C4B Machine Learning

8 Lectures — Hilary Term 2011

2 Tutorial Sheets — A. Zisserman

Overview:

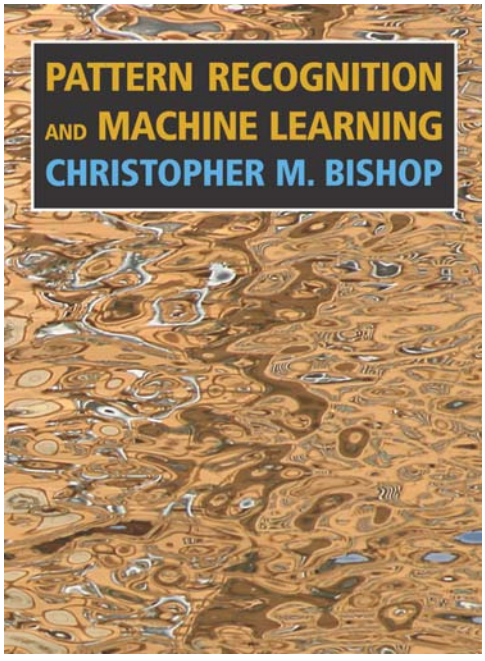
- Supervised classification
  - support vector machine, logistic regression, adaboost, loss functions, kernels
- Supervised regression
  - ridge regression, lasso regression, SVM regression
- Unsupervised learning
  - k-means, PCA, Gaussian Mixture Models, EM, pLSA

Recommended book

- **Pattern Recognition and Machine Learning**

Christopher Bishop, Springer, 2006.

  - Excellent on classification and regression

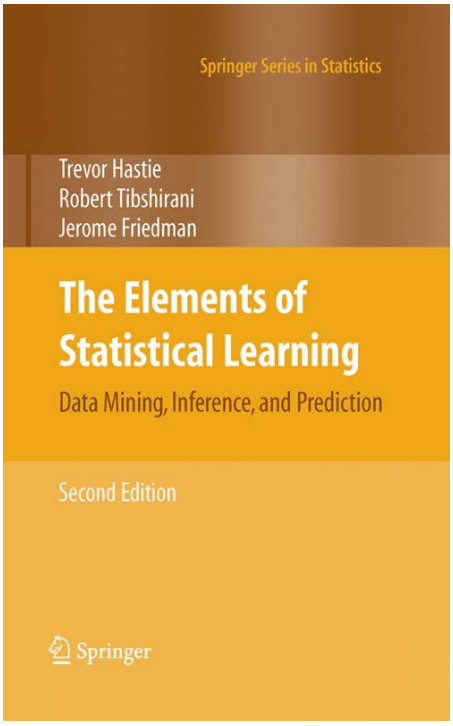


Textbooks 2

- **Elements of Statistical Learning**

Hastie, Tibshirani, Friedman, Springer, 2009, second edition

- Good explanation of algorithms
- pdf available online



One more book for background reading …

- Data Mining: Practical Machine Learning Tools and Techniques (Second Edition)

Ian Witten & Eibe Frank, Morgan Kaufmann, 2005.

- Very readable and practical guide

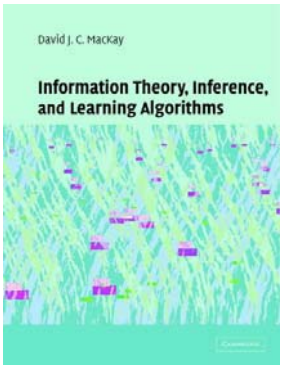
Web resources

- On line book:

Information Theory, Inference, and Learning Algorithms.

David J. C. MacKay, CUP, 2003

  - Covers some of the course material though at an advanced level



- Further reading (www addresses) and the lecture notes are on

http://www.robots.ox.ac.uk/~az/lectures/ml

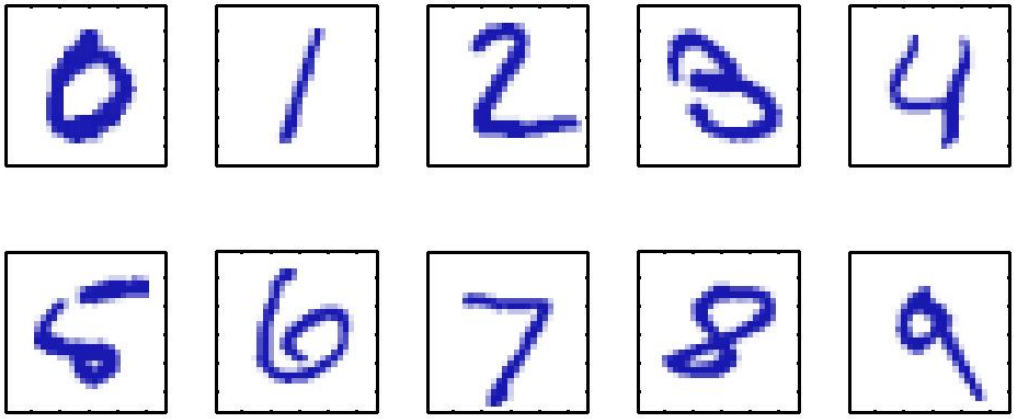
Introduction: What is Machine Learning?

Algorithms that can improve their performance using training data

- Typically the algorithm has a (large) number of parameters whose values are learnt from the data

- Can be applied in situations where it is very challenging (= impossible) to define rules by hand, e.g.:
  - Face detection
  - Speech recognition
  - Stock prediction

Example 1: hand-written digit recognition



Images are 28 x 28 pixels

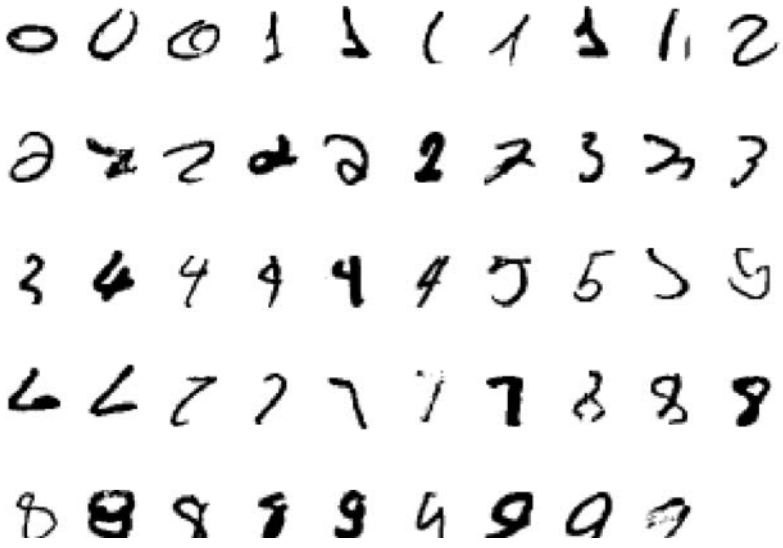
Represent input image as a vector $\mathbf{x} \in \mathbb{R}^{784}$
Learn a classifier $f(\mathbf{x})$ such that,

$$f : \mathbf{x} \rightarrow \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$$

How to proceed ...

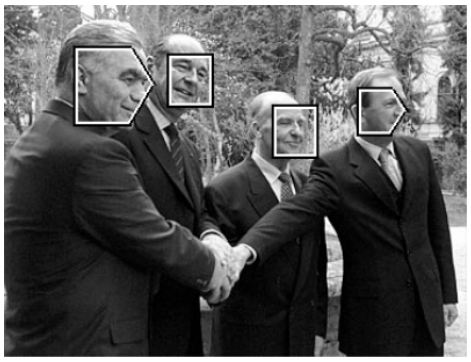
As a supervised classification problem

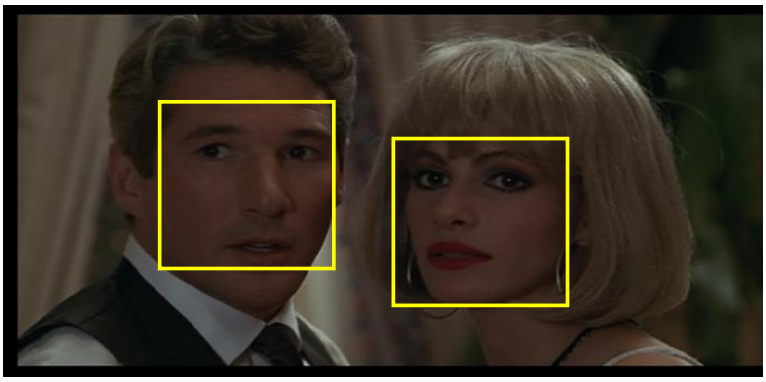
Start with training data, e.g. 6000 examples of each digit



- Can achieve testing error of 0.4%
- One of first commercial and widely used ML systems (for zip codes & checks)

Example 2: Face detection





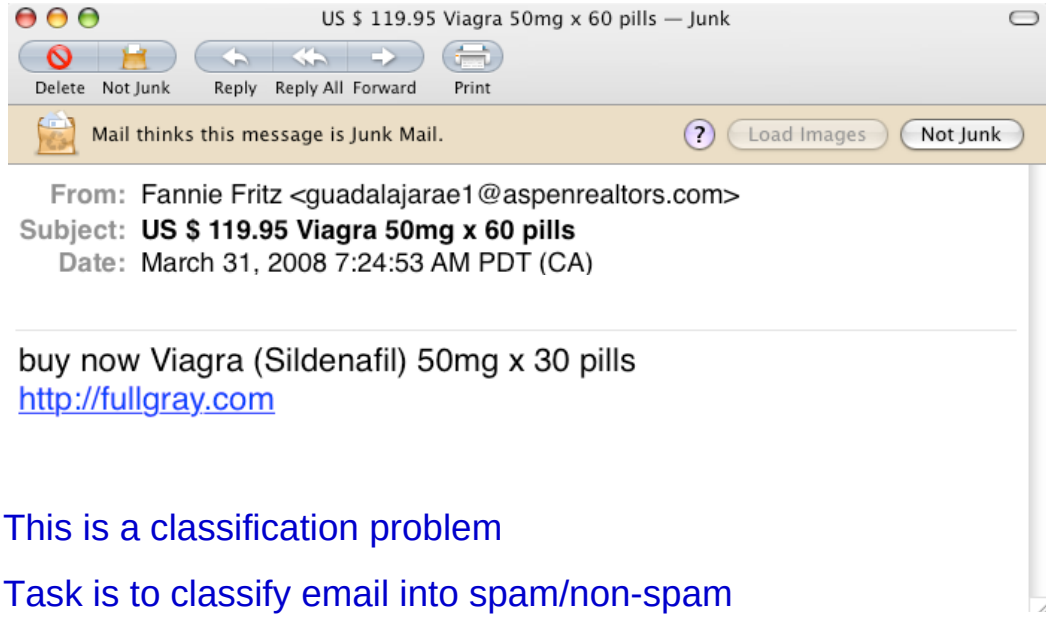
- Again, a supervised classification problem
- Need to classify an image window into three classes:
  - non-face
  - frontal-face
  - profile-face

Classifier is learnt from labelled data

Training data for frontal faces

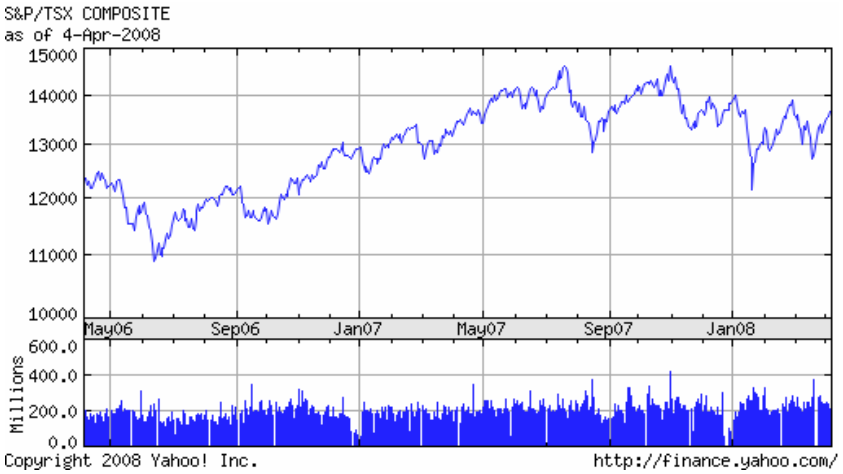
- 5000 faces
  - All near frontal
  - Age, race, gender, lighting
- 10^8 non faces
- faces are normalized
  - scale, translation

Example 3: Spam detection



- This is a classification problem
- Task is to classify email into spam/non-spam
- Data x_i is word count, e.g. of viagra, outperform, "you may be surprized to be contacted" …
- Requires a learning system as "enemy" keeps innovating

Example 4: Stock price prediction



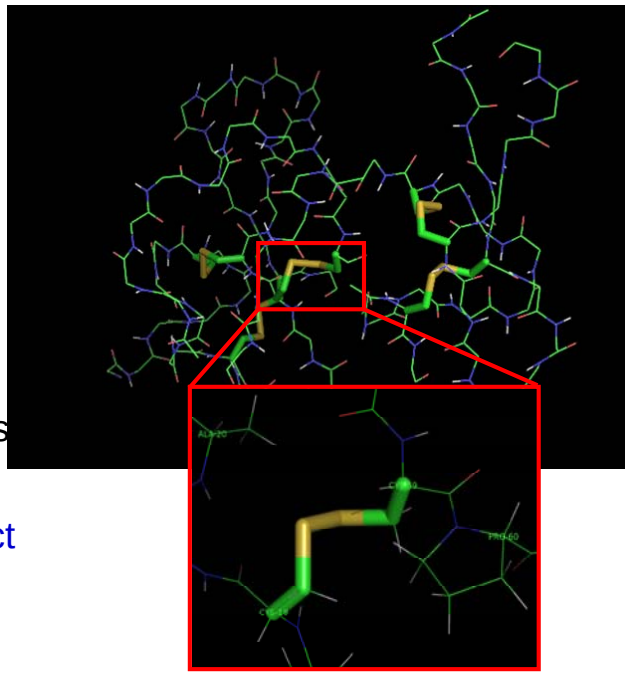
- Task is to predict stock price at future date
- This is a regression task, as the output is continuous

Example 5: Computational biology

x

y

AVITGACERDLQCG
KGTCCAVSLWIKSV
RVCTPVGTSGEDCH
PASHKIPFSGQRMH
HTCPCAPNLACVQT
SPKKFKCLSK



Protein Structure and Disulfide Bridges

Regression task: given sequence predict 3D structure

Protein: 1IMT

Web examples: Machine translation

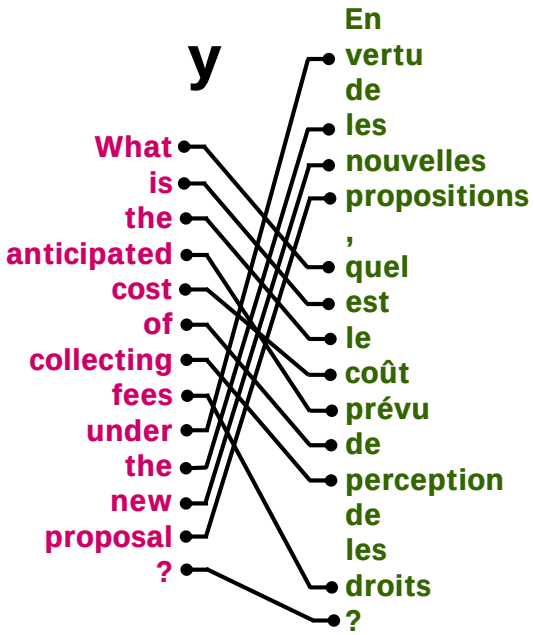
Use of aligned text

x

What is the anticipated cost of collecting fees under the new proposal?

En vertu des nouvelles propositions, quel est le coût prévu de perception des droits?

y



e.g. Google translate

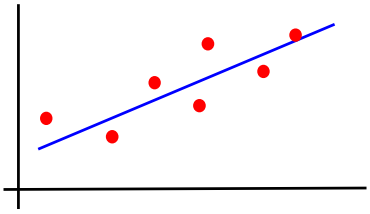
What is the anticipated cost of collecting fees under the new proposal?

Web examples: Recommender systems

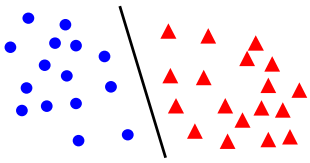
People who bought Hastie …

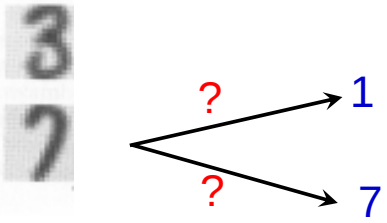
Three canonical learning problems

1. Regression - supervised
   - estimate parameters, e.g. of weight vs height

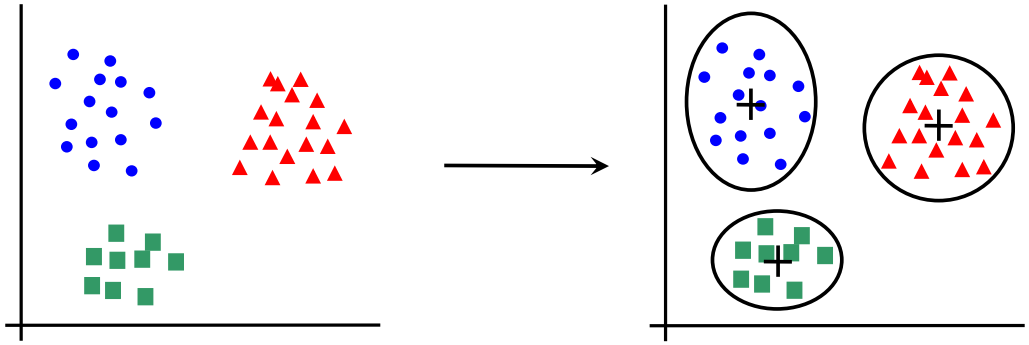


2. Classification - supervised
   - estimate class, e.g. handwritten digit classification

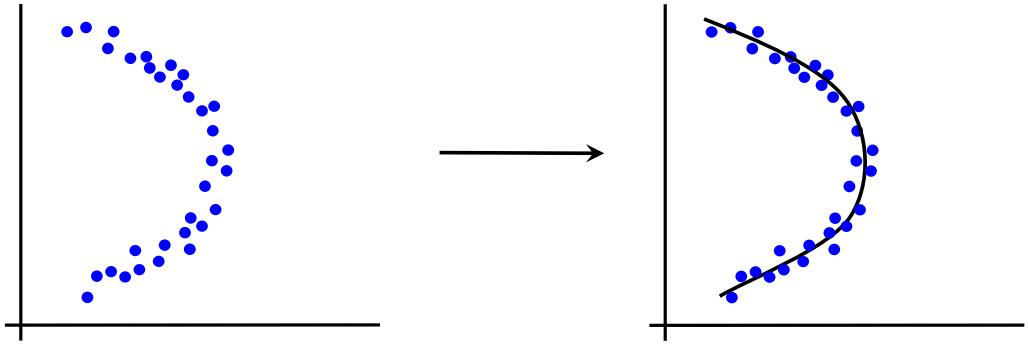




3. Unsupervised learning
   - clustering



- dimensionality reduction



Supervised Learning: Overview

Functions $\mathcal{F}$

$f : \mathcal{X} \rightarrow \mathcal{Y}$

Training data

$\{(x_i, y_i) \in \mathcal{X} \times \mathcal{Y}\}$

LEARNING

find $\hat{f} \in \mathcal{F}$
s.t. $y_i \approx \hat{f}(x_i)$

Learning machine

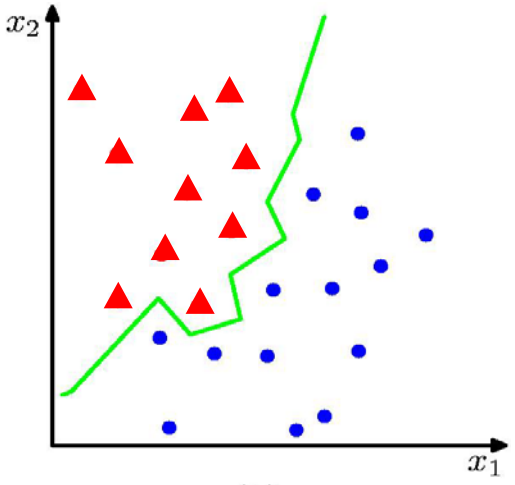
PREDICTION

$y = \hat{f}(x)$

New data

x

Classification



- Suppose we are given a training set of N observations

$(x_1, \ldots, x_N)$ and $(y_1, \ldots, y_N), x_i \in \mathbb{R}^d, y_i \in \{-1, 1\}$

- Classification problem is to estimate f(x) from this data such that

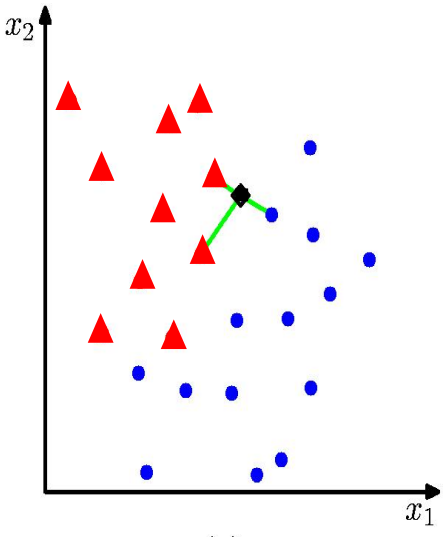
$$f(x_i) = y_i$$

K Nearest Neighbour (K-NN) Classifier

Algorithm

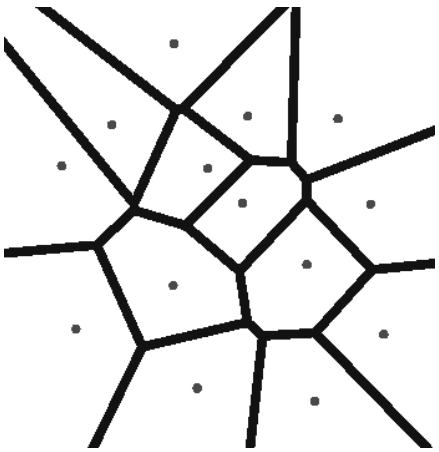
- For each test point, x, to be classified, find the K nearest samples in the training data
- Classify the point, x, according to the majority vote of their class labels

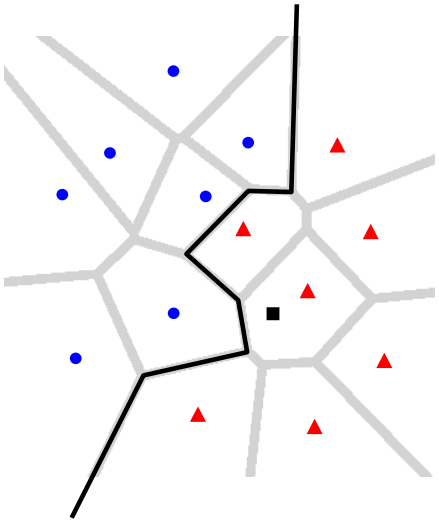
e.g. K = 3



• applicable to multi-class case

K = 1





Voronoi diagram:

- partitions the space into regions
- boundaries are equal distance from training points

Classification boundary:

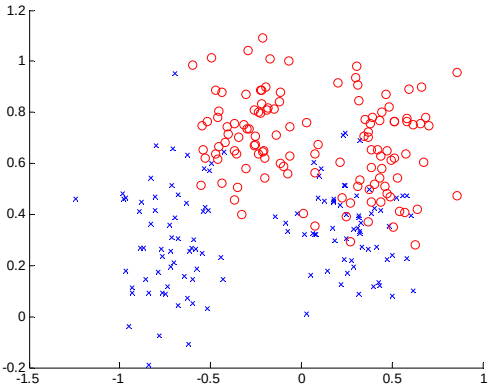
- non-linear

A sampling assumption: training and test data

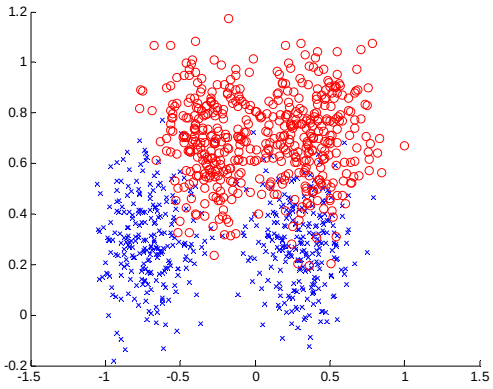
• Assume that the training examples are drawn independently from the set of all possible examples.

• This makes it very unlikely that a strong regularity in the training data will be absent in the test data.

• Measure classification error as $= \frac{1}{N}\sum_{i=1}^{N}\underbrace{[\mathbf{y}_i \neq f(\mathbf{x}_i)]}_{\text{loss function}}$ The “risk”



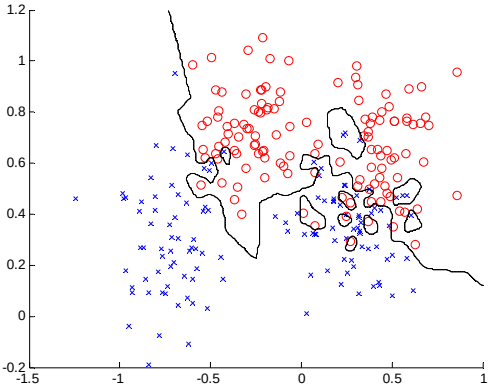
Training data



Testing data

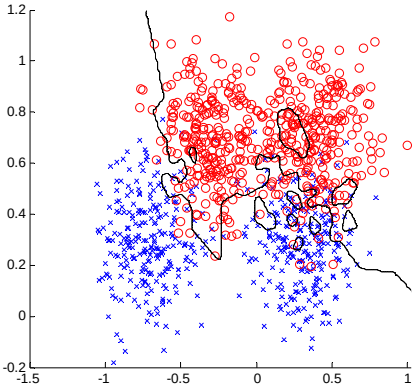
K = 1

Training data



error = 0.0

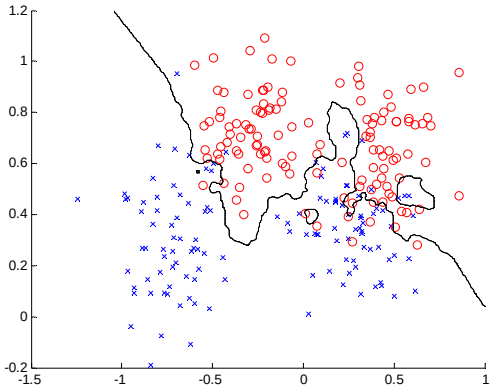
Testing data



error = 0.15

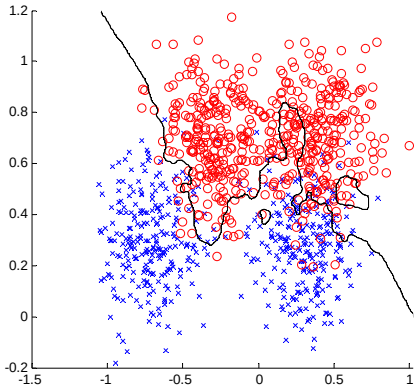
K = 3

Training data



error = 0.0760

Testing data



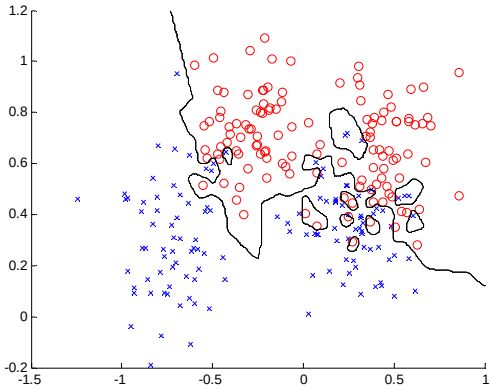
error = 0.1340

Generalization

- The real aim of supervised learning is to do well on test data that is not known during learning
- Choosing the values for the parameters that minimize the loss function on the training data is not necessarily the best policy
- We want the learning machine to model the true regularities in the data and to ignore the noise in the data.

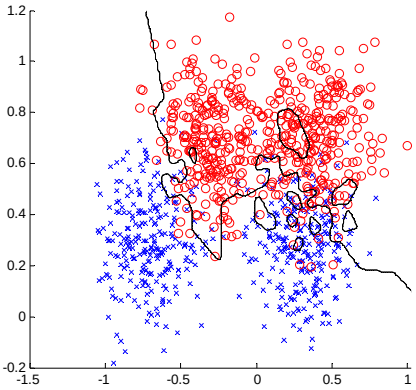
K = 1

Training data



error = 0.0

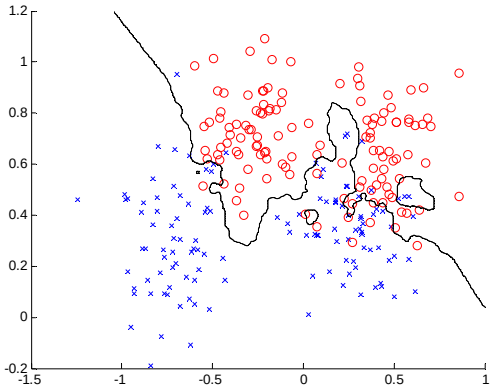
Testing data



error = 0.15

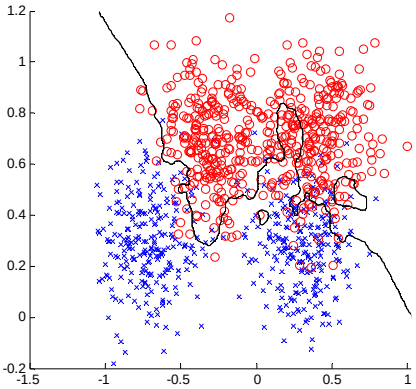
K = 3

Training data



error = 0.0760

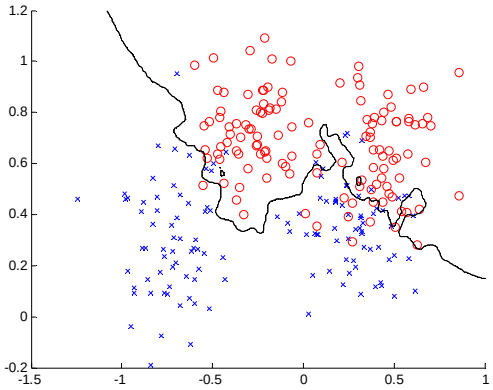
Testing data



error = 0.1340

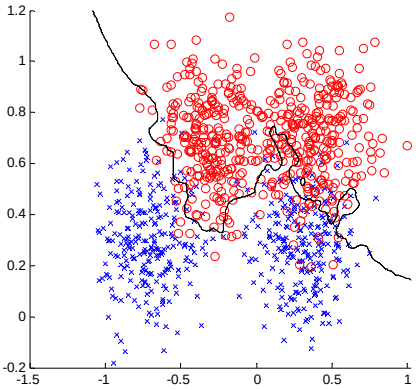
K = 7

Training data



error = 0.1320

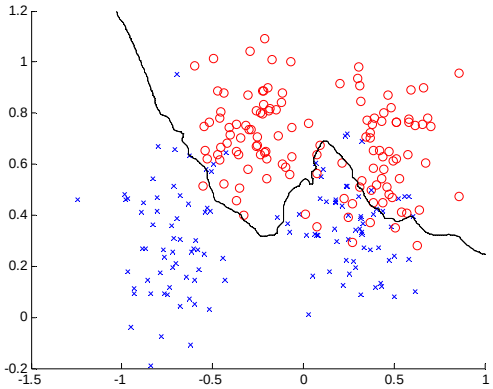
Testing data



error = 0.1110

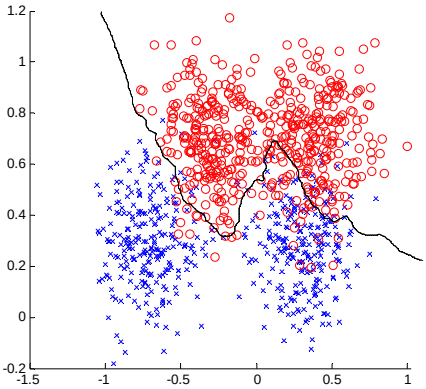
K = 21

Training data



error = 0.1120

Testing data



error = 0.0920

Properties and training

As K increases:

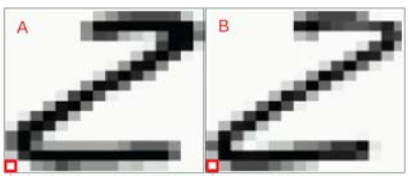
- Classification boundary becomes smoother
- Training error can increase

Choose (learn) K by cross-validation

- Split training data into training and validation
- Hold out validation data and measure error on this

Example: hand written digit recognition

- MNIST data set
- Distance = raw pixel distance between images
- 60K training examples
- 10K testing examples
- K-NN gives 5% classification error

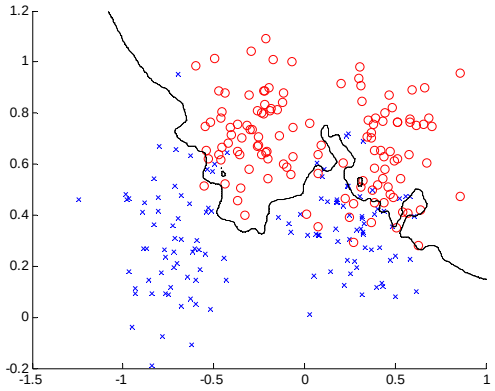


$$D(\mathbf{A},\mathbf{B}) = \sum_{ij} \sqrt{\left(a_{ij} - b_{ij}\right)^2}$$

Summary

Advantages:

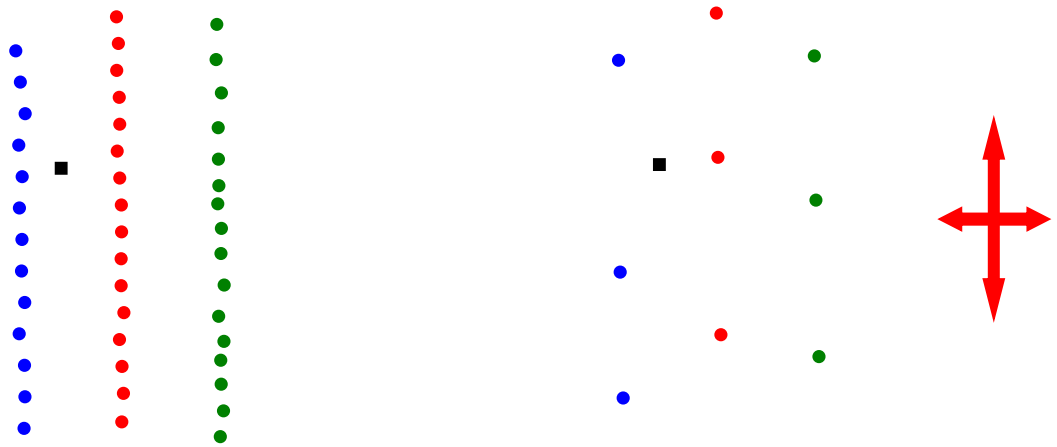
- K-NN is a simple but effective classification procedure
- Applies to multi-class classification
- Decision surfaces are non-linear
- Quality of predictions automatically improves with more training data
- Only a single parameter, K; easily tuned by cross-validation



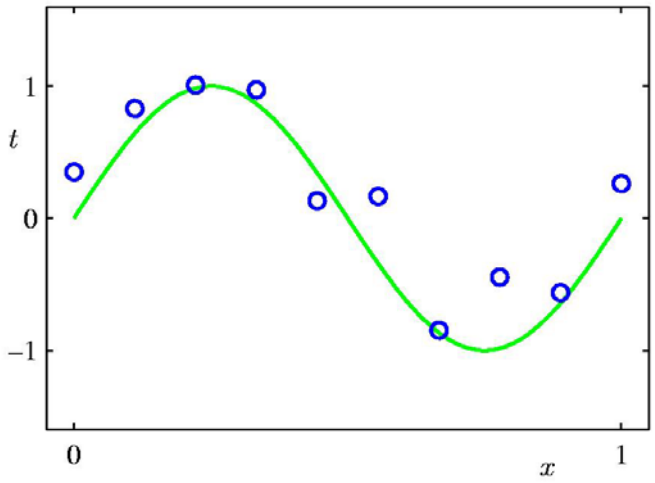
Summary

Disadvantages:

- What does nearest mean? Need to specify a distance metric.
- Computational cost: must store and search through the entire training set at test time. Can alleviate this problem by thinning, and use of efficient data structures like KD trees.



Regression



• Suppose we are given a training set of N observations

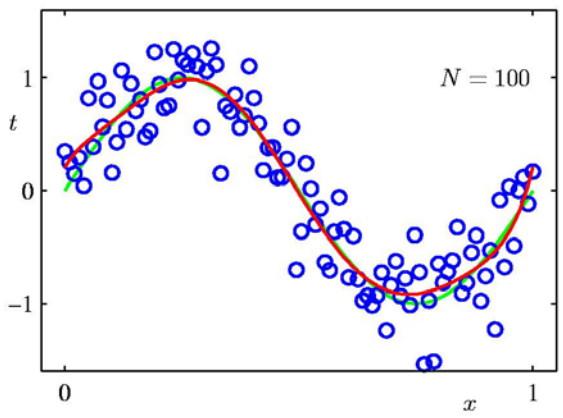
$$(x_1, \ldots, x_N) \text{ and } (y_1, \ldots, y_N), x_i, y_i \in \mathbb{R}$$

• Regression problem is to estimate y(x) from this data

K-NN Regression

Algorithm

- For each test point, x, find the K nearest samples x_i in the training data and their values y_i
- Output is mean of their values $f(\mathbf{x}) = \frac{1}{K} \sum_{i=1}^{K} y_i$
- Again, need to choose (learn) K



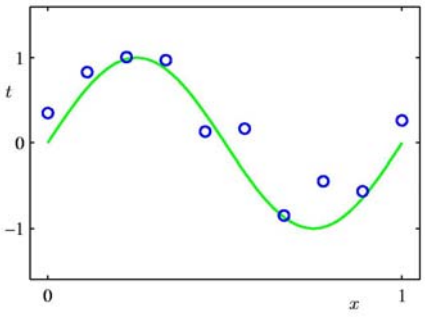
Regression example: polynomial curve fitting

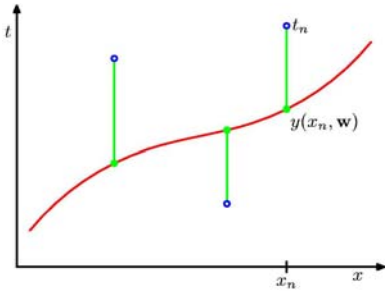
• The green curve is the true function (which is not a polynomial)

• The data points are uniform in x but have noise in y.

• We will use a loss function that measures the squared error in the prediction of y(x) from x. The loss for the red polynomial is the sum of the squared vertical errors.

from Bishop





$$E(\mathbf{w}) = \frac{1}{2}\sum_{i=1}^{N}\{y(x_i, \mathbf{w}) - t_i\}^2$$

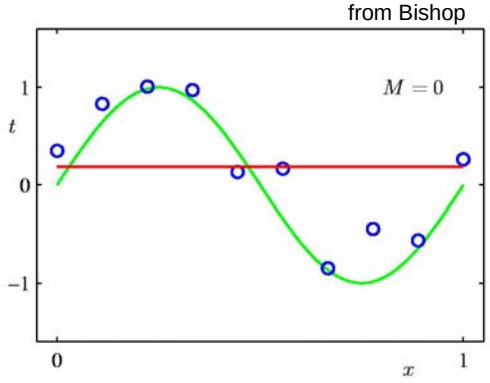
target value

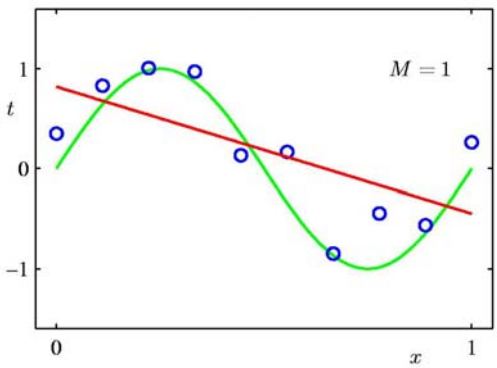
$$y(x, \mathbf{w}) = w_0 + w_1 x + w_2 x^2 + \ldots + w_M x^M = \sum_{j=0}^{M} w_j x^j$$

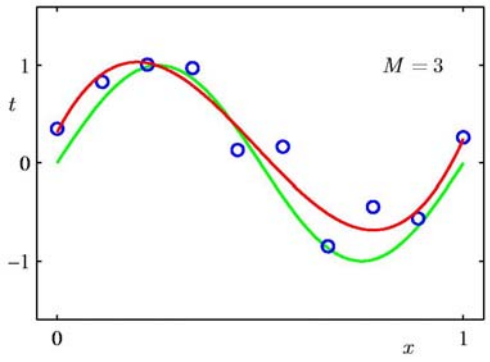
polynomial regression

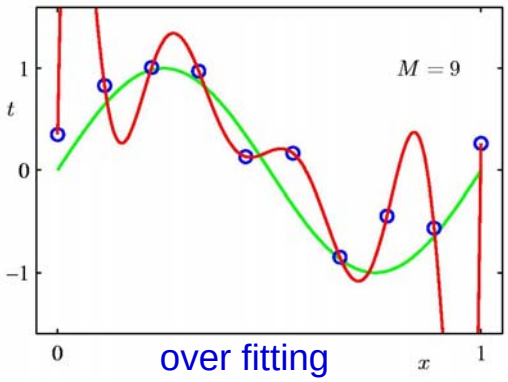
Some fits to the data: which is best?

from Bishop





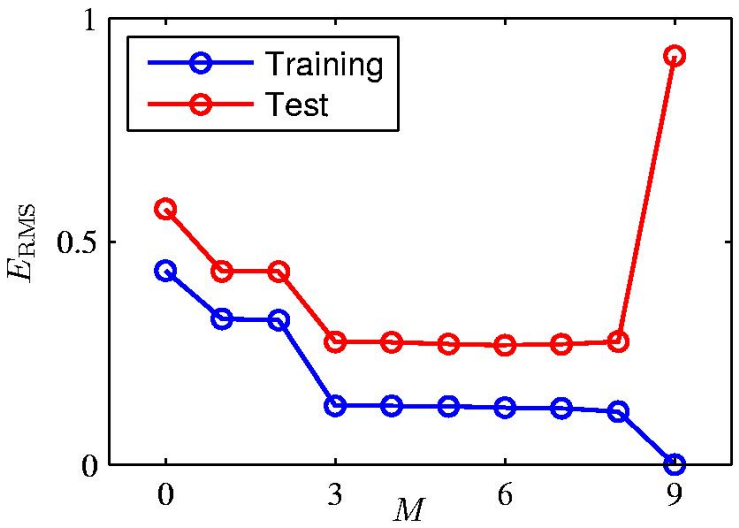




over fitting

Over-fitting

• test data: a different sample from the same true function



Root-Mean-Square (RMS) Error: $E_{\mathrm{RMS}} = \sqrt{2E(\mathbf{w}^\star)/N}$

• training error goes to zero, but test error increases with M

Trading off goodness of fit against model complexity

• If the model has as many degrees of freedom as the data, it can fit the training data perfectly

• But the objective in ML is generalization

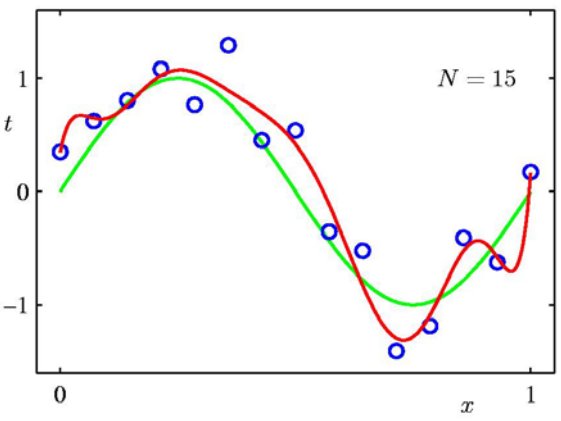
• Can expect a model to generalize well if it explains the training data surprisingly well given the complexity of the model.

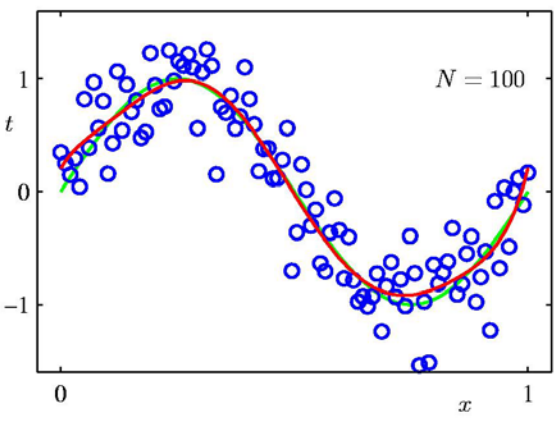
Polynomial Coefficients

	$M=0$	$M=1$	$M=3$	$M=9$
$w_0^\star$	0.19	0.82	0.31	0.35
$w_1^\star$		-1.27	7.99	232.37
$w_2^\star$			-25.43	-5321.83
$w_3^\star$			17.37	48568.31
$w_4^\star$				-231639.30
$w_5^\star$				640042.26
$w_6^\star$				-1061800.52
$w_7^\star$				1042400.18
$w_8^\star$				-557682.99
$w_9^\star$				125201.43

How to prevent over fitting? I

- Add more data than the model “complexity”
- For 9th order polynomial:





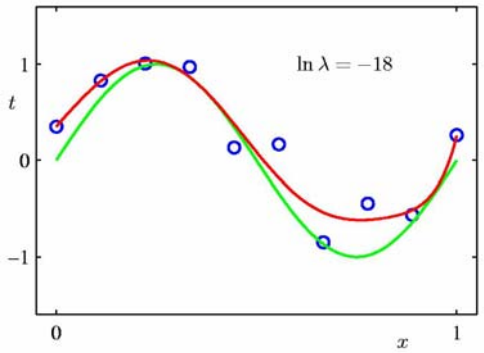
How to prevent over fitting? II

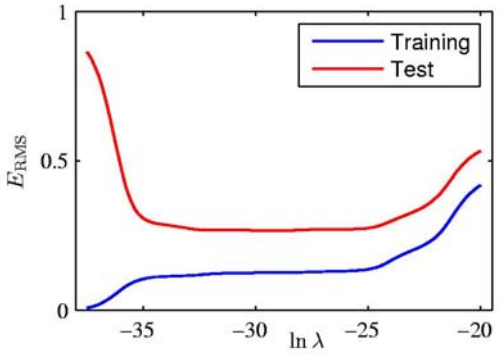
- Regularization: penalize large coefficient values

$$\widetilde{E}(\mathbf{w}) = \frac{1}{2}\sum_{i=1}^{N}\{y(x_i,\mathbf{w}) - t_i\}^2 + \frac{\lambda}{2}\|\mathbf{w}\|^2$$

"ridge" regression

loss function regularization





In practice use validation data to choose λ (not test)

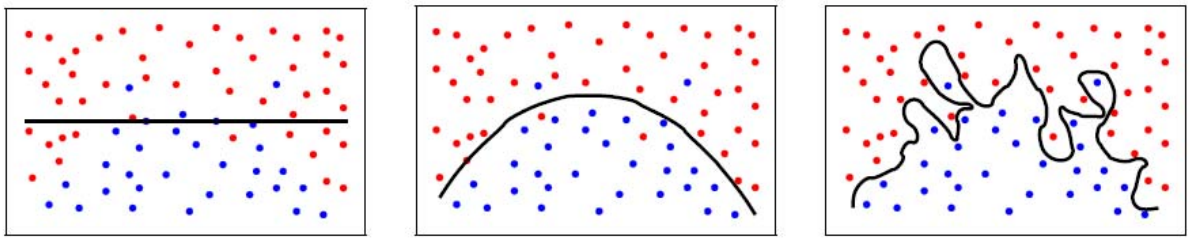
- cf with KNN classification as N increases
- we will return to regularization for regression later

Polynomial Coefficients

	$\ln\lambda = -\infty$	$\ln\lambda = -18$	$\ln\lambda = 0$
$w_0^\star$	0.35	0.35	0.13
$w_1^\star$	232.37	4.74	-0.05
$w_2^\star$	-5321.83	-0.77	-0.06
$w_3^\star$	48568.31	-31.97	-0.05
$w_4^\star$	-231639.30	-3.89	-0.03
$w_5^\star$	640042.26	55.28	-0.02
$w_6^\star$	-1061800.52	41.32	-0.01
$w_7^\star$	1042400.18	-45.95	-0.00
$w_8^\star$	-557682.99	-91.53	0.00
$w_9^\star$	125201.43	72.68	0.01

Generalization Problem in Classification

Underfitting $\longleftrightarrow$ Overfitting



• Again, need to control the complexity of the (discriminant) function

What comes next?

• Learning by optimizing a cost function:

$$\widetilde{E}(\mathbf{w}) = \frac{1}{2}\sum_{i=1}^{N}\{y(x_i,\mathbf{w}) - t_i\}^2 + \frac{\lambda}{2}\|\mathbf{w}\|^2$$

loss function regularization

• In general Minimize with respect to $f \in \mathcal{F}$

$$\sum_{i=1}^{N} l\left(f(x_i), y_i\right) + \lambda R\left(f\right)$$

• choose loss function for: classification, regression, clustering …

• choose regularization function

Background reading

- Bishop, chapter 1
- Hastie et al, chapter 2
- Witten & Frank, chapter 1 for example applications
- More on web page:
  http://www.robots.ox.ac.uk/~az/lectures/ml