Q&A Session

EEC454 Introduction to Machine Learning Systems

2023 Fall, Jaeho Lee

Disclaimer

- Today, we do not review math-heavy parts.
    - However, super important!

Supervised Learning & Unsupervised Learning

Supervised Learning

- Learning from data of form $\{(\mathbf{x}_i, y_i)\}_{i=1}^{n}$ (i.e., input-label pairs)
  - Linear Regression
  - Naïve Bayes
  - Perceptrons
  - Logistic Regression
  - K-NN
  - Decision Trees
  - SVMs

Unsupervised Learning

- Learning from data of form $\{\mathbf{x}_i\}_{i=1}^n$ (i.e., no labels)
    - K-Means
    - Gaussian Mixture Models
    - Principal Component Analysis

Anatomy of ML algorithms

- Three core elements.
  - Hypothesis space $\mathcal{F}$
  - Optimization algorithm
  - Loss function (& regularizer?)
- Given the dataset $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$,
  we perform the empirical risk minimization:

$$\min_{f\in\mathcal{F}} \frac{1}{n}\sum_{i=1}^{n} \ell(f(\mathbf{x}_i), y_i)$$

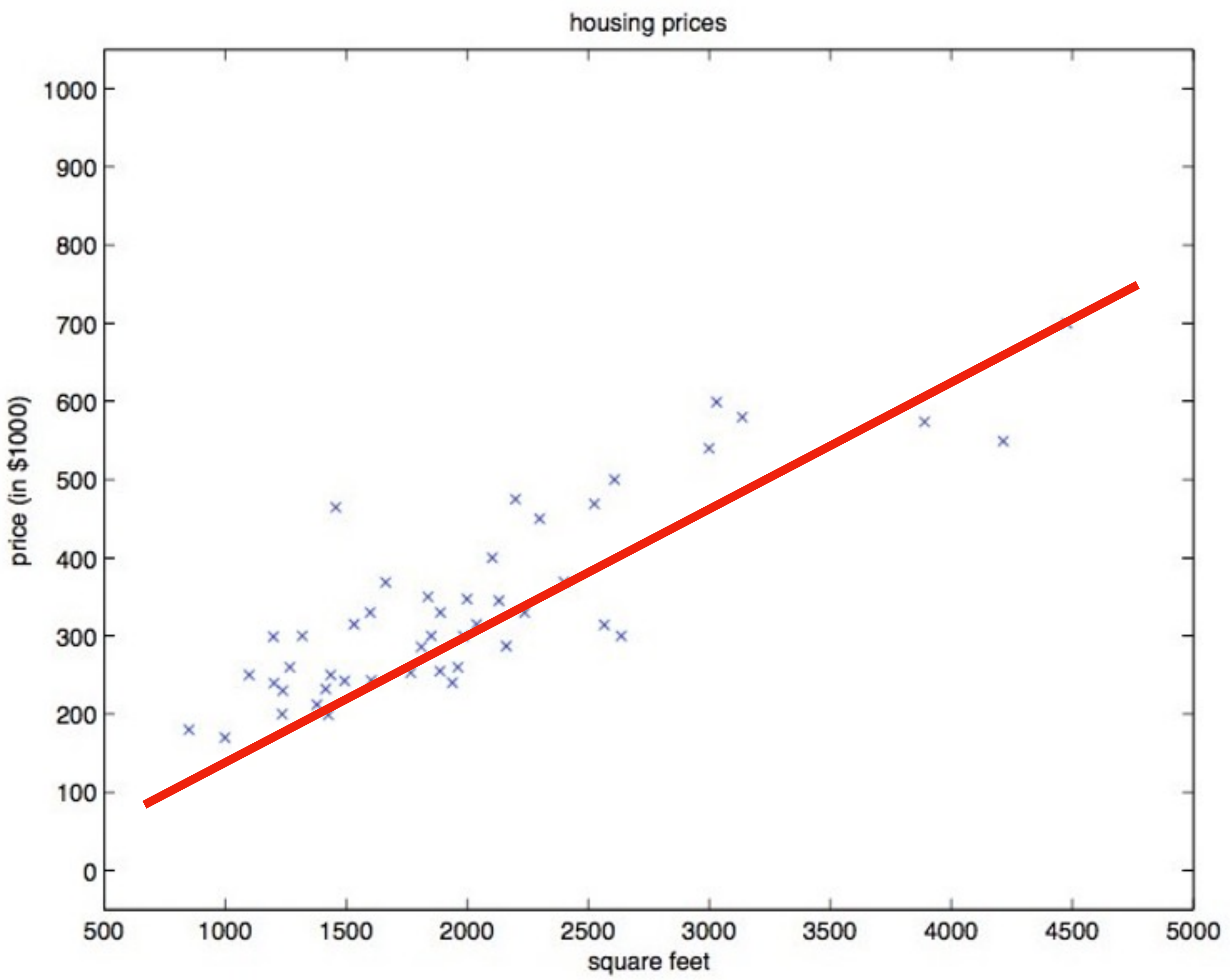
Linear Regression

Linear Regression

- If $\mathbf{x} \in \mathbb{R}^d$ and $y \in \mathbb{R}$, we solve

$$\min_{\mathbf{w} \in \mathbb{R}^d, b \in \mathbb{R}} \frac{1}{n} \sum_{i=1}^{n} (y_i - \mathbf{w}^\top \mathbf{x}_i + b)^2$$

- **Optimization**
    - Critical point analysis
        - Requires some pseudoinverse
    - Gradient descent



Naïve Bayes

Naïve Bayes

- Given the data and "model" (likelihood & prior), maximizes the joint probability

$$\max_{\theta} p_\theta(\mathbf{x}_{1:n}, y_{1:n})$$

- Assuming conditional independence of data, this is equivalent to

$$\min_{\theta} \left(\sum_{i=1}^{n} \log \frac{1}{p_\theta(\mathbf{x}_i \,|\, y_i)} + \log \frac{1}{p_\theta(y_{1:n})} \right)$$

- **Optimization.** Critical point analysis

Perceptrons

Perceptron

- If $\mathbf{x} \in \mathbb{R}^d$ and $y \in \{0,1\}$, we solve

$$\min_{\theta} \frac{1}{n} \sum_{i=1}^{n} (f_\theta(\mathbf{x}) - y) \cdot \theta^\top \tilde{\mathbf{x}}$$

where

$$f_\theta(\mathbf{x}) = \mathbf{1}\{\theta^\top \tilde{\mathbf{x}} > 0\}$$

- **Optimization.** Online learning

(Stochastic gradient descent)

Logistic Regression

Logistic Regression

- We solve

$$\min_{\theta} \frac{1}{n} \sum_{i=1}^{n} \left(\log(f_\theta(\mathbf{x}_i))^{-\mathbf{y}_i} + \log(1 - f_\theta(\mathbf{x}_i))^{\mathbf{y}_i - 1} \right)$$

where

$$f_\theta(\mathbf{x}) = \sigma(\theta^\top \tilde{\mathbf{x}})$$
$$\sigma(t) = \frac{1}{1 + \exp(-t)}$$

- **Optimization.** Gradient Descent

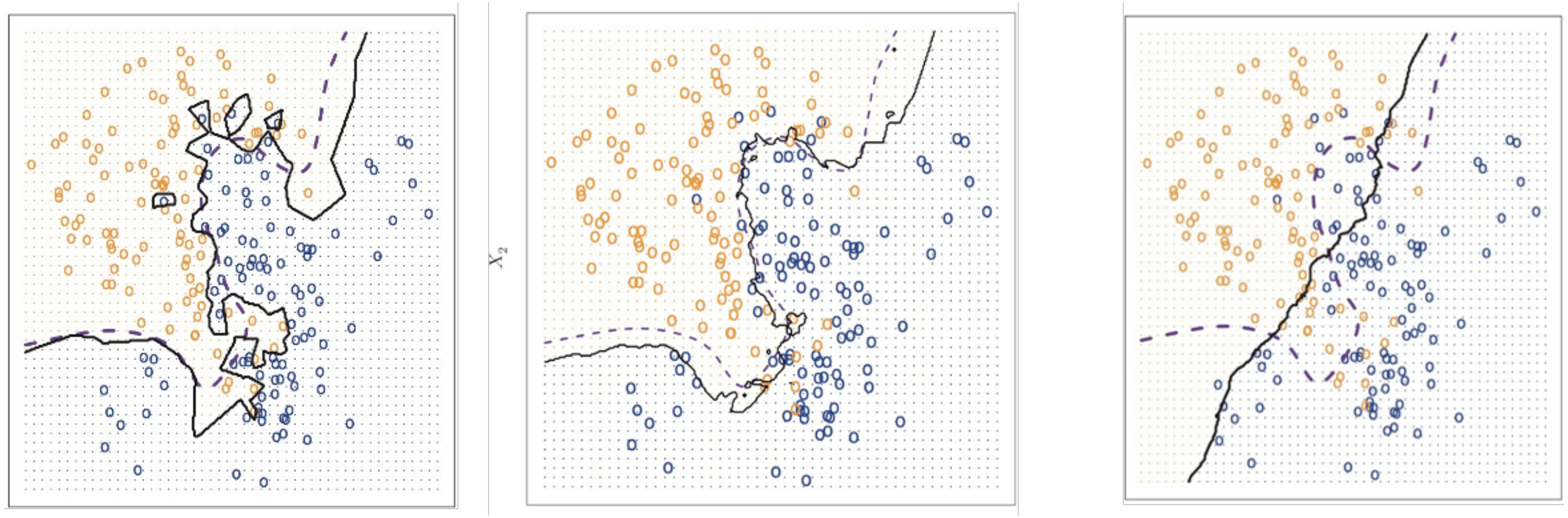
Nearest Neighbors

Nearest Neighbor

- **Idea.** For any test data $\mathbf{x}$, do the majority voting (or averaging) of k training samples with the smallest

$$\|\mathbf{x} - \mathbf{x}_i\|$$

  - First appearance of "nonparametric alg." & "hyperparameters"



Decision Trees

Decision Tree

- **Idea.** Partition the input space with axis-aligned boundaries, so that some uncertainty in each cell is minimized.
  - **Optimization.** Greedy construction, with bagging / boosting

until all leaf node is stopped:

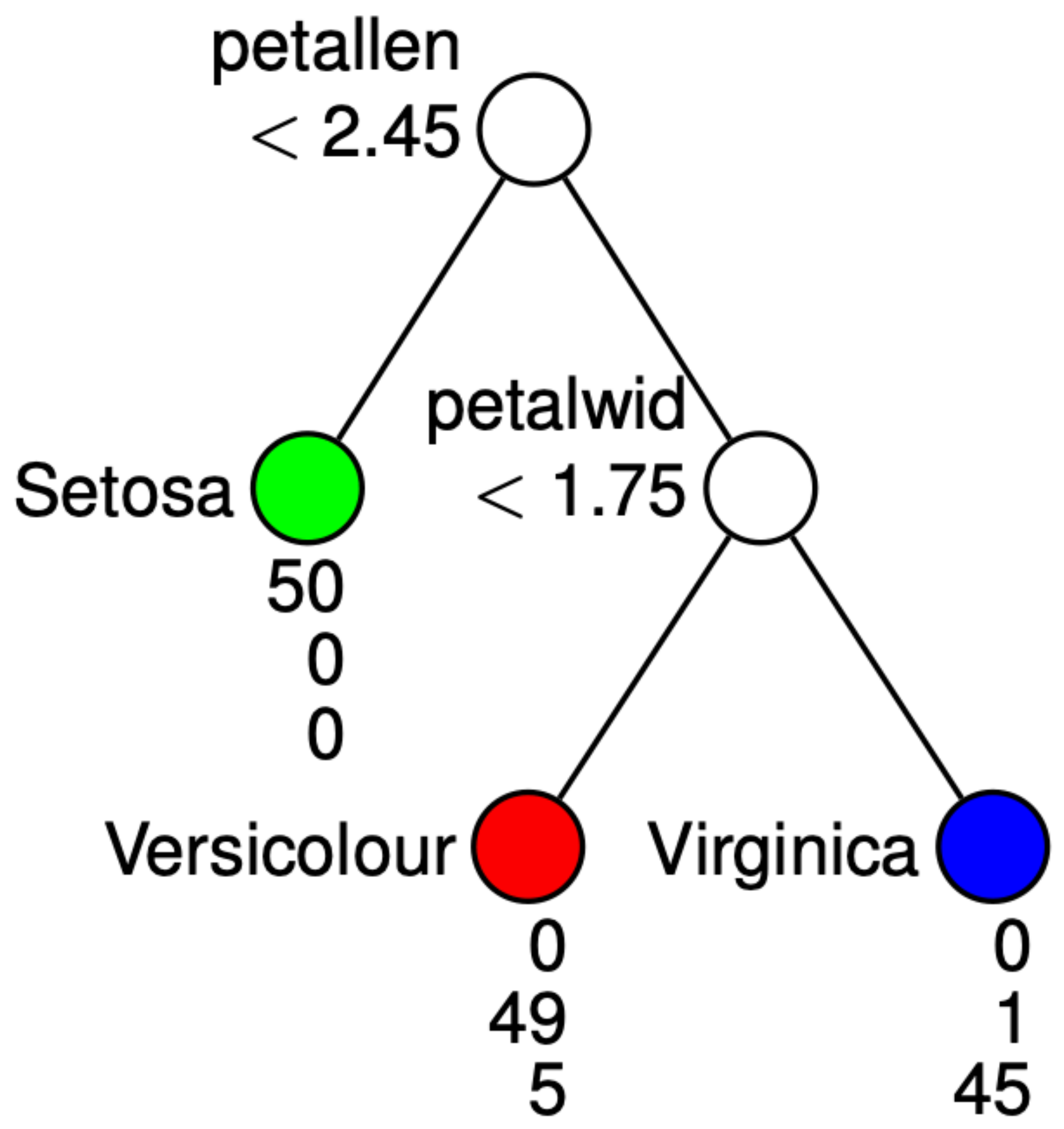
    visit a leaf node

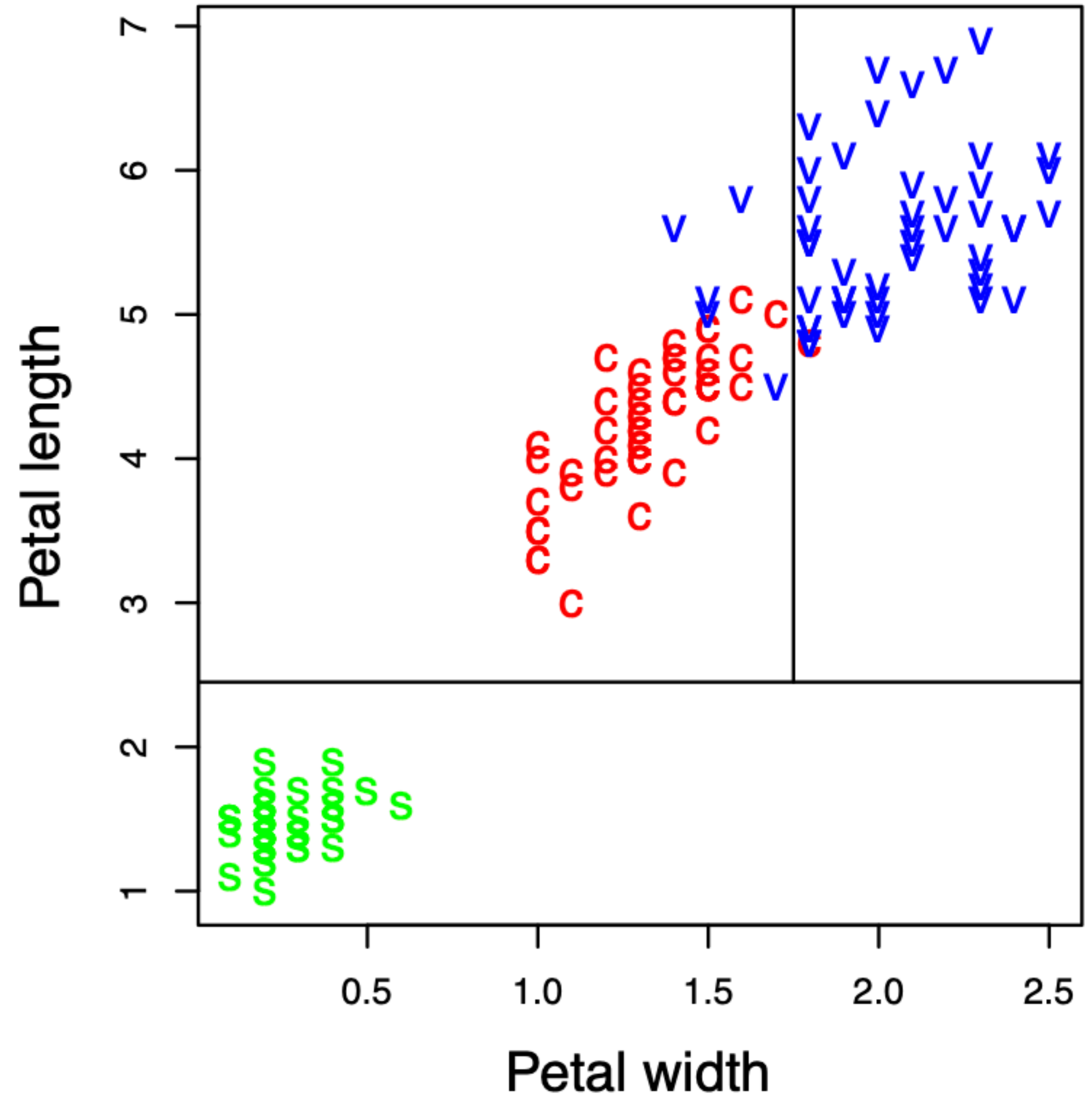
    if(stopping_rule(node) = True):

        apply prediction rule
        stop the node

    else:

        split the node, using the splitting rule





SVMs

SVM

- **Idea.** Linear model, but maximize margin; solves

$$\ell^* = \min_{\mathbf{w},b} \frac{\|\mathbf{w}\|^2}{2} \qquad \text{subject to} \quad y_i(\mathbf{w}^\top \mathbf{x}_i + b) \geq 1$$

- **Optimization.** The method of Lagrangian multipliers
  - Solve the quadratic problem with a solver.

- Softer version, with hyperparameters
- Kernel version

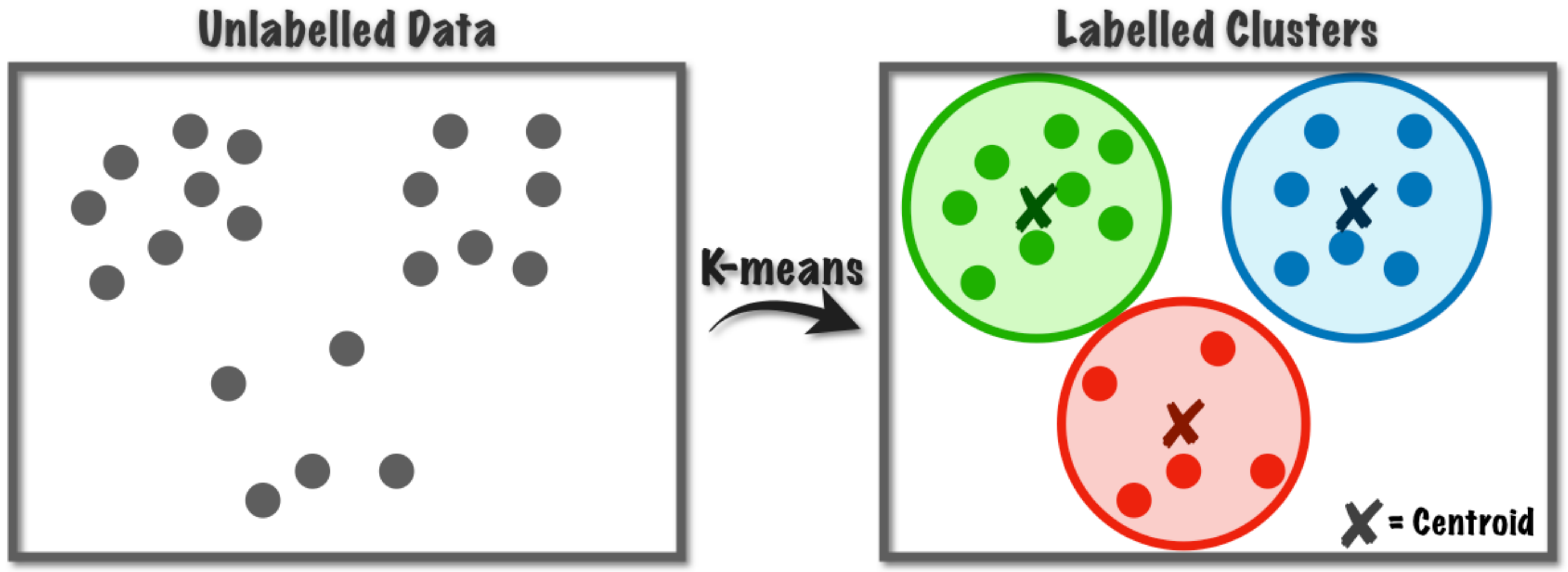
K-means

K-Means

- Solves

$$\min_{\{\mu_k\}} \min_{\{r_{ik}\}} \sum_{i=1}^{n} r_{ik} \|\mathbf{x}_i - \mu_k\|_2^2$$

- **Optimization.** Alternating minimization (general version: EM)



Algorithm 1 k-means algorithm

Specify the number k of clusters to assign.
Randomly initialize k centroids.
repeat
    expectation: Assign each point to its closest centroid.
    maximization: Compute the new centroid (mean) of each cluster.
until The centroid positions do not change.

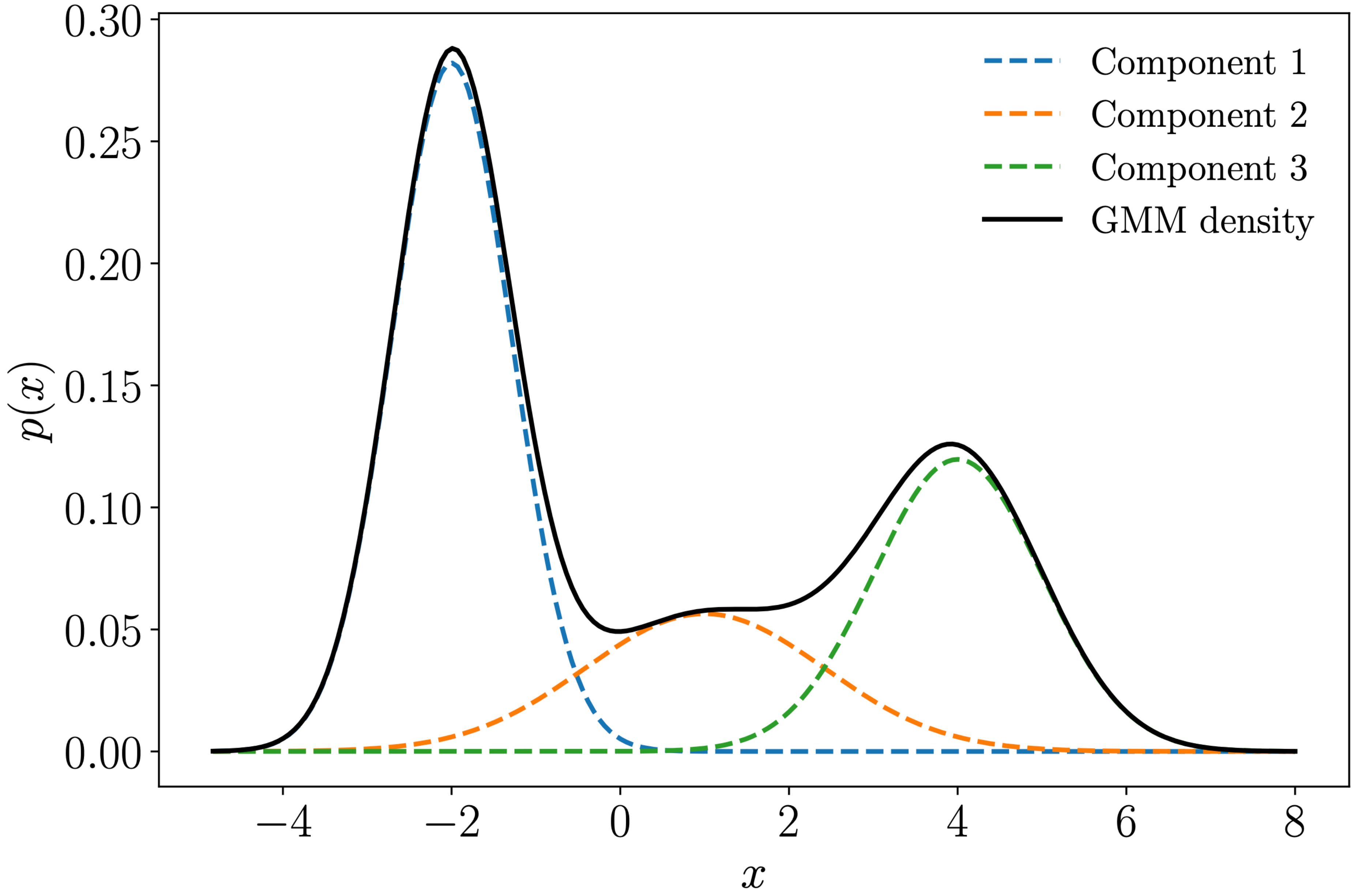
GMMs

GMM

- With Gaussian likelihood models, solves the maximum likelihood

$$\max_{\pi,\mu,\Sigma} \prod_{i=1}^{n} \sum_{k=1}^{K} \pi_k \cdot \mathcal{N}(\mathbf{x}_i \,|\, \mu_k, \Sigma_k)$$

- **Optimization.** Expectation-Maximization (EM) algorithm.



1. Initialize $\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k, \pi_k$.
2. *E-step:* Evaluate responsibilities r_{nk} for every data point $\boldsymbol{x}_n$ using current parameters $\pi_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k$:

$$r_{nk} = \frac{\pi_k \mathcal{N}(\boldsymbol{x}_n \,|\, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_j \pi_j \mathcal{N}(\boldsymbol{x}_n \,|\, \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)} \,. \tag{11.53}$$

3. *M-step:* Reestimate parameters $\pi_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k$ using the current responsibilities r_{nk} (from E-step):

$$\boldsymbol{\mu}_k = \frac{1}{N_k} \sum_{n=1}^{N} r_{nk} \boldsymbol{x}_n \,, \tag{11.54}$$

$$\boldsymbol{\Sigma}_k = \frac{1}{N_k} \sum_{n=1}^{N} r_{nk} (\boldsymbol{x}_n - \boldsymbol{\mu}_k)(\boldsymbol{x}_n - \boldsymbol{\mu}_k)^\top \,, \tag{11.55}$$

$$\pi_k = \frac{N_k}{N} \,. \tag{11.56}$$

PCA

PCA

- Minimize the reconstruction error from projection:

$$\min_{\mathbf{U},\mathbf{b}} \frac{1}{n} \sum_{i=1}^{n} \|\mathbf{x}_i - \mathbf{U}\mathbf{x}_i - \mathbf{b}\|^2$$

- **Optimization.** Greedy selection of basis
  - The method of Lagrangian multipliers + critical point analysis
    - Reduces to the SVD.