EE4685 Machine Learning: a Bayesian Perspective

Lecture 2: Online Learning & Classic Classification Algorithms

Delft University of Technology

February 16, 2024

Content

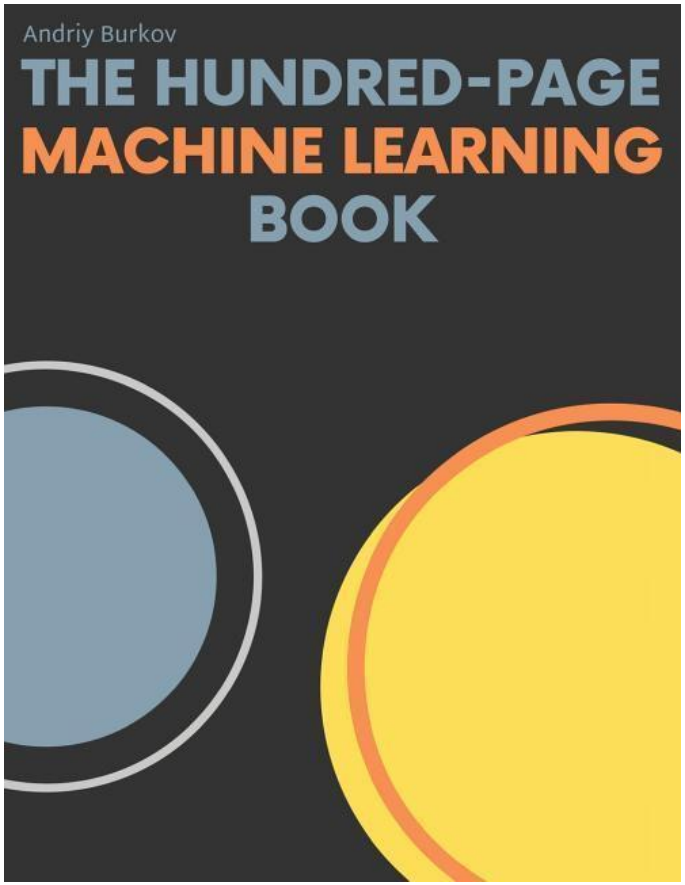
Lecture 2

- **Course logistics**
- Review of Lecture 1
- Online learning (Chapter 5)
- Classic classification algorithms (Chapter 7)
- Preparations for next week

Course logistics

- Book about ML basics
- Material for learning Python and scikit-learn
- Assignment 1

Book about ML basics

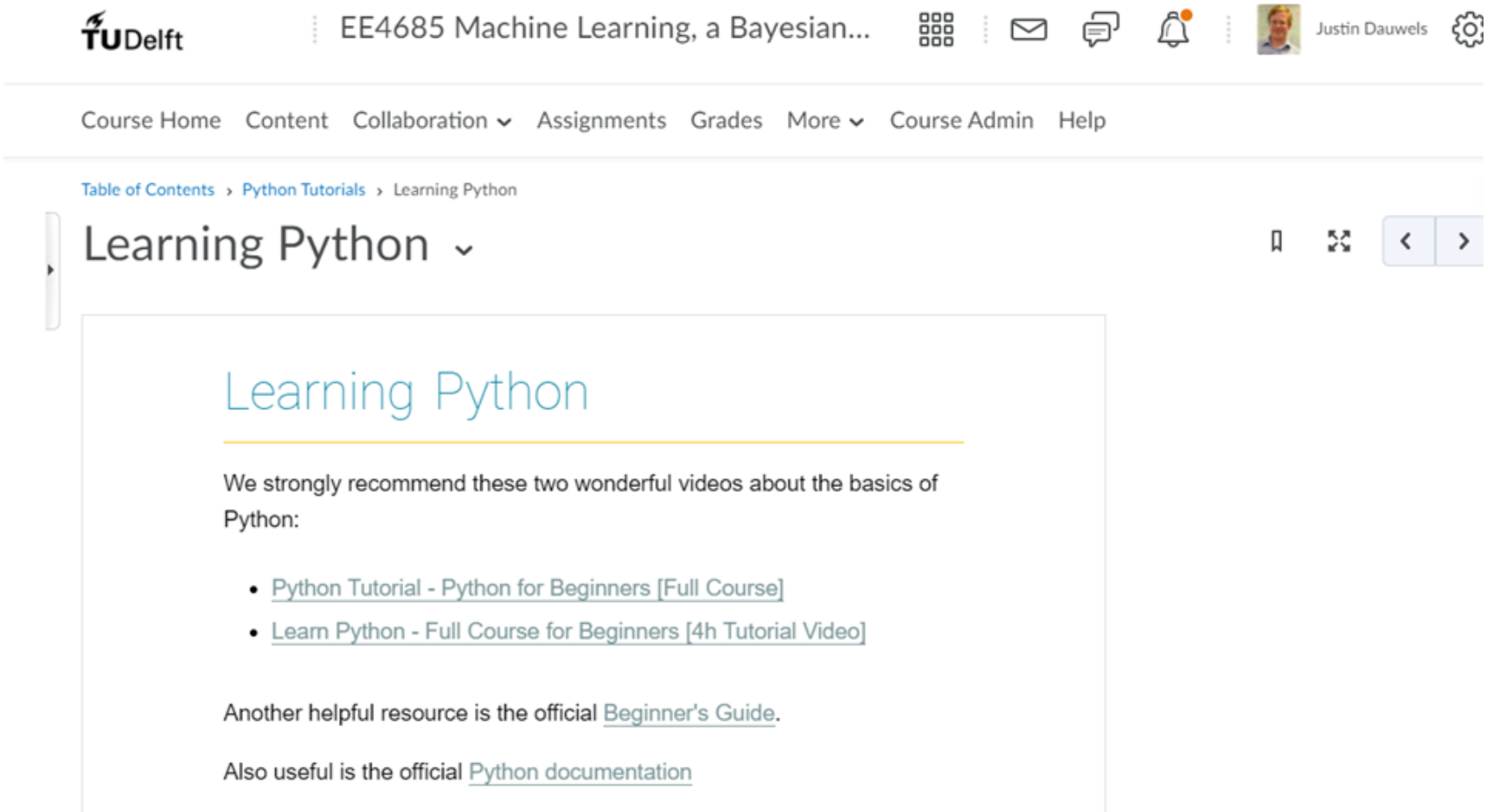


The Hundred-Page Machine Learning Book

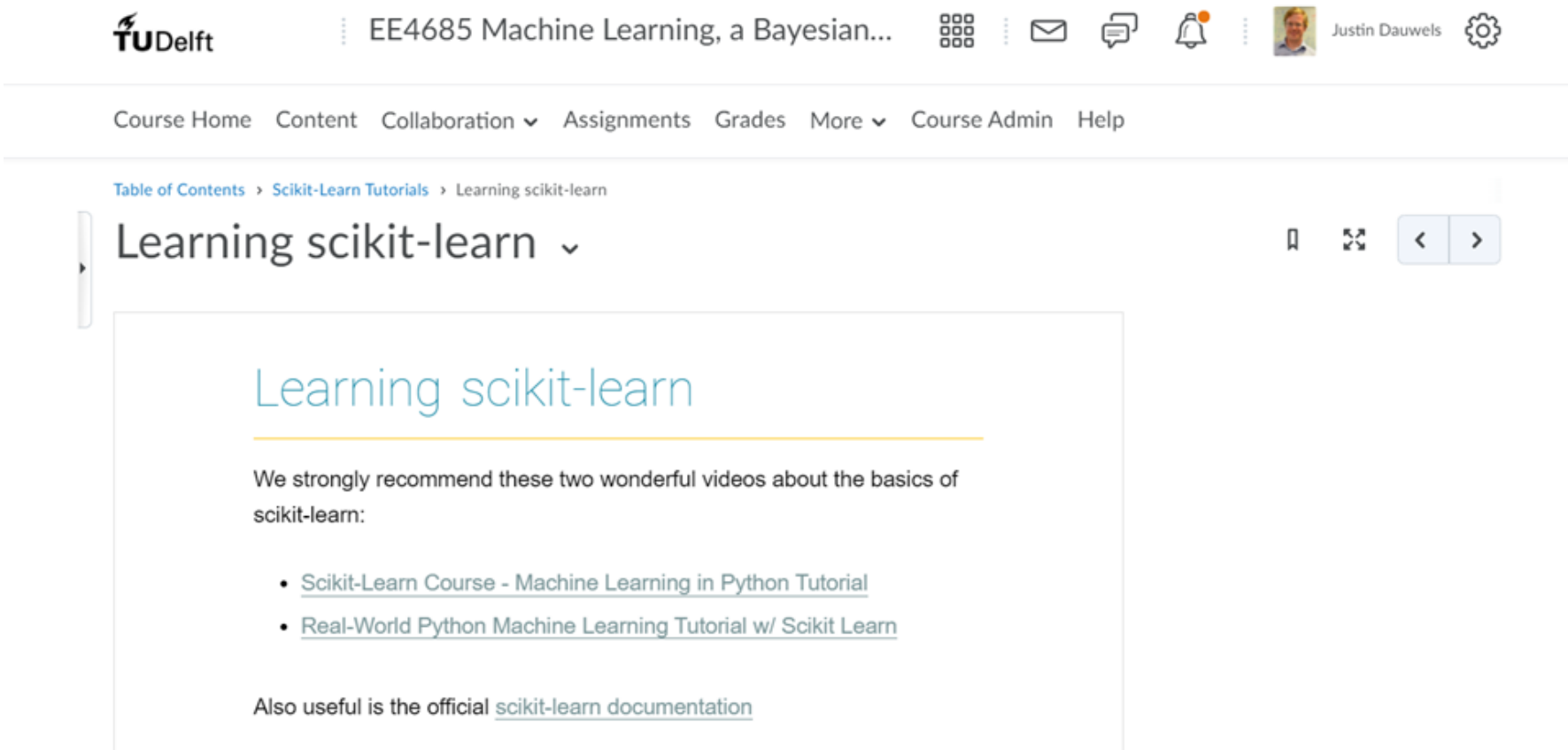
Andriy Burkov

- Succinct overview of ML:
  - Paradigms
  - Algorithms
  - Assessment
- Little to no theory (BSc level)
- Helpful
  - To get started with ML
  - To brainstorm about various solutions for an ML problem
- PDF and code available from book website

Material for learning Python



Material for learning scikit-learn



How to get help?

Table of Contents › Course Information › How to Get Help? › How to Get Help?

How to Get Help?

- **After-hours 'hotline'**
  - Course coordinator (Justin) will be available for online meeting in Google Meet on:
    - Tue: 1:00PM – 2:00PM
    - Wed: 9:00AM – 11:30AM
  - Purpose:
    - To seek help about:
      - course material
      - assignments (essays, projects)
      - general study/career advice
      - …
    - To provide feedback, suggestions, comments
  - Book a timeslot to consult with **Justin** here (provide your TU Delft email)
- **Email:**
  - **Justin**: J.H.G.Dauwels@tudelft.nl
  - We will try to respond within 24 hours.
  - We request you to send emails from your TU Delft account.
- **Forum:**
  - Under *Collaboration* > *Discussion* or click here
  - For questions about course material or logistical matters (e.g., submission deadline of an assignment)

Assignment 1: Introduction

- Machine learning can lead to better services, and generally, a better quality of life. However, this requires careful and ethical design of machine learning systems.
- Critical aspects of ML are as follows:
  - **Robustness**: Will the system work well under any circumstances?
  - **Safety**: Could the system cause harm?
  - **Privacy**: Is the system relying on confidential information from users?
  - **Biases**: Does the system favor certain categories of users?
  - **Ethics**: Is the application doing good for society at large?
  - **Quality control**: Do the datasets and the models meet certain standards?
- In assignment 1 we ask you to write a 5-page essay on societal aspects of ML. You can choose from 9 topics described on the Brightspace page ("Assignment 1")

Assignment 1: Topics

- Explainable AI
- Access to AI
- Impact of AI
- Dependence on AI
- Business of AI
- Responsible AI
- AI & I
- Control over AI
- AI and economy

Helpful book

Kush R. Varshney is a distinguished research staff member at IBM Research – T. J. Watson Research Center where he leads the machine learning group in the Foundations of Trustworthy AI department and co-directs the IBM Science for Social Good initiative. He has invented several new methods in the fairness, interpretability, robustness, transparency, and safety of machine learning systems and applied them with numerous private corporations and social change organizations. His team developed the AI Fairness 360, AI Explainability 360, and Uncertainty Quantification 360 open-source toolkits.

Trustworthy Machine Learning

Accuracy is not enough when you're developing machine learning systems for consequential application domains. You also need to make sure that your models are fair, have not been tampered with, will not fall apart in different conditions, and can be understood by people. Your design and development process has to be transparent and inclusive. You don't want the systems you create to be harmful, but to help people flourish in ways they consent to. All of these considerations beyond accuracy that make machine learning safe, responsible, and worthy of our trust have been described by many experts as the biggest challenge of the next five years. I hope this book equips you with the thought process to meet this challenge.

This book is most appropriate for project managers, data scientists, and other practitioners in high-stakes domains who care about the broader impact of their work, have the patience to think about what they're doing before they jump in, and do not shy away from a little math.

In writing the book, I have taken advantage of the dual nature of my job as an applied data scientist part of the time and a machine learning researcher the other part of the time. Each chapter focuses on a different use case that technologists tend to face when developing algorithms for financial services, health care, workforce management, social change, and other areas. These use cases are fictionalized versions of real engagements I've worked on. The contents bring in the latest research from trustworthy machine learning, including some that I've personally conducted as a machine learning researcher.

—Kush

Trustworthy Machine Learning • Varshney

Trustworthy Machine Learning

concepts for developing accurate, fair, robust, explainable, transparent, inclusive, empowering, and beneficial machine learning systems

Kush R. Varshney

Free PDF available on

http://www.trustworthymachinelearning.com/

Assignment 1: Specifications

- You are expected to work in **groups of 2**. Therefore, please sign up here for a project as a group of 2.
- We are expecting a 5-page essay (excluding references) + Ouriginal similarity report (similarity needs to be less than 10%)
- To be submitted by **Fri 15 March (midnight)**.
- Assignment 1 contributes **15%** to the total grade.

Assignment 1: Structure of the essay

- Introduction of the topic (e.g., responsible AI)
- Motivation: Why is the topic important?
- Examples: Provide a few examples to illustrate the topic
- Literature review: Summarize existing viewpoints on the topic
- Discussion
  - Compare the different existing viewpoints
  - What is your personal viewpoint?
- Conclusions (wrap-up and outlook for the future)
  - Brief summary
  - Suggest ideas for future research
- References from the literature

Assignment 1: Assessment

Assignment 1 will be **assessed** according to a variety of criteria:

- originality and complexity of the analysis and argumentation,
- depth of the literature review and discussion,
- quality of the writing (grammar, spelling, structure).

Assignment 1: Peer review

- Each group of 2 students is asked to **assess two essays from their peers**; for conducting those peer reviews, the students can obtain a maximum of **3% of the total grade**.
- In each peer review, the students are advised to include **3 strengths** of the essay, **3 weaknesses** of the essay (areas for improvement), and suggest a few ideas for future explorations.

Assignment 1: Feedback

- We have created a thread on the **forum** for this assignment. We recommend the students to discuss your essays with your peers and instructors.
- We recommend the students to **discuss** their project with the instructor(s) on Google Meet, at least once during the course period. The students can book a timeslot for this purpose.

Content

Lecture 2

- Course logistics
- **Review of Lecture 1**
- Online learning (Chapter 5)
- Classic classification algorithms (Chapter 7)
- Preparations for next week

Review of Lecture 1

- Learn a **function** $\mathbf{f}(\mathbf{x}, \theta)$ that maps input data x to prediction y (regression & classification) from training data $D = \{(x_k, y_k)\}_{k=1,\ldots,K}$
- What are the fundamental principles and approaches of
  - optimization-based machine learning?
  - probabilistic machine learning?
- What are challenges in machine learning?

Review of Lecture 1

- **Optimization**: Determine that function f by minimizing a loss function L based on D
- **Probabilistic ML**: The function f is derived from probabilistic models:
  - $p(y \mid x; \theta) \rightarrow$ frequentist approach; θ : unknown **fixed parameter**
  - $p(\theta \mid D) \rightarrow$ Bayesian approach ['inference']; θ : **random variable** with pdf $p(\theta)$.
  - Make predictions with:
    - $f(x) = \int f(x, \theta) p(\theta \mid D) d\theta$ [full Bayesian] or approximation [approximate Bayes]
    - $f(x, \hat{\theta})$ [simpler, but uncertainty in θ is discarded] with $\hat{\theta}$ = mode, mean, median, etc. of $p(y; \theta)$ or $p(\theta \mid y)$ [estimation]
    - ML and MAP estimation: $\hat{\theta}$ = mode of $p(y; x, \theta)$ and $p(\theta \mid D)$ respectively

Review of Lecture 1

- **Challenge**: Loss L based on D (empirical loss) is only approximation of *expected* loss
  - *Overfitting* if number of samples K is small and θ is high-dimensional
  - *Curse of dimensionality*: Many samples needed if θ is high-dimensional
  - Proper *validation* is required to assess the predictor f
- **Solution**:
  - Regularization of loss L (extra term in θ) or suitable priors on θ
  - Grow the dimensionality of θ as you get more samples (data)

Discussion

The previous narrative is abstract and **many questions remain** . . .

What aspects of optimization-based and probabilistic ML remain **unclear** for you?

Review of Lecture 1-What's next?

- How to choose the loss L?
- How to minimize loss L?
- How to choose $f(x, \theta), p(y \mid x; \theta), p(\theta \mid y)$?
- How to compute $f(x) = \int f(x, \theta) p(\theta \mid D) d\theta$?
- How to compute ML/MAP estimates $\hat{\theta}$ from $p(y \mid x; \theta), p(\theta \mid D)$ resp.
- How to avoid overfitting by regularization or choosing priors on θ?
- How to grow the dimensionality of θ as you get more samples (data)?

Lecture 2: Preview

- How to minimize loss L?
  - Online Learning: the Stochastic Gradient Descent Family of Algorithms (Chapter 5)
- What are common choices for $f(x, \theta), p(y \mid x; \theta), p(\theta \mid y)$ for classification?
  - Classification (Chapter 7):
    - Bayesian classifier; intersection of hypersurfaces
    - Naïve Bayes classifier
    - Linear/Quatratic Discriminant Algorithm (LDA, QDA)

Content

Lecture 2

- Course logistics
- Review of Lecture 1
- **Online learning (Chapter 5)**
- Classic classification algorithms (Chapter 7)
- Preparations for next week

Online learning-Introduction

- Online learning algorithms:
  - Update the available estimate every time a measurement set (input-output pair of observations) is acquired
  - Operate on a single data point at a time → Do not require the training set to be known and stored in advance
  - Have the agility to learn and track slow time variations of the statistics of the involved processes/variables [adaptive]
- Batch processing methods: Process the whole block of data as a single entity

Starting point-Gradient Descent

- Gradient descent is one of the most widely used methods for iterative minimization of a differentiable cost function, $J(\boldsymbol{\theta})$, $\boldsymbol{\theta} \in \mathbb{R}^l$
- Starts from an initial estimate $\boldsymbol{\theta}^{(0)}$
- Generates a sequence $\boldsymbol{\theta}^{(i)}, i = 1, 2, \ldots$, such that

$$\boldsymbol{\theta}^{(i)} = \boldsymbol{\theta}^{(i-1)} + \mu_i \Delta\boldsymbol{\theta}^{(i)}, i > 0,$$

where $\mu_i > 0$.

- μ_i: step-size
- $\Delta\boldsymbol{\theta}^{(i)}$: update direction

Gradient Descent

- The choice of $\Delta\boldsymbol{\theta}^{(i)}$ is done such that to guarantee that

$$J\left(\boldsymbol{\theta}^{(i)}\right) < J\left(\boldsymbol{\theta}^{(i-1)}\right),$$

  except at a minimizer, $\boldsymbol{\theta}_*$.

- Assume that at the $i-1$ iteration step the value $\boldsymbol{\theta}^{(i-1)}$ has been obtained. Then, mobilizing a first order Taylor's expansion, we can write:

$$J\left(\boldsymbol{\theta}^{(i-1)} + \mu_i \Delta\boldsymbol{\theta}^{(i)}\right) \approx J\left(\boldsymbol{\theta}^{(i-1)}\right) + \mu_i \nabla^T J\left(\boldsymbol{\theta}^{(i-1)}\right) \Delta\boldsymbol{\theta}^{(i)}.$$

- Selecting the search direction so that

$$\nabla^T J\left(\boldsymbol{\theta}^{(i-1)}\right) \Delta\boldsymbol{\theta}^{(i)} < 0.$$

Quiz

- To satisfy the condition

$$\nabla^T J\left(\boldsymbol{\theta}^{(i-1)}\right) \Delta\boldsymbol{\theta}^{(i)} < 0,$$

the angle α between $\Delta\boldsymbol{\theta}^{(i)}$ and $\nabla^T J\left(\boldsymbol{\theta}^{(i-1)}\right)$ can be:

  - A. $\alpha \in [0, 90°]$
  - B. $\alpha \in (90°, 180°]$
  - C. $\alpha \in (180°, 270°)$

vevox

Join at `vevox.app`
ID: 193-199-435

Quiz

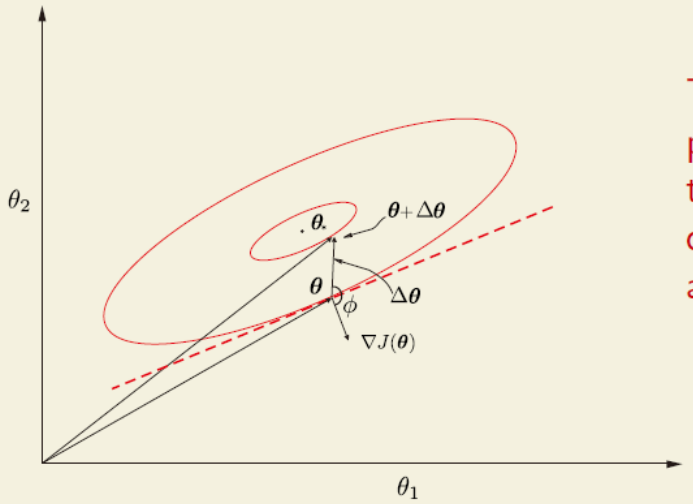
- To satisfy the condition

$$\nabla^T J\left(\boldsymbol{\theta}^{(i-1)}\right) \Delta\boldsymbol{\theta}^{(i)} < 0,$$

the angle α between $\Delta\boldsymbol{\theta}^{(i)}$ and $\nabla^T J\left(\boldsymbol{\theta}^{(i-1)}\right)$ can be:

  - A. $\alpha \in [0, 90°]$
  - B. $\alpha \in (90°, 180°]$
  - C. $\alpha \in (180°, 270°]$

Gradient Descent



The gradient vector at a point θ is perpendicular to the tangent plane at the isovalue curve crossing θ. The descent direction forms an **obtuse** angle, ϕ, with the gradient vector.

- Two issues:
  - To choose the best search direction along which to move
  - To compute how far along this direction one can go

Gradient Descent-Search direction

- Search direction:
  - Assume that $\mu_i = 1$ and search for all vectors, $\mathbf{z}$, with unit Euclidean norm, $\|\mathbf{z}\| = 1$. Then, the direction that gives the most negative value of $\nabla^T J\left(\boldsymbol{\theta}^{(i-1)}\right)\boldsymbol{z}$ is that of the negative gradient, i.e.,

$$\mathbf{z} = -\frac{\nabla J\left(\boldsymbol{\theta}^{(i-1)}\right)}{\left\|\nabla J\left(\boldsymbol{\theta}^{(i-1)}\right)\right\|}.$$

  - This comprises the gradient or steepest descent direction, and it leads to,

$$\boldsymbol{\theta}^{(i)} = \boldsymbol{\theta}^{(i-1)} - \mu_i \nabla J\left(\boldsymbol{\theta}^{(i-1)}\right).$$

Application to the MSE loss function

- Let us now apply the gradient descent scheme to derive an iterative algorithm to minimize the mean square error (MSE) cost function,

$$J(\boldsymbol{\theta}) = \mathbb{E}\left[\left(\mathrm{y} - \boldsymbol{\theta}^T \mathbf{x}\right)^2\right]$$
$$= \sigma_y^2 - 2\boldsymbol{\theta}^T \boldsymbol{p} + \boldsymbol{\theta}^T \Sigma_x \boldsymbol{\theta},$$

where

$$\Sigma_x := \mathbb{E}\left[\mathbf{x}\mathbf{x}^T\right], \quad \boldsymbol{p} := \mathbb{E}[\mathrm{y}\mathbf{x}]$$

are the covariance matrix and the cross-correlation vector and σ_y^2 the variance of y.

- The gradient of the cost function w.r. to $\boldsymbol{\theta}$ is readily seen to be

$$\nabla J(\boldsymbol{\theta}) = 2\Sigma_x \boldsymbol{\theta} - 2\boldsymbol{p}.$$

Application to the MSE loss function

- The fixed step-size case:

$$\boldsymbol{\theta}^{(i)} = \boldsymbol{\theta}^{(i-1)} + \mu\left(\boldsymbol{p} - \Sigma_x \boldsymbol{\theta}^{(i-1)}\right).$$

- The values of the step-size that guarantee convergence lie in the interval

$$0 < \mu < 2/\lambda_{\max}.$$

- The optimal value:

$$\mu_o = \frac{2}{\lambda_{\max} + \lambda_{\min}},$$

where $\lambda_{\max}$ and $\lambda_{\min}$ denotes the maximum and minimum eigenvalue of Σ_x, respectively.

Example

- The aim of the example is to demonstrate what we have said so far, concerning the convergence issues of the gradient descent scheme

$$\boldsymbol{\theta}^{(i)} = \boldsymbol{\theta}^{(i-1)} + \mu\left(\boldsymbol{p} - \Sigma_x \boldsymbol{\theta}^{(i-1)}\right).$$

- The cross-correlation vector was chosen to be

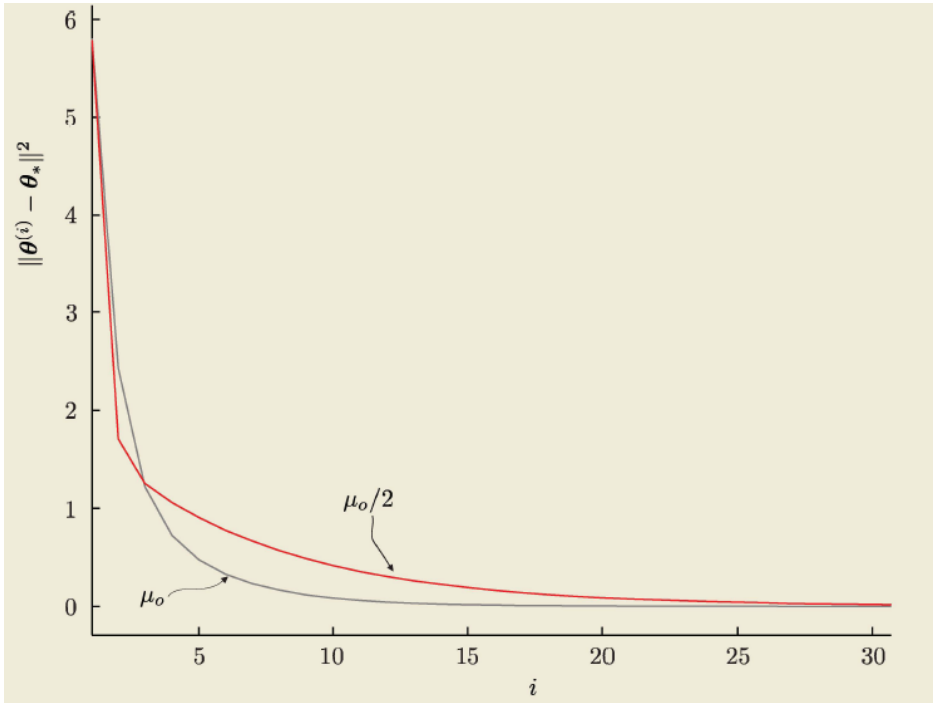
$$\boldsymbol{p} = [0.05, 0.03]^T,$$

and we consider two different covariance matrices,

$$\Sigma_1 = \begin{bmatrix} 1 & 0 \\ 0 & 0.1 \end{bmatrix} \quad \Sigma_2 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}.$$

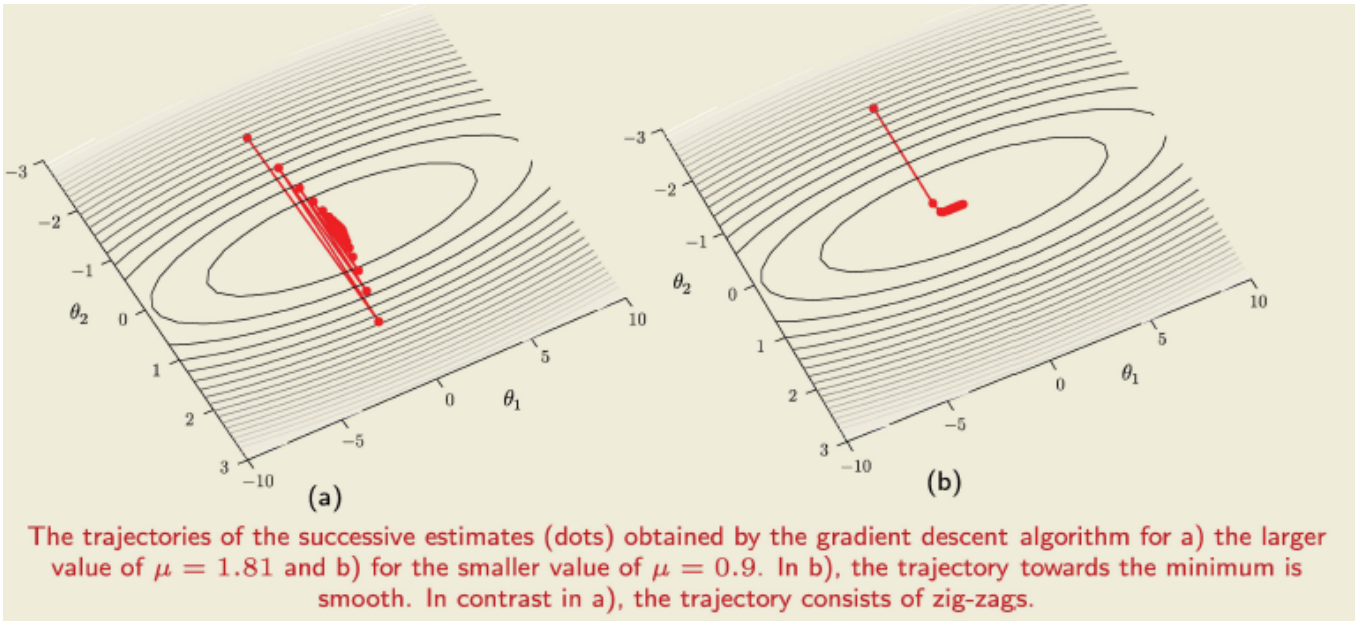
Example

- The error curves for two values of μ, for the case of Σ_1
  - Gray: $\mu_o = 1.81$ (optimum value)
  - Red: $\mu = \mu_o/2 = 0.9$.
- It may happen that the initial convergence for some $\mu \neq \mu_o$ to be faster compared to μ_o.
- What the theory guarantees is that, eventually, the curve corresponding to the optimal will tend to zero faster than for any other value of μ.



Example

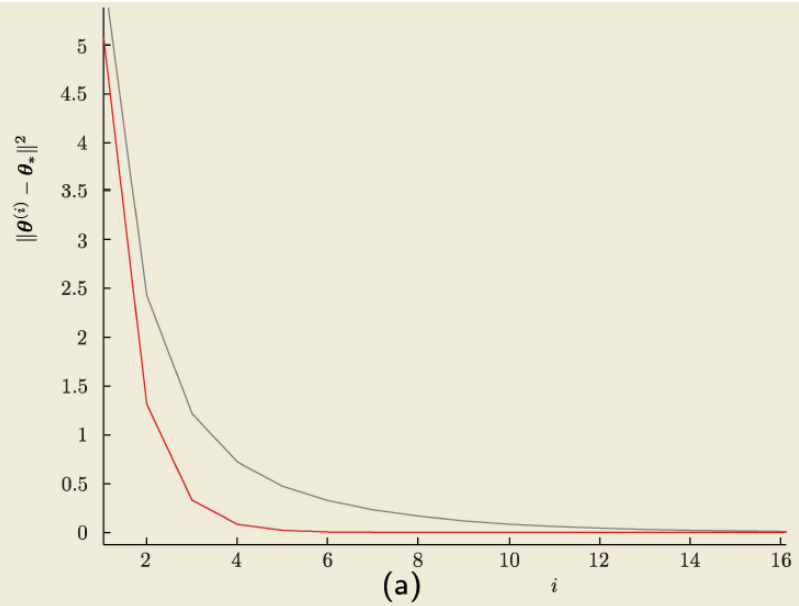
- Observe the zig-zag path, which corresponds to the larger value of $\mu = 1.81$ compared to the smoother one obtained for the smaller step size $\mu = 0.9$.



The trajectories of the successive estimates (dots) obtained by the gradient descent algorithm for a) the larger value of $\mu = 1.81$ and b) for the smaller value of $\mu = 0.9$. In b), the trajectory towards the minimum is smooth. In contrast in a), the trajectory consists of zig-zags.

Example

- The error curves using the same step size, $\mu = 1.81$, for both cases, Σ_1 and Σ_2.
- Observe that large eigenvalues spread of the input covariance matrix slows down the convergence rate.



For the same value of $\mu = 1.81$, the error curves for the case of unequal eigenvalues ($\lambda_1 = 1$ and $\lambda_2 = 0.1$) (gray) and for equal eigenvalues ($\lambda_1 = \lambda_2 = 1$).

Gradient Descent: Time-varying step-sizes

- Time-varying step-sizes: The previous analysis cannot be carried out for the case of an iteration-dependent step-size. It can be shown that, in this case, the gradient descent algorithm converges if
  - $\mu_i \to 0$, as $i \to \infty$
  - $\sum_{i=1}^{\infty} \mu_i = \infty$.

Gradient Descent: Time-varying step-sizes

- Time-varying step-sizes: The previous analysis cannot be carried out for the case of an iteration-dependent step-size. It can be shown that, in this case, the gradient descent algorithm converges if
  - $\mu_i \to 0$, as $i \to \infty$
  - $\sum_{i=1}^{\infty} \mu_i = \infty$.

  Why do we need those two conditions?

Gradient Descent: Time-varying step-sizes

- Time-varying step-sizes: The previous analysis cannot be carried out for the case of an iteration-dependent step-size. It can be shown that, in this case, the gradient descent algorithm converges if
  - $\mu_i \to 0$, as $i \to \infty$
  - $\sum_{i=1}^{\infty} \mu_i = \infty$.
- A typical example of sequences, which comply with both conditions, are those which satisfy the following:

$$\sum_{i=1}^{\infty} \mu_i^2 < \infty, \quad \sum_{i=1}^{\infty} \mu_i = \infty.$$

Quiz

- Which sequence satisfy the condition:

$$\sum_{i=1}^{\infty} \mu_i^2 < \infty, \quad \sum_{i=1}^{\infty} \mu_i = \infty$$

  - A. $\mu_i = 1/i$
  - B. $\mu_i = 1/i^2$
  - C. $\mu_i = 2^i$

vevox

Join at `vevox.app`
ID: 193-199-435

Quiz

- Which sequence satisfy the condition:

$$\sum_{i=1}^{\infty} \mu_i^2 < \infty, \quad \sum_{i=1}^{\infty} \mu_i = \infty$$

  - A. $\mu_i = 1/i$
  - B. $\mu_i = 1/i^2$
  - C. $\mu_i = 2^i$

Stochastic approximation

- Solving for the normal equations as well as using the gradient descent iterative scheme (for the case of the MSE), one has to have access to the second order statistics of the involved processes/variables.
- However, in most of the cases, this is not known and it has to be approximated using a set of measurements.
- We now turn our attention to algorithms that can learn the statistics iteratively via the training set. Such techniques evolve around the method of stochastic approximation, or the Robbins-Monro algorithm.

Stochastic approximation

- Let us consider the case of a function which is defined in terms of the expected value of another one, i.e.,

$$f(\boldsymbol{\theta}) = \mathbb{E}[\phi(\boldsymbol{\theta}, \boldsymbol{\eta})], \boldsymbol{\theta} \in \mathbb{R}^l,$$

  where η is a random vector of unknown statistics. The goal is to compute a root of $f(\boldsymbol{\theta})$.
  - If the statistics were known → root-finding algorithms
  - However, the statistics are unknown → the Robbins-Monro algorithm.

Stochastic approximation

- The Robbins-Monro algorithm: Robbins and Monro proved that the following iterative scheme

$$\boldsymbol{\theta}_n = \boldsymbol{\theta}_{n-1} - \mu_n \phi(\boldsymbol{\theta}_{n-1}, \boldsymbol{\eta}_n),$$

starting from an arbitrary initial condition, $\boldsymbol{\theta}_{-1}$, converges (in probability) to a root of $f(\boldsymbol{\theta})$, under some general conditions and provided that

$$\sum_n \mu_n^2 < \infty, \quad \sum_n \mu_n \to \infty.$$

Stochastic approximation

- In other words, in the previous iterative scheme, we get rid of the expectation operation and use the value of $\phi(\cdot, \cdot)$, which is computed using the **current** observations/measurements and the currently available estimate.
- That is, the algorithm learns both the statistics as well as the root; two into one!

Stochastic approximation

- Cost function optimization: In the context of optimizing a general differentiable cost function of the form:

$$J(\boldsymbol{\theta}) = \mathbb{E}[\mathcal{L}(\boldsymbol{\theta}, \mathrm{y}, \mathbf{x})],$$

the Robbins-Monro scheme can be mobilized to find a root of the respected gradient, i.e,

$$\nabla J(\boldsymbol{\theta}) = \mathbb{E}[\nabla \mathcal{L}(\boldsymbol{\theta}, \mathrm{y}, \mathbf{x})],$$

where the expectation is w.r. to the pair $(\mathrm{y}, \mathbf{x})$.

- Given the sequence of observations $(y_n, \boldsymbol{x}_n)$, $n = 0, 1, \ldots$, the Robbins-Monro recursion now becomes

$$\boldsymbol{\theta}_n = \boldsymbol{\theta}_{n-1} - \mu_n \nabla \mathcal{L}\left(\boldsymbol{\theta}_{n-1}, y_n, \boldsymbol{x}_n\right).$$

Stochastic approximation

- When to seize iterations?
  - Two quantities are of interest, namely the mean and the covariance matrix of the estimate **at iteration** n, i.e.,

$$\mathbb{E}[\boldsymbol{\theta}_n], \mathrm{Cov}(\boldsymbol{\theta}_n).$$

  - It can be shown that, if $\mu_n = \mathcal{O}(1/n)$ and assuming that iterations have brought the estimate close to the optimal value, then

$$\mathbb{E}[\boldsymbol{\theta}_n] = \boldsymbol{\theta}_* + \frac{1}{n}\mathbf{c}, \mathrm{Cov}(\boldsymbol{\theta}_n) = \frac{1}{n}V + \mathcal{O}(1/n^2),$$

    where $\mathbf{c}$ and V are constants that depend on the cost function.

Stochastic approximation

- That is, both the mean as well as the standard deviations of the components follow an $\mathcal{O}(1/n)$ pattern. Moreover, these formulae indicate that the parameter vector estimate **fluctuates** around the optimal value.
- This fluctuation depends on the choice of the sequence μ_n, being smaller for smaller values of the step-size sequence. However, μ_n cannot be made to decrease very fast due to the two convergence conditions, as discussed before. This is the price one pays for using the **noisy version** of the gradient and it is the reason that such schemes suffer from relatively slow convergence rates.

Quiz

- Given the training sequence of observations, $(y_n, \boldsymbol{x}_n)$, which are assumed to be i.i.d. drawn from the joint distribution of $(\mathrm{y}, \mathbf{x})$. Write down the Robbins-Monro recursion for the MSE linear estimation problem.
  Hint: The loss function is

$$J(\boldsymbol{\theta}) = \mathbb{E}\left[\left(\mathrm{y} - \boldsymbol{\theta}^T \mathbf{x}\right)^2\right]$$

Quiz

- The gradient of the cost function can be written as

$$\nabla J(\boldsymbol{\theta}) = \mathbb{E}[\mathbf{x}(\mathbf{x}^T\boldsymbol{\theta} - y)].$$

- Therefore the Robbins-Monro algorithm becomes

$$\boldsymbol{\theta}_n = \boldsymbol{\theta}_{n-1} + \mu_n \boldsymbol{x}_n \left(y_n - \boldsymbol{x}_n^T \boldsymbol{\theta}_{n-1}\right).$$

- Compare the above Robbins-Monro recursion with the gradient descent one, i.e.,

$$\boldsymbol{\theta}^{(i)} = \boldsymbol{\theta}^{(i-1)} + \mu \left(\boldsymbol{p} - \Sigma_x \boldsymbol{\theta}^{(i-1)}\right).$$

  The former equation results from the latter one by dropping out the expectation operations and using an iteration-dependent step-size.

Example

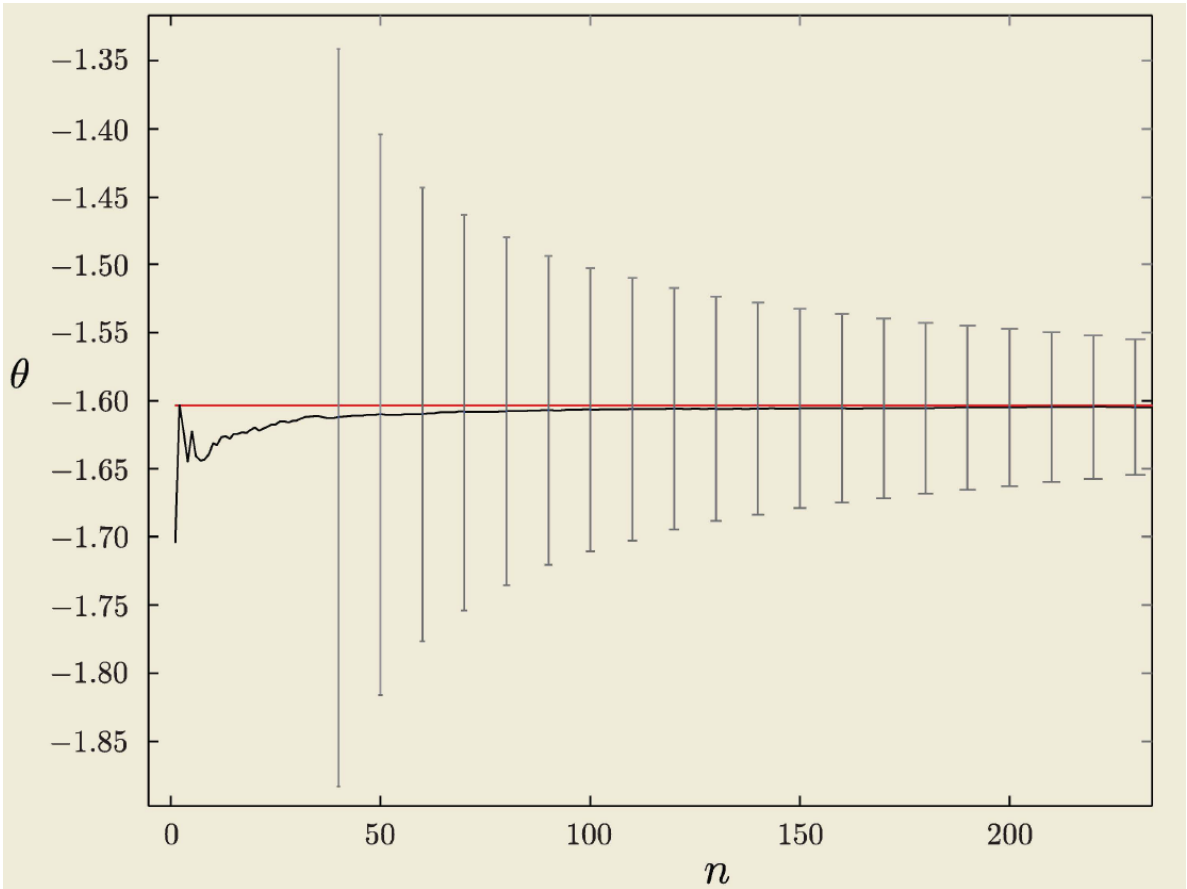
- Data samples were first generated according to the regression model

$$y_n = \boldsymbol{\theta}^T \boldsymbol{x}_n + \eta_n,$$

  where, $\boldsymbol{\theta} \in \mathbb{R}^2$ was randomly chosen and then fixed. The elements of $\boldsymbol{x}_n$ were i.i.d generated via a normal distribution $\mathcal{N}(0,1)$ and η_n is a white noise sequence with variance equal to $\sigma^2 = 0.1$.
- Then, the observations $(y_n, \boldsymbol{x}_n)$ were used in the Robbins-Monro recursive scheme in order to obtain an estimate of $\boldsymbol{\theta}$.

Example

- The figure below shows the resulting curve for one of the parameters (the other one being similar).
- Observe that the mean values of the estimates tend to the true value, and the standard deviation keeps decreasing as n grows. The step-size was chosen equal $\mu_n = l/n$.



Quiz

- Can the SGD algorithm track changes?
  - A. Yes
  - B. No

Join at `vevox.app`
ID: 193-199-435

Quiz

- Can the SGD algorithm track changes?
  - A. Yes
  - B. No

Quiz

- Once the algorithm has converged, it "locks" at the obtained solution.
- If the statistics of the involved variables/ unknown parameters start changing, the error term

$$e_n = y_n - \boldsymbol{\theta}_{n-1}^T \mathbf{x}_n,$$

  gets larger values; however, since μ_n is very small, the increased value of the error will not lead to corresponding changes of the estimate.
- Therefore, the SGD algorithm cannot track the changes.

The least-mean-square adaptive algorithms (LMS)

- This can be overcome if one sets the value of μ_n to a preselected fixed value, μ. The resulting algorithm is the celebrated LMS algorithm.
- **The LMS Algorithm**
  - Initialize
    - $\theta_{-1} = \mathbf{0} \in \mathbb{R}^l$; other values can also be used.
    - Select the value of μ.
  - For $n = 0, 1, \ldots,$ Do
    - $e_n = y_n - \boldsymbol{\theta}_{n-1}^T \boldsymbol{x}_n$
    - $\theta_n = \theta_{n-1} + \mu e_n x_n$
  - End For

Content

Lecture 2

- Course logistics
- Review of Lecture 1
- Online learning (Chapter 5)
- **Classic classification algorithms (Chapter 7)**
- Preparations for next week

Bayesian classification

- Bayesian Classification Rule: Given a set of M classes, $\omega_i, i = 1, 2, \ldots, M$, as well as the respective posterior probabilities, $P(\omega_i \mid \boldsymbol{x})$, classify an unknown feature vector, $\boldsymbol{x}$, according to the rule,

$$\text{Assign } x \text{ to } \omega_i = \arg\max_{\omega_j} P\left(\omega_j \mid x\right), j = 1, 2, \ldots, M.$$

In words, the unknown pattern, represented by $\boldsymbol{x}$, is assigned to the class for which the posterior probability becomes maximum.

Bayesian classification

- Employing Bayes' theorem,

$$P\left(\omega_j \mid \boldsymbol{x}\right) = \frac{p\left(\boldsymbol{x} \mid \omega_j\right) P\left(\omega_j\right)}{p(\boldsymbol{x})}, \quad j = 1, 2, \ldots, M,$$

where $p\left(\boldsymbol{x} \mid \omega_j\right)$ are the respective conditional probability distribution densities (pdf), the Bayesian classification rule becomes,

$$\text{Assign } x \text{ to } \omega_i = \arg\max_{\omega_j} p\left(x \mid \omega_j\right) P\left(\omega_j\right), \quad j = 1, 2 \ldots, M.$$

In other words, the classifier depends on the a-priori class probabilities as well as the respective conditional pdfs.

Bayesian classification

- Training Bayesian Classifiers: Let us assume that we are given a set of training points, $(y_n, \boldsymbol{x}_n) \in D \times \mathbb{R}^l, n = 1, 2, \ldots, N$, and consider the general task comprising M classes. Assume that each class, $\omega_i, i = 1, 2, \ldots, M$, is represented by N_i points in the training set, with $\sum_{i=1}^{M} N_i = N$.
- Then, the a-priori probabilities can be approximated by,

$$P(\omega_i) \approx \frac{N_i}{N}, i = 1, 2, \ldots, M$$

- For the conditional pdfs, $p(\boldsymbol{x} \mid \omega_i), i = 1, 2 \ldots, M$, any method for estimating pdfs can be mobilized. For example,
  - Maximum Likelihood method (ML) for pdf estimation
  - Non-parametric histogram-like techniques

Bayesian classification

- For a two-class task, let $\mathcal{R}_1, \mathcal{R}_2$ be the two regions in, say, $\mathbb{R}^l$, where we decide in favor of class ω_1 and ω_2, respectively. The probability of classification error is given by

$$P_e = P\left(\boldsymbol{x} \in \mathcal{R}_1, \boldsymbol{x} \in \omega_2\right) + P\left(\boldsymbol{x} \in \mathcal{R}_2, \boldsymbol{x} \in \omega_1\right).$$

  That is, it is equal to the probability of the feature vector to belong to class $\omega_1\,(\omega_2)$ and at the same time to lie in the "wrong" region $\mathcal{R}_2\,(\mathcal{R}_1)$ in the feature space.

Bayesian classification

- The previous equation can be written as,

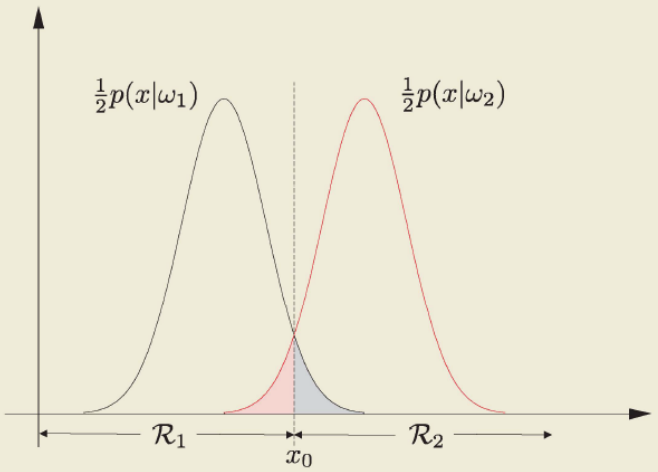
$$P_e = P(\omega_2)\, P(x \in \mathcal{R}_1 \mid x \in \omega_2) + P(\omega_1)\, P(x \in \mathcal{R}_2 \mid x \in \omega_1)$$

or

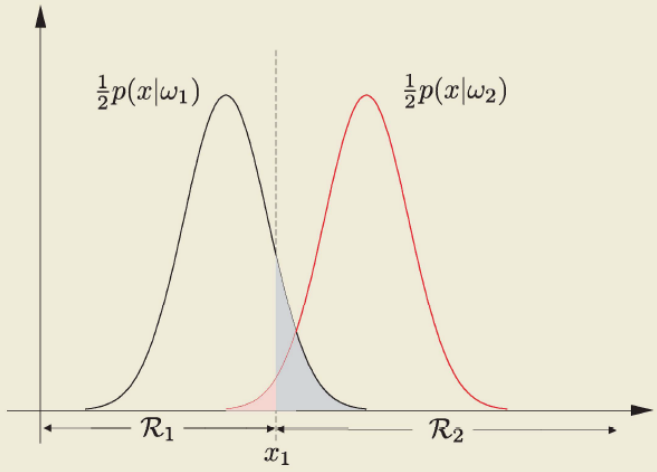
$$P_e = P(\omega_2) \int_{\mathcal{R}_1} p(\boldsymbol{x} \mid \omega_2)\, d\boldsymbol{x} + P(\omega_1) \int_{\mathcal{R}_2} p(\boldsymbol{x} \mid \omega_1)\, d\boldsymbol{x}$$

- The Bayesian classifier minimizes P_e with respect to $\mathcal{R}_1$ and $\mathcal{R}_2$. This is also true for the general case of M classes.

Bayesian classification



(a)



(b)

a) The classification error probability for partitioning the feature space for two equiprobable classes, according to the Bayesian optimal classifier, is equal to the area of the shaded region. b) Moving the threshold value away from the value corresponding to the optimal Bayes rule increases the probability of error, as it is indicated by the increase of the area of the corresponding shaded region.

Bayesian classification

- However, minimizing the classification error probability is not a good option in some cases.
- For example, in a medical diagnosis system, committing an error by predicting the class of a finding in an x-ray image as being "malignant", while its true class is "normal" , is less significant than an error in the other way round.
- Why?

Average risk rule

- For such cases, relative weights on the errors according to their importance have to be used. The resulting function is known as the average risk.
- For the two-class case, the risk or loss associated with each one of the two classes is defined as,

$$r_1 = \lambda_{11} \int_{\mathcal{R}_1} p(\boldsymbol{x} \mid \omega_1)\, d\boldsymbol{x} + \lambda_{12} \underbrace{\int_{\mathcal{R}_2} p(\boldsymbol{x} \mid \omega_1)\, d\boldsymbol{x}}_{\text{error}}$$

$$r_2 = \lambda_{21} \underbrace{\int_{\mathcal{R}_1} p(\boldsymbol{x} \mid \omega_2)\, d\boldsymbol{x}}_{\text{error}} + \lambda_{22} \int_{\mathcal{R}_2} p(\boldsymbol{x} \mid \omega_2)\, d\boldsymbol{x}$$

Typically, $\lambda_{11} = \lambda_{22} = 0$, since they correspond to correct decisions.

Example of Bayesian classification & average risk rule

- The average risk, to be minimised, is given by

$$r = P(\omega_1)\, r_1 + P(\omega_2)\, r_2.$$

For the two-class case, it is readily seen that the rule becomes

$$\text{Assign } \boldsymbol{x} \text{ to } \omega_1 \,(\omega_2) \text{ if : } \lambda_{12} P(\omega_1 \mid \boldsymbol{x}) > (<) \lambda_{21} P(\omega_2 \mid \boldsymbol{x}),$$

or equivalently

$$\text{Assign } \boldsymbol{x} \text{ to } \omega_1 \,(\omega_2) \text{ if :}$$
$$\underbrace{\lambda_{12} P(\omega_1)}_{P'(\omega_1)}\, p(\boldsymbol{x} \mid \omega_1) > (<) \underbrace{\lambda_{21} P(\omega_2)}_{P'(\omega_2)}\, p(\boldsymbol{x} \mid \omega_2).$$

Quiz

Assign $\boldsymbol{x}$ to ω_1 (ω_2) if :

$$\underbrace{\lambda_{12} P(\omega_1)}_{P'(\omega_1)} p(\boldsymbol{x} \mid \omega_1) > (<) \underbrace{\lambda_{21} P(\omega_2)}_{P'(\omega_2)} p(\boldsymbol{x} \mid \omega_2).$$

- If ω_1 denotes "malignant" and ω_2 denotes "normal", how would you choose λ_{12} and λ_{21}?
  - A. $\lambda_{12} = \lambda_{21}$
  - B. $\lambda_{12} > \lambda_{21}$
  - C. $\lambda_{12} < \lambda_{21}$

vevox

Join at `vevox.app`

ID: 193-199-435

Quiz

Assign $\boldsymbol{x}$ to ω_1 (ω_2) if :

$$\underbrace{\lambda_{12} P(\omega_1)}_{P'(\omega_1)} p(\boldsymbol{x} \mid \omega_1) > (<) \underbrace{\lambda_{21} P(\omega_2)}_{P'(\omega_2)} p(\boldsymbol{x} \mid \omega_2).$$

- If ω_1 denotes "malignant" and ω_2 denotes "normal", how would you choose λ_{12} and λ_{21}?
  - A. $\lambda_{12} = \lambda_{21}$
  - B. $\lambda_{12} > \lambda_{21}$
  - C. $\lambda_{12} < \lambda_{21}$

Example of Bayesian classification & average risk rule

- In a two-class one-dimensional classification task, the data in the two classes are distributed according to the following two Gaussians,

$$p(x \mid \omega_1) = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{x^2}{2}\right)$$

and

$$p(x \mid \omega_2) = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{(x-1)^2}{2}\right)$$

The problem is more sensitive with respect to errors committed on patterns from class ω_1, which is expressed via the following loss matrix, which is the matrix comprising the respective weights,

$$L := \begin{bmatrix} \lambda_{11} & \lambda_{12} \\ \lambda_{21} & \lambda_{22} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0.5 & 0 \end{bmatrix}.$$

The two classes are considered equiprobable.

Example of Bayesian classification & average risk rule

- The goal is to derive the threshold values, x_r, and x_B, for the average risk and the Bayesian classification respectively.
- **Solution**: According to the average risk rule, the region for which we decide in favor of class ω_1 is given by:

$$\mathcal{R}_1 : \lambda_{12}\frac{1}{2}p\left(x \mid \omega_1\right) > \lambda_{21}\frac{1}{2}p\left(x \mid \omega_2\right),$$

and the respective threshold value, x_r, is computed by the equation,

$$\exp\left(-\frac{x_r^2}{2}\right) = 0.5\exp\left(-\frac{\left(x_r-1\right)^2}{2}\right),$$

which after taking the logarithm and solving the respective equation, trivially results in

$$x_r = \frac{1}{2}(1 - 2\ln 0.5)$$

Quiz

- The goal is to derive the threshold values, x_r, and x_B, for the average risk and the Bayesian classification respectively.
- **Solution**: According to the average risk rule, the region for which we decide in favor of class ω_1 is given by:

$$\mathcal{R}_1 : \lambda_{12}\frac{1}{2}p\left(x \mid \omega_1\right) > \lambda_{21}\frac{1}{2}p\left(x \mid \omega_2\right),$$

and the respective threshold value, x_r, is computed by the equation,

$$\exp\left(-\frac{x_r^2}{2}\right) = 0.5\exp\left(-\frac{\left(x_r-1\right)^2}{2}\right),$$

which after taking the logarithm and solving the respective equation, trivially results in

$$x_r = \frac{1}{2}(1 - 2\ln 0.5)$$

Is x_r larger than 0.5?

- A. Yes
- B. No

Quiz

- The goal is to derive the threshold values, x_r, and x_B, for the average risk and the Bayesian classification respectively.
- **Solution**: According to the average risk rule, the region for which we decide in favor of class ω_1 is given by:

$$\mathcal{R}_1 : \lambda_{12}\frac{1}{2}p\left(x \mid \omega_1\right) > \lambda_{21}\frac{1}{2}p\left(x \mid \omega_2\right),$$

and the respective threshold value, x_r, is computed by the equation,

$$\exp\left(-\frac{x_r^2}{2}\right) = 0.5\exp\left(-\frac{\left(x_r-1\right)^2}{2}\right),$$

which after taking the logarithm and solving the respective equation, trivially results in

$$x_r = \frac{1}{2}(1 - 2\ln 0.5)$$

Is x_r larger than 0.5?

- A. Yes
- B. No

Example of Bayesian classification

- The threshold for the Bayesian classifier results if we set $\lambda_{21} = 1$, which gives

$$x_B = \frac{1}{2}.$$

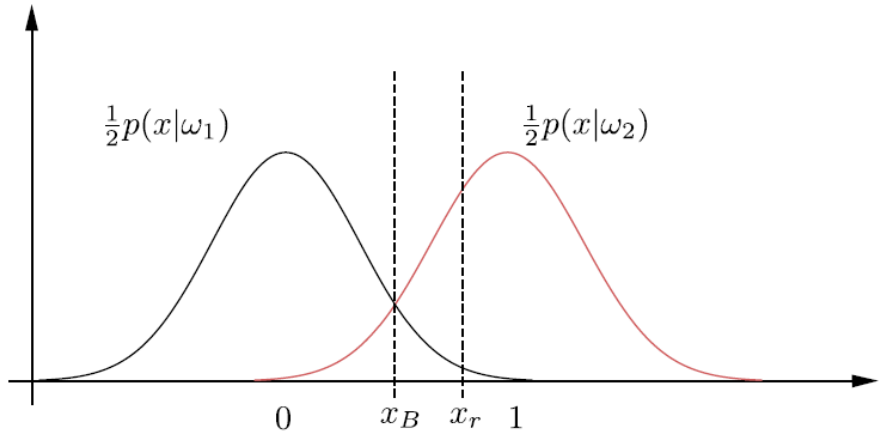


FIGURE 7.2

The class distributions and the resulting threshold values for the two cases of Example 7.1. Note that minimizing the average risk enlarges the region in which we decide in favor of the most sensitive class, ω_1.

Exercise

- The threshold for the Bayesian classifier results if we set $\lambda_{21} = 1$, which gives

$$x_B = \frac{1}{2}.$$

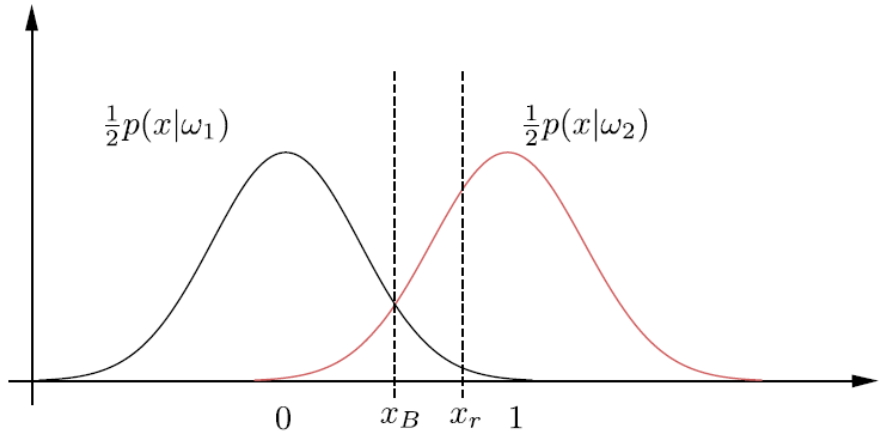


FIGURE 7.2

The class distributions and the resulting threshold values for the two cases of Example 7.1. Note that minimizing the average risk enlarges the region in which we decide in favor of the most sensitive class, ω_1.

Draw $\lambda_i \frac{1}{2} p(x|\omega_i)$

Decision hypersurfaces

- The goal of any classifier is to **partition** the feature space into regions. The partition is achieved via points in $\mathbb{R}$, curves in $\mathbb{R}^2$, surfaces in $\mathbb{R}^3$ and hypersurfaces in $\mathbb{R}^l$.
  - Any hypersurface, S, is expressed in terms of a function,

$$g : \mathbb{R}^l \longmapsto \mathbb{R},$$

    and it comprises all the points such that

$$S = \left\{ \boldsymbol{x} \in \mathbb{R}^l : g(\boldsymbol{x}) = 0 \right\}.$$

  - All points lying on one side of this hypersurface score $g(x) > 0$ and all the points on the other side score $g(x) < 0$. The resulting (hyper)surfaces are known as decision (hyper)surfaces.

Decision hypersurfaces

- For example, the respective decision hypersurface, which is formed by the Bayesian classifier for a two class classification task, is given by

$$g(\mathbf{x}) := P(\omega_1 \mid \mathbf{x}) - P(\omega_2 \mid \mathbf{x}) = 0.$$

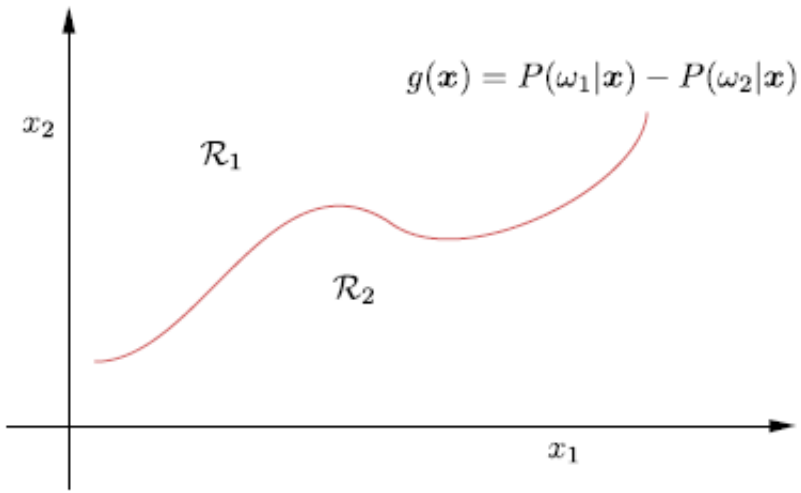


FIGURE 7.3

The Bayesian classifier implicitly forms hypersurfaces defined by $g(\boldsymbol{x}) = P(\omega_1|\boldsymbol{x}) - P(\omega_2|\boldsymbol{x}) = 0$.

Decision hypersurfaces-The Gaussian distribution case

- Plugging the Gaussian pdf into the conditionals in the products, $P(\omega_i)\, p(\boldsymbol{x} \mid \omega_i)$, and taking the respective logarithms, define

$$g_i(\boldsymbol{x}) := \ln \frac{P(\omega_i)}{(2\pi)^{l/2} |\Sigma_i|^{1/2}} \exp\left\{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_i)^T \Sigma_i^{-1} (\boldsymbol{x}-\boldsymbol{\mu}_i)\right\}.$$

  These are also known as discriminant functions.

- Thus, the decision hypersurface can be written as

$$\begin{aligned} g(\boldsymbol{x}) =& g_1(\boldsymbol{x}) - g_2(\boldsymbol{x}) = \ln \frac{P(\omega_1 \mid \mathbf{x})}{P(\omega_2 \mid \mathbf{x})} \\ =& \underbrace{\frac{1}{2}\left(\boldsymbol{x}^T \Sigma_2^{-1} \boldsymbol{x} - \boldsymbol{x}^T \Sigma_1^{-1} \boldsymbol{x}\right)}_{\text{quadratic terms}} + \underbrace{\boldsymbol{\mu}_1^T \Sigma_1^{-1} \boldsymbol{x} - \boldsymbol{\mu}_2^T \Sigma_2^{-1} \boldsymbol{x}}_{\text{linear terms}} \\ & \underbrace{-\frac{1}{2}\boldsymbol{\mu}_1^T \Sigma_1^{-1} \boldsymbol{\mu}_1 + \frac{1}{2}\boldsymbol{\mu}_2^T \Sigma_2^{-1} \boldsymbol{\mu}_2 + \ln \frac{P(\omega_1)}{P(\omega_2)} + \frac{1}{2} \ln \frac{|\Sigma_2|}{|\Sigma_1|}}_{\text{constant terms}} = 0. \end{aligned}$$

Quiz

$$
\begin{aligned}
g(\boldsymbol{x}) =& g_1(\boldsymbol{x}) - g_2(\boldsymbol{x}) \\
=& \underbrace{\frac{1}{2}\left(\boldsymbol{x}^T\Sigma_2^{-1}\boldsymbol{x} - \boldsymbol{x}^T\Sigma_1^{-1}\boldsymbol{x}\right)}_{\text{quadratic terms}} + \underbrace{\boldsymbol{\mu}_1^T\Sigma_1^{-1}\boldsymbol{x} - \boldsymbol{\mu}_2^T\Sigma_2^{-1}\boldsymbol{x}}_{\text{linear terms}} \\
& \underbrace{-\frac{1}{2}\boldsymbol{\mu}_1^T\Sigma_1^{-1}\boldsymbol{\mu}_1 + \frac{1}{2}\boldsymbol{\mu}_2^T\Sigma_2^{-1}\boldsymbol{\mu}_2 + \ln\frac{P(\omega_1)}{P(\omega_2)} + \frac{1}{2}\ln\frac{|\Sigma_2|}{|\Sigma_1|}}_{\text{constant terms}} = 0.
\end{aligned}
$$

- When does the discriminant function become linear?

Quiz

- When does the discriminant function become linear?

$$\Sigma_1 = \Sigma_2$$

- The corresponding hypersurface is a hyperplane:

$$g(\boldsymbol{x}) = \boldsymbol{\theta}^T (\boldsymbol{x} - \boldsymbol{x}_0) = 0, \text{ where}$$

$$\boldsymbol{\theta} := \Sigma^{-1} (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)$$

$$\boldsymbol{x}_0 := \frac{1}{2} (\boldsymbol{\mu}_1 + \boldsymbol{\mu}_2) - \ln \frac{P(\omega_1)}{P(\omega_2)} \frac{\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2}{\|\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2\|^2_{\Sigma^{-1}}}$$

where Σ is the common to the two classes covariance matrix and

$$\|\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2\|_{\Sigma^{-1}} := \sqrt{(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)^T \Sigma^{-1} (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)}$$

is the so-called Σ^{-1}-norm of the vector $(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)$.

(Proof: start from $g(\boldsymbol{x}) = 0$ and substitute $\boldsymbol{\theta}$ and $\boldsymbol{x}_0$)

Minimum distance classifiers

- Minimum Euclidean Distance Classifier: Under the assumptions of
  - Gaussian distributed data in each one of the classes,
  - equiprobable classes,
  - common covariance matrix in all classes of the special form $\Sigma = \sigma^2 I$

  the Bayesian classification rule is equivalent with

$$\text{Assign } \mathbf{x} \text{ to class } \omega_1 : i = \arg\min_j \left(\boldsymbol{x} - \boldsymbol{\mu}_j\right)^T \left(\boldsymbol{x} - \boldsymbol{\mu}_j\right)$$

- In other words, the Euclidean distance of x is computed from the mean values of all classes and it is assigned to the class for which this distance becomes smaller.

Minimum distance classifiers

- Minimum Mahalanobis Distance Classifier: Under the previously adopted assumptions, but with the covariance matrix being of the more general form, $\Sigma \neq \sigma^2 I$,

$$\text{Assign } \mathbf{x} \text{ to class } \omega_1 : i = \arg\min_j \left(\boldsymbol{x} - \boldsymbol{\mu}_j\right)^T \Sigma^{-1} \left(\boldsymbol{x} - \boldsymbol{\mu}_j\right)$$

- Can such model work well for high-dimensional $\mathbf{x}$?

The naive Bayes classifier

- Naive Bayes Classifier: The basic assumption is that the components (features) in the feature vector are statistically independent; hence, the joint pdf can be written as a product of l marginals, i.e.,

$$p(\boldsymbol{x} \mid \omega_i) = \prod_{k=1}^{l} p(x_k \mid \omega_i), i = 1, 2, \ldots, M$$

  - E.g., adopting the Gaussian assumption, each one of the marginals is described by two parameters, i.e., the mean and the variance; the covariance matrix is **diagonal**.
  - This leads to a total of $2l$, per class, unknown parameters to be estimated.
  - This is a substantial saving compared to the $\mathcal{O}\left(l^2/2\right)$ number of parameters, needed for an $l \times l$ covariance matrix.

Logistic regression

- In Bayesian classification, the posteriors are estimated via the respective conditional pdfs, which is not, in general, an easy task.
- The goal now becomes to model the posterior probabilities directly, via the so-called logistic regression method.

Logistic regression

- We start with the two-class case. The kick-off point is to model the ratio of the posteriors as:

$$\ln \frac{P(\omega_1 \mid \mathbf{x})}{P(\omega_2 \mid \mathbf{x})} = \boldsymbol{\theta}^T \mathbf{x},$$

  where the constant term, θ_0, has been absorbed in $\boldsymbol{\theta}$.

- Taking into account that $P(\omega_1 \mid \boldsymbol{x}) + P(\omega_2 \mid \boldsymbol{x}) = 1$ and defining

$$t := \boldsymbol{\theta}^T \boldsymbol{x},$$

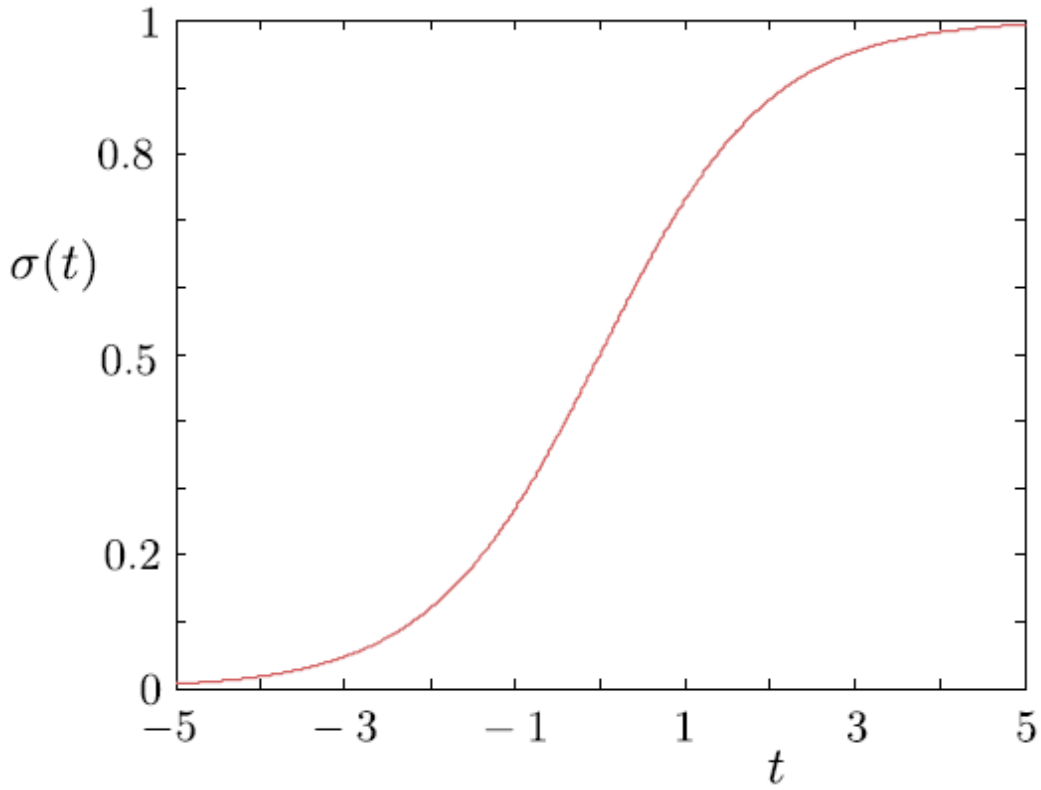
  it is readily seen that the previous model is equivalent to

$$P(\omega_1 \mid \mathbf{x}) = \sigma(t) := \frac{1}{1 + \exp(-t)},$$

$$P(\omega_2 \mid \mathbf{x}) = 1 - \sigma(t) = \frac{\exp(-t)}{1 + \exp(-t)}.$$

Logistic regression

- The function $\sigma(t)$ is known as the logistic sigmoid or sigmoid link function.



Training the Logistic regression model

- The parameter vector, $\boldsymbol{\theta}$, is estimated via the Maximum Likelihood method applied on a set of training samples, $(y_n, \boldsymbol{x}_n)$, $n = 1, \ldots, N, y_n \in \{0, 1\}$. The likelihood function can be written as,

$$P(y_1, \ldots, y_N; \boldsymbol{\theta}) = \prod_{n=1}^{N} \left(\sigma\left(\boldsymbol{\theta}^T \boldsymbol{x}_n\right)\right)^{y_n} \left(1 - \sigma\left(\boldsymbol{\theta}^T \boldsymbol{x}_n\right)\right)^{1-y_n}.$$

- Minimize the negative log-likelihood given by,

$$L(\boldsymbol{\theta}) = -\sum_{n=1}^{N} \left(y_n \ln s_n + (1 - y_n) \ln(1 - s_n)\right), s_n := \sigma\left(\boldsymbol{\theta}^T \boldsymbol{x}_n\right)$$

  The above log-likelihood is also known as the cross-entropy cost function.
- The solution is achieved iteratively, starting from a $\theta^{(0)}$, e.g., by employing the gradient descent method,

$$\boldsymbol{\theta}^{(i)} = \boldsymbol{\theta}^{(i-1)} - \mu_i \nabla L\left(\boldsymbol{\theta}^{(i-1)}\right).$$

Selecting a classifier

- We have discussed a number of classifiers and more methods will be discussed later on, such as support vector machines, Bayesian methods and neural/deep networks. The obvious question that may be naturally raised is:

WHICH CLASSIFIER THEN?

Unfortunately, there is no definite answer to it.

No-Free lunch theorem

- No Free Lunch Theorem: There is **no context-independent or usage-independent reasons to support one learning technique from another**.
- Equivalently, according to this theorem, averaged over all possible data generating distributions, **every** classification scheme results in the **same** error rate on data outside the training set.
- In other words, there is no learning algorithm that is universally optimal. However, note that these results hold only when one averages over **all** possible data generating distributions.

No-Free lunch theorem

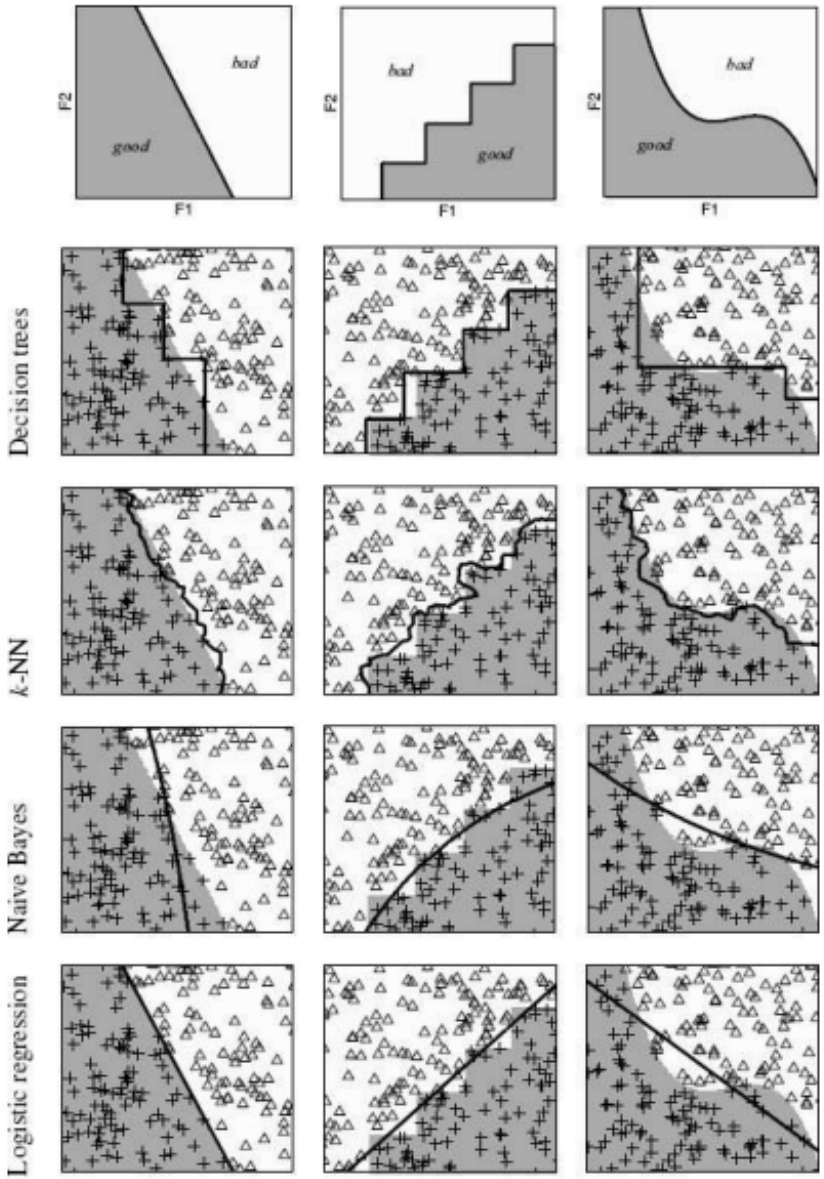


Figure 11.2

An illustration of the decision boundaries learned by different machine learning algorithms for three artificial datasets.

Lecture 2: Summary

- How to minimize loss L?
  - Online Learning: the Stochastic Gradient Descent Family of Algorithms (Chapter 5)
- What are common choices for $f(x, \theta), p(y \mid x; \theta), p(\theta \mid y)$ for classification?
  - Classification (Chapter 7):
    - Bayesian classifier; intersection of hypersurfaces
    - Naïve Bayes classifier
    - Linear/Quatratic Discriminant Algorithm (LDA, QDA)

Content

Lecture 2

- Course logistics
- Review of Lecture 1
- Online learning (Chapter 5)
- Classic classification algorithms (Chapter 7)
- **Preparations for next week**

Preparations for next week

- Participate in the **pre-lecture survey**; this will help us to better plan the course and make adjustments if needed.
- Bring your **laptop** to class for the **Mon and Fri** Sessions.
- **Read** in the book
  - Chapters 1, 2, and 3 (skip sections 2.6, 3.6, and 3.15)
  - Sections 5.1-5.5 (skip 5.3.1, 5.5.1 and 5.5.2)
  - Sections 7.1-7.7.2.
- Attempt the **theoretical exercises** 2.2, 2.8, 3.1, 3.24, 3.27, 5.2, 5.5, 7.1, and 7.3 in the book. We will discuss those exercises in the **Mon session**.
- Before the Mon session, take a look at **Computer Exercise 1 (Bayesian Regression)**, in particular, the *least squares approach*. As the course progresses, we will explore the latter parts of this exercise.
- **Familiarize** yourself with
  - Jupyter notebooks
  - Google colab
  - Python
  - Scikit-learn.