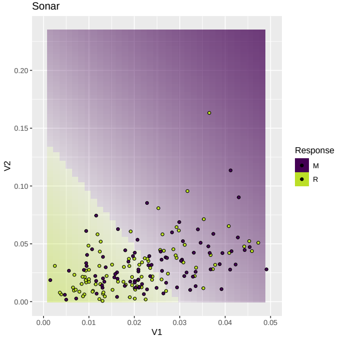
Einführung in das Statistische Lernen

Classification: Tasks

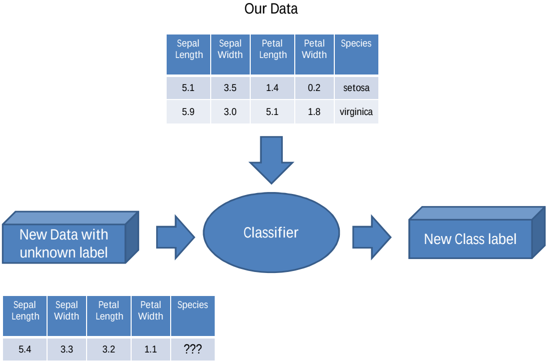


Learning goals

- Understand the main difference between regression and classification
- Know that classification can be binary or multiclass
- Know some examples of classification tasks

CLASSIFICATION

Learn functions that assign class labels to observation / feature vectors. Each observation belongs to exactly one class. The main difference to regression is the scale of the output / label.



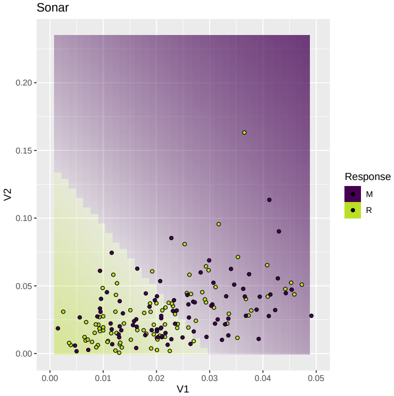
Our Data

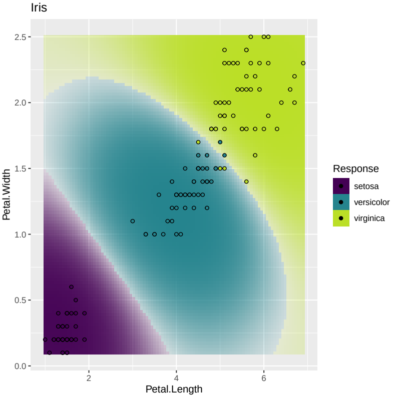
Sepal Length	Sepal Width	Petal Length	Petal Width	Species
5.1	3.5	1.4	0.2	setosa
5.9	3.0	5.1	1.8	virginica

Sepal Length	Sepal Width	Petal Length	Petal Width	Species
5.4	3.3	3.2	1.1	???

BINARY AND MULTICLASS TASKS

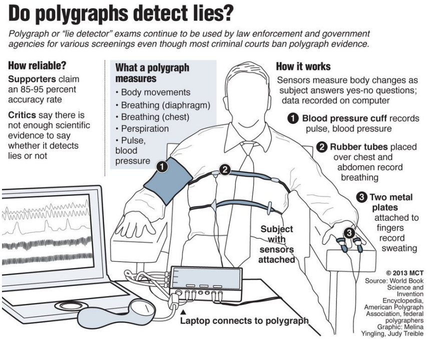
The task can contain 2 classes (binary) or multiple (multiclass).





BINARY CLASSIFICATION TASK - EXAMPLES

- Credit risk prediction, based on personal data and transactions
- Spam detection, based on textual features
- Churn prediction, based on customer behavior
- Predisposition for specific illness, based on genetic data



Do polygraphs detect lies?

Polygraph or "lie detector" exams continue to be used by law enforcement and government agencies for various screenings even though most criminal courts ban polygraph evidence.

How reliable?

Supporters claim an 85-95 percent accuracy rate

Critics say there is not enough scientific evidence to say whether it detects lies or not

What a polygraph measures

- Body movements
- Breathing (diaphragm)
- Breathing (chest)
- Perspiration
- Pulse, blood pressure

How it works

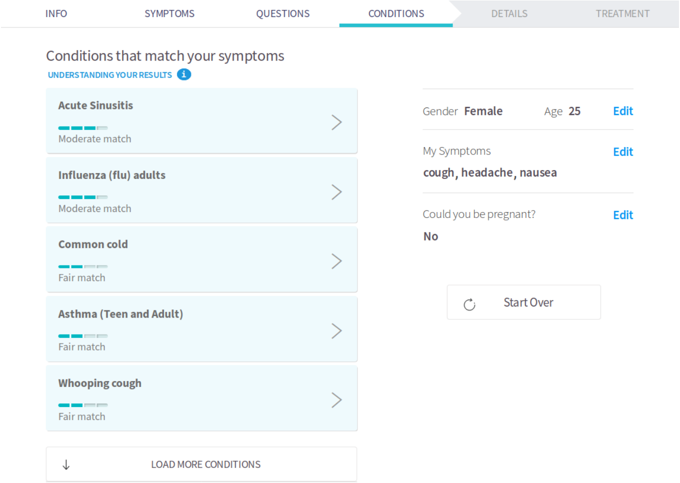
Sensors measure body changes as subject answers yes-no questions; data recorded on computer

1. **Blood pressure cuff** records pulse, blood pressure
2. **Rubber tubes** placed over chest and abdomen record breathing
3. **Two metal plates** attached to fingers record sweating

© 2013 MCT
Source: World Book Science and Invention Encyclopedia, American Polygraph Association, federal polygraphers
Graphic: Melina Yingling, Judy Treible

https://www.bendbulletin.com/localstate/deschutescounty/3430324-151/fact-or-fiction-polygraphs-just-an-investigative-tool

MULTICLASS TASK - MEDICAL DIAGNOSIS



Conditions that match your symptoms

UNDERSTANDING YOUR RESULTS

Acute Sinusitis
Moderate match

Influenza (flu) adults
Moderate match

Common cold
Fair match

Asthma (Teen and Adult)
Fair match

Whooping cough
Fair match

LOAD MORE CONDITIONS

Gender **Female** Age **25** Edit

My Symptoms Edit
cough, headache, nausea

Could you be pregnant? Edit
No

Start Over

https://symptoms.webmd.com

MULTICLASS TASK - IRIS

The iris dataset was introduced by the statistician Ronald Fisher and is one of the most frequent used data sets. Originally, it was designed for linear discriminant analysis.

Setosa

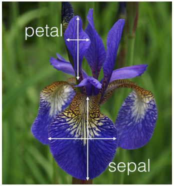
Versicolor

Virginica

Source:
`https://en.wikipedia.org/wiki/Iris_flower_data_set`

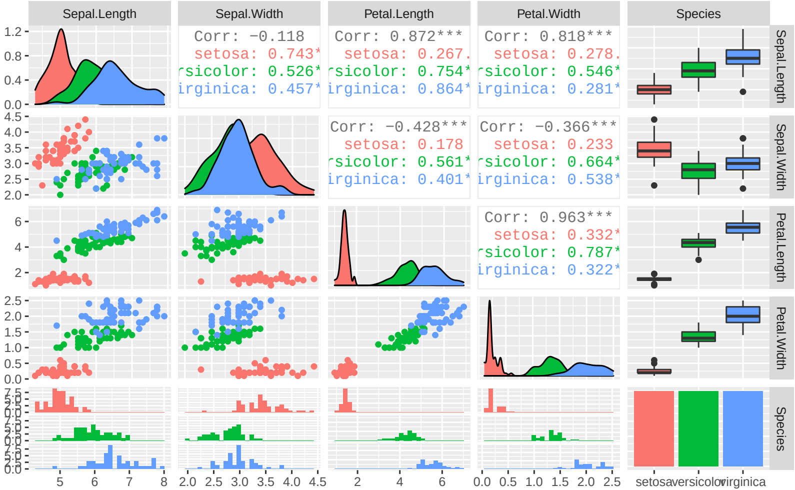
MULTICLASS TASK - IRIS

- 150 iris flowers
- Predict subspecies
- Based on sepal and petal length / width in [cm]



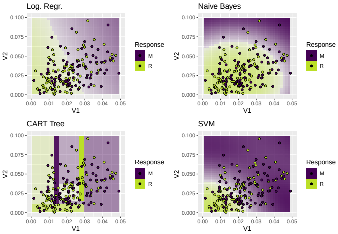
```
##      Sepal.Length Sepal.Width Petal.Length Petal.Width   Species
##   1:          5.1         3.5          1.4         0.2    setosa
##   2:          4.9         3.0          1.4         0.2    setosa
##   3:          4.7         3.2          1.3         0.2    setosa
##   4:          4.6         3.1          1.5         0.2    setosa
##   5:          5.0         3.6          1.4         0.2    setosa
##  ---
## 146:          6.7         3.0          5.2         2.3 virginica
## 147:          6.3         2.5          5.0         1.9 virginica
## 148:          6.5         3.0          5.2         2.0 virginica
## 149:          6.2         3.4          5.4         2.3 virginica
## 150:          5.9         3.0          5.1         1.8 virginica
```

MULTICLASS TASK - IRIS



Einführung in das statistische Lernen

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 $f_1(\mathbf{x}), \dots, f_g(\mathbf{x})$ are transformed into classes by choosing the class with the maximum score

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

- 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

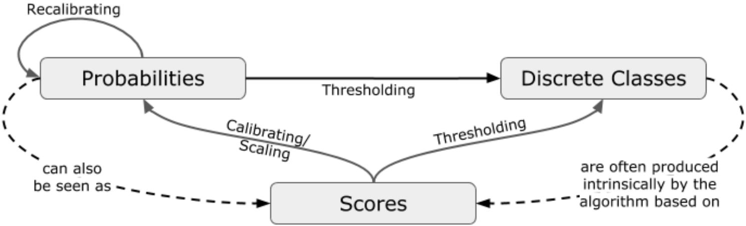
- Construct g **probability functions** $\pi_1, ..., \pi_g : \mathcal{X} \to [0, 1],\ \sum_i \pi_i = 1$
- Probabilities $\pi_1(\mathbf{x}), \dots, \pi_g(\mathbf{x})$ are transformed into labels by predicting the class with the maximum probability

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

- 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

- 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**

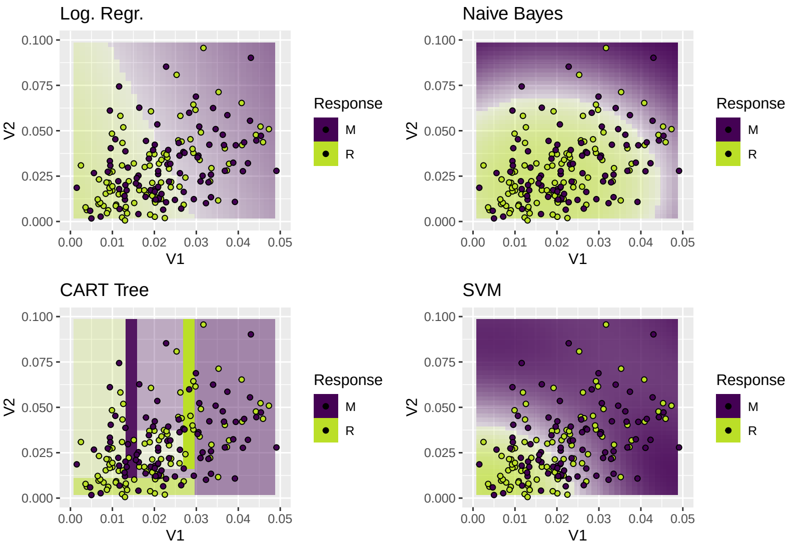
$$\{\mathbf{x} \in \mathcal{X} : \quad \exists\, i \neq j \text{ s.t. } f_i(\mathbf{x}) = f_j(\mathbf{x}) \\ \text{and } f_i(\mathbf{x}), f_j(\mathbf{x}) \geq f_k(\mathbf{x})\ \forall k \neq i, j\}$$

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:

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

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.

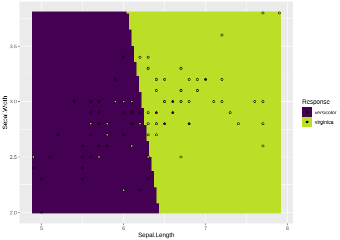
$$\hat{f} = \arg\min_{f \in \mathcal{H}} \mathcal{R}_{\text{emp}}(f) = \arg\min_{f \in \mathcal{H}} \sum_{i=1}^{n} L\left(y^{(i)}, f\left(\mathbf{x}^{(i)}\right)\right).$$

Examples:

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

Introduction to Machine Learning

Classification: Linear Classifiers



Learning goals

- Know the definition of a linear classifier

LINEAR CLASSIFIERS

Linear classifiers are an important subclass of classification models. If the discriminant function(s) $f_k(\mathbf{x})$ can be specified as linear function(s) (possibly through a rank-preserving, monotone transformation $g : \mathbb{R} \to \mathbb{R}$), i. e.

$$g(f_k(\mathbf{x})) = \boldsymbol{w}_k^\top \mathbf{x} + b_k,$$

we will call the classifier a **linear classifier**.

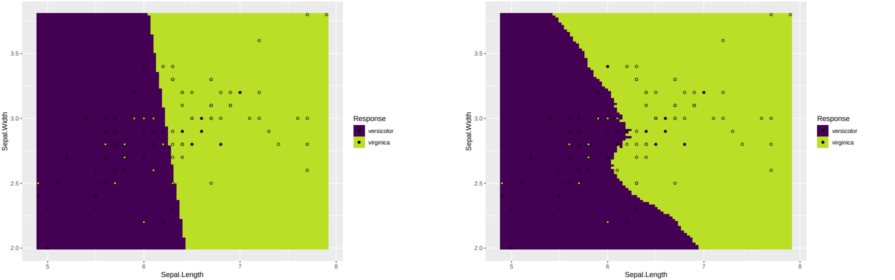
LINEAR CLASSIFIERS

We can also easily show that the decision boundary between classes *i* and *j* is a hyperplane. For every **x** where there is a tie in scores:

$$\begin{aligned} f_i(\mathbf{x}) &= f_j(\mathbf{x}) \\ g(f_i(\mathbf{x})) &= g(f_j(\mathbf{x})) \\ \boldsymbol{w}_i^\top \mathbf{x} + b_i &= \boldsymbol{w}_j^\top \mathbf{x} + b_j \\ (\boldsymbol{w}_i - \boldsymbol{w}_j)^\top \mathbf{x} + (b_i - b_j) &= 0 \end{aligned}$$

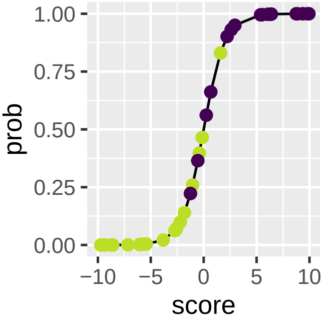
This is a **hyperplane** separating two classes.

LINEAR VS NONLINEAR DECISION BOUNDARY



Introduction to Machine Learning

Classification: Logistic Regression



Learning goals

- Understand the definition of the logit model
- Understand how a reasonable loss function for binary classification can be derived
- Know the hypothesis space that belongs to the logit model

MOTIVATION

A **discriminant** approach for directly modeling the posterior probabilities $\pi(\mathbf{x} \mid \boldsymbol{\theta})$ of the labels is **logistic regression**.
For now, let's focus on the binary case $y \in \{0, 1\}$ and use empirical risk minimization.

$$\underset{\boldsymbol{\theta} \in \Theta}{\arg\min}\, \mathcal{R}_{\text{emp}}(\boldsymbol{\theta}) = \underset{\boldsymbol{\theta} \in \Theta}{\arg\min} \sum_{i=1}^{n} L\left(y^{(i)}, \pi\left(\mathbf{x}^{(i)} \mid \boldsymbol{\theta}\right)\right).$$

A naive approach would be to model

$$\pi(\mathbf{x} \mid \boldsymbol{\theta}) = \boldsymbol{\theta}^T \mathbf{x}.$$

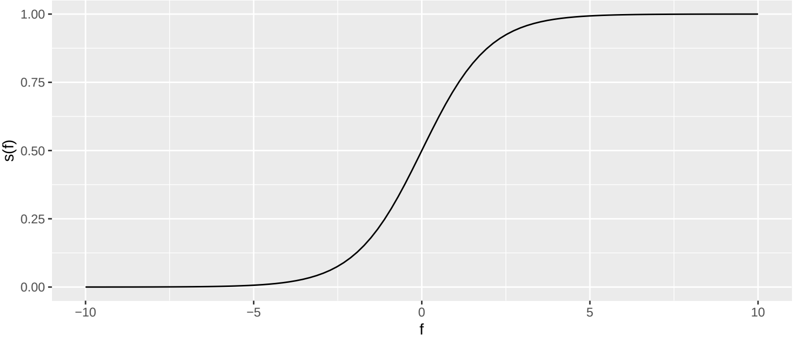
NB: We will often suppress the intercept in notation.

Obviously this could result in predicted probabilities $\pi(\mathbf{x} \mid \boldsymbol{\theta}) \not\in [0, 1]$.

LOGISTIC FUNCTION

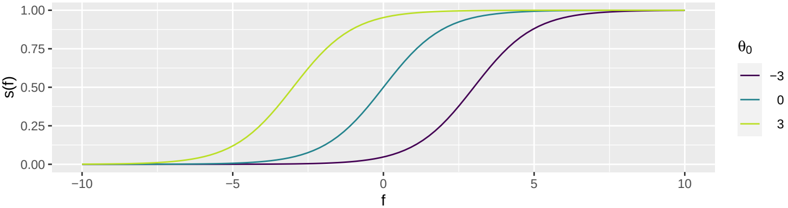
To avoid this, logistic regression “squashes” the estimated linear scores $\boldsymbol{\theta}^T\mathbf{x}$ to $[0, 1]$ through the **logistic function** s:

$$\pi(\mathbf{x} \mid \boldsymbol{\theta}) = \frac{\exp\left(\boldsymbol{\theta}^T\mathbf{x}\right)}{1 + \exp\left(\boldsymbol{\theta}^T\mathbf{x}\right)} = \frac{1}{1 + \exp\left(-\boldsymbol{\theta}^T\mathbf{x}\right)} = s\left(\boldsymbol{\theta}^T\mathbf{x}\right)$$

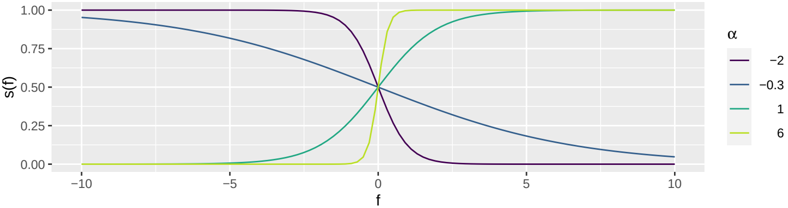


LOGISTIC FUNCTION

The intercept shifts $s(f)$ horizontally $s(\theta_0 + f) = \frac{\exp(\theta_0+f)}{1+\exp(\theta_0+f)}$



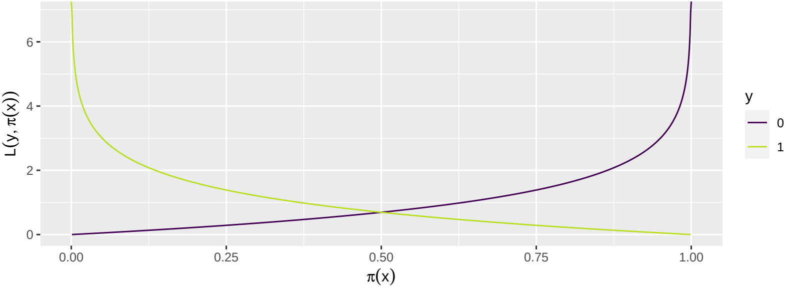
Scaling f like $s(\alpha f) = \frac{\exp(\alpha f)}{1+\exp(\alpha f)}$ controls the slope and direction.



BERNOULLI / LOG LOSS

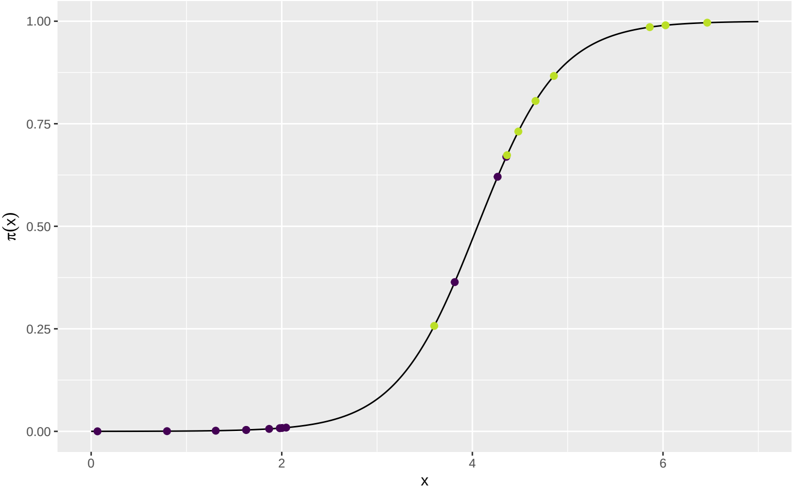
We need to define a loss function for the ERM approach:

- $L(y, \pi(\mathbf{x})) = -y \ln(\pi(\mathbf{x})) - (1 - y) \ln(1 - \pi(\mathbf{x}))$
- Penalizes confidently wrong predictions heavily
- Called Bernoulli, log or cross-entropy loss
- We can derive it from the negative log-likelihood of Bernoulli / logistic regression model in statistics
- Used for many other classifiers, e.g., in NNs or boosting



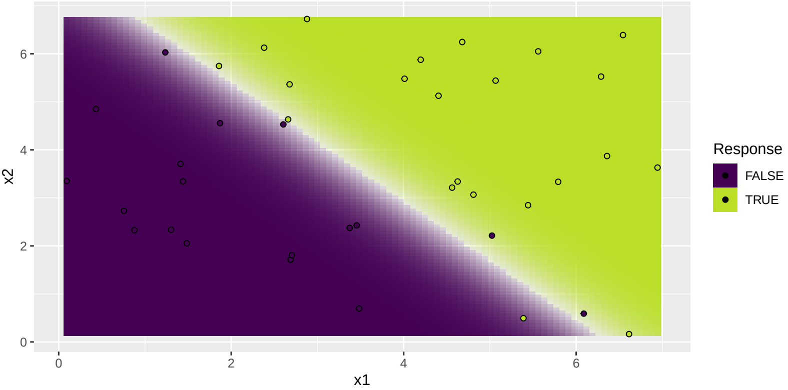
LOGISTIC REGRESSION IN 1D

With one feature $\mathbf{x} \in \mathbb{R}$. The figure shows data and $\mathbf{x} \mapsto \pi(\mathbf{x})$.

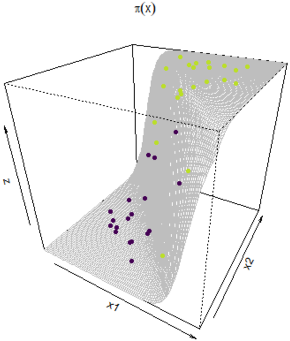


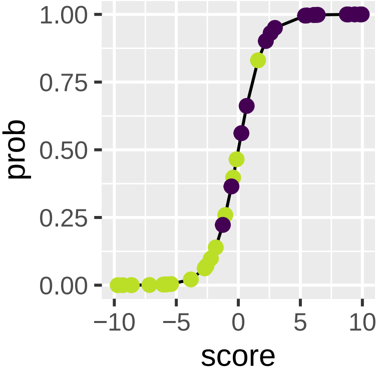
LOGISTIC REGRESSION IN 2D

Obviously, logistic regression is a linear classifier, as $\pi(\mathbf{x} \mid \boldsymbol{\theta}) = s\left(\boldsymbol{\theta}^T \mathbf{x}\right)$ and s is isotonic.



LOGISTIC REGRESSION IN 2D





SUMMARY

Hypothesis Space:

$$\mathcal{H} = \left\{\pi : \mathcal{X} \to [0,1] \mid \pi(\mathbf{x}) = s(\boldsymbol{\theta}^T\mathbf{x})\right\}$$

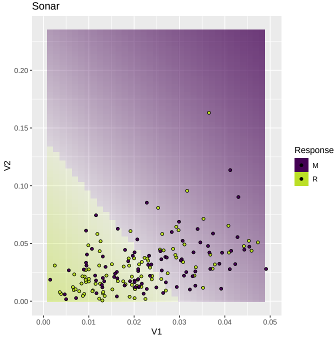
Risk: Logistic/Bernoulli loss function.

$$L\left(y, \pi(\mathbf{x})\right) = -y\ln(\pi(\mathbf{x})) - (1-y)\ln(1-\pi(\mathbf{x}))$$

Optimization: Numerical optimization, typically gradient-based methods.

Einführung in das Statistische Lernen

Multiclass Classification



Learning goals

- Understand the definition of multiclass classification
- Understand how to extend logistic regression to softmax regression

FROM LOGISTIC REGRESSION ...

Remember **logistic regression** ($\mathcal{Y} = \{0, 1\}$): We combined the hypothesis space of linear functions, transformed by the logistic function $s(z) = \frac{1}{1+\exp(-z)}$

$$\mathcal{H} = \left\{\pi : \mathcal{X} \to \mathbb{R} \mid \pi(\mathbf{x}) = s(\boldsymbol{\theta}^\top \mathbf{x})\right\}$$

with the Bernoulli (logarithmic) loss:

$$L(y, \pi(\mathbf{x})) = -y \log\left(\pi(\mathbf{x})\right) - (1 - y) \log\left(1 - \pi(\mathbf{x})\right).$$

Remark: We suppress the intercept term for better readability. The intercept term can be easily included via $\boldsymbol{\theta}^\top \tilde{\mathbf{x}}$, $\boldsymbol{\theta} \in \mathbb{R}^{p+1}$, $\tilde{\mathbf{x}} = (1, \mathbf{x})$.

... TO SOFTMAX REGRESSION

There is a straightforward generalization to the multiclass case:

- Instead of a single linear discriminant function we have g linear discriminant functions

$$f_k(\mathbf{x}) = \boldsymbol{\theta}_k^\top \mathbf{x}, \quad k = 1, 2, ..., g,$$

  each indicating the confidence in class k.

- The g score functions are transformed into g probability functions by the **softmax** function $s : \mathbb{R}^g \to \mathbb{R}^g$

$$\pi_k(\mathbf{x}) = s(f(\mathbf{x}))_k = \frac{\exp(\boldsymbol{\theta}_k^\top \mathbf{x})}{\sum_{j=1}^g \exp(\boldsymbol{\theta}_j^\top \mathbf{x})}$$

  instead of the **logistic** function for $g = 2$. The probabilities are well-defined: $\sum \pi_k(\mathbf{x}) = 1$ and $\pi_k(\mathbf{x}) \in [0, 1]$ for all k.

... TO SOFTMAX REGRESSION

- The softmax function is a generalization of the logistic function. For $g = 2$, the logistic function and the softmax function are equivalent.
- Instead of the **Bernoulli** loss, we use the multiclass **logarithmic loss**

$$L(y, \pi(\mathbf{x})) = -\sum_{k=1}^{g} \mathbb{1}_{\{y=k\}} \log\left(\pi_k(\mathbf{x})\right).$$

- Note that the softmax function is a "smooth" approximation of the arg max operation, so $s((1, 1000, 2)^T) \approx (0, 1, 0)^T$ (picks out 2nd element!).
- Furthermore, it is invariant to constant offsets in the input:

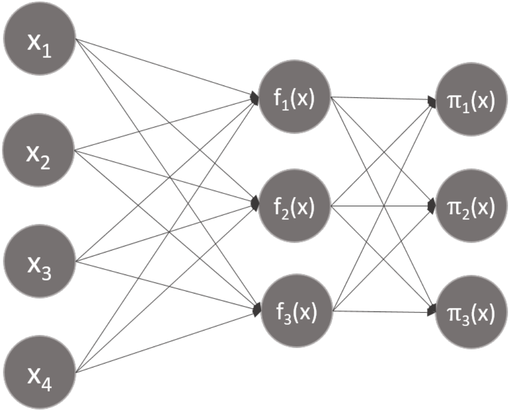
$$s(f(\mathbf{x}) + \mathbf{c}) = \frac{\exp(\boldsymbol{\theta}_k^\top \mathbf{x} + c)}{\sum_{j=1}^{g} \exp(\boldsymbol{\theta}_j^\top \mathbf{x} + c)} = \frac{\exp(\boldsymbol{\theta}_k^\top \mathbf{x}) \cdot \exp(c)}{\sum_{j=1}^{g} \exp(\boldsymbol{\theta}_j^\top \mathbf{x}) \cdot \exp(c)} = s(f(\mathbf{x}))$$

LOGISTIC VS. SOFTMAX REGRESSION

	Logistic Regression	Softmax Regression
$\mathcal{Y}$	$\{0, 1\}$	$\{1, 2, ..., g\}$
Discriminant fun.	$f(\mathbf{x}) = \boldsymbol{\theta}^\top \mathbf{x}$	$f_k(\mathbf{x}) = \boldsymbol{\theta}_k^\top \mathbf{x}, k = 1, 2, ..., g$
Probabilities	$\pi(\mathbf{x}) = \frac{1}{1+\exp\left(-\boldsymbol{\theta}^\top \mathbf{x}\right)}$	$\pi_k(\mathbf{x}) = \frac{\exp(\boldsymbol{\theta}_k^\top \mathbf{x})}{\sum_{j=1}^g \exp(\boldsymbol{\theta}_j^\top \mathbf{x})}$
$L(y, \pi(\mathbf{x}))$	Bernoulli / logarithmic loss $-y \log\left(\pi(\mathbf{x})\right) - (1-y) \log\left(1 - \pi(\mathbf{x})\right)$	Multiclass logarithmic loss $-\sum_{k=1}^g [y = k] \log\left(\pi_k(\mathbf{x})\right)$

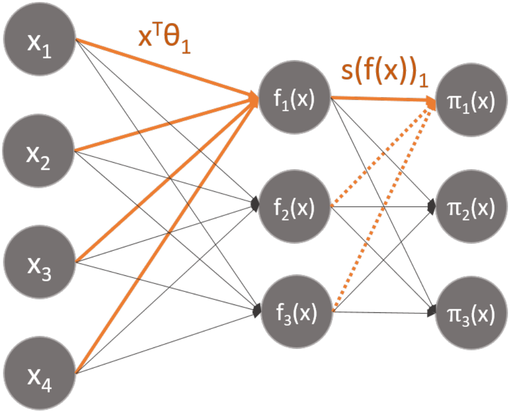
LOGISTIC VS. SOFTMAX REGRESSION

We can schematically depict softmax regression as follows:



LOGISTIC VS. SOFTMAX REGRESSION

We can schematically depict softmax regression as follows:



LOGISTIC VS. SOFTMAX REGRESSION

Further comments:

- We can now, for instance, calculate gradients and optimize this with standard numerical optimization software.
- Softmax regression has an unusual property in that it has a “redundant” set of parameters. If we subtract a fixed vector from all $\boldsymbol{\theta}_k$, the predictions do not change at all. I.e., our model is “over-parameterized”. For any hypothesis we might fit, there are multiple parameter vectors that give rise to exactly the same hypothesis function. This also implies that the minimizer of $\mathcal{R}_{\text{emp}}(\boldsymbol{\theta})$ above is not unique (but $\mathcal{R}_{\text{emp}}(\boldsymbol{\theta})$ is convex)! Hence, a numerical trick is to set $\theta_g = 0$ and only optimize the other θ_k.
- A similar approach is used in many ML models: multiclass LDA, naive Bayes, neural networks and boosting.

SOFTMAX: LINEAR DISCRIMINANT FUNCTIONS

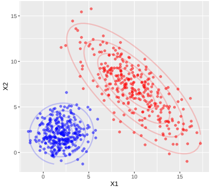
Softmax regression gives us a **linear classifier**.

- The softmax function $s(\boldsymbol{z})_k = \frac{\exp(\boldsymbol{z}_k)}{\sum_{j=1}^{g} \exp(\boldsymbol{z}_j)}$ is
  - a rank-preserving function, i.e. the ranks among the elements of the vector $\boldsymbol{z}$ are the same as among the elements of $s(\boldsymbol{z})$. This is because softmax transforms all scores by taking the $\exp(\cdot)$ (rank-preserving) and divides each element by **the same** normalizing constant.

  Thus, the softmax function has a unique inverse function $s^{-1} : \mathbb{R}^g \to \mathbb{R}^g$ that is also monotonic and rank-preserving. Applying s_k^{-1} to $\pi_k(\mathbf{x}) = \frac{\exp(\boldsymbol{\theta}_k^\top \mathbf{x})}{\sum_{j=1}^{n} \boldsymbol{\theta}_j^\top \mathbf{x}}$ gives us $f_k(\mathbf{x}) = \theta_k^\top \mathbf{x}$. Thus softmax regression is a linear classifier.

Einführung in das statistische Lernen

Classification: Discriminant Analysis



Learning goals

- Understand the ideas of linear and quadratic discriminant analysis
- Understand how parameteres are estimated for LDA and QDA
- Understand how decision boundaries are computed for LDA and QDA

LINEAR DISCRIMINANT ANALYSIS (LDA)

LDA follows a generative approach

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

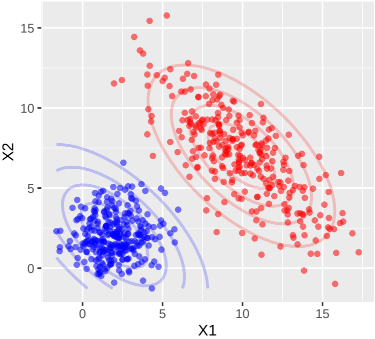
where we now have to pick a distributional form for $p(\mathbf{x}|y = k)$.

LINEAR DISCRIMINANT ANALYSIS (LDA)

LDA assumes that each class density is modeled as a *multivariate Gaussian*:

$$p(\mathbf{x}|y = k) = \frac{1}{(2\pi)^{\frac{p}{2}}|\Sigma|^{\frac{1}{2}}} \exp\left(-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu_k})^T\Sigma^{-1}(\mathbf{x} - \boldsymbol{\mu_k})\right)$$

with equal covariance, i. e. $\Sigma_k = \Sigma \quad \forall k$.



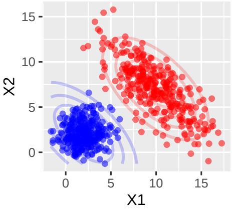
LINEAR DISCRIMINANT ANALYSIS (LDA)

Parameters $\boldsymbol{\theta}$ are estimated in a straightforward manner by estimating

$$\hat{\pi}_k = \frac{n_k}{n}, \text{ where } n_k \text{ is the number of class-}k\text{ observations}$$

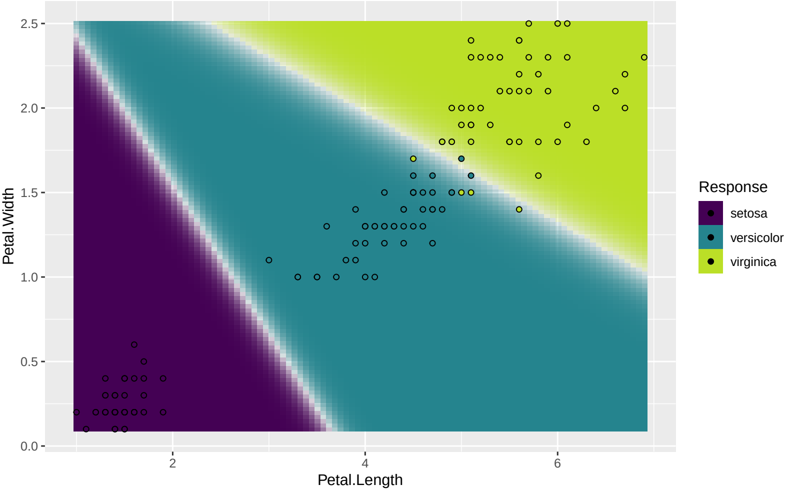
$$\hat{\boldsymbol{\mu}}_{\boldsymbol{k}} = \frac{1}{n_k} \sum_{i: y^{(i)} = k} \mathbf{x}^{(i)}$$

$$\hat{\Sigma} = \frac{1}{n - g} \sum_{k=1}^{g} \sum_{i: y^{(i)} = k} (\mathbf{x}^{(i)} - \hat{\boldsymbol{\mu}}_{\boldsymbol{k}})(\mathbf{x}^{(i)} - \hat{\boldsymbol{\mu}}_{\boldsymbol{k}})^T$$



LDA AS LINEAR CLASSIFIER

Because of the equal covariance structure of all class-specific Gaussian, the decision boundaries of LDA are linear.



LDA AS LINEAR CLASSIFIER

We can formally show that LDA is a linear classifier, by showing that the posterior probabilities can be written as linear scoring functions - up to any isotonic / rank-preserving transformation.

$$\pi_k(\mathbf{x}) = \frac{\pi_k \cdot p(\mathbf{x}|y=k)}{p(\mathbf{x})} = \frac{\pi_k \cdot p(\mathbf{x}|y=k)}{\sum\limits_{j=1}^{g} \pi_j \cdot p(\mathbf{x}|y=j)}$$

As the denominator is the same for all classes we only need to consider

$$\pi_k \cdot p(\mathbf{x}|y=k)$$

and show that this can be written as a linear function of **x**.

LDA AS LINEAR CLASSIFIER

$$
\begin{aligned}
& \pi_k \cdot p(\mathbf{x}|y=k) \\
\propto \quad & \pi_k \exp\left(-\tfrac{1}{2}\mathbf{x}^T\Sigma^{-1}\mathbf{x} - \tfrac{1}{2}\boldsymbol{\mu}_{\boldsymbol{k}}^T\Sigma^{-1}\boldsymbol{\mu}_{\boldsymbol{k}} + \mathbf{x}^T\Sigma^{-1}\boldsymbol{\mu}_{\boldsymbol{k}}\right) \\
= \quad & \exp\left(\log\pi_k - \tfrac{1}{2}\boldsymbol{\mu}_{\boldsymbol{k}}^T\Sigma^{-1}\boldsymbol{\mu}_{\boldsymbol{k}} + \mathbf{x}^T\Sigma^{-1}\boldsymbol{\mu}_{\boldsymbol{k}}\right)\exp\left(-\tfrac{1}{2}\mathbf{x}^T\Sigma^{-1}\mathbf{x}\right) \\
= \quad & \exp\left(\boldsymbol{\theta}_{0k} + \mathbf{x}^T\boldsymbol{\theta}_k\right)\exp\left(-\tfrac{1}{2}\mathbf{x}^T\Sigma^{-1}\mathbf{x}\right) \\
\propto \quad & \exp\left(\boldsymbol{\theta}_{0k} + \mathbf{x}^T\boldsymbol{\theta}_k\right)
\end{aligned}
$$

by defining $\boldsymbol{\theta}_{0k} := \log\pi_k - \frac{1}{2}\boldsymbol{\mu}_{\boldsymbol{k}}^T\Sigma^{-1}\boldsymbol{\mu}_{\boldsymbol{k}}$ and $\boldsymbol{\theta}_k := \Sigma^{-1}\boldsymbol{\mu}_{\boldsymbol{k}}$.

We have again left out all constants which are the same for all classes k, so the normalizing constant of our Gaussians and $\exp\left(-\frac{1}{2}\mathbf{x}^T\Sigma^{-1}\mathbf{x}\right)$.

By finally taking the log, we can write our transformed scores as linear:

$$f_k(\mathbf{x}) = \boldsymbol{\theta}_{0k} + \mathbf{x}^T\boldsymbol{\theta}_k$$

QUADRATIC DISCRIMINANT ANALYSIS (QDA)

QDA is a direct generalization of LDA, where the class densities are now Gaussians with unequal covariances Σ_k.

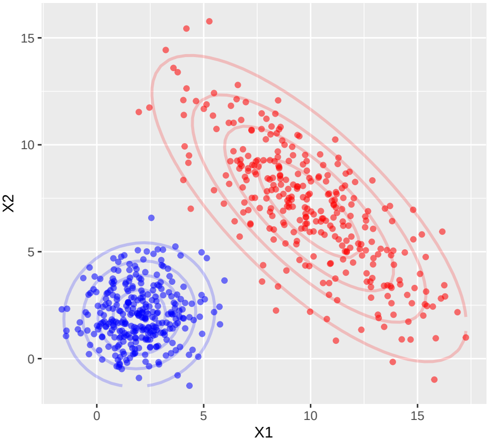
$$p(\mathbf{x}|y = k) = \frac{1}{(2\pi)^{\frac{p}{2}}|\Sigma_k|^{\frac{1}{2}}} \exp\left(-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu_k})^T\Sigma_k^{-1}(\mathbf{x} - \boldsymbol{\mu_k})\right)$$

Parameters are estimated in a straightforward manner by:

$$\begin{aligned}
\hat{\pi}_k &= \frac{n_k}{n}, \text{ where } n_k \text{ is the number of class-}k\text{ observations} \\
\hat{\boldsymbol{\mu}}_{\boldsymbol{k}} &= \frac{1}{n_k}\sum_{i:y^{(i)}=k} \mathbf{x}^{(i)} \\
\hat{\Sigma}_k &= \frac{1}{n_k - 1}\sum_{i:y^{(i)}=k} (\mathbf{x}^{(i)} - \hat{\boldsymbol{\mu}}_{\boldsymbol{k}})(\mathbf{x}^{(i)} - \hat{\boldsymbol{\mu}}_{\boldsymbol{k}})^T
\end{aligned}$$

QUADRATIC DISCRIMINANT ANALYSIS (QDA)

- Covariance matrices can differ over classes.
- Yields better data fit but also requires estimation of more parameters.



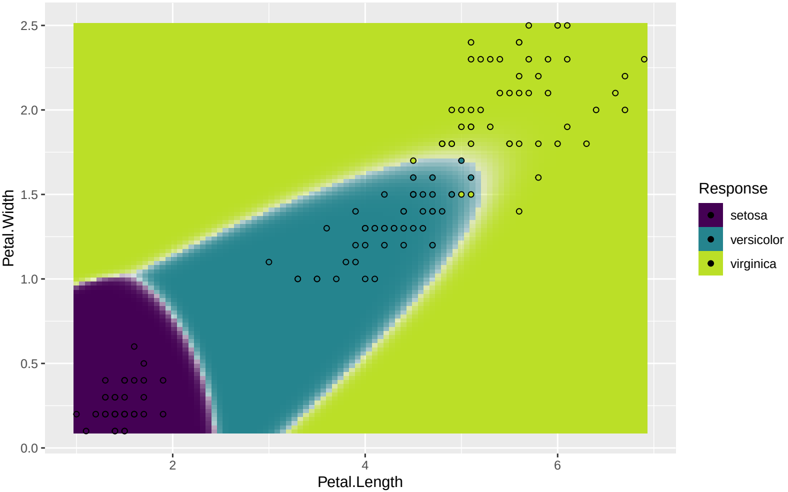
QUADRATIC DISCRIMINANT ANALYSIS (QDA)

$$\begin{aligned}\pi_k(\mathbf{x}) &\propto \pi_k \cdot p(\mathbf{x}|y=k)\\ &\propto \pi_k|\Sigma_k|^{-\frac{1}{2}}\exp(-\frac{1}{2}\mathbf{x}^T\Sigma_k^{-1}\mathbf{x}-\frac{1}{2}\boldsymbol{\mu}_{\boldsymbol{k}}^T\Sigma_k^{-1}\boldsymbol{\mu}_{\boldsymbol{k}}+\mathbf{x}^T\Sigma_k^{-1}\boldsymbol{\mu}_{\boldsymbol{k}})\end{aligned}$$

Taking the log of the above, we can define a discriminant function that is quadratic in x.

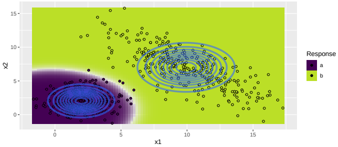
$$\log\pi_k-\frac{1}{2}\log|\Sigma_k|-\frac{1}{2}\boldsymbol{\mu}_{\boldsymbol{k}}^T\Sigma_k^{-1}\boldsymbol{\mu}_{\boldsymbol{k}}+\mathbf{x}^T\Sigma_k^{-1}\boldsymbol{\mu}_{\boldsymbol{k}}-\frac{1}{2}\mathbf{x}^T\Sigma_k^{-1}\mathbf{x}$$

QUADRATIC DISCRIMINANT ANALYSIS (QDA)



Einführung in das statistische Lernen

Classification: Naive Bayes



Learning goals

- Understand the idea of Naive Bayes
- Understand in which sense Naive Bayes is a special QDA model

NAIVE BAYES CLASSIFIER

NB is a generative multiclass technique. Remember: We use Bayes' theorem and only need $p(\mathbf{x}|y = k)$ to compute the posterior as:

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

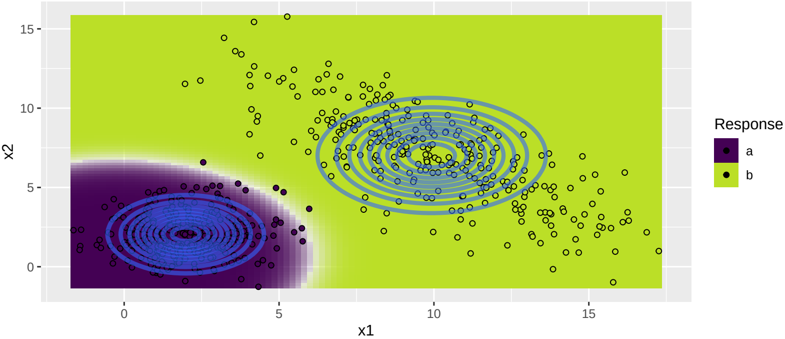
NB is based on a simple **conditional independence assumption**: the features are conditionally independent given class y.

$$p(\mathbf{x}|y = k) = p((x_1, x_2, ..., x_p)|y = k) = \prod_{j=1}^{p} p(x_j|y = k).$$

So we only need to specify and estimate the distribution $p(x_j|y = k)$, which is considerably simpler as this is univariate.

NB: NUMERICAL FEATURES

We use a univariate Gaussian for $p(x_j|y=k)$, and estimate (μ_j, σ_j^2) in the standard manner. Because of $p(\mathbf{x}|y=k) = \prod_{j=1}^{p} p(x_j|y=k)$, the joint conditional density is Gaussian with diagonal but non-isotropic covariance structure, and potentially different across classes. Hence, NB is a (specific) QDA model, with quadratic decision boundary.



NB: CATEGORICAL FEATURES

We use a categorical distribution for $p(x_j|y = k)$ and estimate the probabilities p_{kjm} that, in class k, our j-th feature has value m, $x_j = m$, simply by counting the frequencies.

$$p(x_j|y = k) = \prod_m p_{kjm}^{[x_j = m]}$$

Because of the simple conditional independence structure it is also very easy to deal with mixed numerical / categorical feature spaces.

LAPLACE SMOOTHING

If a given class and feature value never occur together in the training data, then the frequency-based probability estimate will be zero.

This is problematic because it will wipe out all information in the other probabilities when they are multiplied.

A simple numerical correction is to set these zero probabilities to a small value to regularize against this case.

NAIVE BAYES: APPLICATION AS SPAM FILTER

- In the late 90s, Naive Bayes became popular for e-mail spam filter programs
- Word counts were used as features to detect spam mails (e.g., "Viagra" often occurs in spam mail)
- Independence assumption implies: occurrence of two words in mail is not correlated
- Seems naive ("Viagra" more likely to occur in context with "Buy now" than "flower"), but leads to less required parameters and therefore better generalization, and often works well in practice.