Machine Learning 2007: Lecture 13

Instructor: Tim van Erven (Tim.van.Erven@cwi.nl)
Website: `www.cwi.nl/~erven/teaching/0708/ml/`

December 12, 2007

Overview

- **Organisational Matters**
- Basic Concepts
- Methods
- Overfitting
- Probability Theory
- More Methods
- Student Discussion

Organisational Matters

Enrolling for the final exam:

- Someone reported getting an error when enrolling for the final exam on TISVu. Anyone else having troubles?

Today's Overview:

- I will ask things on the exam that are not in today's overview.
- The overview is only intended to give you a high-level view of the course, which should help you organise all the information.
- A few additional explanations of things that I think may have been unclear. For example, (the role of) random variables.

MDL lecture:

- Peter did his lecture on the blackboard, so there are no slides.

Organisational Matters

Enrolling for the final exam:

- Someone reported getting an error when enrolling for the final exam on TISVu. Anyone else having troubles?

Today's Overview:

- I will ask things on the exam that are not in today's overview.
- The overview is only intended to give you a high-level view of the course, which should help you organise all the information.
- A few additional explanations of things that I think may have been unclear. For example, (the role of) random variables.

MDL lecture:

- Peter did his lecture on the blackboard, so there are no slides.
- To help you study, I will make slides myself.
- These will be ready before Saturday evening.

Overview

- Organisational Matters
- **Basic Concepts**
- Methods
- Overfitting
- Probability Theory
- More Methods
- Student Discussion

Machine Learning Categories

Prediction: Given data $D = y_1, \ldots, y_n$, predict how the sequence continues with y_{n+1}.

Regression: Given data $D = \begin{pmatrix} y_1 \\ \mathbf{x}_1 \end{pmatrix}, \ldots, \begin{pmatrix} y_n \\ \mathbf{x}_n \end{pmatrix}$, learn to predict the value of the label y for any new feature vector $\mathbf{x}$. Typically y can take infinitely many values. Acceptable if your prediction is close to the correct y.

Classification: Given data $D = \begin{pmatrix} y_1 \\ \mathbf{x}_1 \end{pmatrix}, \ldots, \begin{pmatrix} y_n \\ \mathbf{x}_n \end{pmatrix}$, learn to predict the class label y for any new feature vector $\mathbf{x}$. Only finitely many categories. Your prediction is either correct or wrong.

- Not all machine learning problems fit into these categories.

Hypothesis Spaces and Models

- Hypothesis h: candidate description of regularity in the data
- Hypothesis space $\mathcal{H}$: set of hypotheses being considered
- Model $\mathcal{M}$: Hypothesis space that contains only probabilistic hypotheses

Example:

Suppose we use the following hypothesis space with deterministic hypotheses for prediction of binary outcomes $y_1, y_2, \ldots$

$$\mathcal{H} = \{h_1, h_2\} \qquad \begin{aligned} h_1: &\quad y_n = 0 \\ h_2: &\quad y_n = 1 \end{aligned}$$

If our hypotheses are not so sure about what is going to happen, then we should use probabilistic hypotheses:

$$\mathcal{M} = \{P_1, P_2\} \qquad \begin{aligned} P_1: &\quad P_1(y_n = 1) = 0.3 \\ P_2: &\quad P_2(y_n = 1) = 0.8 \end{aligned}$$

Overview

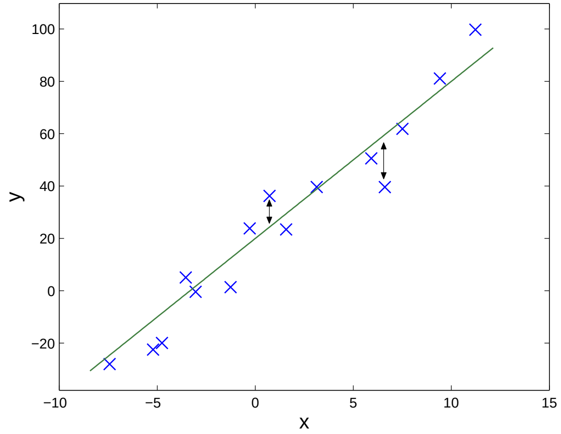
- Organisational Matters
- Basic Concepts
- **Methods**
- Overfitting
- Probability Theory
- More Methods
- Student Discussion

Least squares regression

- Method for regression
- Selects the hypothesis from $\mathcal{H}$ that minimizes the sum of squared errors on the data.
- Relies on representation bias to generalise.

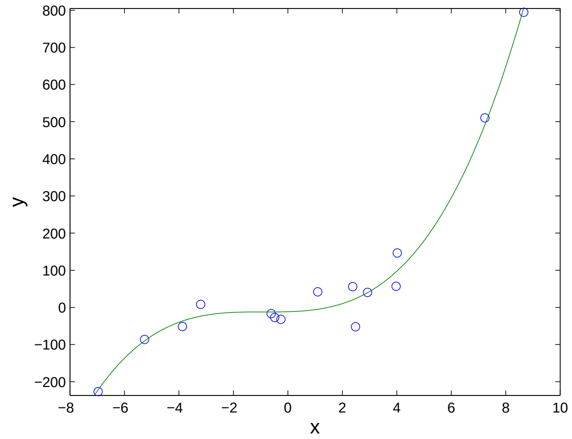
Linear regression:

- For d-dimensional $\mathbf{x}$
- $\mathcal{H} = \{w_0 + w_1 x_1 + \ldots + w_d x_d \mid \mathbf{w} \in \mathbb{R}^{d+1}\}$
- In example $d = 1$.



Polynomial regression with k-degree polynomials:

- For 1-dimensional $\mathbf{x}$
- $\mathcal{H} = \{w_0 + w_1 x_1 + w_2 x_1^2 + \ldots w_d x_1^k | \mathbf{w} \in \mathbb{R}^{k+1}\}$
- In example $k = 3$.



LIST-THEN-ELIMINATE *algorithm*

Method:

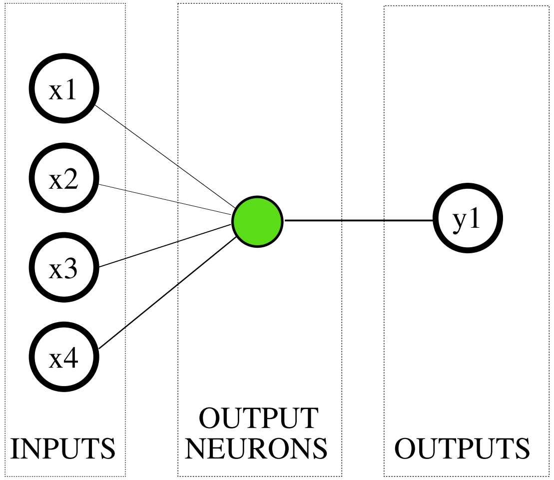
- Method for classification/concept learning (e.g. for EnjoySport data)
- Finds the set, VersionSpace, of hypotheses in $\mathcal{H}$ that are consistent with the training data.
- Can classify a new instance $\mathbf{x}$ if all hypotheses in VersionSpace agree on its classification.

Inductive bias:

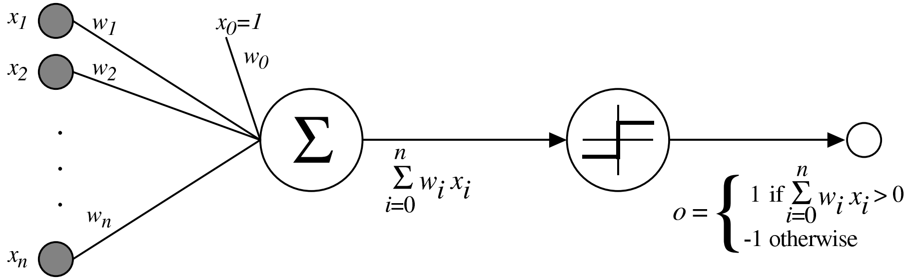
- Relies on representation bias to generalise:
- With $\mathcal{H}$ containing a list of constraints on attributes, it has a strong representation bias.
- With $\mathcal{H}$ containing all possible hypotheses it cannot generalise: bias is unavoidable!

The Perceptron

Simple Neural Network:



Mitchell's Drawing:

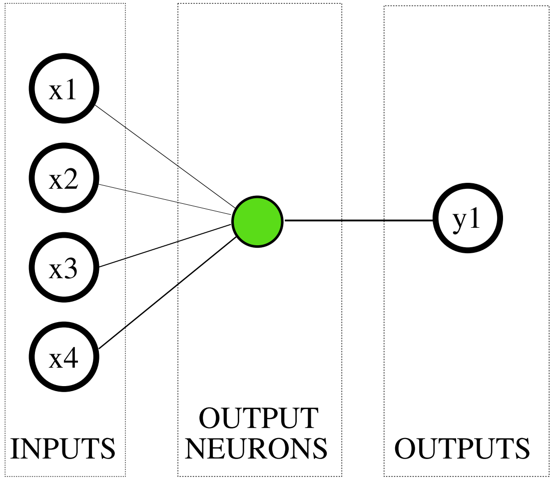


Equation:

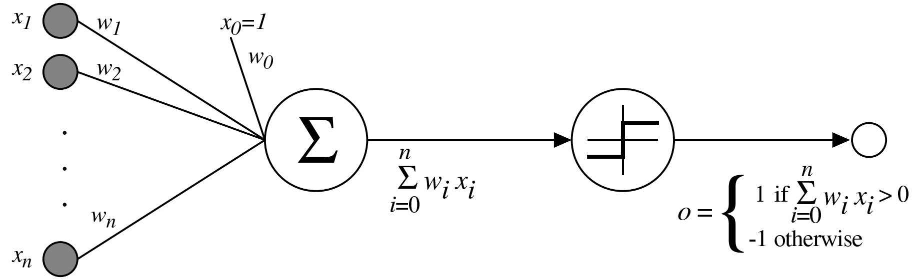
$$h_{\mathbf{w}}(\mathbf{x}) = \begin{cases} 1 & \text{if } w_0 + w_1 x_1 + \ldots w_d x_d > 0, \\ -1 & \text{otherwise.} \end{cases}$$

The Perceptron

Simple Neural Network:



Mitchell's Drawing:



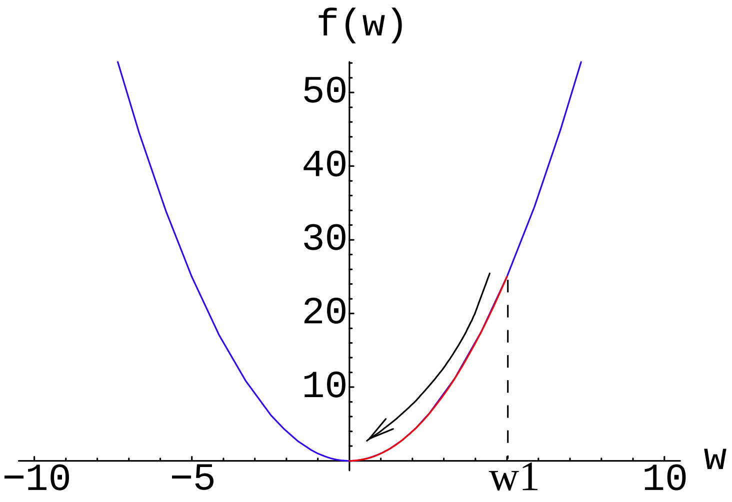
Equation:

$$h_{\mathbf{w}}(\mathbf{x}) = \begin{cases} 1 & \text{if } w_0 + w_1 x_1 + \ldots w_d x_d > 0, \\ -1 & \text{otherwise.} \end{cases}$$

- A perceptron does **classification**.
- It is a linear function with a threshold.
- Can learn functions with a linear decision boundary, but there are some functions that it can never represent (e.g. xor).

Gradient Descent

- Gradient descent is a method to find the minimum of a function: $\min_{\mathbf{w}} f(\mathbf{w})$.
- It works for convex functions and also some other functions, but not for functions that have local minima.
- It can be used, for example, to find the weights that minimize the error in least squares regression.



General Idea:

1. Pick some starting point w_1.
2. Keep taking small steps downhill: $f(w_1) > f(w_2) > f(w_3) > \ldots$
3. The negative derivative $-f'(w)$ points the way. (The gradient generalises the derivative in case $\mathbf{w}$ has dimension ≥ 2.)
4. Stop at the minimum. (Here $f'(w) = 0$.)

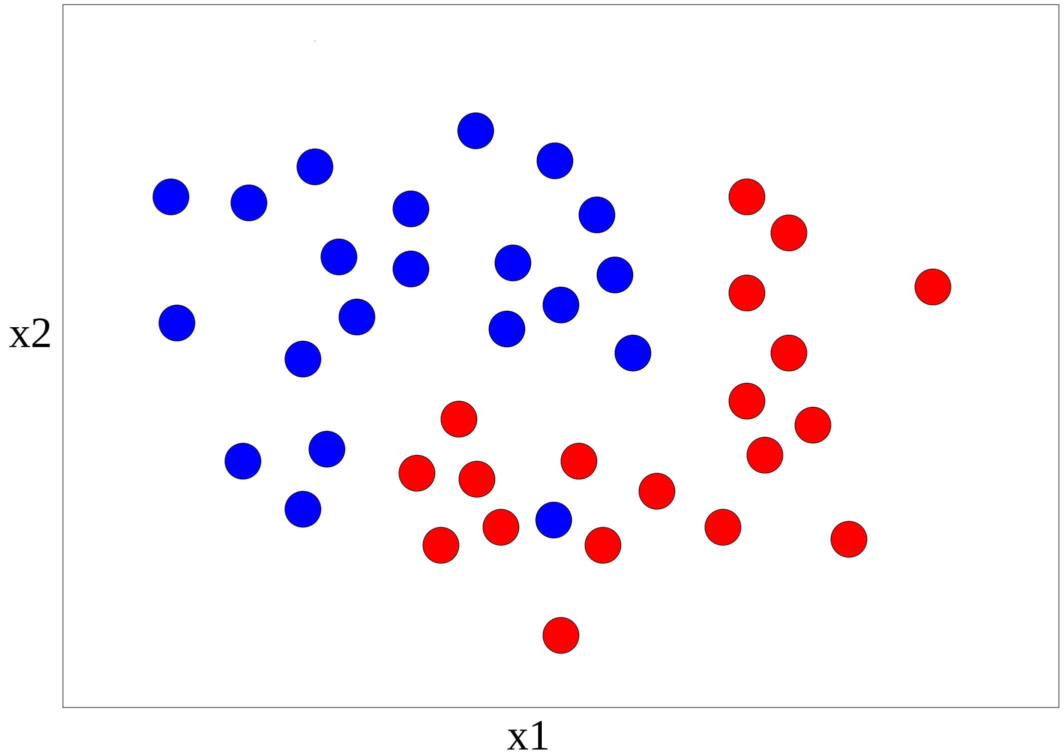
k-Nearest Neighbour

- Given train set D, k-**nearest neighbour** classifies a new vector $\mathbf{x}$ by voting among the k examples in D that are closest to $\mathbf{x}$.
- Distance is the essential ingredient.

k-Nearest Neighbour

- Given train set D, k-**nearest neighbour** classifies a new vector $\mathbf{x}$ by voting among the k examples in D that are closest to $\mathbf{x}$.
- Distance is the essential ingredient.

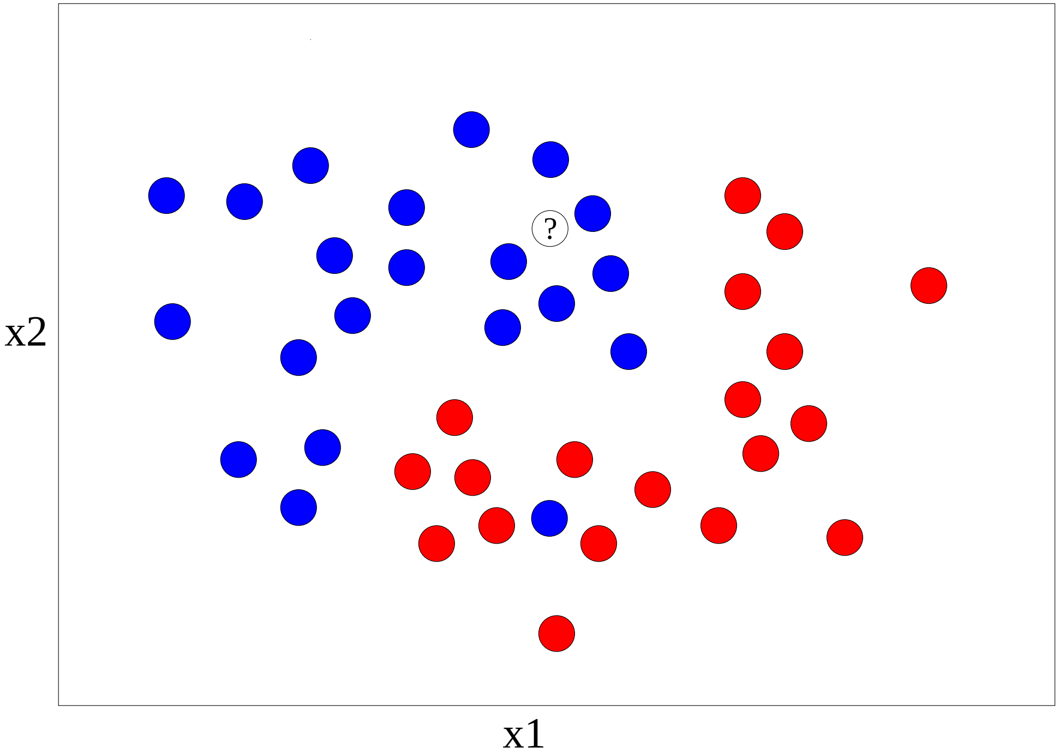
Example 5-nearest neighbour:



k-Nearest Neighbour

- Given train set D, k**-nearest neighbour** classifies a new vector $\mathbf{x}$ by voting among the k examples in D that are closest to $\mathbf{x}$.
- Distance is the essential ingredient.

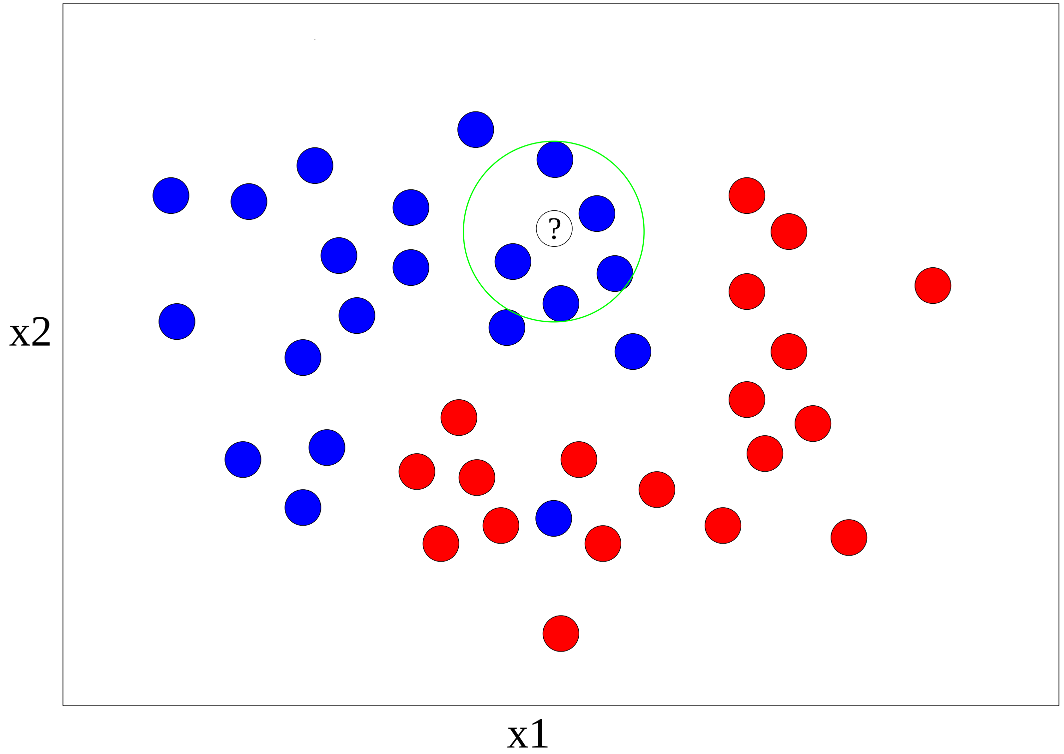
Example 5-nearest neighbour:



k-Nearest Neighbour

- Given train set D, k**-nearest neighbour** classifies a new vector $\mathbf{x}$ by voting among the k examples in D that are closest to $\mathbf{x}$.
- Distance is the essential ingredient.

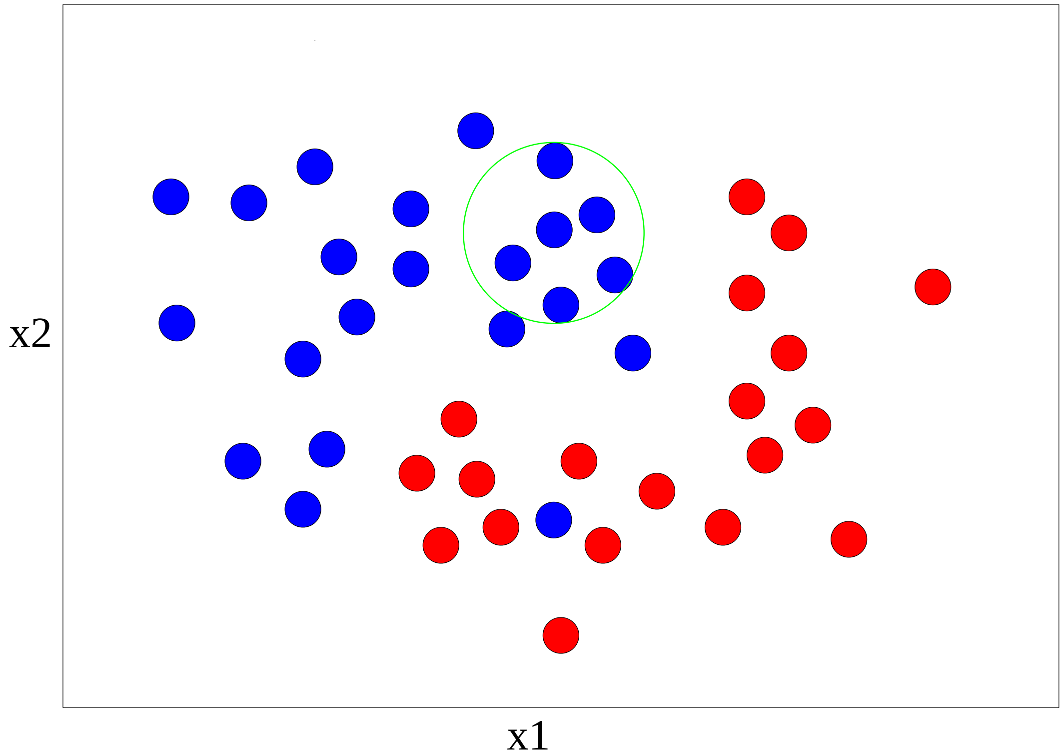
Example 5-nearest neighbour:



k-Nearest Neighbour

- Given train set D, k-**nearest neighbour** classifies a new vector $\mathbf{x}$ by voting among the k examples in D that are closest to $\mathbf{x}$.
- Distance is the essential ingredient.

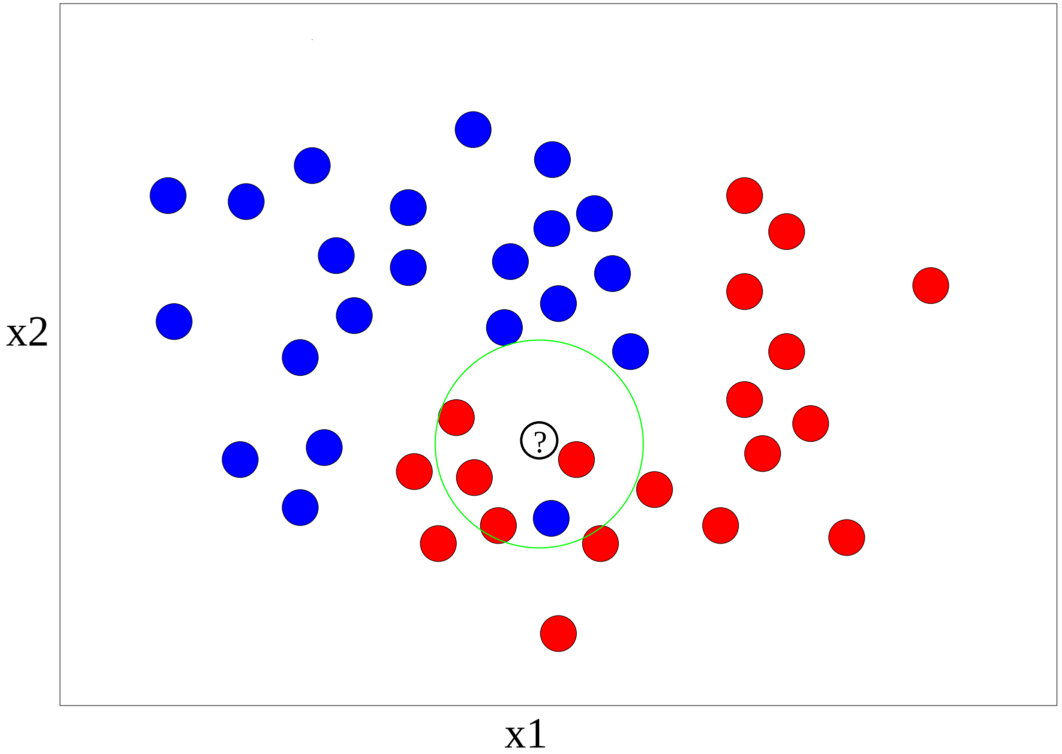
Example 5-nearest neighbour:



k-Nearest Neighbour

- Given train set D, k-**nearest neighbour** classifies a new vector $\mathbf{x}$ by voting among the k examples in D that are closest to $\mathbf{x}$.
- Distance is the essential ingredient.

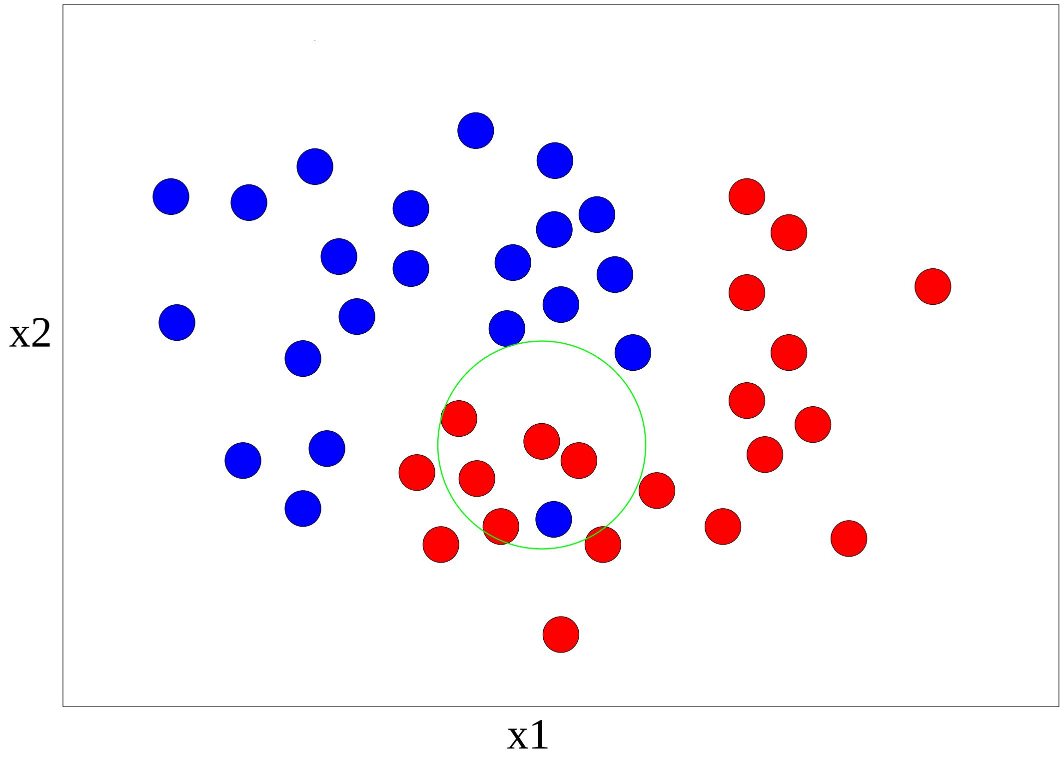
Example 5-nearest neighbour:



k-Nearest Neighbour

- Given train set D, k**-nearest neighbour** classifies a new vector $\mathbf{x}$ by voting among the k examples in D that are closest to $\mathbf{x}$.
- Distance is the essential ingredient.

Example 5-nearest neighbour:



The ID3 *Algorithm*

- Does classification by learning a decision tree from data.
- Decision trees can only handle attributes with a finite number of values.

Main ideas:

1. Start by selecting a root attribute for the tree.
2. Then construct the tree recursively by adding more and more attributes to it.
3. Attributes are chosen greedily based on their estimated mutual information with the class labels (information gain).
4. Stop growing the tree when it is consistent with all the data.

The ID3 *Algorithm*

- Does classification by learning a decision tree from data.
- Decision trees can only handle attributes with a finite number of values.

Main ideas:

1. Start by selecting a root attribute for the tree.
2. Then construct the tree recursively by adding more and more attributes to it.
3. Attributes are chosen greedily based on their estimated mutual information with the class labels (information gain).
4. Stop growing the tree when it is consistent with all the data.

Inductive bias:

- No representation bias
- Preference bias: prefers shorter trees with attributes that have a higher information gain closer to the root.
- After running ID3, post-pruning reduces overfitting.

Overview

- Organisational Matters
- Basic Concepts
- Methods
- **Overfitting**
- Probability Theory
- More Methods
- Student Discussion

Train, Test and Validation Sets

- **Train set:** data D_{train} used to train a machine learning algorithm (e.g. to select a hypothesis h)
- **Validation set:** used to estimate parameters.
- **Test set:** data D_{test} used to evaluate the performance of the algorithm (e.g. by evaluating Error(h,D_{test}))

Overfitting: One of the Main Problems in ML

Definition: A hypothesis $h \in \mathcal{H}$ **overfits** the train set if there exists a hypothesis $h' \in \mathcal{H}$ such that: h performs better than h' on the **train set**:

$$\text{Error}(h, D_{\text{train}}) < \text{Error}(h', D_{\text{train}}), \tag{1}$$

but h generalises less well than h': For a suffiently large **test set**,

$$\text{Error}(h, D_{\text{test}}) > \text{Error}(h', D_{\text{test}}). \tag{2}$$

Overfitting: One of the Main Problems in ML

Definition: A hypothesis $h \in \mathcal{H}$ **overfits** the train set if there exists a hypothesis $h' \in \mathcal{H}$ such that: h performs better than h' on the **train set**:

$$\text{Error}(h, D_{\text{train}}) < \text{Error}(h', D_{\text{train}}), \tag{1}$$

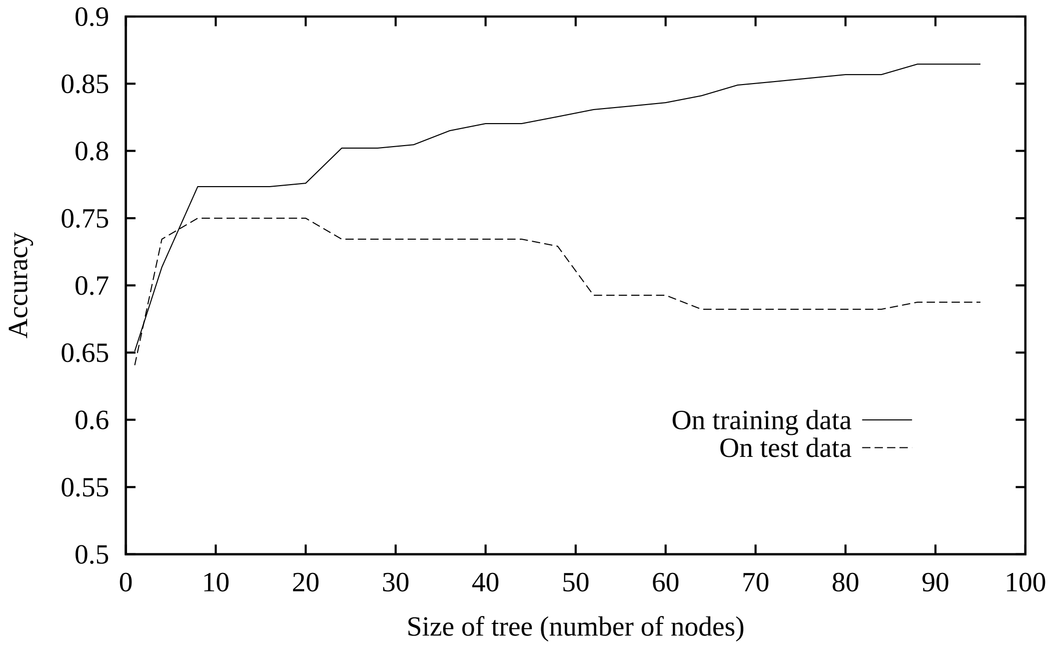
but h generalises less well than h': For a suffiently large **test set**,

$$\text{Error}(h, D_{\text{test}}) > \text{Error}(h', D_{\text{test}}). \tag{2}$$

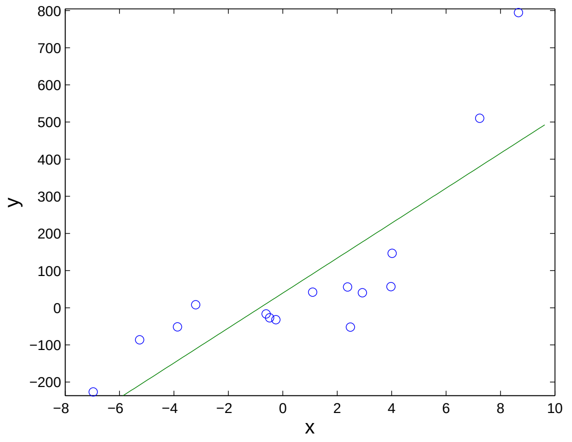
Example in prediction: If many students predict the outcome of a throw with a die, some will predict correctly, even though they have no special insight. They will not predict better than the others if we throw the die again.

Reason for overfitting: When selecting hypotheses (students) from a **large hypothesis space** (class), some will fit the train set well by coincidence.

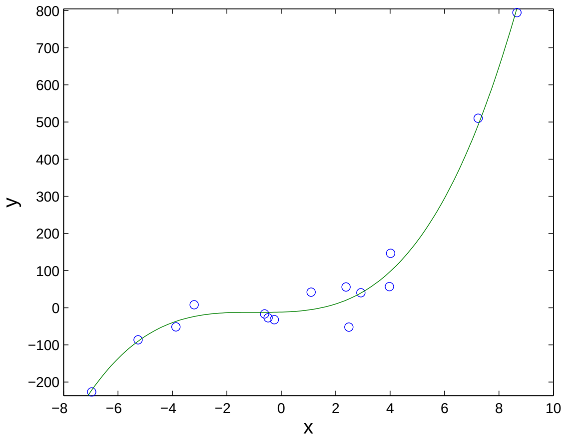
Overfitting with ID3:



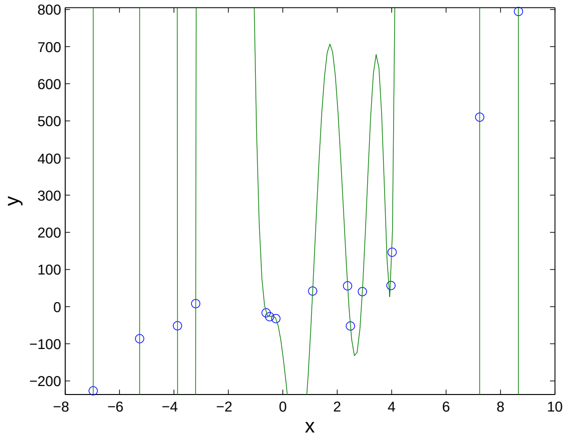
Overfitting in Least Squares with Polynomials::



Linear Function (Simple)



Third Degree Polynomial (Intermediate)



14th Degree polynomial (Complex)

Minimum Description Length Learning

MDL is a theory of learning based on the ideas:

- Learning is looking for structure/regularity in data.
- All regularity can be used to compress the data.

Properties:

- Automatic protection against overfitting:
- We need many bits to describe a “complex” hypothesis.

Examples:

- Grammar learning
- Regression with polynomials

Overview

- Organisational Matters
- Basic Concepts
- Methods
- Overfitting
- **Probability Theory**
- More Methods
- Student Discussion

Probability Distributions

To precisely define a probability distribution (e.g. a probabilistic hypothesis) P we need to specify:

1. Sample space Ω: Which outcomes are possible. E.g. $\Omega = \{a, b, c\}$.
2. Mass function p: What is the probability mass of each of the individual outcomes? E.g. $p(a) = 1/6$, $p(b) = 1/2$, $p(c) = 1/3$.
3. The masses may not be negative and have to sum up to 1.

Probability Distributions

To precisely define a probability distribution (e.g. a probabilistic hypothesis) P we need to specify:

1. Sample space Ω: Which outcomes are possible. E.g. $\Omega = \{a, b, c\}$.
2. Mass function p: What is the probability mass of each of the individual outcomes? E.g. $p(a) = 1/6$, $p(b) = 1/2$, $p(c) = 1/3$.
3. The masses may not be negative and have to sum up to 1.

Now what is the probability that we will either get an a or a c?

- Event A: More abstractly, what is the probability that we will observe an outcome in the set $A = \{a, c\}$?
- Any set of possible outcomes is an event, even the empty set or the set containing all outcomes. Hence $A \subseteq \Omega$.
- Probability distribution: $P(\{a, c\}) = p(a) + p(c) = 1/2$.
- In general, for any event A: $P(A) = \sum_{\omega \in A} p(\omega)$.

Conditional Probability

- Given that we will get an outcome in event B, what is the probability that the outcome is also in event A?

Conditional Probability

- Given that we will get an outcome in event B, what is the probability that the outcome is also in event A?
- For any two events $A, B \subseteq \Omega$ the conditional probability $P(A \mid B)$ of A given B is defined as:

$$P(A \mid B) = \frac{P(A \cap B)}{P(B)}$$

Example: $P(\{a, d\} \mid \{a, c\}) = \frac{P(\{a\})}{P(\{a,c\})}$

Conditional Probability

- Given that we will get an outcome in event B, what is the probability that the outcome is also in event A?
- For any two events $A, B \subseteq \Omega$ the conditional probability $P(A \mid B)$ of A given B is defined as:

$$P(A \mid B) = \frac{P(A \cap B)}{P(B)}$$

Example: $P(\{a, d\} \mid \{a, c\}) = \frac{P(\{a\})}{P(\{a,c\})}$

Bayes' rule:

- $P(A \mid B) = \frac{P(B|A)P(A)}{P(B)}$
- Bayes' rule is often useful when we want to compute $P(A \mid B)$, but only know $P(B \mid A)$
- (and $P(A)$ and $P(B)$, which are often easier because they only concern a single event).

Independent Events

Independent Events:

- Two events A and B are independent if the probability of one of them doesn't change when we condition on the other: $P(A \mid B) = P(A)$.
- Or equivalently: $P(A \cap B) = P(A)P(B)$

Conditional Independence:

- Two events A and B are conditionally independent given event C if $P(A \cap B \mid C) = P(A \mid C)P(B \mid C)$.

Overview

- Organisational Matters
- Basic Concepts
- Methods
- Overfitting
- Probability Theory
- **More Methods**
- Student Discussion

Naive Bayes:

Classification:

- Naive Bayes is a method for classification.
- It assumes that the outcomes $(y, \mathbf{x})^\top$ that we get are distributed according to some unknown distribution P.
- To classify $\mathbf{x}$ it selects the y with highest conditional probability: $\arg\max_y P(Y = y \mid X = \mathbf{x})$.

Estimating P:

- To maximize $P(Y = y \mid X = \mathbf{x})$ it applies Bayes' rule.
- And it assumes that the attributes of $\mathbf{x}$ are conditionally independent given the class label y.
- Then it estimates the required probabilities from the training data.

Random Variables:

What does the expression $P(Y = y' \mid X = \mathbf{x}')$ mean?

- Here Y and X are random variables.
- $Y = y'$ defines the event "all possible pairs $(y, \mathbf{x})^\top$ such that $y = y'$"
- $X = \mathbf{x}'$ defines the event "all possible pairs $(y, \mathbf{x})^\top$ such that $\mathbf{x} = \mathbf{x}'$"

Example:

- Suppose y can take the possible values 0 and 1.
- And x can take the possible values 10, 20, 30.
- Then $Y = 0$ defines the event $\{(0, 10)^\top, (0, 20)^\top, (0, 30)^\top\}$.
- And $X = 30$ defines the event $\{(0, 30)^\top, (1, 30)^\top\}$.
- If $\mathbf{x}$ is, say, 2-dimensional, then we might have separate random variables X_1 and X_2 to talk about the values of x_1 and x_2.

Maximum Likelihood Parameter Estimation

Training data and model:

$$D = \begin{array}{|c|c|c|c|c|c|c|c|} \hline y_1 & y_2 & y_3 & y_4 & y_5 & y_6 & y_7 & y_8 \\ \hline 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 \\ \hline \end{array}$$

$$\mathcal{M} = \left\{ P_\theta \mid \theta \in \left\{ \frac{1}{4}, \frac{1}{2}, \frac{3}{4} \right\} \right\}$$ where $P_\theta(y_n = 1) = \theta$.

Likelihood:

θ	$1/4$	$1/2$	$3/4$
$P_\theta(D)$	$(1/4)^6(3/4)^2$ $= 9/65536$	$(1/2)^8$ $= 256/65536$	$(3/4)^6(1/4)^2$ $= 729/65536$

Maximum Likelihood Parameter Estimation:

$$\hat{\theta} = \arg\max_\theta P_\theta(D) = 3/4$$

Bayesian Learning

- Given some model with parameter θ, construct a **single** distribution P_{Bayes} on both data D and the parameter θ.
- We have to introduce a prior $\pi(\theta)$, which determines the relative importance of each θ.
- Now we can compute the
  - posterior distribution over parameters given the training data: $\pi(\theta = 3/4 \mid D)$;
  - find the MAP hypothesis: $\theta_{\text{MAP}} = \arg\max_\theta \pi(\theta \mid D)$
  - predictive distribution over the next outcome given the training data: $P_{\text{Bayes}}(y_{n+1} = 1 \mid D) = \sum_\theta P_\theta(y_{n+1})\pi(\theta \mid D)$.

Remarks:

- Automatically protects against overfitting.
- Widely used (much better known than MDL).

Implications of Machine Learning

- Price of car/health insurance made dependent on the learned probability of getting an accident/getting ill, based on your genes.
- Learning a persons preferences from search queries to improve search results. I am sure that Google can find out:
  - your personal interests
  - your (lack of) religion
  - your sexual preferences
  - your (controversial?) political views
- Learning racist characteristics. How do you know that your learning method is racist? It may very subtly be using features that are correlated to race.
- Ability to get a job with the government dependent on fit to the profile of a terrorist. Think about overfitting like in the dice prediction game.

References

- A.N. Shiryaev, “Probability”, Second Edition, 1996
- P. Grünwald, “The Minimum Description Length Principle”, 2007
- T.M. Cover and J.A. Thomas, “Elements of Information Theory,” 1991
- T.M. Mitchell, “Machine Learning”, McGraw-Hill, 1997