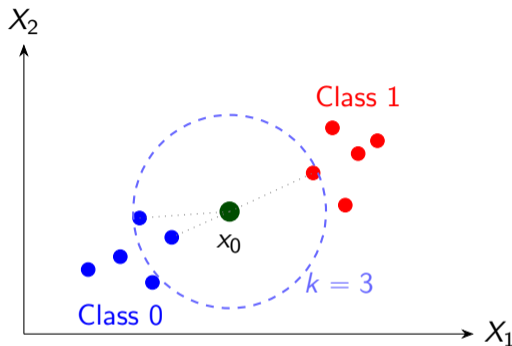
- ▶ Alternative Approach: Instead of a global parametric model for $\Pr(Y | X)$, KNN constructs a local, data-driven approximation.
- ▶ Local Similarity Principle:
 - ▶ For query point x_0 , observations with covariates x_i close to x_0 are most informative.
 - ▶ Closeness is measured by a distance metric $d(x_i, x_0)$.
- ▶ Local Probability Estimation: Let $\mathcal{N}_k(x_0)$ be the k nearest neighbors of x_0 .

$$\hat{p}_c(x_0) = \frac{1}{k} \sum_{i \in \mathcal{N}_k(x_0)} \mathbf{1}(y_i = c).$$

- ▶ Classification Rule:

$$\hat{Y}(x_0) = \arg \max_{c \in \mathcal{C}} \hat{p}_c(x_0).$$

KNN (Illustration)



Example with $k = 3$:

- ▶ 3 nearest neighbors: 2 from Class 0 (blue), 1 from Class 1 (red).
- ▶ Majority vote \Rightarrow Predict Class 0. $\hat{P}(\text{Class 0}) = 2/3$.

KNN: Algorithmic Implementation

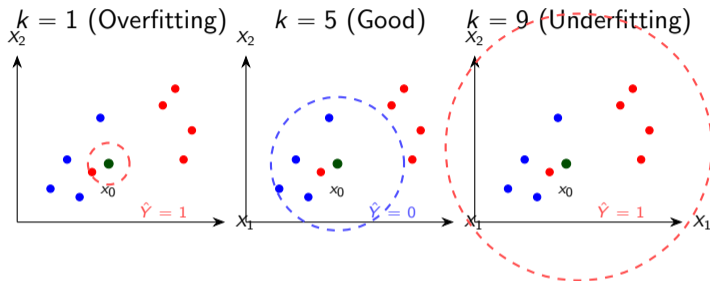
- ▶ Given the local probability estimator defined previously, KNN operationalizes it through the following steps. For a query point x_0 :
 1. Choose the number of neighbors k .
 2. Compute distances $d(x_0, x_i)$ for all training observations.
 3. Identify the neighbor set $\mathcal{N}_k(x_0)$.
 4. Compute $\hat{p}_c(x_0)$ using neighbor frequencies.
 5. Assign class $\hat{Y}(x_0)$ by maximizing $\hat{p}_c(x_0)$.
- ▶ Remarks:
 - ▶ No parameters are estimated during training.
 - ▶ All computation occurs at prediction time (“lazy learning”).
 - ▶ hyperparameters: k (number of neighbors), d (distance metric)

KNN: Choice of k and bias-variance trade-off

The hyperparameter k controls the bias-variance trade-off.

- ▶ Small k (e.g., $k = 1$):
 - ▶ Highly flexible, jagged boundary. Low bias, high variance.
 - ▶ Sensitive to noise and outliers. Risk of overfitting.
- ▶ Large k (e.g., $k = n$):
 - ▶ Smoother, more stable boundary. High bias, low variance.
 - ▶ Robust to noise. Risk of underfitting.
- ▶ Selecting k in Practice:
 - ▶ Use cross-validation to find optimal k .
 - ▶ Common starting point: $k = \sqrt{n}$. Odd k avoids ties in binary classification.

KNN: Effect of k (Illustration)



- ▶ $k = 1$: Only the red outlier \Rightarrow misclassification.
- ▶ $k = 5$: 4 blue, 1 red \Rightarrow correct (Class 0).
- ▶ $k = 9$: 4 blue, 5 red \Rightarrow predicts global majority, ignores local structure.

KNN: Choice of d and curse of dimensionality

- ▶ The definition of “nearest” depends on the distance metric $d(x, x')$.
- ▶ Common choices:
 - ▶ Euclidean (L2): $d(x, x') = \sqrt{\sum_{j=1}^p (x_j - x'_j)^2}$ (most common)
 - ▶ Manhattan (L1): $d(x, x') = \sum_{j=1}^p |x_j - x'_j|$ (robust to outliers)
 - ▶ Cosine: $d(x, x') = 1 - \frac{x \cdot x'}{\|x\| \|x'\|}$ (text/high-dim sparse data)
- ▶ Critical preprocessing — Feature Scaling:
 - ▶ Features with larger scales dominate the distance calculation.
 - ▶ Standardization is essential: $\tilde{x}_j = \frac{x_j - \mu_j}{\sigma_j}$.

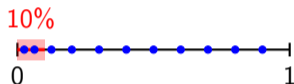
KNN: Dimensionality and Computation

- ▶ Curse of Dimensionality:
 - ▶ As the dimension p increases, observations become sparse in the feature space.
 - ▶ “Local” neighborhoods may no longer be meaningfully local.
 - ▶ To see this, note that to capture a fraction r of the data in p dimensions, a hypercube must have edge length $r^{1/p}$. Consider the example $r = 0.1$:
 - ▶ $p = 2 \Rightarrow 0.32$
 - ▶ $p = 10 \Rightarrow 0.80$
 - ▶ $p = 100 \Rightarrow 0.98$
- ▶ Computational Implications:
 - ▶ Training cost is negligible (data storage only).
 - ▶ Prediction requires distance computation to all n observations: $O(np)$ per query.
- ▶ Practical Responses:
 - ▶ Dimensionality reduction (feature selection, PCA).
 - ▶ Distance metrics adapted to high-dimensional data (e.g. cosine similarity).
 - ▶ Approximate nearest-neighbor methods for large datasets.

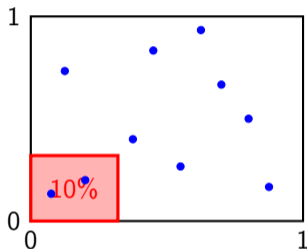
KNN: Curse of Dimensionality (Illustration)

To capture fraction r of data, neighborhood edge length $= r^{1/p}$.

$p = 1$: Edge = 0.1



$p = 2$: Edge $= \sqrt{0.1} \approx 0.32$



- ▶ In 1D, capturing 10% of data requires only 10% of the range.
- ▶ In 2D, capturing 10% of data requires 32% of each axis.
- ▶ As $p \rightarrow \infty$, "local" neighborhoods span nearly the entire space.

Unsupervised Learning: Overview

- ▶ Setup: We observe input vectors X_1, \dots, X_n , but no response Y .

$$\mathcal{D} = \{x_1, x_2, \dots, x_n\}.$$

- ▶ Challenge: Without an observed target, there is no explicit loss function to guide learning.
- ▶ Goal: Using only the joint distribution of X , we aim to:
 - ▶ Identify latent groupings or patterns among observations.
 - ▶ Construct low-dimensional representations preserving essential structure.
 - ▶ Characterize salient features of $P(X)$.

Task 1: Clustering (Definitions)

- ▶ Setup: We observe data $\{x_i\}_{i=1}^n$ drawn from a single, unknown distribution $P(X)$.
- ▶ Definition: Clustering partitions observations into groups using only information in X , without observing class labels or outcomes.
- ▶ Objective: Construct a partition $\{C_1, \dots, C_K\}$ such that:
 - ▶ Observations within a cluster are similar according to a chosen metric.
 - ▶ Observations across clusters are dissimilar.
- ▶ Interpretation:
 - ▶ Clusters provide a discrete summary of structure in $P(X)$ (e.g. regions of high density or distinct geometric patterns).
 - ▶ In some models, clusters may be interpreted as latent components of a mixture distribution.

Task 1: Clustering (Example & Mechanism)

- ▶ Example: *Market Segmentation*. Given customer spending histories X , group customers with similar behavior.
- ▶ Mechanism: K-Means Clustering. Approximate the data distribution by K representative centers and solve

$$\min_{C_1, \dots, C_K} \sum_{k=1}^K \sum_{i \in C_k} \|x_i - \mu_k\|^2,$$

where μ_k is the centroid of cluster k .

- ▶ Algorithm (Lloyd's Algorithm):
 - ▶ Initialize centroids $\{\mu_k^{(0)}\}_{k=1}^K$.
 - ▶ Assignment:

$$i \in C_k^{(t)} \iff \|x_i - \mu_k^{(t)}\| \leq \|x_i - \mu_j^{(t)}\| \quad \forall j.$$

- ▶ Update:

$$\mu_k^{(t+1)} = \frac{1}{|C_k^{(t)}|} \sum_{i \in C_k^{(t)}} x_i.$$

- ▶ Iterate until assignments stabilize.

Task 2: Dimensionality Reduction (Definitions)

- ▶ Setup: Observations $\{x_i\}_{i=1}^n$ lie in a high-dimensional space \mathbb{R}^p .
- ▶ Definition: Dimensionality reduction constructs a low-dimensional representation of X using only information in X .
- ▶ Objective: Find a mapping

$$x_i \in \mathbb{R}^p \longmapsto z_i \in \mathbb{R}^q, \quad q \ll p,$$

such that essential structure in the data is preserved.

- ▶ Interpretation:
 - ▶ The low-dimensional representation captures dominant patterns in $P(X)$.
 - ▶ Reduces noise and redundancy while retaining informative variation.

Task 2: Dimensionality Reduction (Example & Mechanism)

- ▶ Example: *Visualizing Genetic Data*. Gene expression levels for $p = 10,000$ genes observed across $n = 50$ patients; direct visualization in \mathbb{R}^p is infeasible.
- ▶ Mechanism: Principal Component Analysis (PCA)

PCA seeks low-dimensional projections of the form $Z_j = \phi_j^\top X$, where directions $\{\phi_j\}$ are chosen sequentially.

- ▶ First Principal Component:
 - ▶ Choose ϕ_1 to maximize variance:

$$\phi_1 = \arg \max_{\|\phi\|=1} \text{Var}(\phi^\top X).$$

- ▶ Subsequent Components:
 - ▶ For $j \geq 2$, choose

$$\phi_j = \arg \max_{\|\phi\|=1} \text{Var}(\phi^\top X) \quad \text{s.t.} \quad \phi^\top \phi_\ell = 0 \quad \forall \ell < j.$$

- ▶ This enforces orthogonality and ensures components are uncorrelated.

Task 3: Density Estimation (Definitions)

- ▶ Setup: Observations $\{x_i\}_{i=1}^n$ are drawn from an unknown distribution $P(X)$.
- ▶ Definition: Density estimation aims to estimate the probability distribution $P(X)$ itself.
- ▶ Objective: Construct an estimator $\hat{p}(x)$ of the data density $p(x)$ using only the observed sample.
- ▶ Interpretation:
 - ▶ Provides a global, probabilistic description of $P(X)$.
 - ▶ Serves as a foundation for anomaly detection, simulation, and likelihood-based analysis.

Task 3: Density Estimation (Example & Mechanism)

- ▶ Example: Estimate the distribution of income, asset returns, or firm productivity using only observed data $\{x_i\}_{i=1}^n$.
- ▶ Mechanism: Kernel Density Estimation (KDE)

KDE estimates the density at a point x by locally averaging nearby observations:

$$\hat{p}(x) = \frac{1}{nh} \sum_{i=1}^n K\left(\frac{x - x_i}{h}\right),$$

where $K(\cdot)$ is a kernel function and $h > 0$ is a bandwidth.

- ▶ Algorithmic Interpretation:
 - ▶ Center a kernel K at each observation x_i .
 - ▶ Evaluate proximity using the scaled distance $(x - x_i)/h$.
 - ▶ Average the contributions across all observations.
- ▶ Bandwidth Choice:
 - ▶ Small h : highly local averaging (low bias, high variance).
 - ▶ Large h : strong smoothing (high bias, low variance).

References



James, G., Witten, D., Hastie, T., Tibshirani, R. (2021).

An Introduction to Statistical Learning with Applications in Python.

Springer.

<https://www.statlearning.com/>