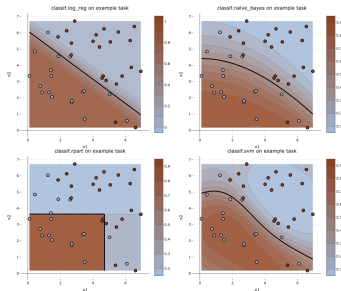


# Introduction to Machine Learning

## Classification

## Basic Definitions



## Learning goals

- Basic notation
- Hard labels vs. probabilities vs. scores
- Decision regions and boundaries
- Generative vs. discriminant approaches

# NOTATION AND TARGET ENCODING

- In classification, we aim at predicting a discrete output

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

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

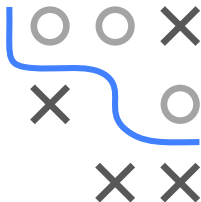
- For convenience, we often encode these classes differently
- Binary case**,  $g = 2$ : Usually use  $\mathcal{Y} = \{0, 1\}$  or  $\mathcal{Y} = \{-1, +1\}$
- Multiclass case**,  $g \geq 3$ : Could use  $\mathcal{Y} = \{1, \dots, g\}$ , but often use **one-hot encoding**  $o(y)$ , i.e.,  $g$ -length vector with  $o_k(y) = \mathbb{I}(y = k) \in \{0, 1\}$ :

ID	Features	Species	one-hot encoding	$o(\text{Species})$
1	...	Setosa		(1, 0, 0)
2	...	Setosa		(1, 0, 0)
3	...	Versicolor		(0, 1, 0)
4	...	Virginica		(0, 0, 1)
5	...	Setosa		(1, 0, 0)



# CLASSIFICATION MODELS

- While for regression the model  $f : \mathcal{X} \rightarrow \mathbb{R}$  simply maps to the label space  $\mathcal{Y} = \mathbb{R}$ , classification is slightly more complicated.
- We sometimes like our models to output (hard) classes, sometimes probabilities, sometimes class scores. The latter 2 are vectors.
- The most basic / common form is the score-based classifier, this is why we defined models already as  $f : \mathcal{X} \rightarrow \mathbb{R}^g$ .
- To minimize confusion, we distinguish between all 3 in notation:  $h(\mathbf{x})$  for hard labels,  $\pi(\mathbf{x})$  for probabilities and  $f(\mathbf{x})$  for scores
- Why all of that and not only hard labels? a) Scores / probabilities are more informative than hard class predictions; b) from an optimization perspective, it is much (!) easier to work with continuous values.

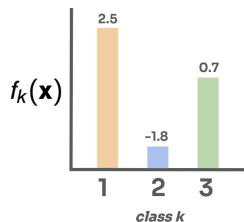


# SCORING CLASSIFIERS

- Construct  $g$  **discriminant / scoring functions**  $f_1, \dots, f_g : \mathcal{X} \rightarrow \mathbb{R}$
- Predicted class is usually the one with max score

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

- For  $g = 2$ , a single discriminant function  $f(\mathbf{x}) = f_1(\mathbf{x}) - f_{-1}(\mathbf{x})$  is sufficient (here, it's natural to label classes with  $\{-1, +1\}$  and we used slight abuse of notation for the subscripts), class labels are constructed by  $h(\mathbf{x}) = \text{sgn}(f(\mathbf{x}))$
- $|f(\mathbf{x})|$  or  $|f_k(\mathbf{x})|$  is loosely called “confidence”



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



# PROBABILISTIC CLASSIFIERS

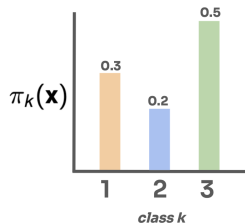
- Construct  $g$  **probability functions**

$$\pi_1, \dots, \pi_g : \mathcal{X} \rightarrow [0, 1], \quad \sum_{k=1}^g \pi_k(\mathbf{x}) = 1$$

- Predicted class is usually the one with max probability

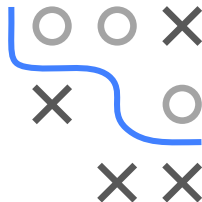
$$h(\mathbf{x}) = \arg \max_{k \in \{1, \dots, g\}} \pi_k(\mathbf{x})$$

- For  $g = 2$ , single  $\pi(\mathbf{x})$  is constructed, which models the predicted probability for the positive class (natural to encode  $\mathcal{Y} = \{0, 1\}$ )



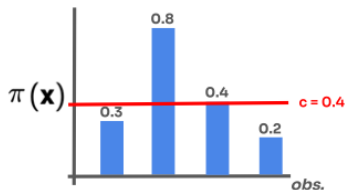
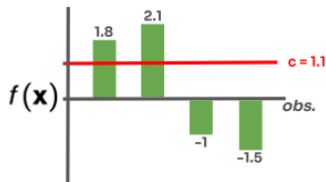
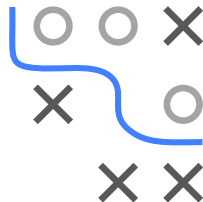
predict class  
with highest prob.

$$h(\mathbf{x}) = \arg \max_{k \in \{1, \dots, g\}} \pi_k(\mathbf{x}) = 3$$



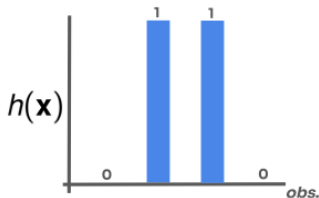
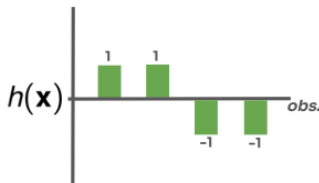
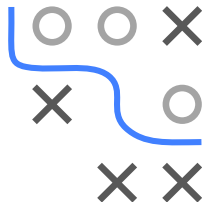
# THRESHOLDING

- For imbalanced cases or class with costs, we might want to deviate from the standard conversion of scores to classes
- Introduce basic concept (for binary case) and add details later
- Convert scores or probabilities to class outputs by thresholding:  
 $h(\mathbf{x}) := [\pi(\mathbf{x}) \geq c]$  or  $h(\mathbf{x}) := [f(\mathbf{x}) \geq c]$  for some threshold  $c$
- Standard thresholds:  $c = 0.5$  for probabilities,  $c = 0$  for scores



# THRESHOLDING

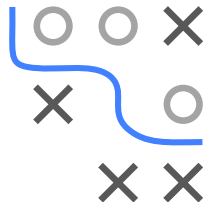
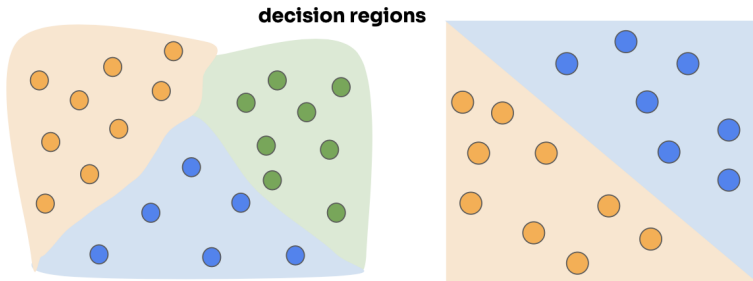
- For imbalanced cases or class with costs, we might want to deviate from the standard conversion of scores to classes
- Introduce basic concept (for binary case) and add details later
- Convert scores or probabilities to class outputs by thresholding:  
 $h(\mathbf{x}) := [\pi(\mathbf{x}) \geq c]$  or  $h(\mathbf{x}) := [f(\mathbf{x}) \geq c]$  for some threshold  $c$
- Standard thresholds:  $c = 0.5$  for probabilities,  $c = 0$  for scores



# DECISION REGIONS

Set of points  $\mathbf{x}$  where class  $k$  is predicted:

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



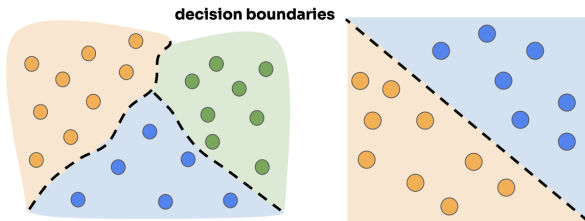


## DECISION BOUNDARIES

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

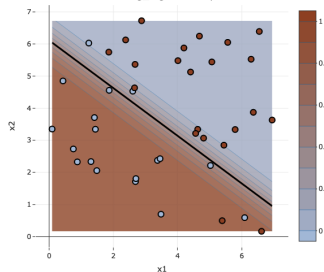
In binary case we can simply use the threshold:

$c = 0$  for scores and  $c = 0.5$  for probs is consistent with the above.

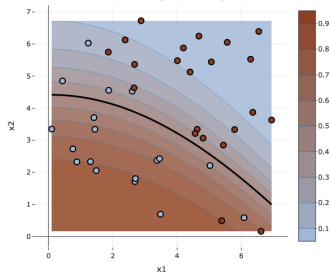


# DECISION BOUNDARY EXAMPLES

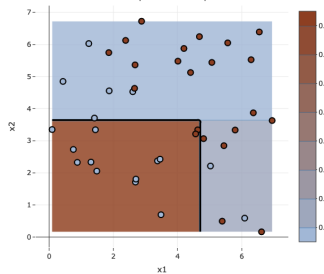
classif.log\_reg on example task



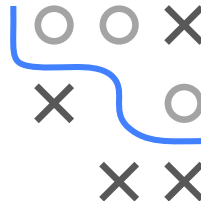
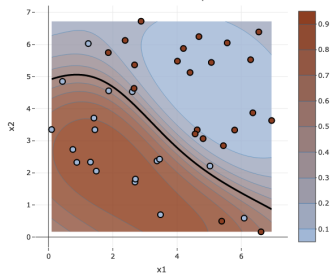
classif.naive\_bayes on example task



classif.rpart on example task



classif.svm on example task



# GENERATIVE APPROACH

Models class-conditional  $p(\mathbf{x}|y = k)$ , and employs Bayes' theorem:

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

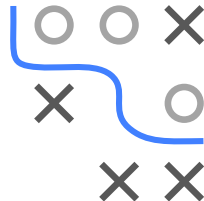
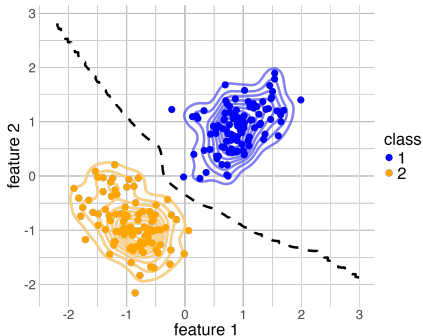
Prior probs  $\pi_k = \mathbb{P}(y = k)$  can easily be estimated from training data as relative frequencies of each class:

ID	Sex	Age	Class	Survived the Titanic
1	male	49	2nd	no
2	female	23	1st	yes
3	male	32	3rd	no
4	male	51	2nd	no
5	female	49	1st	yes

$$\hat{\pi} = \frac{2}{5}$$

# GENERATIVE APPROACH

Decision boundary implicitly defined via the conditional distributions

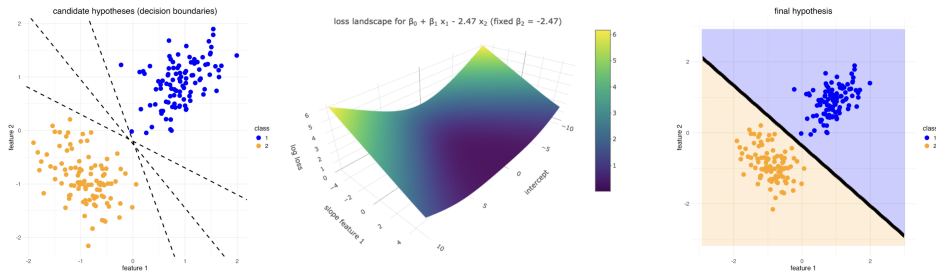


Examples are Naive Bayes, LDA and QDA.

NB: LDA and QDA have 'discriminant' in their name, but are generative!

Here we optimize the discriminant functions (or better: their parameters) directly, usually via ERM:

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



Examples are neural networks, logistic regression and SVMs

