Machine Learning I

Lecture 1: Introduction

7 September 2019

Outline: What, Why, How and When?

Practical Matters: organisation, schedule, grading, ...

What is Machine Learning?

Why learn from data?

How: forms of Machine Learning
- Classification and Regression
- Clustering and Dimensionality reduction

When is a machine learning
- Overfitting
- Evaluation

Outline: What, Why, How and When?

Practical Matters: organisation, schedule, grading, ...

What is Machine Learning?

Why learn from data?

How: forms of Machine Learning
- Classification and Regression
- Clustering and Dimensionality reduction

When is a machine learning
- Overfitting
- Evaluation

Practical matters: Organisation

- ML I (1st quarter) and ML II (2nd quarter) courses
- The course consists of lectures, labs and homeworks

- The final grade is weighed as: 50% exam, 50% homeworks.
- Advanced course is graded based on a project

Practical matters: Organisation

- ML I (1st quarter) and ML II (2nd quarter) courses
- The course consists of lectures, labs and homeworks
  - Lectures are on Tuesday afternoon
  - The lectures are in person, recordings will be put on canvas after the lecture
  - Don't hesitate to interrupt to ask questions
- The final grade is weighed as: 50% exam, 50% homeworks.
- Advanced course is graded based on a project

Practical matters: Organisation

- ML I (1st quarter) and ML II (2nd quarter) courses
- The course consists of lectures, labs and homeworks
  - Lab exercises are in **groups of 3**
    - Register your group in Canvas by Thursday
  - Python 3.x in jupyter notebooks
  - Lab sessions are on **Thursday** morning or afternoon
    - Morning session: in person, afternoon session: online
    - Sessions are not compulsory
    - Organise your group so everyone can attend at the same time
  - Labs are not graded
- The final grade is weighed as: 50% exam, 50% homeworks.
- Advanced course is graded based on a project

Practical matters: Organisation

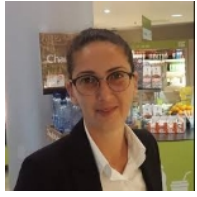
- ML I (1st quarter) and ML II (2nd quarter) courses
- The course consists of lectures, labs and homeworks
  - Homeworks are done in the same **groups of 3** as the labs
    - Register your group in Canvas by Thursday
  - Homeworks are graded (starting in week 2)
    - This week's homework is not graded
    - All other homeworks all carry the same weight
- The final grade is weighed as: 50% exam, 50% homeworks.
- Advanced course is graded based on a project

Teachers/Instructors

Gwenn Englebienne

g.englebienne
@utwente.nl

Elena Mocanu



e.mocanu
@utwente.nl

Docebal Mocanu

d.mocanu
@utwente.nl

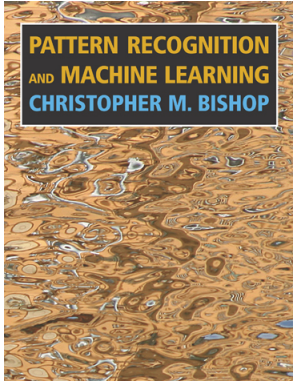
Jur van Geel

j.g.j.a.vangeel
@utwente.nl

Practical matters

Book:

- **Pattern Recognition and Machine Learning**, Christopher M. Bishop, Springer (2006)



- Everything else will be available from Canvas

Schedule: Basic course

Wk.	Subject	Exercise/Lab
1	Introduction	Familiarising Python & libraries
2	Linear Discriminants	Linear discriminants, overfitting
3	Neural Networks	Exercises Neural networks
4	Deep networks	Regression and classification with (deep) NN
5	Decision Trees	Decision Trees
6	Kernels and Support Vector Machines	SVM
7	Probabilistic models	Implement the E.M. algorithm
8	Dimensionality reduction	PCA, ...
9	Rehearsal, example exam	
10	Written exam	

Outline: What, Why, How and When?

Practical Matters: organisation, schedule, grading, ...

What is Machine Learning?

Why learn from data?

How: forms of Machine Learning
- Classification and Regression
- Clustering and Dimensionality reduction

When is a machine learning
- Overfitting
- Evaluation

What's it about?

Machine Learning Make machines learn from examples

Pattern Recognition Find patterns in data

Course objectives:

- Introduction to state-of-the art methods for machine learning and data modeling
- When relevant, to refer back to human learning
- Today's lecture: Introduction to the field, overview of the course.
- This week's homework: familiarization with basic mathematics
- This week's lab: familiarization with the Python libraries

What's it about?

Machine Learning Make machines learn from examples

Pattern Recognition Find patterns in data

Course objectives:

- Introduction to state-of-the art methods for machine learning and data modeling
- When relevant, to refer back to human learning
- Today's lecture: Introduction to the field, overview of the course.
- This week's homework: familiarization with basic mathematics
- This week's lab: familiarization with the Python libraries

What is learning?

"extract information from observations that is valid and relevant for future observations"

Informally...

- Training data **cannot** tell us what will be true in future data!
  - Distinguishing between the trainingset-specific and the general requires **bias**
  - Occam's Razor
- Generalization: figuring out what will be valid for future data
- Overfitting: learning aspects from training data that are not true in future data

The world

The world

The Machine

The world

The Machine

Machines that learn

The world

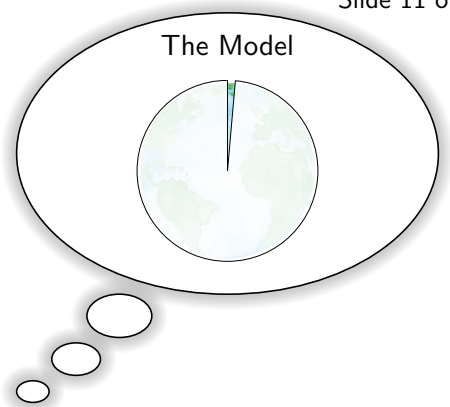
The Machine

The Model

Machines that learn

The world

The Machine



- Reading house numbers
- Interpreting pictures
- Playing Go at master level
- Kaggle competitions
- Deep learning

VOCABULARY

Niiiiiiiice: How Tweets Reveal Your Age

As they say, you are what you tweet

By Katy Steinmetz @katysteinmetz | July 17, 2013

Writing about her new study on Twitter and age, researcher Dong Nguyen starts off with a little quiz. How old do you think the people are who sent these respective tweets?

AS LONG AS YOU LOVE ME❤

Interesting article about usability design on mobile search [LINK]

That might seem like a test for advanced Twitterati, but in a paper published this month—titled "'How Old Do You Think I Am?': A Study of Language and Age in Twitter"—four Dutch researchers reveal stylistic tics associated with younger and older tweeters. Nguyen's team also discovered that, based on such tweet-tics, an automated program can better predict your age than a fellow human can.

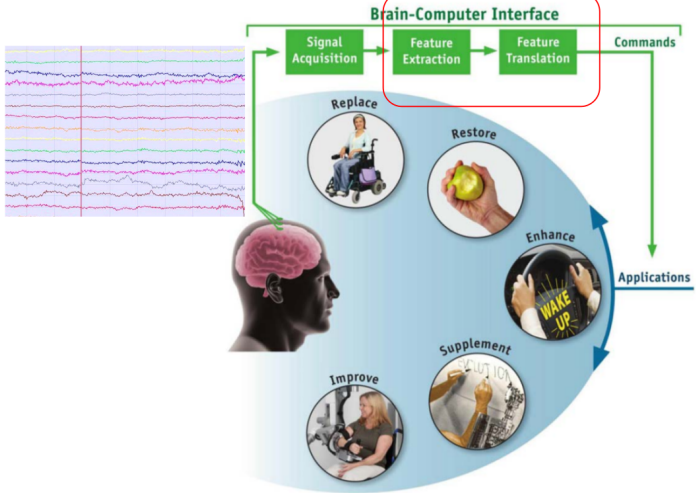
Michael DeLeon / Getty Images

RELATED

The Edward Snowden Name Game: Whistle-Blower, Traitor, Leaker

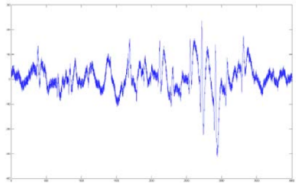
Brain-Computer interfaces

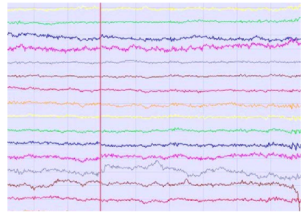
ML part



Goal of BCI

- Challenge: translate brain signals to intentions or mental state

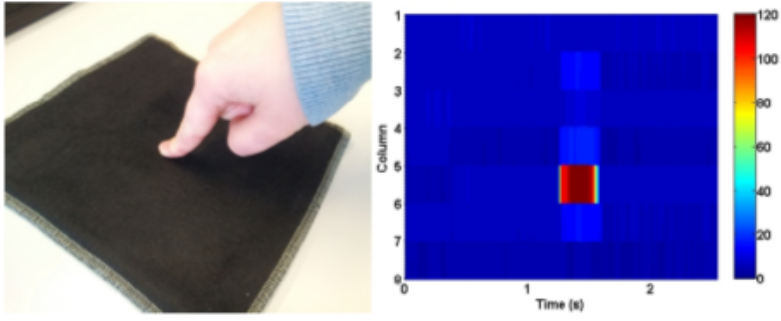




- Issues: Ground truth, signal-to-noise ratio

Touch interaction: integrating touch in HCI

Can we detect, recognise, interpret touch gestures



Outline: What, Why, How and When?

How (and why) do we build the model?

- Traditionally: humans code it up
- Machine learning
- Issues

How (and why) do we build the model?

- Traditionally: humans code it up
  - Hard-coded rules or equations
  - Include expert knowledge
  - Difficult, time-consuming, limited success
- Machine learning
- Issues

How (and why) do we build the model?

- Traditionally: humans code it up
- Machine learning
  - observe and learn from observations
  - Requires (lots) of examples: data
  - Solves problems we don't know how to solve!
  - There are many problems **we can solve, but don't know how**

    Speech recognition, object recognition, bipedal locomotion, driving a vehicle, ...

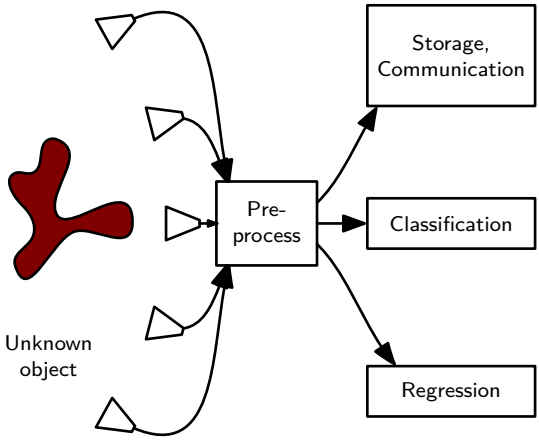
  - With machine learning: provide examples, let the machine figure it out
- Issues

How (and why) do we build the model?

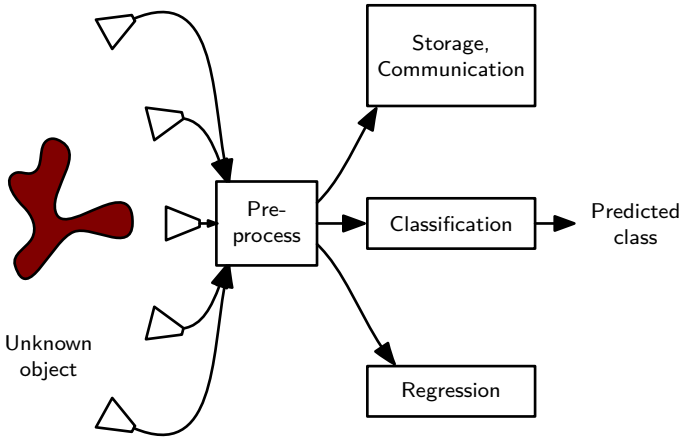
- Traditionally: humans code it up
- Machine learning
- Issues
  - **How** is the world perceived?
    - Sensors (camera, microphones, chemical analysis, DNA sequencing, . . .)
    - Results in "features" or "attributes"
  - **What** is the purpose of the model?
    - What is the task? Recognition, identification, prediction, . . .
    - How does the machine know what is "right"?
  - **Which task-related** information does the machine get?
    - Does the machine get feedback on every example? (supervised learning)
    - Does the machine get feedback on some examples? (semi-supervised learning)
    - Does the machine get any task-related information? (unsupervised learning)
    - Can the machine choose the examples for which it gets feedback (active learning)
    - Does the machine actively explore the world (reinforcement learning)

Outline: What, Why, How and When?

Basic Framework

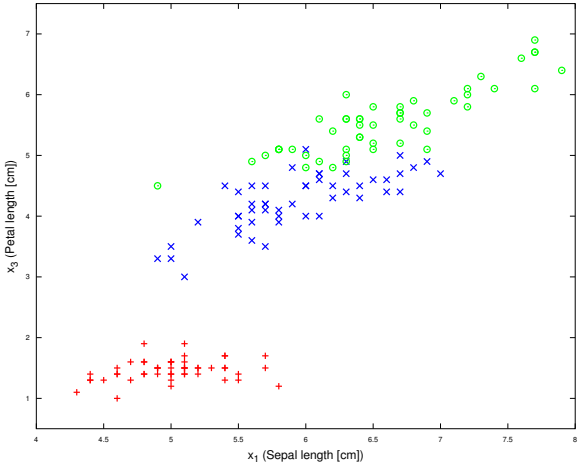


Basic Framework

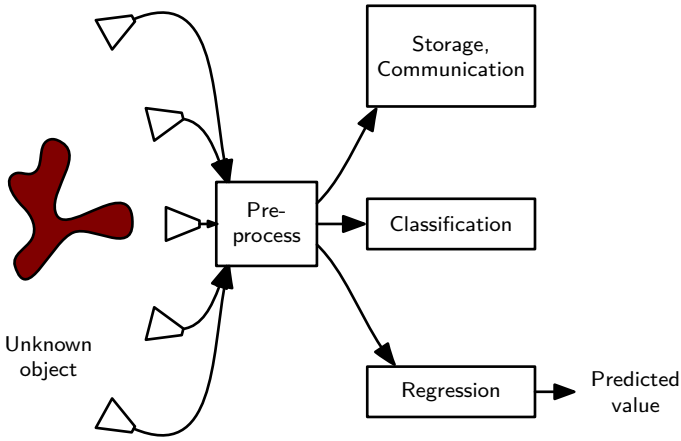


An example of classification

Example: Iris classification

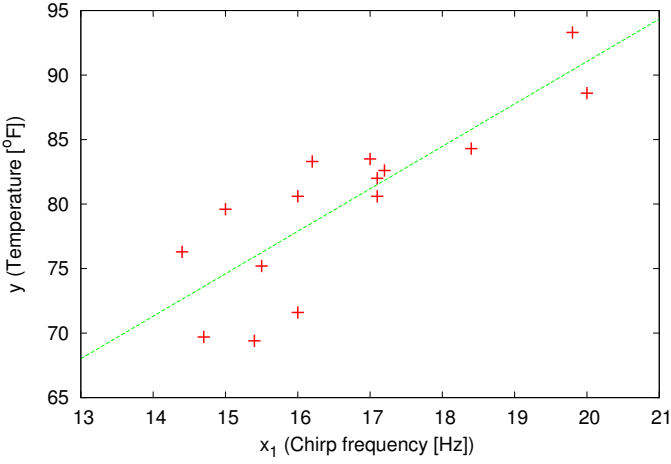


Basic Framework



Regression example

Example: Evaluating temperature from cricket activity



Classification vs. Regression

- **Classification**: Predict a discrete label from features

  Example

  - Medicine: classify X-rays as “cancer” or “healthy”
  - SPAM detection: classify emails as spam or not
  - Face recognition, speech recognition, . . .
  - Fall risk estimation

- **Regression**: Predict a continuous value

  Example

  - Weather forecasting (wind speed, mm rainfall, . . .)
  - In financial markets: predict tomorrow’s stock price from past evolution and external factors
  - A robot learning its location in an environment

Classification vs. Regression

- **Classification**: Predict a discrete label from features

  Example

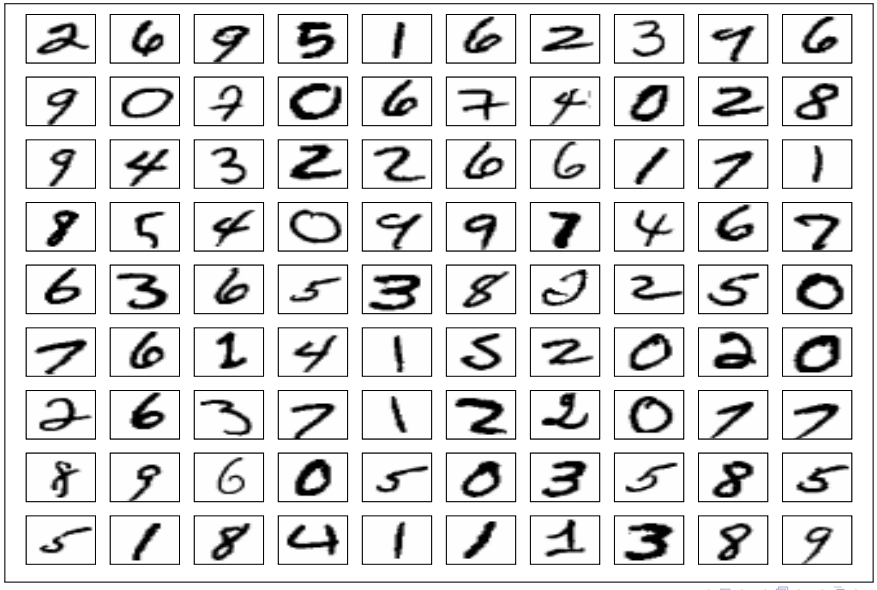
  - Medicine: classify X-rays as “cancer” or “healthy”
  - SPAM detection: classify emails as spam or not
  - Face recognition, speech recognition, . . .
  - Fall risk estimation

- **Regression**: Predict a continuous value

  Example

  - Weather forecasting (wind speed, mm rainfall, . . .)
  - In financial markets: predict tomorrow’s stock price from past evolution and external factors
  - A robot learning its location in an environment

Another classification example



In functional form

$f($  $) = f($  $) = f($  $) = \cdots = \mathcal{C}_0$

$f($  $) = f($  $) = f($  $) = \cdots = \mathcal{C}_1$

$f($  $) = f($  $) = f($  $) = \cdots = \mathcal{C}_2$

$f($  $) = f($  $) = f($  $) = \cdots = \mathcal{C}_3$

$f($  $) = f($  $) = f($  $) = \cdots = \mathcal{C}_4$

$f($  $) = f($  $) = f($  $) = \cdots = \mathcal{C}_5$

$f($  $) = f($  $) = f($  $) = \cdots = \mathcal{C}_6$

$f($  $) = f($  $) = f($  $) = \cdots = \mathcal{C}_7$

$f($  $) = f($  $) = f($  $) = \cdots = \mathcal{C}_8$

$f($  $) = f($  $) = f($  $) = \cdots = \mathcal{C}_9$

In functional form

$f($  $, \theta) = f($  $, \theta) = f($  $, \theta) = \cdots = \mathcal{C}_0$

$f($  $, \theta) = f($  $, \theta) = f($  $, \theta) = \cdots = \mathcal{C}_1$

$f($  $, \theta) = f($  $, \theta) = f($  $, \theta) = \cdots = \mathcal{C}_2$

$f($  $, \theta) = f($  $, \theta) = f($  $, \theta) = \cdots = \mathcal{C}_3$

$f($  $, \theta) = f($  $, \theta) = f($  $, \theta) = \cdots = \mathcal{C}_4$

$f($  $, \theta) = f($  $, \theta) = f($  $, \theta) = \cdots = \mathcal{C}_5$

$f($  $, \theta) = f($  $, \theta) = f($  $, \theta) = \cdots = \mathcal{C}_6$

$f($  $, \theta) = f($  $, \theta) = f($  $, \theta) = \cdots = \mathcal{C}_7$

$f($  $, \theta) = f($  $, \theta) = f($  $, \theta) = \cdots = \mathcal{C}_8$

$f($  $, \theta) = f($  $, \theta) = f($  $, \theta) = \cdots = \mathcal{C}_9$

In functional form

vector: bold lowercase

θ: traditional symbol for parameters (but sometimes **w** for "weights")

$$f(\mathbf{x}_1, \boldsymbol{\theta}) = f(\mathbf{x}_2, \boldsymbol{\theta}) = f(\mathbf{x}_3, \boldsymbol{\theta}) = \cdots = \mathcal{C}_0$$

$$f(\mathbf{x}_4, \boldsymbol{\theta}) = f(\mathbf{x}_5, \boldsymbol{\theta}) = f(\mathbf{x}_6, \boldsymbol{\theta}) = \cdots = \mathcal{C}_1$$

$$f(\mathbf{x}_7, \boldsymbol{\theta}) = f(\mathbf{x}_8, \boldsymbol{\theta}) = f(\mathbf{x}_9, \boldsymbol{\theta}) = \cdots = \mathcal{C}_2$$

$$f(\mathbf{x}_{10}, \boldsymbol{\theta}) = f(\mathbf{x}_{11}, \boldsymbol{\theta}) = f(\mathbf{x}_{12}, \boldsymbol{\theta}) = \cdots = \mathcal{C}_3$$

$$f(\mathbf{x}_{13}, \boldsymbol{\theta}) = f(\mathbf{x}_{14}, \boldsymbol{\theta}) = f(\mathbf{x}_{15}, \boldsymbol{\theta}) = \cdots = \mathcal{C}_4$$

$$f(\mathbf{x}_{16}, \boldsymbol{\theta}) = f(\mathbf{x}_{17}, \boldsymbol{\theta}) = f(\mathbf{x}_{18}, \boldsymbol{\theta}) = \cdots = \mathcal{C}_5$$

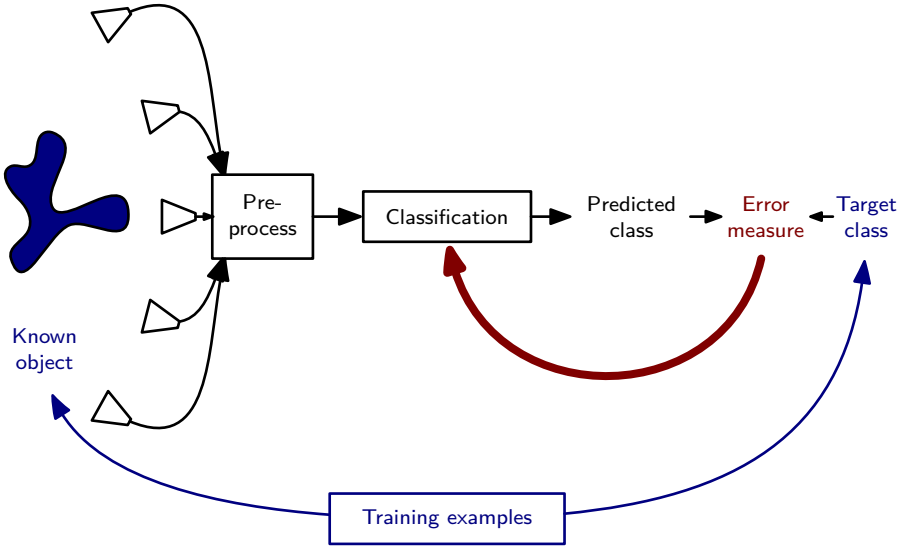
$$f(\mathbf{x}_{19}, \boldsymbol{\theta}) = f(\mathbf{x}_{20}, \boldsymbol{\theta}) = f(\mathbf{x}_{21}, \boldsymbol{\theta}) = \cdots = \mathcal{C}_6$$

$$f(\mathbf{x}_{22}, \boldsymbol{\theta}) = f(\mathbf{x}_{23}, \boldsymbol{\theta}) = f(\mathbf{x}_{24}, \boldsymbol{\theta}) = \cdots = \mathcal{C}_7$$

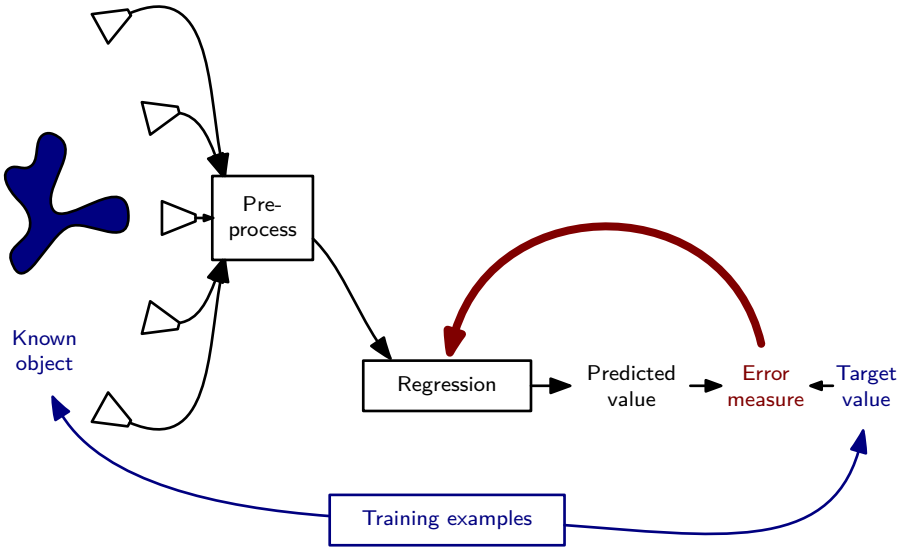
$$f(\mathbf{x}_{25}, \boldsymbol{\theta}) = f(\mathbf{x}_{26}, \boldsymbol{\theta}) = f(\mathbf{x}_{27}, \boldsymbol{\theta}) = \cdots = \mathcal{C}_8$$

$$f(\mathbf{x}_{28}, \boldsymbol{\theta}) = f(\mathbf{x}_{29}, \boldsymbol{\theta}) = f(\mathbf{x}_{30}, \boldsymbol{\theta}) = \cdots = \mathcal{C}_9$$

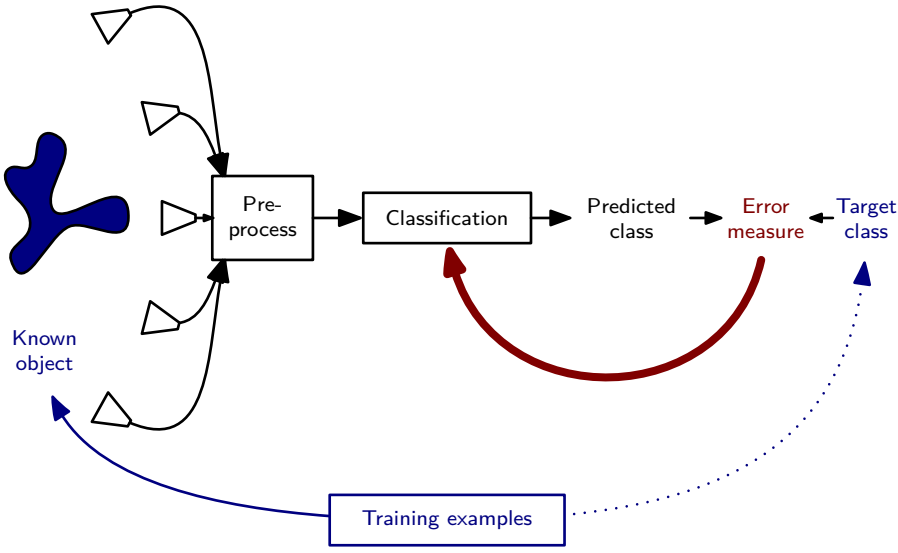
Supervised Training — Classification



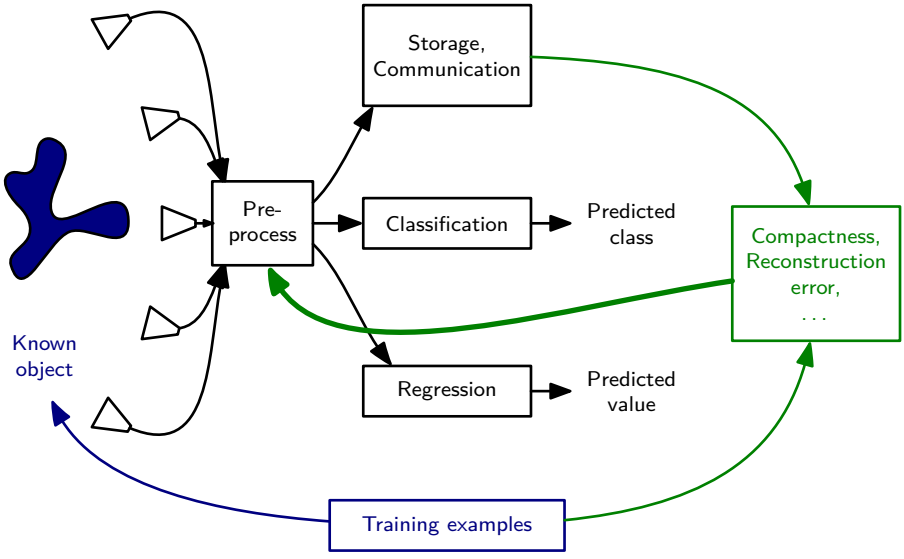
Supervised Training — Regression



Semi-supervised Training



Unsupervised Training



Basic issues of classification:

1. Given:
   - Classes, $\mathcal{C} \in \{\mathcal{C}_1, \ldots, \mathcal{C}_k\}$
   - Data elements / Feature values: $\mathbf{x} = (x_1, \ldots, x_d)^\top$
2. What are the best features / Should we use all features?
3. How do we *learn* to classify unseen data from a set of training examples $\{(\mathbf{x}^{(i)}, \mathcal{C}^{(i)}), i = 1, \ldots, n\}$
   - What *kind* of function can provide the right answer?
   - How do we *train* that function?
   - *How much training data* do we need to learn a good function?

For regression, we predict a continuous value rather than a discrete label

Unsupervised: Clustering

Goal: divide the data in groups, such that:

- Items in each group are similar
- Dissimilar items are in different groups

Example

Customer/product clustering

- Identify groups of customers with similar buying patterns for targeted marketing campaigns: send mailings only to likely buyers
- Identify groups of products that are often bought together, offer packages of products for reduced price
- Recommender systems: Jointly cluster users of movies, books, CD's,... (e.g. Amazon, Netflix, ...)

Unsupervised: Clustering

Goal: divide the data in groups, such that:

- Items in each group are similar
- Dissimilar items are in different groups

Example

Customer/product clustering

- Identify groups of customers with similar buying patterns for targeted marketing campaigns: send mailings only to likely buyers
- Identify groups of products that are often bought together, offer packages of products for reduced price
- Recommender systems: Jointly cluster users of movies, books, CD's,... (e.g. Amazon, Netflix, ...)

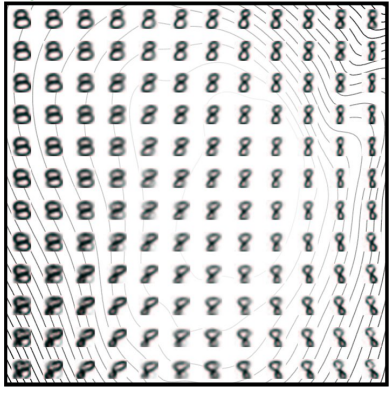
Unsupervised: Dimensionality reduction

- MNIST example: 16×16 pixels, 256 intensities
  - $256^{256} \approx 10^{616}$ possible images
  - If you tried to list all such images, and generated them at the rate of one per second, you'd need (a lot) more time than the lifespan of the universe ($\approx 10^{157} s$) to list them all.
  - Notice that doing it faster does not help much: a supercomputer generating 10 billion billion billion images per second would still need 10^{589} seconds, or 10^{432} universes ...
- However most of these possible images are not meaningful
  - In this 256D space, only limited locations are used
- It is therefore possible to reduce the size of the description, without losing information

Dimensionality reduction

- Used for data compressing and reconstruction
- Used as a pre-processing step, to reduce classifier complexity

Example



Learning with delayed feedback

Sometimes the learning is part of a process

Example

- Recommender systems, search engines: are the recommendations valuable?
- Game systems: what moves lead to a win?
- Robotics: What combinations of actions improve performance?

Reinforcement learning

- Such problems can be formalised as a sequence of steps (system states) that lead to a reward
- Reinforcement learning theory allows us to use that reward to learn good state transitions
- Complex states cannot be represented exactly $\Rightarrow$ dimensionality reduction, ...

Outline: What, Why, How and When?

Training

Supervised learning:

- We *learn* from examples
  - Training data: inputs and outputs
  - Representation of the input
  - Representation of the output
- Find a function that maps inputs to outputs
  - That also applies to data we've never seen: **generalisation**

  Assumption

  Both training data and future data are sampled independently from the same distribution (Independent and Identically Distributed — i.i.d.)

  - We cannot consider all functions: **inductive bias**

Error measures: regression

Typically: **sum-of-squares** error function:

$$E_{SSE}(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}\left(y(x_n, \mathbf{w}) - t_n\right)^2 \tag{1}$$

- Minimising the SSE is equivalent with maximising the log-likelihood under the assumption of zero-mean Gaussian noise.

Sometimes more convenient: **root-mean-square** error:

$$E_{RMS}(\mathbf{w}) = \sqrt{2E_{SSE}(\mathbf{w})/N} \tag{2}$$

- Square root ensures that the error has same scale as target
- Division by N allows comparison over data sets of different size

Error measures: regression

For convenience when differentiating

Typically: **sum-of-squares** error function:

Target value

$$E_{SSE}(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}\left(y(x_n, \mathbf{w}) - t_n\right)^2 \tag{1}$$

Model output

- Minimising the SSE is equivalent with maximising the likelihood under the assumption of zero-mean Gaussian noise.

Sometimes more convenient: **root-mean-square** error:

$$E_{RMS}(\mathbf{w}) = \sqrt{2E_{SSE}(\mathbf{w})/N} \tag{2}$$

- Square root ensures that the error has same scale as target
- Division by N allows comparison over data sets of different size

Number of datapoints

Overfitting and Generalisation

- Generalisation: learn, from known examples, about unseen examples
- Overfitting: learn properties from the given examples which do not apply to unseen examples
- Evaluate on separate set

Example: polynomial regression

- Process: $y = \sin(2\pi x)$
- Observations: corrupted by Gaussian noise def:

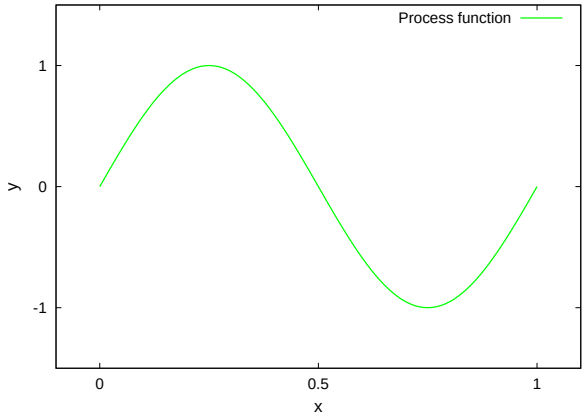
$$y = \sin(2\pi x) + \xi$$

with

$$\xi \sim \mathcal{N}(0, 0.3)$$

- Attempt to recover a description of the process, using a polynomial function

$$y = w_0 + w_1 x + w_2 x^2 + \cdots$$



Example: polynomial regression

- Process: $y = \sin(2\pi x)$
- Observations: corrupted by Gaussian noise def:

$$y = \sin(2\pi x) + \xi$$

with

$$\xi \sim \mathcal{N}(0, 0.3)$$

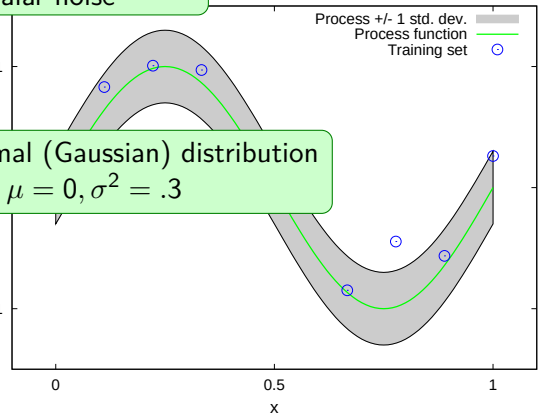
- Attempt to recover a description of y using a polynomial function

$$y = w_0 + w_1 x + w_2 x^2 + \cdots$$

Scalar noise

Normal (Gaussian) distribution with $\mu = 0, \sigma^2 = .3$

"Distributed as"



Example: polynomial regression

- Process: $y = \sin(2\pi x)$
- Observations: corrupted by Gaussian noise def:

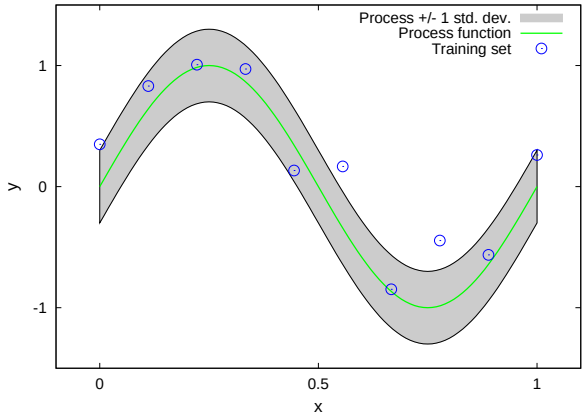
$$y = \sin(2\pi x) + \xi$$

with

$$\xi \sim \mathcal{N}(0, 0.3)$$

- Attempt to recover a description of the process, using a polynomial function

$$y = w_0 + w_1 x + w_2 x^2 + \cdots$$



Example: polynomial regression

- Process: $y = \sin(2\pi x)$
- Observations: corrupted by Gaussian noise def:

$$y = \sin(2\pi x) + \xi$$

with

$$\xi \sim \mathcal{N}(0, 0.3)$$

- Attempt to recover a description of the process, using a polynomial function

$$y = w_0 + w_1 x + w_2 x^2 + \cdots$$

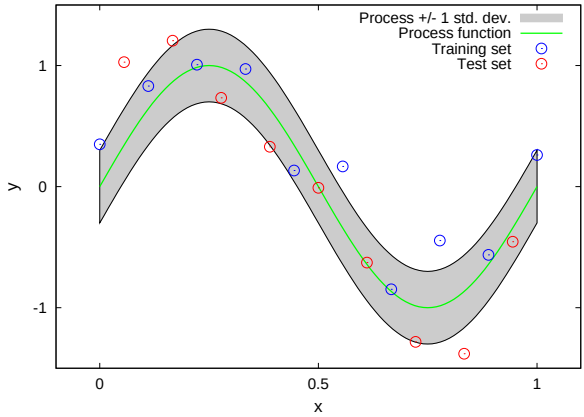
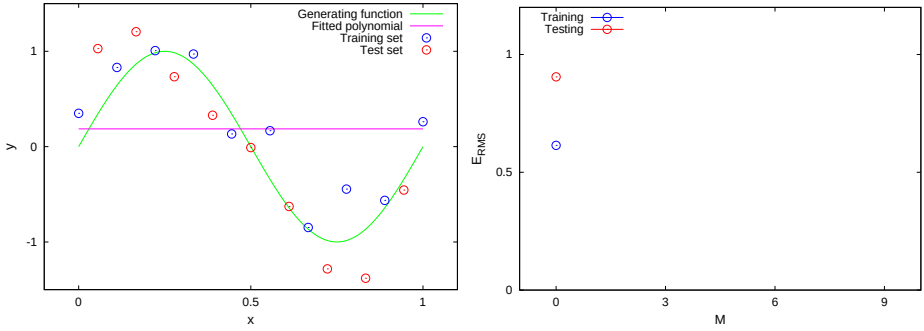


Illustration of overfitting

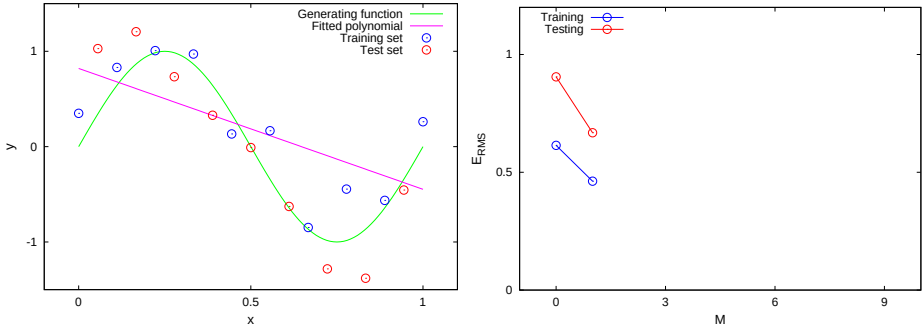
Example



$M = 0$

Illustration of overfitting

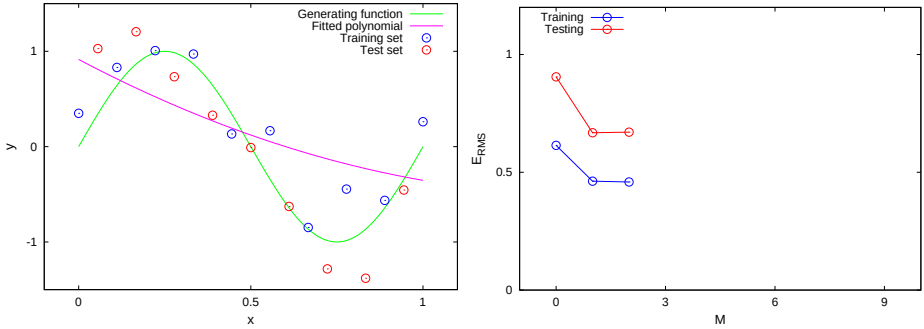
Example



$M = 1$

Illustration of overfitting

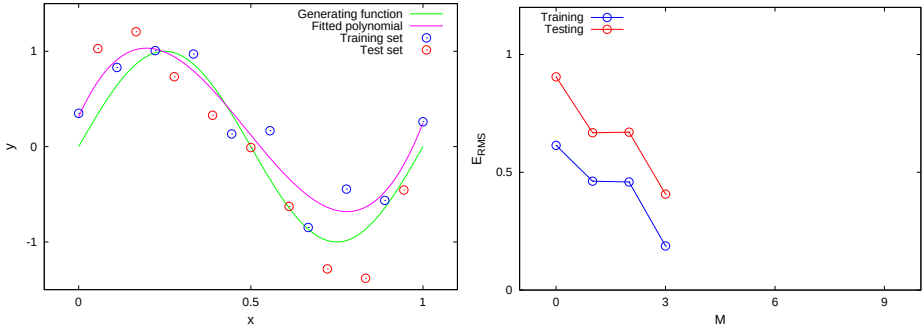
Example



$M = 2$

Illustration of overfitting

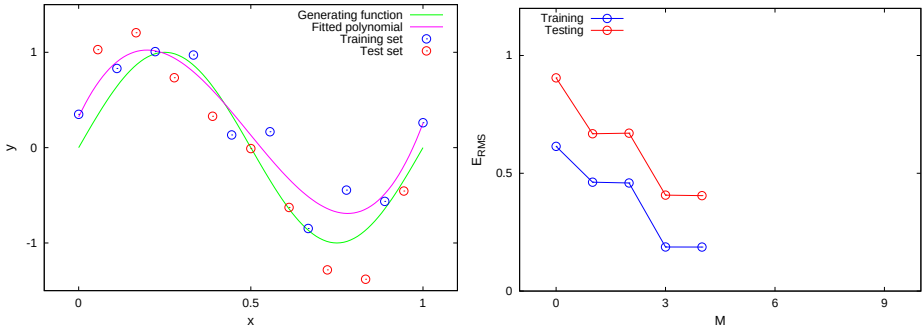
Example



$M = 3$

Illustration of overfitting

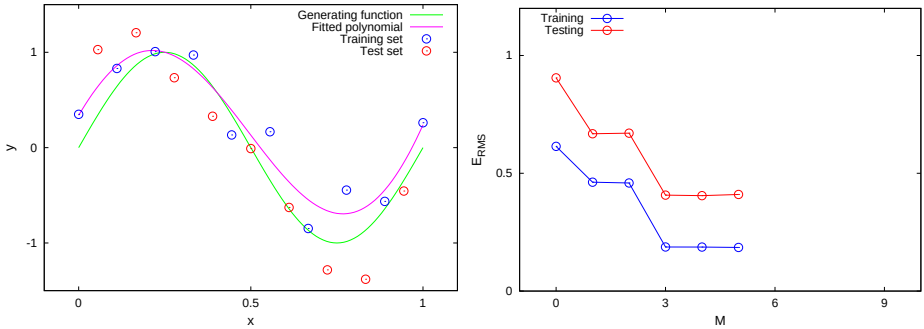
Example



$M = 4$

Illustration of overfitting

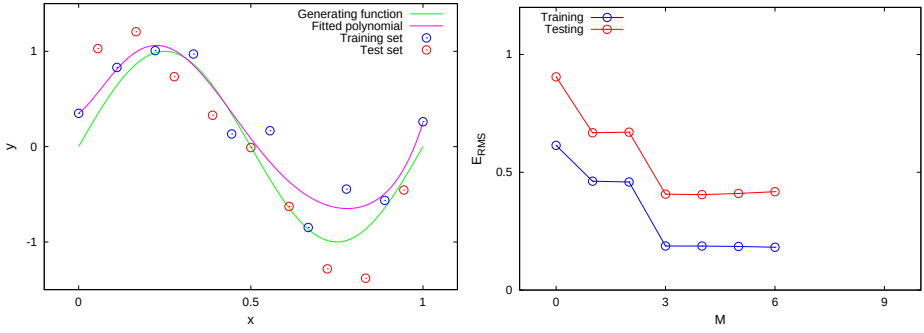
Example



$M = 5$

Illustration of overfitting

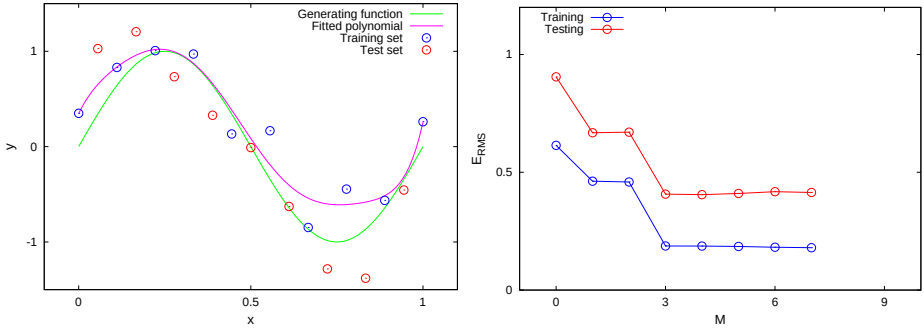
Example



$M = 6$

Illustration of overfitting

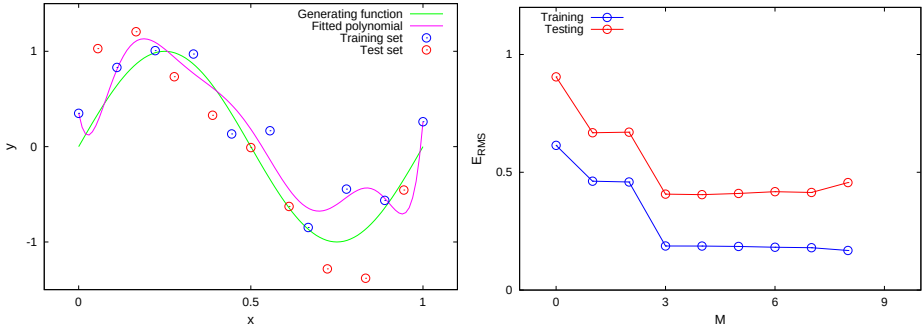
Example



$M = 7$

Illustration of overfitting

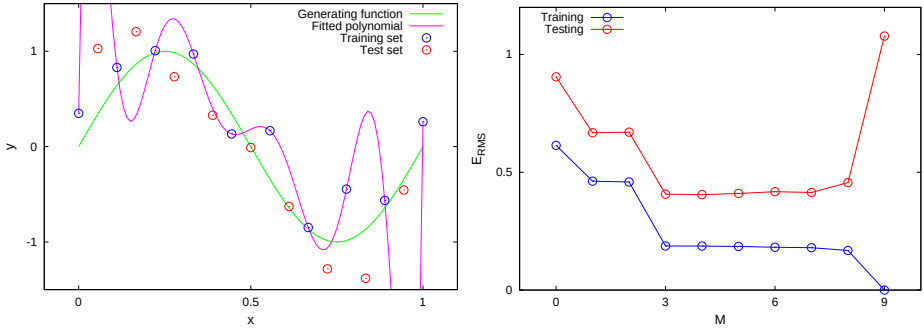
Example



$M = 8$

Illustration of overfitting

Example



$M = 9$

Train and Test sets: Avoid overfitting

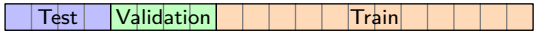
- Use a training set to train the machine
- Use a separate data set to avoid overfitting
- **However:** This biases the machine towards the separate set
  - Performance on this set is not an unbiased estimate of real-world performance
- **Solution:** Separate the data into three distinct sets

Train	Optimise the objective function
Validation	Model selection
Test	Estimate performance

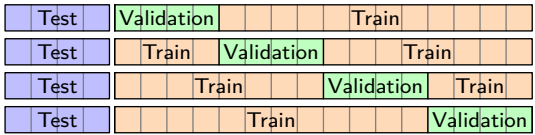
Cross-Validation

In practice, available training data is often limited

- Splitting the data in sets further reduces this



- Solution: k-fold cross-validation
- Repeatedly split the data and average the results (here, $k = 4$)



Error measures: Classification — Two-class problems

- Confusion Matrix

		Estimated	
		Positive	Negative
True	Positive (P)	TP	FN
	Negative (N)	FP	TN

- Performance measures

$$\text{Accuracy} = A = \frac{TP + TN}{P + N}$$

$$\text{Precision} = P = \frac{TP}{TP + FP}$$

$$\text{Recall} = R = \frac{TP}{TP + FN}$$

$$\text{F-measure} = F_1 = \frac{2 \cdot \text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$

$$\text{Specificity} = \frac{TN}{N}$$

Error measures: Classification — Two-class problems: Example

- Confusion Matrix

		Estimated: Positive	Estimated: Negative
True	Positive (P)	7	3
True	Negative (N)	2	8

- Performance measures

$$\text{Accuracy} = A = \frac{TP + TN}{P + N}$$

$$\text{Precision} = P = \frac{TP}{TP + FP}$$

$$\text{Recall} = R = \frac{TP}{TP + FN}$$

$$\text{F-measure} = F_1 = \frac{2 \cdot \text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$

$$\text{Specificity} = \frac{TN}{N}$$

Error measures: Classification — Two-class problems: Example

- Confusion Matrix

		Estimated	
		Positive	Negative
True	Positive (P)	7	3
	Negative (N)	2	8

- Performance measures

$$\text{Accuracy} = A = \frac{TP + TN}{P + N} = \frac{7+8}{20} = 0.75$$

$$\text{Precision} = P = \frac{TP}{TP + FP} = \frac{7}{7+2} = 0.78$$

$$\text{Recall} = R = \frac{TP}{TP + FN} = \frac{7}{7+3} = \frac{7}{10}$$

$$\text{F-measure} = F_1 = \frac{2 \cdot \text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} = \frac{2 \times \frac{7}{9} \times \frac{7}{10}}{\frac{7}{9} + \frac{7}{10}} = \frac{1}{4}$$

$$\text{Specificity} = \frac{TN}{N} = \frac{8}{10} = 0.8$$

Error measures: Classification — Two-class problems: Example

- Confusion Matrix

		Estimated: Positive	Estimated: Negative
True: Positive (P)		290	0
True: Negative (N)		10	0

- Performance measures

$$\text{Accuracy} = A = \frac{TP + TN}{P + N}$$

$$\text{Precision} = P = \frac{TP}{TP + FP}$$

$$\text{Recall} = R = \frac{TP}{TP + FN}$$

$$\text{F-measure} = F_1 = \frac{2 \cdot \text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$

$$\text{Specificity} = \frac{TN}{N}$$

Error measures: Classification — Two-class problems: Example

- Confusion Matrix

		Estimated: Positive	Estimated: Negative
True	Positive (P)	290	0
True	Negative (N)	10	0

- Performance measures

$$\text{Accuracy} = A = \frac{TP + TN}{P + N} = \frac{290}{300} = 0.97$$

$$\text{Precision} = P = \frac{TP}{TP + FP} = \frac{290}{300} = 0.97$$

$$\text{Recall} = R = \frac{TP}{TP + FN} = \frac{290}{290} = 1$$

$$\text{F-measure} = F_1 = \frac{2 \cdot \text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} = \frac{2 \times .97 \times 1}{.97 + 1} = 0.98$$

$$\text{Specificity} = \frac{TN}{N} = \frac{0}{10} = 0$$

Error measures: classification

- Confusion matrix (m classes)

$$\begin{array}{cc} & \text{Estimated class} \\ & \begin{array}{ccc} \mathcal{C}_1 & \dots & \mathcal{C}_m \end{array} \\ \text{True class} \begin{array}{c} \mathcal{C}_1 \\ \vdots \\ \mathcal{C}_m \end{array} & \begin{bmatrix} n_{11} & \dots & n_{1m} \\ \vdots & \ddots & \vdots \\ n_{m1} & \dots & n_{mm} \end{bmatrix} \end{array}$$

- Error measures:

$$\begin{aligned} \text{Accuracy} &= \#_{correct}/\#_{datapoints} = \frac{\sum_i n_{ii}}{\sum_{ij} n_{ij}} \\ \text{Error rate} &= 1 - \text{accuracy} \\ \text{Macro-Averaged Precision} &= \frac{1}{M}\sum_{m\in 1\dots M} \text{precision}(m) \\ \text{Macro-Averaged Recall} &= \frac{1}{M}\sum_{m\in 1\dots M} \text{recall}(m) \\ \text{Macro-Averaged } \mathrm{F}_1 &= \frac{1}{M}\sum_{m\in 1\dots M} \mathrm{F}_1(m) \end{aligned}$$

Decision Threshold

- Sometimes we don't want to "just" minimise the error rate

  Example: Cancer diagnosis

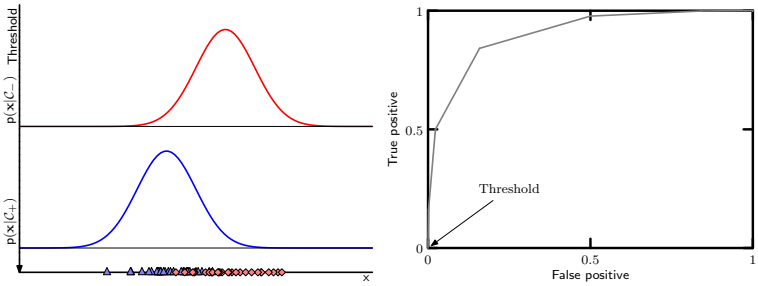
  Misclassifying a diseased person as healthy (FN) results in death, while misclassifying a healthy person (FP) results in additional tests.

- Cost/Loss function: weigh the errors according to type
  - Example: loss matrix $L = \begin{array}{c} \\ \text{Cancer} \\ \text{Normal} \end{array} \begin{array}{c} \begin{array}{cc} \text{Cancer} & \text{Normal} \end{array} \\ \begin{bmatrix} 0 & 1000 \\ 1 & 0 \end{bmatrix} \end{array}$
  - Minimise the expected loss: for each $\mathbf{x}$, assign to class j for which $\sum_k L_{kj} p(\mathcal{C}_k|\mathbf{x})$ is minimal

Receiver Operating Characteristic (ROC)

- Plot of True Positive Rate against False Positive Rate
- Each point of the curve corresponds to a different threshold

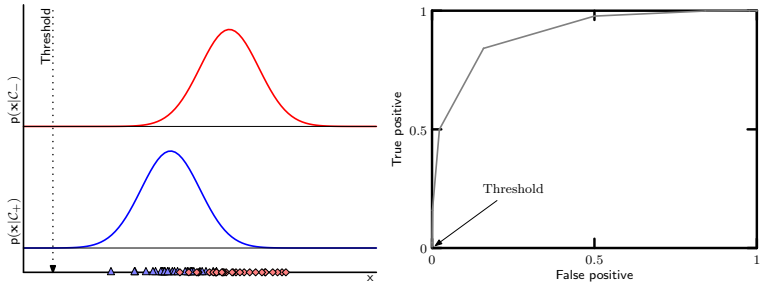
Example



Receiver Operating Characteristic (ROC)

- Plot of True Positive Rate against False Positive Rate
- Each point of the curve corresponds to a different threshold

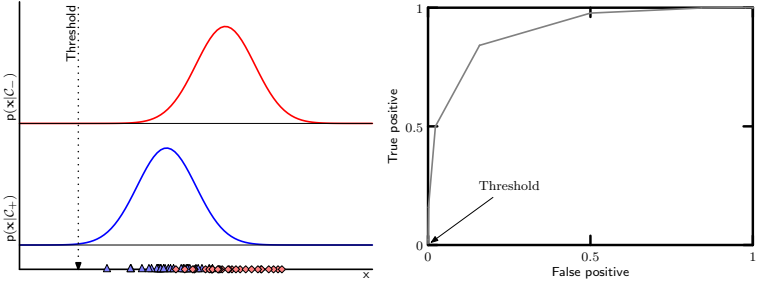
Example



Receiver Operating Characteristic (ROC)

- Plot of True Positive Rate against False Positive Rate
- Each point of the curve corresponds to a different threshold

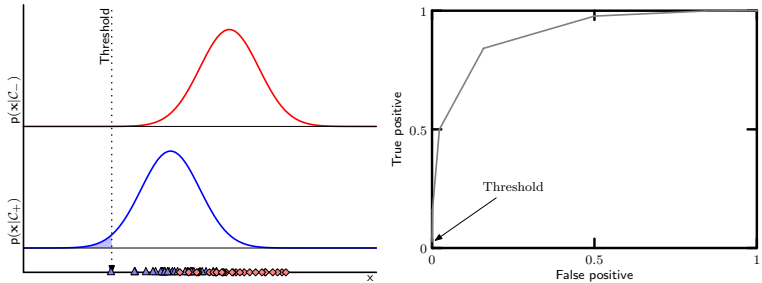
Example



Receiver Operating Characteristic (ROC)

- Plot of True Positive Rate against False Positive Rate
- Each point of the curve corresponds to a different threshold

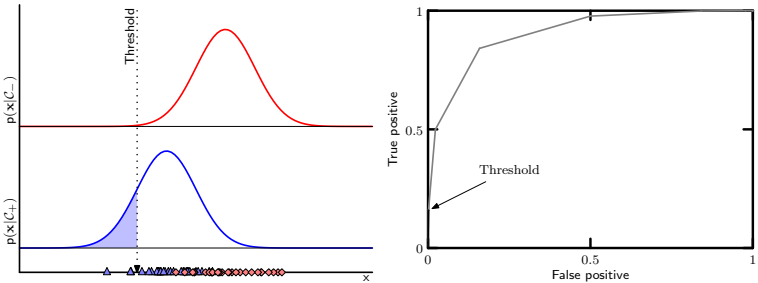
Example



Receiver Operating Characteristic (ROC)

- Plot of True Positive Rate against False Positive Rate
- Each point of the curve corresponds to a different threshold

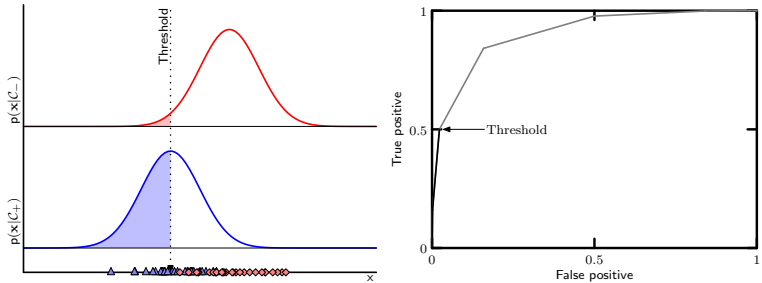
Example



Receiver Operating Characteristic (ROC)

- Plot of True Positive Rate against False Positive Rate
- Each point of the curve corresponds to a different threshold

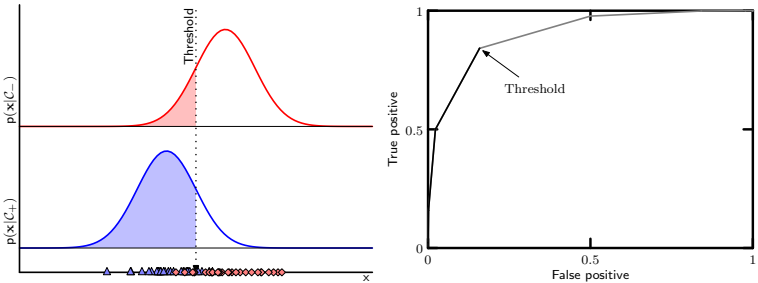
Example



Receiver Operating Characteristic (ROC)

- Plot of True Positive Rate against False Positive Rate
- Each point of the curve corresponds to a different threshold

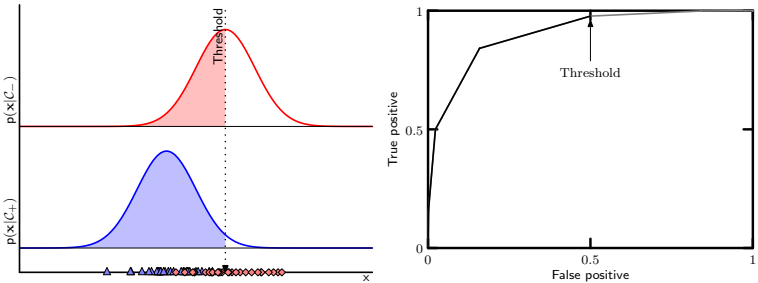
Example



Receiver Operating Characteristic (ROC)

- Plot of True Positive Rate against False Positive Rate
- Each point of the curve corresponds to a different threshold

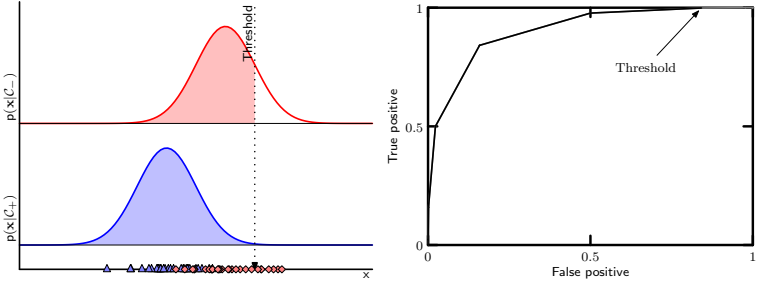
Example



Receiver Operating Characteristic (ROC)

- Plot of True Positive Rate against False Positive Rate
- Each point of the curve corresponds to a different threshold

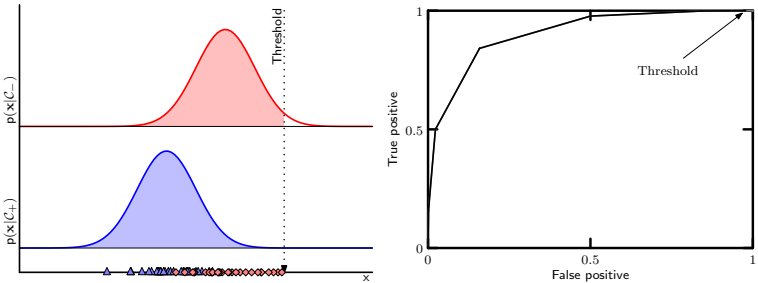
Example



Receiver Operating Characteristic (ROC)

- Plot of True Positive Rate against False Positive Rate
- Each point of the curve corresponds to a different threshold

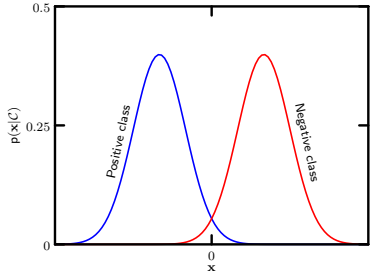
Example

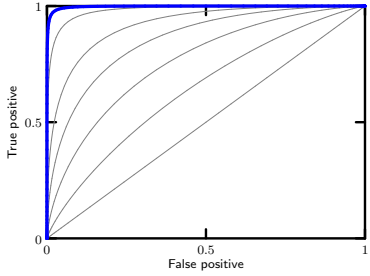


Area under the curve (AUC)

- The ROC gives a measure of the classification performance
- The Area Under the Curve reflects how well the classifier performs
  - Independently from the specific cost function used

Example

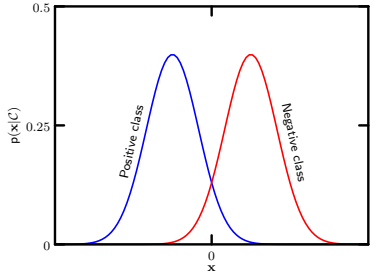


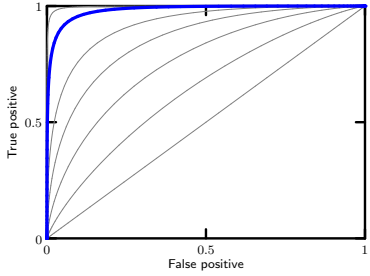


Area under the curve (AUC)

- The ROC gives a measure of the classification performance
- The Area Under the Curve reflects how well the classifier performs
  - Independently from the specific cost function used

Example

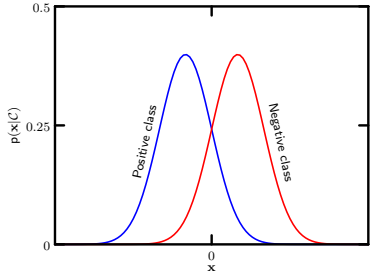


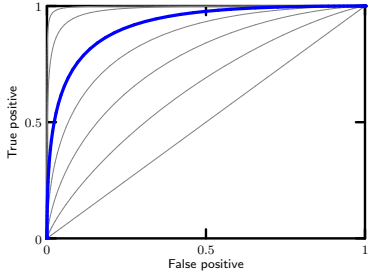


Area under the curve (AUC)

- The ROC gives a measure of the classification performance
- The Area Under the Curve reflects how well the classifier performs
  - Independently from the specific cost function used

Example

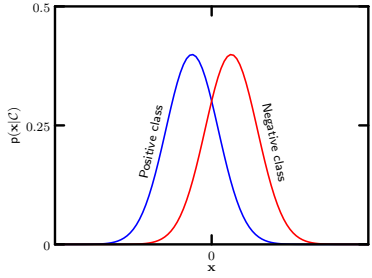


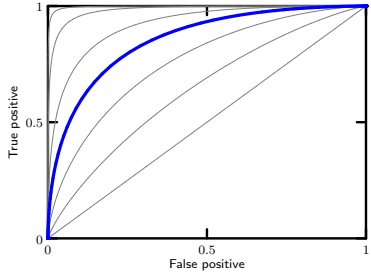


Area under the curve (AUC)

- The ROC gives a measure of the classification performance
- The Area Under the Curve reflects how well the classifier performs
  - Independently from the specific cost function used

Example

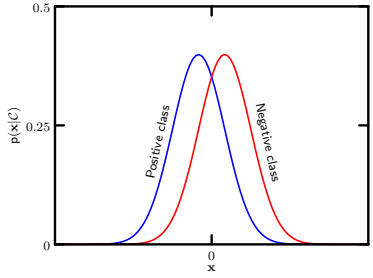


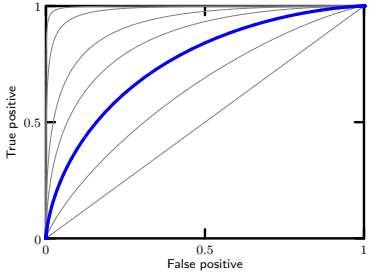


Area under the curve (AUC)

- The ROC gives a measure of the classification performance
- The Area Under the Curve reflects how well the classifier performs
  - Independently from the specific cost function used

Example

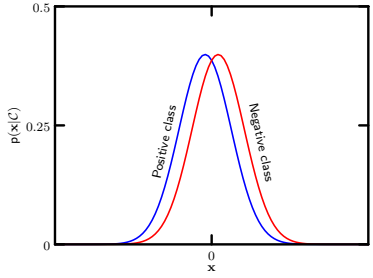


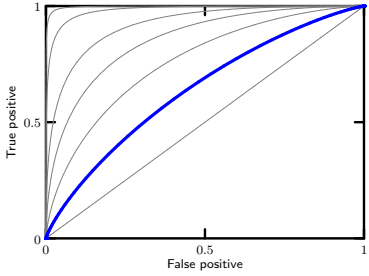


Area under the curve (AUC)

- The ROC gives a measure of the classification performance
- The Area Under the Curve reflects how well the classifier performs
  - Independently from the specific cost function used

Example

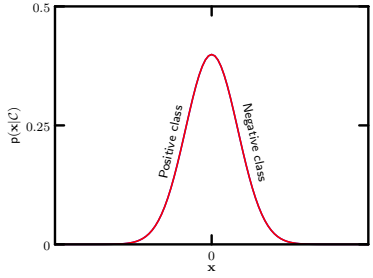


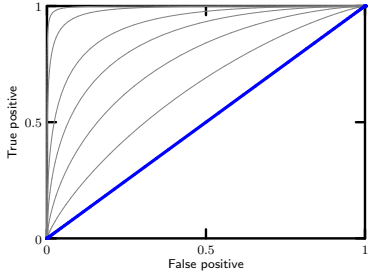


Area under the curve (AUC)

- The ROC gives a measure of the classification performance
- The Area Under the Curve reflects how well the classifier performs
  - Independently from the specific cost function used

Example





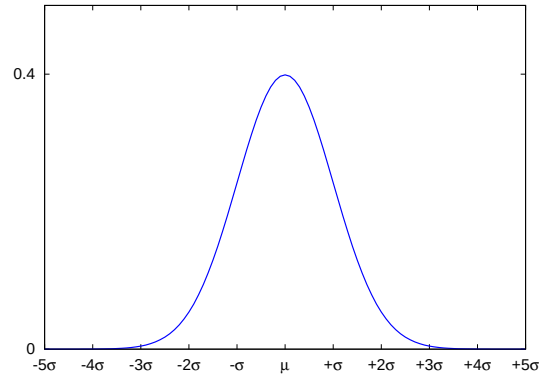
Summary

- We introduced Machine Learning
  - What, why, how, when
- Homework: recapitulation of some basic maths
- Lab: Introduction to the software environment
- Next week: linear discriminants


Are there any questions?

Outline: What, Why, How and When?

The Gaussian (Normal) Distribution



back

Notation

- $x \sim \mathcal{N}(\mu, \sigma)$

Properties

- $p(x) = \frac{1}{\sqrt{2\pi}\sigma} \exp - \frac{(x-\mu)^2}{2\sigma^2}$
- $\mathbb{E}[x] = \mu$
- $\text{var}[x] = \sigma^2$