

Marcel Gehrke

Data Understanding vs. Machine Training

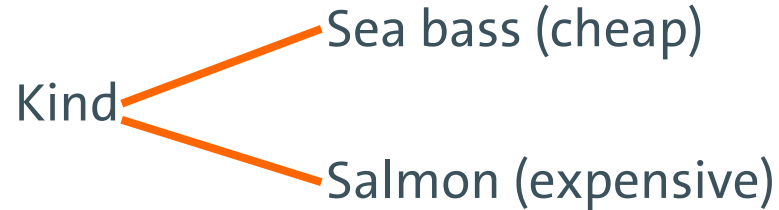
Classification and Regression

Supervised learning

Classification and Combination of Features

Example

“Sorting of fish on a conveyor belt by species by image processing”



Analysis of the Problem

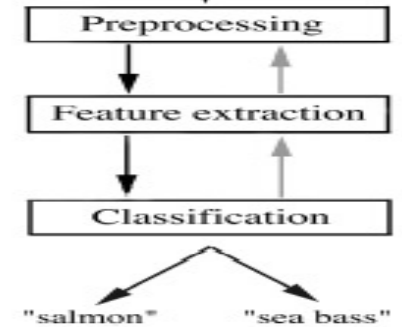
Use camera and take pictures to determine features:

- Length
- Brightness
- Width
- Number and shape of fins
- Position of the mouth, etc.

Set of all possible features

Goal: Select the relevant ones

Classification



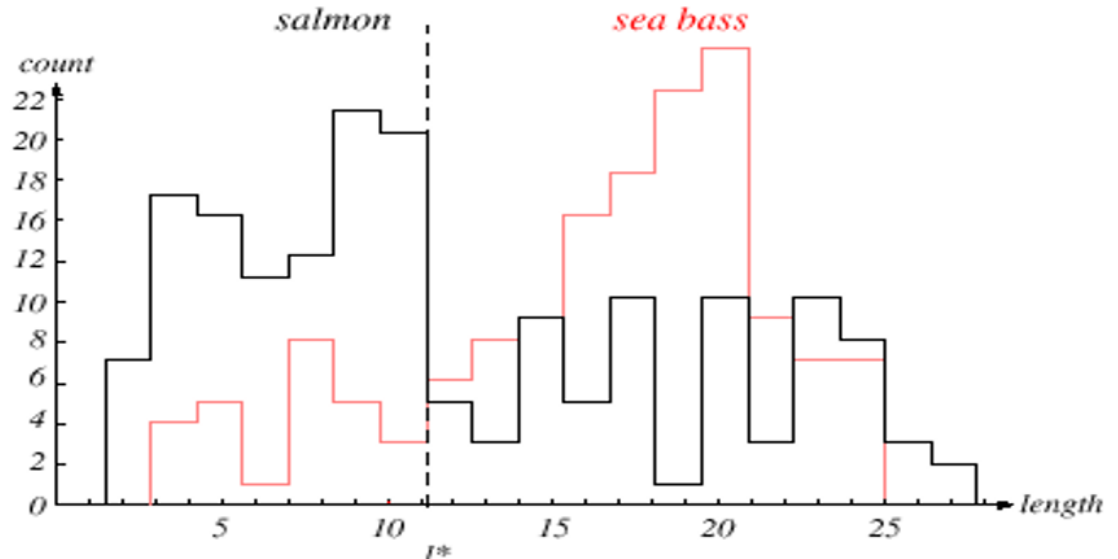
Selection of Suitable Features

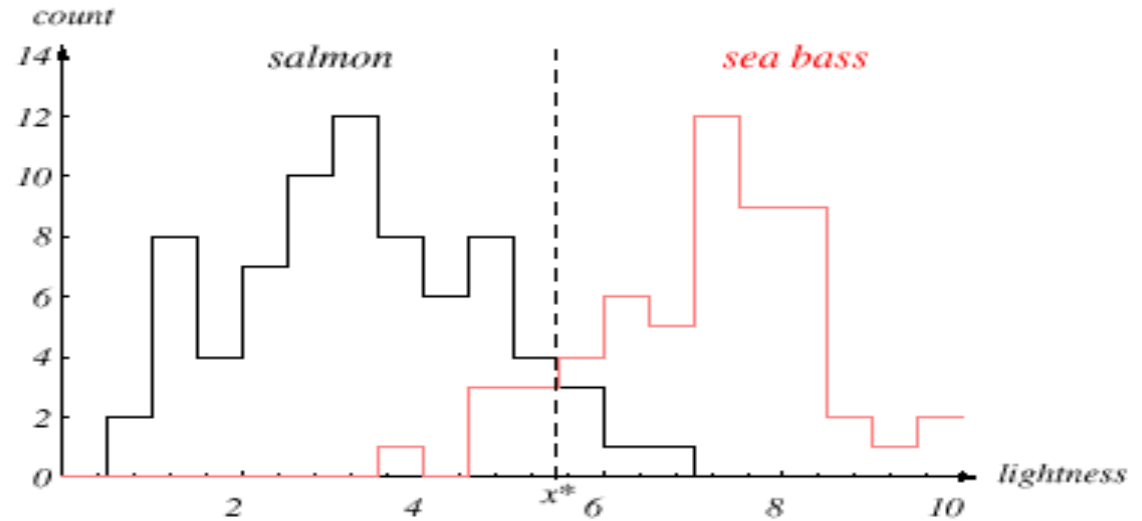
We need an expert to determine the characteristics that can be used to properly classify sea bass and salmon.

How about length as a differentiating feature?

Length Alone is not a Good Feature!

- High costs in case of wrong decision
 - How about lightness?





Decision Boundary based on Threshold and Induced Costs

- Decisions based on a threshold close to the medium lightness values minimises the costs (of misclassification)

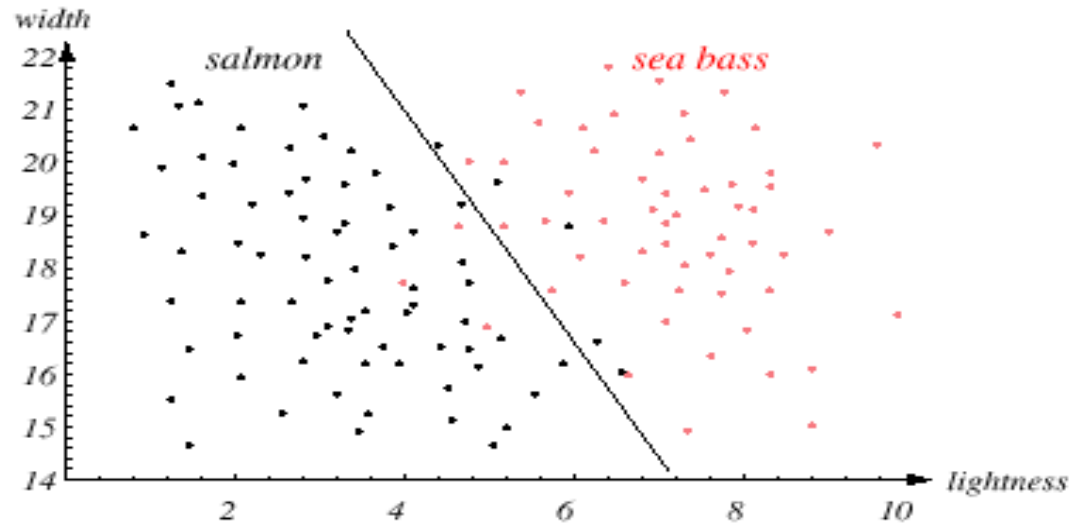


Investigation in the so-called decision theory

The aim is to automatically determine functions for suitable characteristics

Lightness and Additional Width of the Fish?



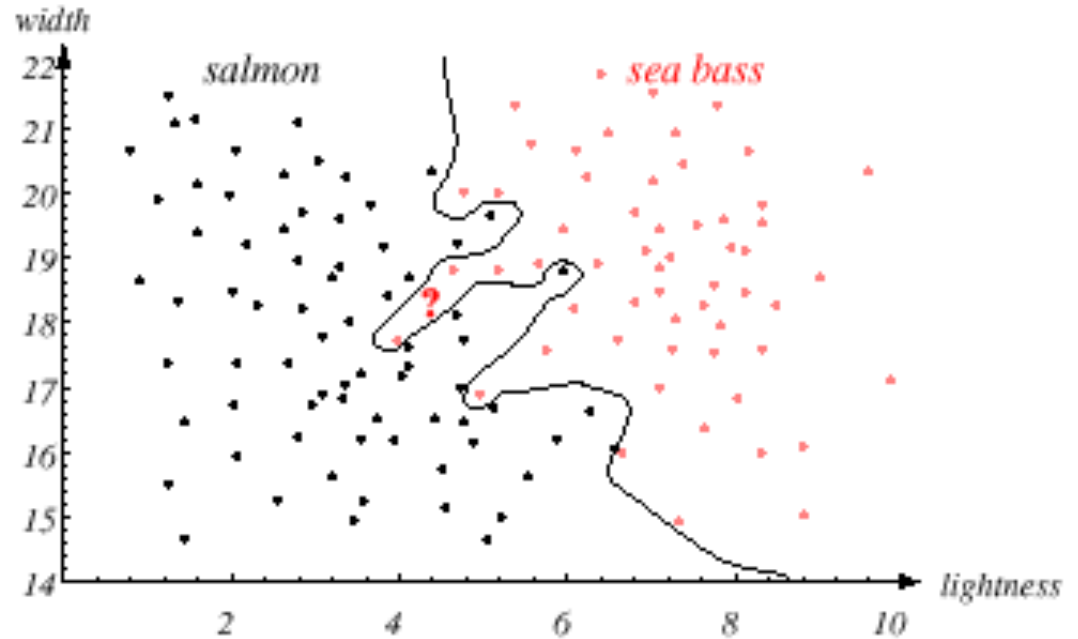


Additional features that are not directly related to brightness and width could be added

- Beware, however, of reduction due to "noisy features"

Desirably, the best **decision boundary** results in **optimal performance** (in the sense of minimising losses through misclassification)

Overfitted Decision Boundary



Generalisation

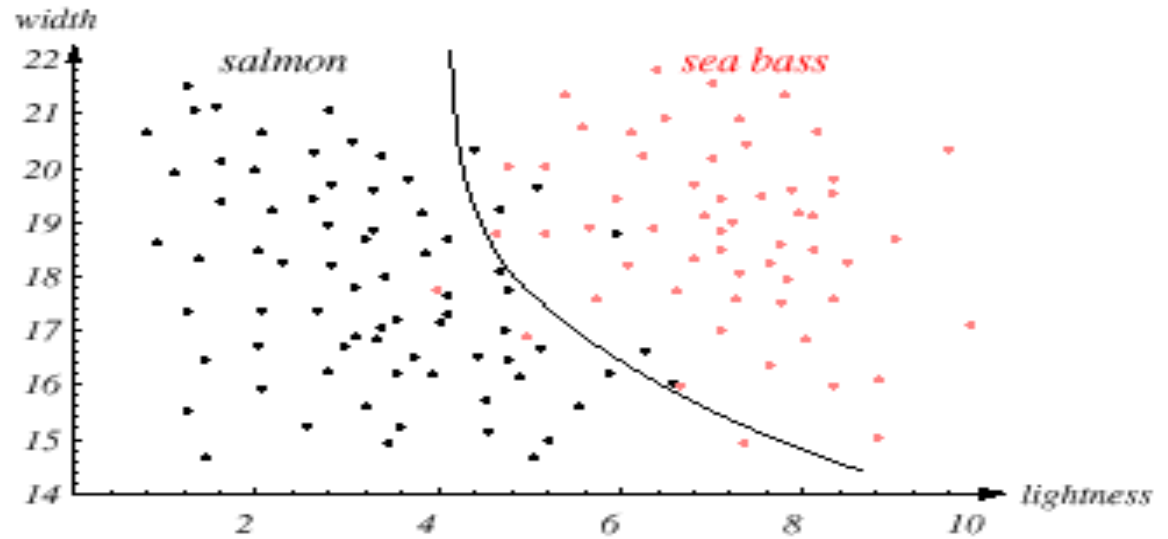
Having good classification performance on test data does not suffice.

Performance on new data is important!



Ability to generalise matters most!

Generalised Decision Boundary



Supervised learning

Support-Vector Machines

Support Vector Machines

Mapping of instances of two classes into a linearly separable space

Mapping function is called kernel function

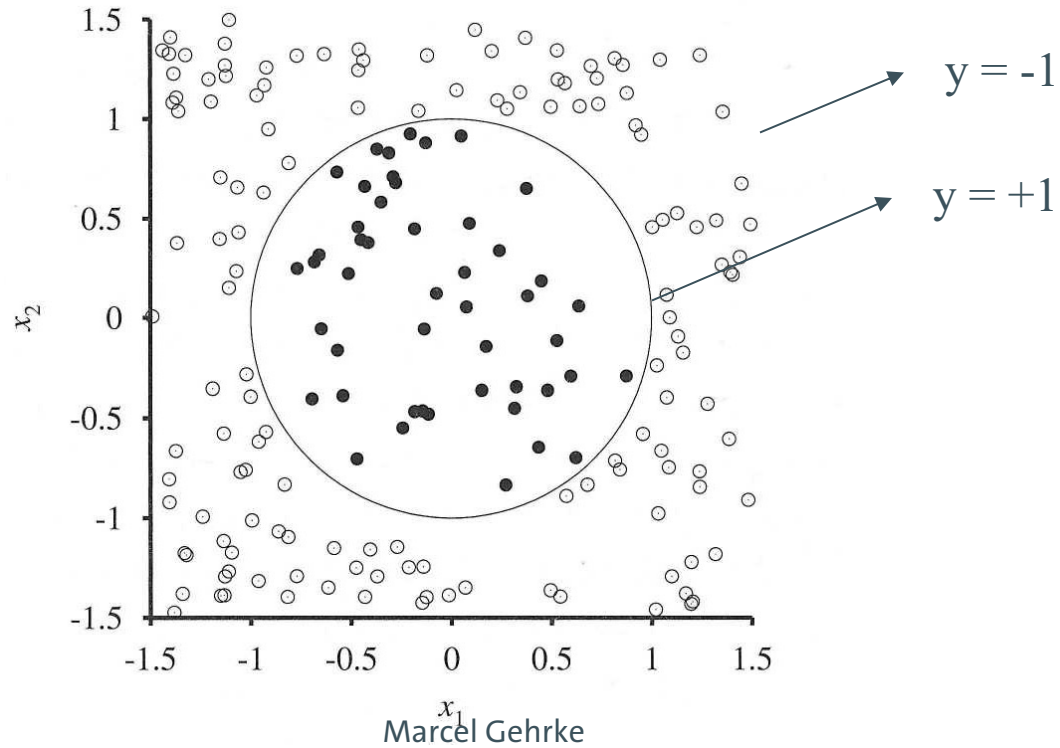
Computation of a separator via an optimisation problem

Formulation as problem not as procedure!

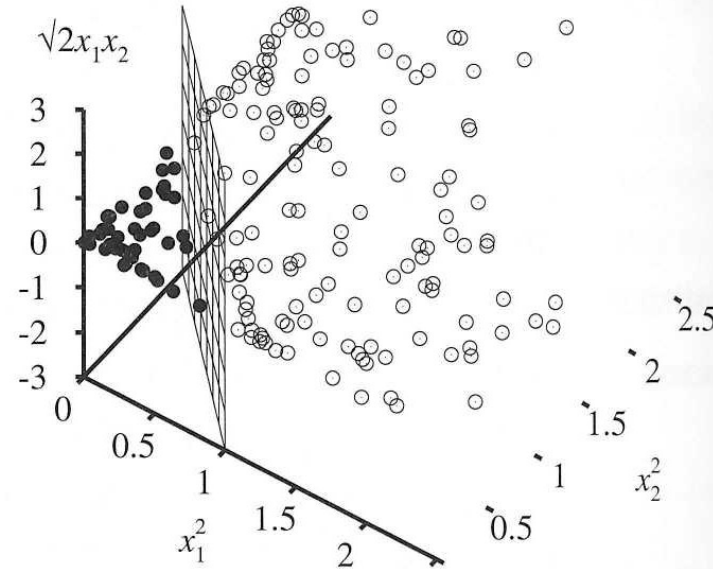
V. Vapnik, A. Chervonenkis, A note on one class of perceptrons. *Automation and Remote Control*, 25, 1964

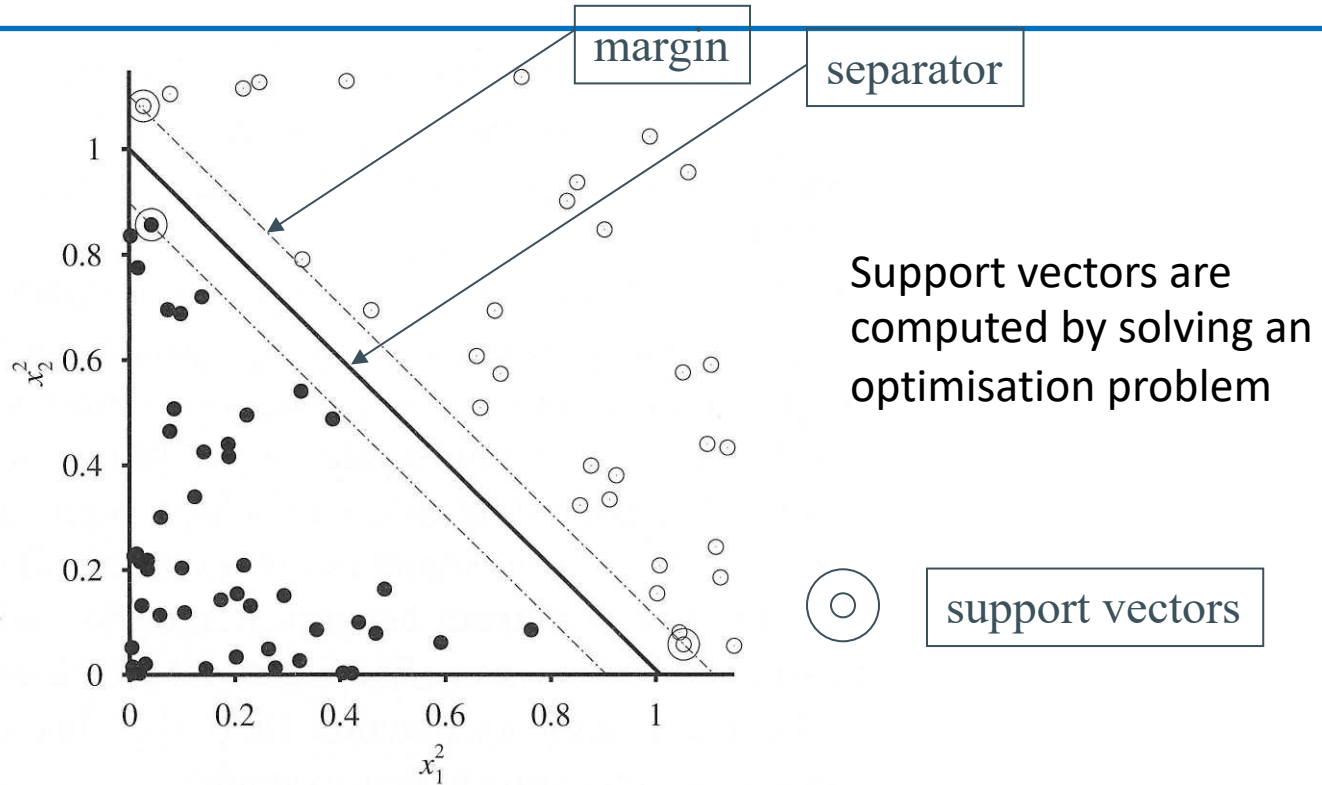
Boser, B. E.; Guyon, I. M.; Vapnik, V. N., A training algorithm for optimal margin classifiers. *Proceedings of the fifth annual workshop on Computational learning theory – COLT '92*. p. 144, 1992

Vapnik, V., Support-vector networks, *Machine Learning*. 20 (3): 273–297, 1995



$$(x_1^2, x_2^2, \sqrt{2x_1x_2})$$





Multi-class SVMs?

SVMs for multiple class labels: Combination of multiple SVMs

- One-versus-all
 - A decision boundary for each class between its "own" data points and all other data points
- One-versus-one
 - For every possible combination of two classes, there is a decision boundary

Development of Classifiers

How to automatically determine relevant features or combination of these feature?

- Will be covered (Gen-AI course)

How to dynamically adjust classifier?

- Without labeled data, i.e., data with known output result (ground truth)
 - Transductive learning

How to transfer a classifier to a new application?

- Trout vs. salmon?
 - Transfer learning

Supervised Learning

Version Space Learning

Performance of Classifiers

Mercedes C-Class as an example of “family cars”

- **Prediction:** Is car x a family car?
- **Knowledge extraction:**
What do people expect from a family car?

Output:

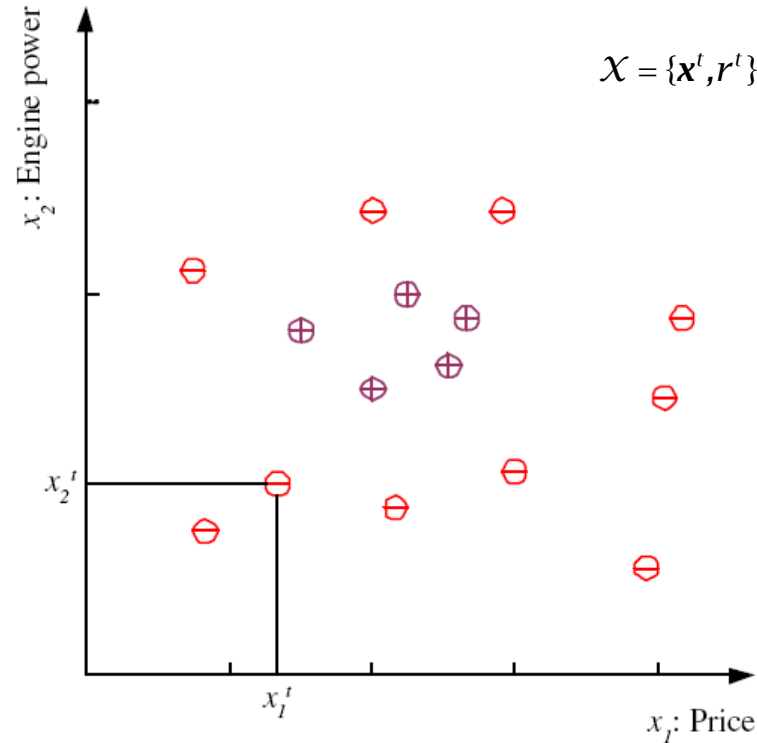
Positive (+) and negative (–) examples (ground truth)

Representation of the input:

x_1 : Price, x_2 : Engine power

Research question: How well does a particular classifier perform?

Trainings set \mathcal{X}

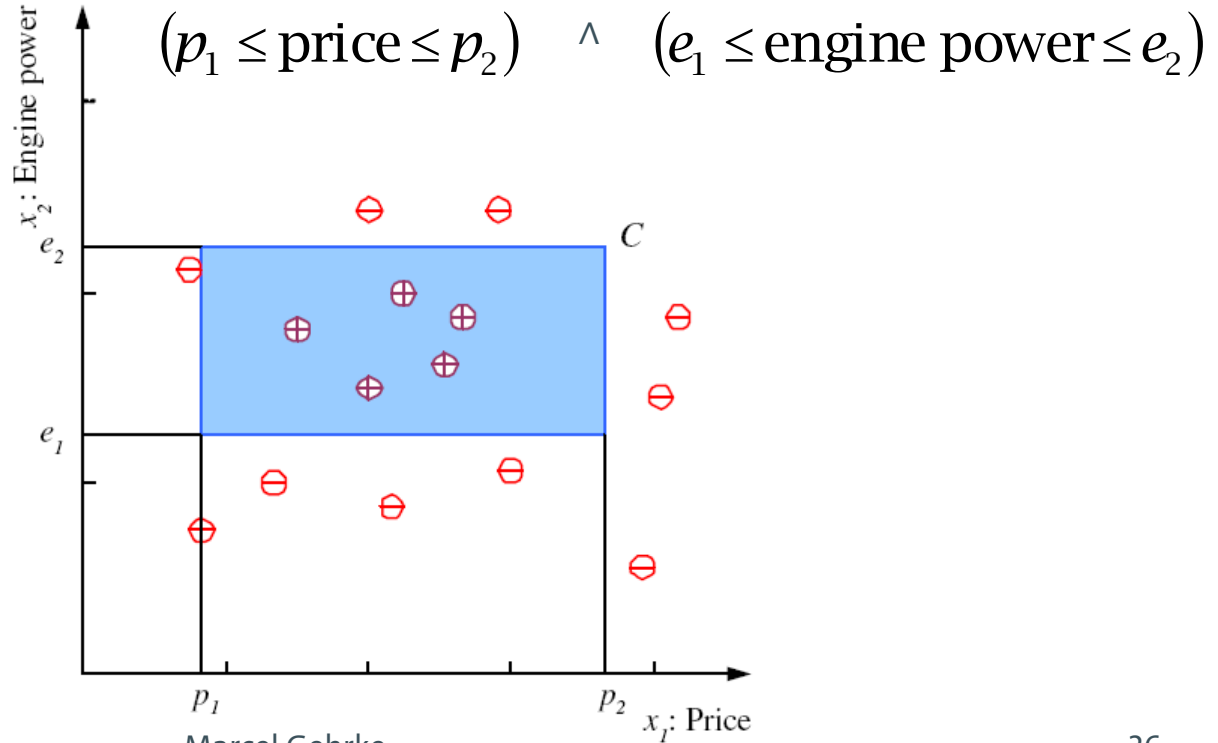


$$\mathcal{X} = \{\mathbf{x}^t, r^t\}_{t=1}^N$$

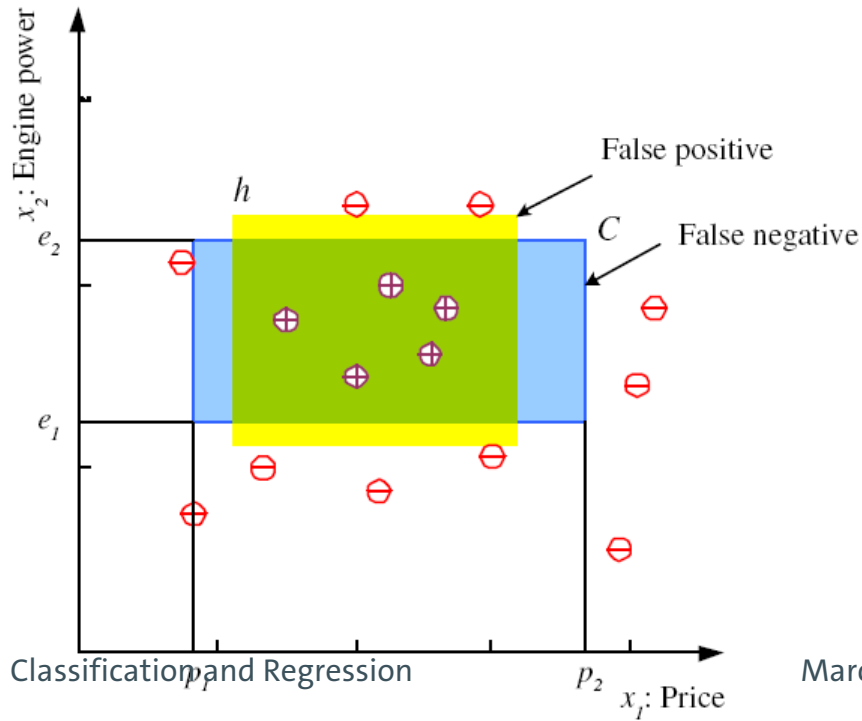
$$r = \begin{cases} 1 & \text{if } \mathbf{x} \text{ is positive} \\ 0 & \text{if } \mathbf{x} \text{ is negative} \end{cases}$$

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

Correct C-Class



Hypothesis class \mathcal{H} (e.g. $h \in \mathcal{H}$ in yellow)



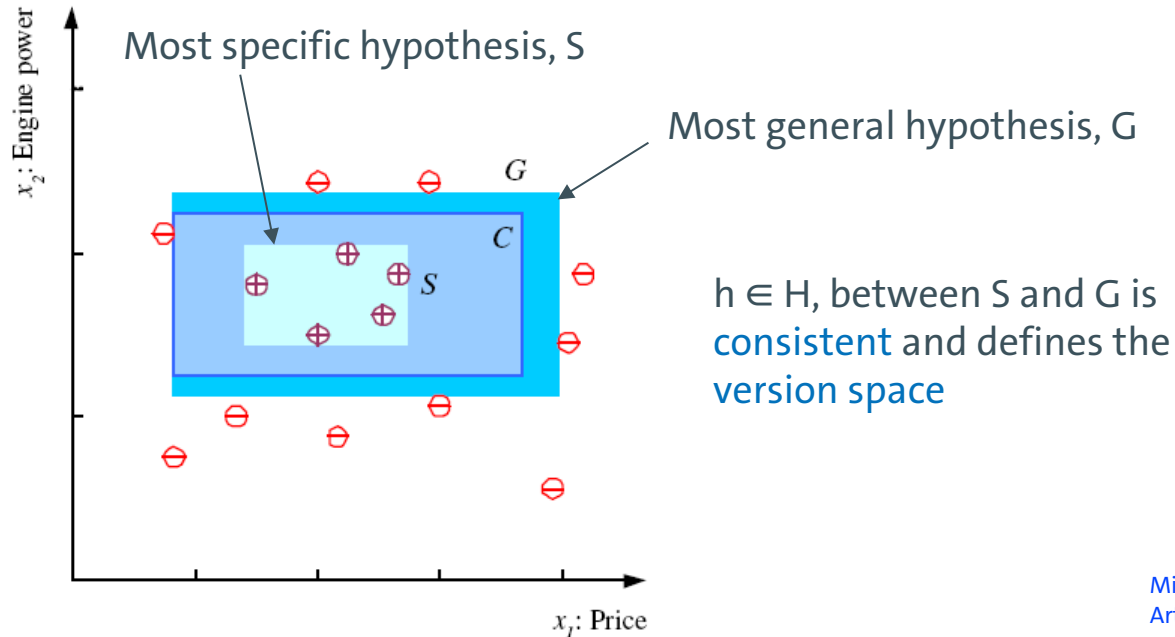
$$h(\mathbf{x}) = \begin{cases} 1 & \text{if } h \text{ classifies } \mathbf{x} \text{ as positive} \\ 0 & \text{if } h \text{ classifies } \mathbf{x} \text{ as negative} \end{cases}$$

Error of h on \mathcal{H}

$$E(h|\mathcal{X}) = (1/N) \sum_{t=1}^N (h(\mathbf{x}^t) \neq r^t)$$

$$(a \neq b) = 1, \text{ otherwise } 0$$

S, G, and the Version Space

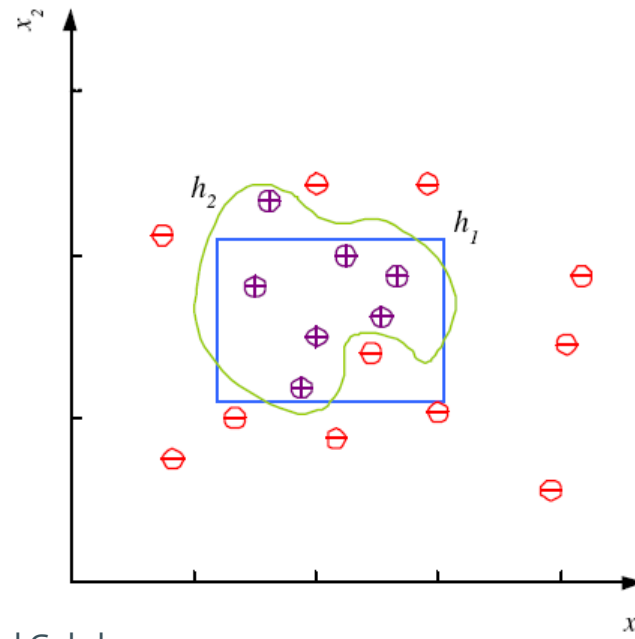


Mitchell, Tom M., *Generalization as search*.
Artificial Intelligence. 18 (2): 203–226, 1982

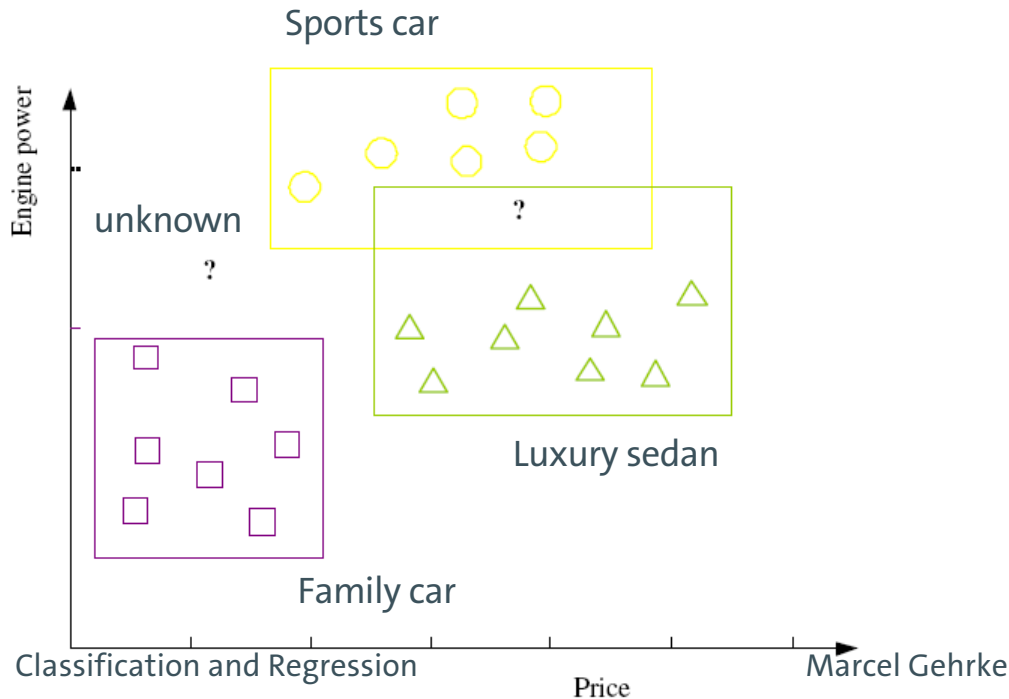
Noise and Model Complexity

Use simple a model:

- Easier to apply
(fewer calculation steps)
- Easier to train
(less data to store)
- Easier to explain
(easier to interpret)
- Better generalisation
(Occam's Razor)



Clustering: Different Classes, $C_i, i = 1, \dots, K$



$$\mathcal{X} = \{\mathbf{x}^t, r^t\}_{t=1}^N$$

$$r_i^t = \begin{cases} 1 & \text{if } \mathbf{x}^t \in C_i \\ 0 & \text{if } \mathbf{x}^t \in C_j, j \neq i \end{cases}$$

Train hypotheses

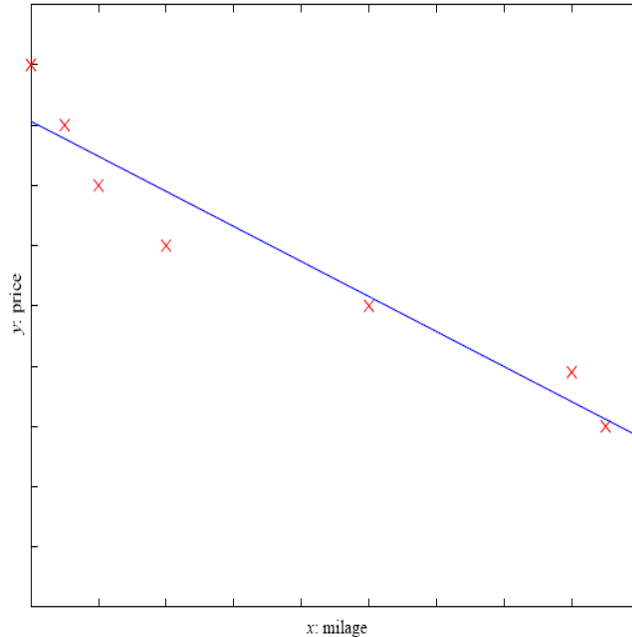
$h_i(\mathbf{x}), i = 1, \dots, K:$

$$h_i(\mathbf{x}^t) = \begin{cases} 1 & \text{if } \mathbf{x}^t \in C_i \\ 0 & \text{if } \mathbf{x}^t \in C_j, j \neq i \end{cases}$$

Supervised learning

Regression

Curve Fitting



Price of used cars

x : Mileage

y : Price

$\hat{y} = g(X | \theta)$: Hypothesis

Curve Fitting: Different Function Types

$$\mathcal{X} = \{x^t, r^t\}_{t=1}^N$$

$$r^t \in \mathfrak{R}$$

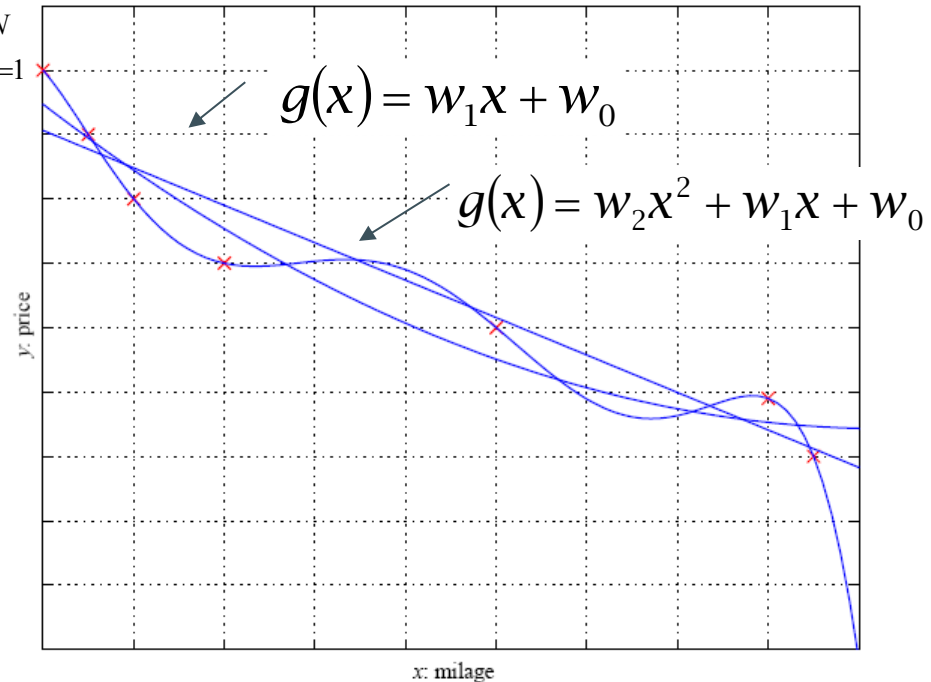
$$r^t = f(x^t)$$

Error function:
Mean squared error

$$E(g | \mathcal{X}) = \frac{1}{N} \sum_{t=1}^N [r^t - g(x^t)]^2$$

$$E(w_1, w_0 | \mathcal{X}) = \frac{1}{N} \sum_{t=1}^N [r^t - (w_1 x^t + w_0)]^2$$

Set partial derivatives of E w.r.t. w_1 and w_0 to 0
→ error minimised



Curve Fitting: Computation of w_0

$$E(w_1, w_0 | \mathcal{X}) = \frac{1}{N} \sum_{t=1}^N [r^t - (w_1 x^t + w_0)]^2$$

$$\frac{\partial E(w_1, w_0 | \mathcal{X})}{\partial w_0} = 0 \Rightarrow \frac{2}{N} \sum_{t=1}^N [r^t - (w_1 x^t + w_0)] = 0$$

$$\sum_{t=1}^N r^t = w_1 \sum_{t=1}^N x^t + N w_0$$

$$w_0 = \frac{1}{N} \sum_{t=1}^N r^t - w_1 \frac{1}{N} \sum_{t=1}^N x^t = \bar{r} - w_1 \bar{x}$$

Curve Fitting: Computation of w_1

$$E(w_1, w_0 | \mathcal{X}) = \frac{1}{N} \sum_{t=1}^N [r^t - (w_1 x^t + w_0)]^2$$

$$\frac{\partial E(w_1, w_0 | \mathcal{X})}{\partial w_1} = 0 \Rightarrow \frac{2}{N} \sum_{t=1}^N [r^t - (w_1 x^t + w_0)] x^t = 0$$

$$\sum_{t=1}^N r^t x^t = w_1 \sum_{t=1}^N (x^t)^2 + w_0 \sum_{t=1}^N x^t$$

$$\sum_{t=1}^N r^t x^t = w_1 \sum_{t=1}^N (x^t)^2 + (\bar{r} - w_1 \bar{x}) \sum_{t=1}^N x^t$$

Curve Fitting: Computation of w_1 cntd.

$$\begin{aligned}\sum_{t=1}^N r^t x^t &= w_1 \sum_{t=1}^N x^{t^2} + (\bar{r} - w_1 \bar{x}) \sum_{t=1}^N x^t \\ \sum_{t=1}^N r^t x^t &= w_1 \sum_{t=1}^N (x^t)^2 + \bar{r} \sum_{t=1}^N x^t - w_1 \bar{x} \sum_{t=1}^N x^t \\ \sum_{t=1}^N r^t x^t - \bar{r} \sum_{t=1}^N x^t &= w_1 \left(\sum_{t=1}^N (x^t)^2 - \bar{x} \sum_{t=1}^N x^t \right) \\ w_1 &= \frac{\sum_{t=1}^N r^t x^t - \bar{r} \sum_{t=1}^N x^t}{\sum_{t=1}^N (x^t)^2 - \bar{x} \sum_{t=1}^N x^t} = \frac{\sum_{t=1}^N r^t x^t - N \bar{r} \bar{x}}{\sum_{t=1}^N (x^t)^2 - N \bar{x}^2}\end{aligned}$$

Curve Fitting: Different Function Types

$$\mathcal{X} = \{x^t, r^t\}_{t=1}^N$$

$$r^t \in \mathfrak{R}$$

$$r^t = f(x^t)$$

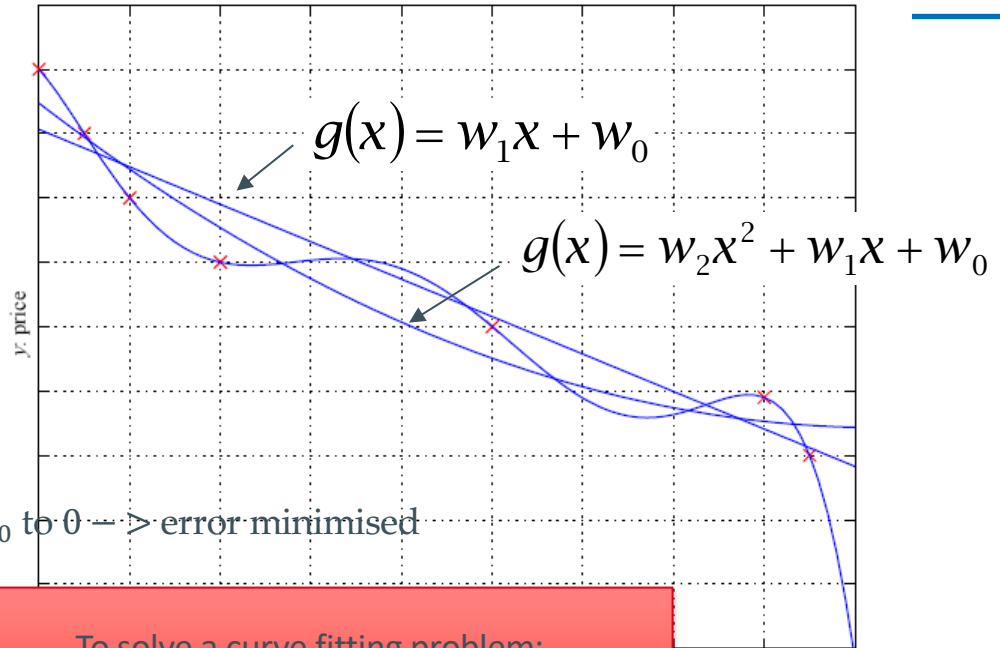
$$E(g | \mathcal{X}) = \frac{1}{N} \sum_{t=1}^N [r^t - g(x^t)]^2$$

$$E(w_1, w_0 | \mathcal{X}) = \frac{1}{N} \sum_{t=1}^N [r^t - (w_1 x^t + w_0)]^2$$

Set partial derivatives of E with respect to w_1 and w_0 to 0 \rightarrow error minimised

$$w_1 = \frac{\sum_t x^t r^t - \bar{x} r N}{\sum_t (x^t)^2 - N \bar{x}^2}$$

$$w_0 = \bar{r} - w_1 \bar{x}$$



To solve a curve fitting problem:
Regression

Regularisation

Approach so far: Least Squares Error $E(\lambda_1, \lambda_2, \dots, \lambda_m) := \sum_{i=1}^n \left(y_i - \sum_{j=1}^m \lambda_j f_j(x_i) \right)^2$

Introduction of punishment terms (Penalised Least Squares):

$PLS(\lambda_1, \lambda_2, \dots, \lambda_m) := E(\lambda_1, \lambda_2, \dots, \lambda_m) + \alpha \cdot \text{pen}(\lambda_1, \lambda_2, \dots, \lambda_m)$

- to minimise w.r.t. $\lambda = \lambda_1, \lambda_2, \dots, \lambda_m$
- $\text{pen}(\lambda)$ measures complexity of regression coefficients
- Smoothing parameter α measures influence of $\text{pen}(\lambda)$

Called: Regularisation

Regularisation in Regression

Ridge Regression

- $pen(\lambda_1, \lambda_2, \dots, \lambda_m) = \sum_{j=1}^m \lambda_j^2$
- Shrinking of coefficients *towards 0*

LASSO (Least Absolute Shrinkage and Selection Operator)

- $pen(\lambda_1, \lambda_2, \dots, \lambda_m) = \sum_{j=1}^m |\lambda_j|$
- Shrinking of coefficients *to 0*
- Used to construct models that are as simple as possible

In all regularisation procedures, the assumption is that the coefficients are comparable in terms of their values

Summary: Supervised Learning

Example: Curve fitting (Regression)¹

- Given data points, determine parameters, such that for given **x-values**, with usually minimal error, **the y-values can be estimated**
- Optimisation problem
Minimisation of a distance measure

Example : Classification

- Given data points each with classification value, determine classification value for data points without classification value (binary or multivalued classifier)
- Can be considered a special case of regression

¹ Regression: Decline or inference from function values to function

Supervised Learning

Generalisation

Model Selection & Generalisation

Learning can be seen as an optimisation problem:

- Compute model such that error function is minimised
- Computing parameters for the representation → "Parametric learning"

Learning usually is a **ill defined problem**

- Usually, the data is not suitable for finding a unique solution to the optimisation problem
- **Making assumptions:** Assumptions regarding H (inductive bias)
- **Can optimal hypothesis classes be determined automatically??**

Generalisation: How well does the model work on new data?

- Generalisation error

Overfitting: H more complex than C or f

Underfitting: H less complex than C or f

H = hypothesis class, C = classifier, f = regression function

The Triple Tradeoff

(Dietterich, 2003):

1. Complexity of \mathcal{H} : $c(\mathcal{H})$,
2. Training data size N ,
3. Generalisation error, E , w.r.t. new data

If $N \uparrow$, $E \downarrow$

If $c(\mathcal{H}) \uparrow$, first the generalisation error decreases $E \downarrow$, then starts to increase $E \uparrow$

Supervised Learning

Cross-validation

Cross-validation

To estimate the generalisation error, we need data that has not been used for training

Splitting of data:

- Trainings set (50%)
- Test set (e.g. for publication) (50%)

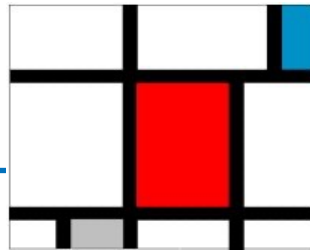
Resampling in case there is little data

Different Layers of Supervised Learning

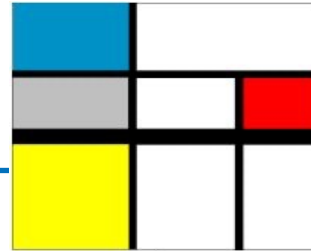
1. Model: $g(\mathbf{x} | \theta)$

2. Error function
(Loss function): $E(\theta | \mathcal{X}) = \sum_t L(r^t, g(\mathbf{x}^t | \theta))$

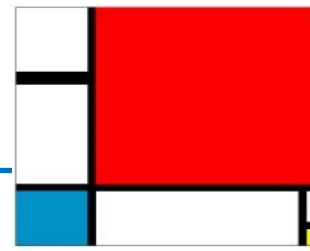
3. Optimisation
problem: $\theta^* = \arg \min_{\theta} E(\theta | \mathcal{X})$



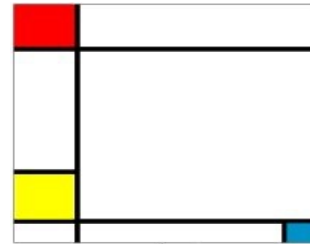
1



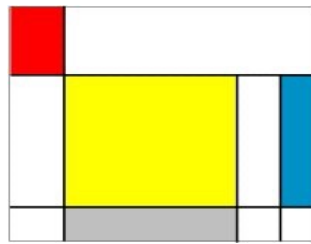
2



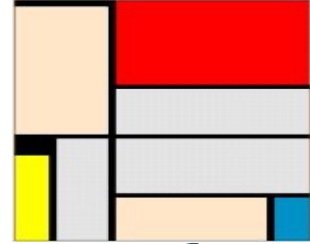
3



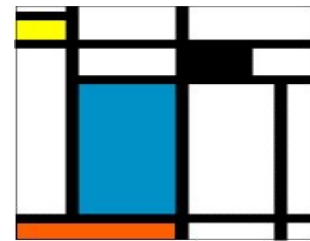
4



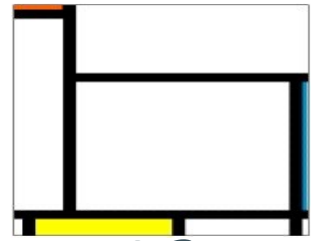
5



6



7



8 ?

Data in Tabular Form

Number	Lines	Line types	Rectangles	Colours	Mondrian?
1	6	1	10	4	No
2	4	2	8	5	No
3	5	2	7	4	Yes
4	5	1	8	4	Yes
5	5	1	10	5	No
6	6	1	8	6	Yes
7	7	1	14	5	No

Query:

Number	Lines	Line types	Rectangles	Colours	Mondrian?
8	7	2	9	4	

How can we Analyse the Data?

- Consider a column x with n values
- Determine the **mean value**: $\bar{x} = \frac{1}{n} \sum_{i=1}^n x_i$
- Large and small values can cancel each other out
- Consider mean deviation from mean value (**Variance**)
- Determine the variance*: $var = \frac{1}{n} \sum_{i=1}^n (\bar{x} - x_i)^2$
- The so-called **standard deviation**: $\sigma = \sqrt{var}$ is considered often

*: Will be refined later on.

Keep Data in Normalised Form

One way to normalise:

$$x_t' \equiv \frac{x_t - \bar{x}_t}{\sigma_t}$$

Average deviation from the mean

Normalised Training Data

Number	Lines	Line types	Rectangles	Colours	Mondrian?
1	0,632	-0,632	0,327	-1,021	No
2	-1,581	1,581	-0,588	0,408	No
3	-0,474	1,581	-1,046	-1,021	Yes
4	-0,474	-0,632	-0,588	-1,021	Yes
5	-0,474	-0,632	0,327	0,408	No
6	0,632	-0,632	-0,588	1,837	Yes
7	1,739	-0,632	2,157	0,408	No

$$d(\mathbf{x}_i, \mathbf{x}_j) = \sqrt{\sum_{t=1}^T [x_{it} - x_{jt}]^2}$$

Number	Lines	Line types	Rectangles	Colours	Mondrian?
8	1,739	1,581	-0,131	-1,021	

Normalised Training Data

Number	Lines	Line types	Rectangles	Colours	Mondrian?
1	0,632	-0,632	0,327	-1,021	No
2	-1,581	1,581	-0,588	0,408	No
3	-0,474	1,581	-1,046	-1,021	Yes
4	-0,474	-0,632	-0,588	-1,021	Yes
5	-0,474	-0,632	0,327	0,408	No
6	0,632	-0,632	-0,588	1,837	Yes
7	1,739	-0,632	2,157	0,408	No

$$\sqrt{(0 + 4,89 + 5,23 + 2,04)} = 3,489$$

Number	Lines	Line types	Rectangles	Colours	Mondrian?
8	1,739	1,581	-0,131	-1,021	

Distance of the Test Instance from the Training Data

Number	Distance to the test instance	Mondrian?
1	2,517	No
2	3,644	No
3	2,395	Yes
4	3,164	Yes
5	3,472	No
6	3,808	Yes
7	3,489	No

Classification

1-NN Yes

3-NN Yes

5-NN No

7-NN No

What do We use as Output for Real-valued Target Functions?

Mean of the k-nearest neighbors

Variant of kNN: Distance-weighted kNN

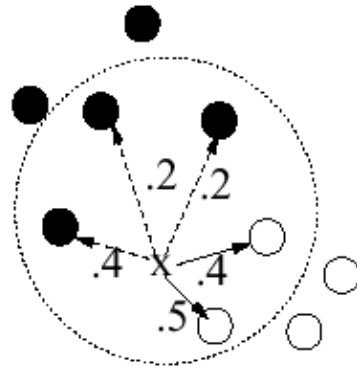
Closer neighbours have more influence

$$f(\mathbf{x}_q) := \frac{\sum_{i=1}^k w_i f(\mathbf{x}_i)}{\sum_{i=1}^k w_i} \quad \text{where } w_i = \frac{1}{d(\mathbf{x}_q, \mathbf{x}_i)^2}$$

Dudani, S.A. The distance-weighted k-nearest-neighbor rule. IEEE
Trans. Syst. Man Cybern., SMC-6:325–327, 1976

Variant of kNN: Distance-weighted kNN

kNN with a weighted voting system



kNN ($k = 5$)

Assign "white" to x because the weighted sum of "whites" is larger than the sum of "blacks".

Each neighbour gets a weight based on proximity

-> Then, in principle, we could use **all** training instances (= examples) instead of just **k**

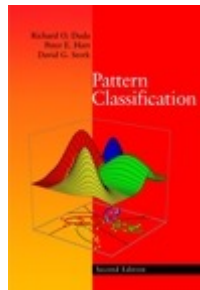
kNN: Summary

- Very simple approach, **non-parametric**
 - Classification (if necessary with threshold)
 - Regression (interpolation)
- Behaves well even if data cannot be easily separated
- Ranked 7 of the 10 most important data mining methods (in a publication in 2007)

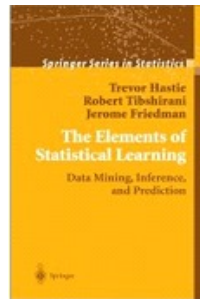


Literature (1)

Mitchell (1989). Machine Learning.
<http://www.cs.cmu.edu/~tom/mlbook.html>

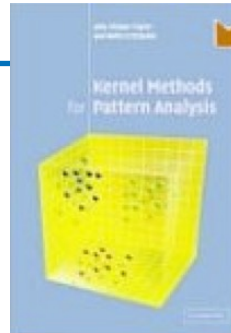


Duda, Hart, & Stork (2000). Pattern Classification.
<http://rii.ricoh.com/~stork/DHS.html>

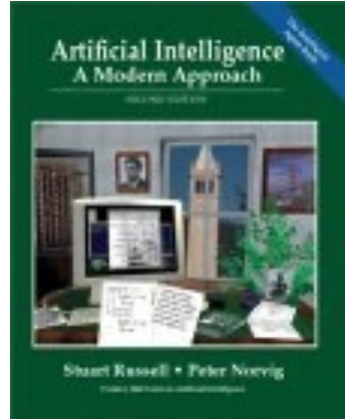


Hastie, Tibshirani, & Friedman (2001). The Elements of Statistical Learning. <http://www-stat.stanford.edu/~tibs/ElemStatLearn/>

Literature (2)

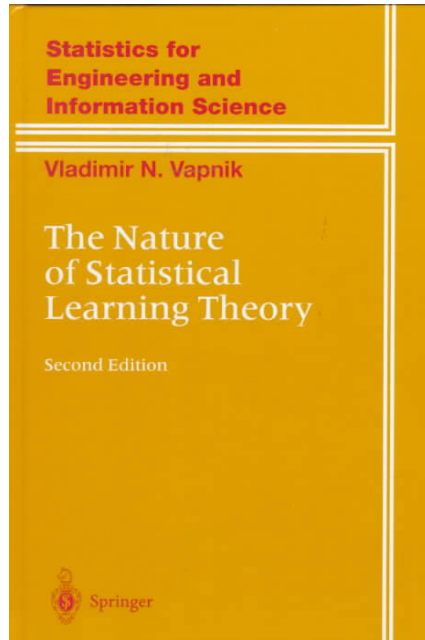


Shawe-Taylor & Cristianini. Kernel Methods for Pattern Analysis.
<http://www.kernel-methods.net/>



Russell & Norvig (2004). Artificial Intelligence.
<http://aima.cs.berkeley.edu/>

Original literature SVM



VAPNIK, Vladimir N.,. The Nature of Statistical Learning Theory. Springer-Verlag New York, Inc.,
1995