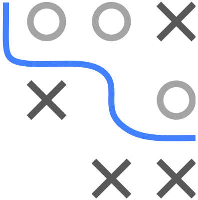
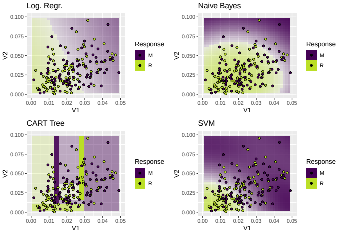
Introduction to Machine Learning

Classification: Basic Definitions

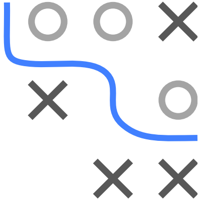




Learning goals

- Understand why classification models have a score / probability as output and not a class
- Understand the difference between scoring and probabilistic classifiers
- Know the concept of decision regions and boundaries
- Know the difference between generative and discriminant approach

CLASSIFICATION TASKS



In classification, we aim at predicting a discrete output

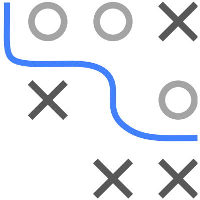
$$y \in \mathcal{Y} = \{C_1, ..., C_g\}$$

with $2 \leq g < \infty$, given data $\mathcal{D}$.

In this course, we assume the classes to be encoded as

- $\mathcal{Y} = \{0, 1\}$ or $\mathcal{Y} = \{-1, +1\}$ (in the binary case $g = 2$)
- $\mathcal{Y} = \{1, \dots, g\}$ (in the multiclass case $g \geq 3$)

CLASSIFICATION MODELS

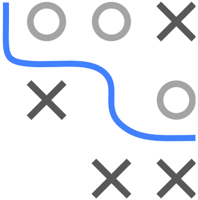


We defined models $f : \mathcal{X} \to \mathbb{R}^g$ as functions that output (continuous) **scores** / **probabilities** and **not** (discrete) classes. Why?

- From an optimization perspective, it is **much** (!) easier to optimize costs for continuous-valued functions
- Scores / probabilities (for classes) contain more information than the class labels alone
- As we will see later, scores can easily be transformed into class labels; but class labels cannot be transformed into scores

We distinguish **scoring** and **probabilistic** classifiers.

SCORING CLASSIFIERS

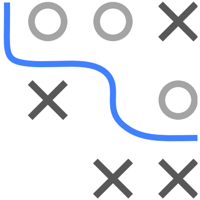


- Construct g **discriminant / scoring functions** $f_1, ..., f_g : \mathcal{X} \to \mathbb{R}$
- Scores $f_1(\mathbf{x}), \ldots, f_g(\mathbf{x})$ are transformed into classes by choosing the class with the maximum score

$$h(\mathbf{x}) = \arg\max_{k \in \{1,...,g\}} f_k(\mathbf{x}).$$

- For $g = 2$, a single discriminant function $f(\mathbf{x}) = f_1(\mathbf{x}) - f_{-1}(\mathbf{x})$ is sufficient (note that it would be natural here to label the classes with $\{-1, +1\}$)
- Class labels are constructed by $h(\mathbf{x}) = \text{sgn}(f(\mathbf{x}))$
- $|f(\mathbf{x})|$ is called “confidence”

PROBABILISTIC CLASSIFIERS

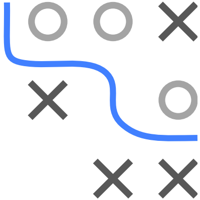


- Construct g **probability functions** $\pi_1, ..., \pi_g : \mathcal{X} \to [0, 1], \ \sum_i \pi_i = 1$
- Probabilities $\pi_1(\mathbf{x}), \dots, \pi_g(\mathbf{x})$ are transformed into labels by predicting the class with the maximum probability

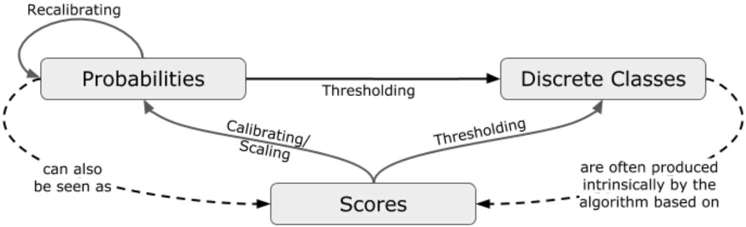
$$h(\mathbf{x}) = \arg\max_{k \in \{1,...,g\}} \pi_k(\mathbf{x})$$

- For $g = 2$ one $\pi(\mathbf{x})$ is constructed (note that it would be natural here to label the classes with $\{0, 1\}$)
- Probabilistic classifiers can also be seen as scoring classifiers
- If we want to emphasize that our model outputs probabilities, we denote the model as $\pi(\mathbf{x}) : \mathcal{X} \to [0, 1]^g$; if we are talking about models in a general sense, we write f, comprising both probabilistic and scoring classifiers (context will make this clear!)

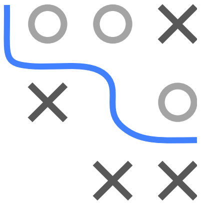
PROBABILISTIC CLASSIFIERS / 2



- Both scoring and probabilistic classifiers can output classes by thresholding (binary case) / selecting the class with the maximum score (multiclass)
- Thresholding: $h(\mathbf{x}) := [\pi(\mathbf{x}) \geq c]$ or $h(\mathbf{x}) = [f(\mathbf{x}) \geq c]$ for some threshold c.
- Usually $c = 0.5$ for probabilistic, $c = 0$ for scoring classifiers.
- There are also versions of thresholding for the multiclass case



DECISION REGIONS AND BOUNDARIES



- A **decision region** for class k is the set of input points $\mathbf{x}$ where class k is assigned as prediction of our model:

$$\mathcal{X}_k = \{\mathbf{x} \in \mathcal{X} : h(\mathbf{x}) = k\}$$

- Points in space where the classes with maximal score are tied and the corresponding hypersurfaces are called **decision boundaries**

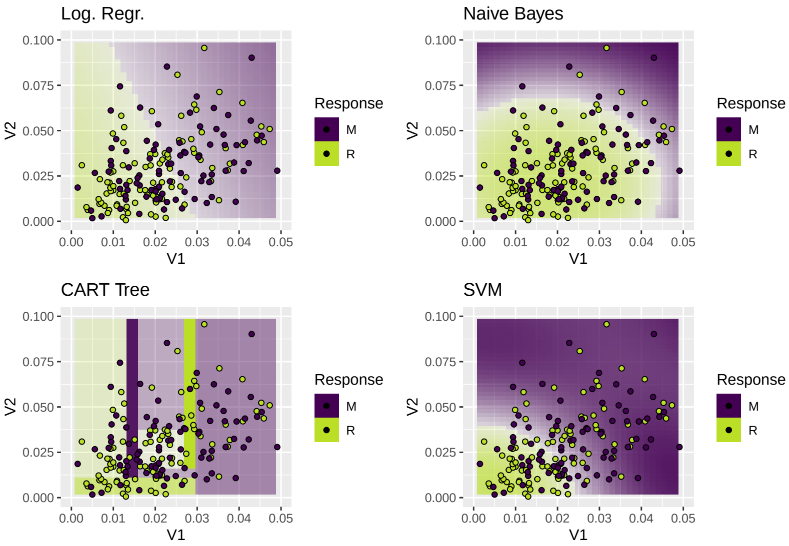
$$\begin{aligned}\{\mathbf{x} \in \mathcal{X} : \quad & \exists\, i \neq j \text{ s.t. } f_i(\mathbf{x}) = f_j(\mathbf{x}) \\ & \text{and } f_i(\mathbf{x}), f_j(\mathbf{x}) \geq f_k(\mathbf{x})\ \forall k \neq i, j\}\end{aligned}$$

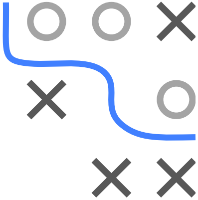
In the binary case we can simplify and generalize to the decision boundary for general threshold c:

$$\{\mathbf{x} \in \mathcal{X} : f(\mathbf{x}) = c\}$$

If we set $c = 0$ for scores and $c = 0.5$ for probabilities, this is consistent with the definition above.

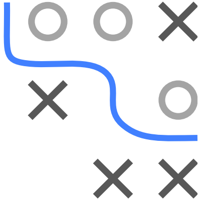
DECISION BOUNDARY EXAMPLES





CLASSIFICATION APPROACHES

Two fundamental approaches exist to construct classifiers:
The **generative approach** and the **discriminant approach**.

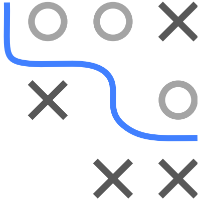


They tackle the classification problem from different angles:

- **Generative** classification approaches assume a data-generating process in which the distribution of the features $\mathbf{x}$ is different for the various classes of the output y, and try to learn these conditional distributions:
  "Which y tends to have $\mathbf{x}$ like these?"

- **Discriminant** approaches use **empirical risk minimization** based on a suitable loss function:
  "What is the best prediction for y given these $\mathbf{x}$?"

GENERATIVE APPROACH



The **generative approach** models $p(\mathbf{x}|y = k)$, usually by making some assumptions about the structure of these distributions, and employs the Bayes theorem:

$$\pi_k(\mathbf{x}) = \mathbb{P}(y = k \mid \mathbf{x}) = \frac{\mathbb{P}(\mathbf{x}|y = k)\mathbb{P}(y = k)}{\mathbb{P}(\mathbf{x})} = \frac{p(\mathbf{x}|y = k)\pi_k}{\sum_{j=1}^{g} p(\mathbf{x}|y = j)\pi_j}$$

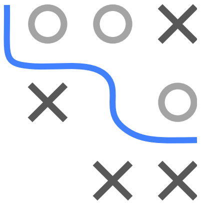
Prior class probabilities π_k are easy to estimate from the training data.

Examples:

- Naive Bayes classifier
- Linear discriminant analysis (generative, linear)
- Quadratic discriminant analysis (generative, not linear)

Note: LDA and QDA have 'discriminant' in their name, but are generative models! (. . . sorry.)

DISCRIMINANT APPROACH



The **discriminant approach** tries to optimize the discriminant functions directly, usually via empirical risk minimization.

$$\hat{f} = \arg\min_{f \in \mathcal{H}} \mathcal{R}_{\text{emp}}(f) = \arg\min_{f \in \mathcal{H}} \sum_{i=1}^{n} L\left(y^{(i)}, f\left(\mathbf{x}^{(i)}\right)\right).$$

Examples:

- Logistic regression (discriminant, linear)
- Neural networks
- Support vector machines