Taal- en spraaktechnologie

Sophia Katrenko

Utrecht University, the Netherlands

June 15, 2012

Outline

1. **Machine learning: what is it?**
   - Learning approaches
   - Evaluation measures
2. **Methods**
   - Naive Bayes (incl. a toy example)
   - Decision tree classifier
   - Memory-based (lazy) learning

Focus

This part of the course focuses on

- meaning representation
- lexical semantics
- distributional similarity
- intro to machine learning
- word sense disambiguation
- information extraction

Machine learning notions

Main learning notions (1)

- Learning involves three components: task **T**, experience **E**, and performance measure **P**.
- The goal of learning is to perform well w.r.t. some performance measure **P** on task **T** given some past experience or observations.
- Consider, for example, weather prediction given previous observations.

Main learning notions (1)

- Learning involves three components: task **T**, experience **E**, and performance measure **P**.
- The goal of learning is to perform well w.r.t. some performance measure **P** on task **T** given some past experience or observations.
- Consider, for example, weather prediction given previous observations.

Main learning notions (1)

- Learning involves three components: task **T**, experience **E**, and performance measure **P**.
- The goal of learning is to perform well w.r.t. some performance measure **P** on task **T** given some past experience or observations.
- Consider, for example, weather prediction given previous observations.

Main learning notions (2)

But

- What is experience? Is it direct or implicit?
- Do given observations reflect the task/goal?
- Does the number of observations matter? What about noisy data?

Main learning notions (2)

But

- What is experience? Is it direct or implicit?
- Do given observations reflect the task/goal?
- Does the number of observations matter? What about noisy data?

Main learning notions (2)

But

- What is experience? Is it direct or implicit?
- Do given observations reflect the task/goal?
- Does the number of observations matter? What about noisy data?

Main learning notions (3)

Today, we consider

1. Learning tasks: regression and classification
2. Learning types: supervised, unsupervised, and semi-supervised
3. Evaluation measures: accuracy, precision, recall and F-score

Main learning notions (4)

- Formally, let observations (training data) (X, Y) be defined as $(X, Y) \in \mathcal{X} \times \mathcal{Y}$ on the input space $\mathcal{X}$ and the output space $\mathcal{Y}$.
- Pairs (X, Y) are random variables distributed according to the unknown distribution D.
- The observed data points we denote by $(\mathbf{x_i}, y_i)$ and say that they are independently and identically distributed according to D.
- The goal is to construct a hypothesis h such that for any instance from the input space $\mathcal{X}$ it predicts its label from the output space $\mathcal{Y}$, i.e. $h : \mathcal{X} \rightarrow \mathcal{Y}$.

Main learning notions (4)

- Formally, let observations (training data) (X, Y) be defined as $(X, Y) \in \mathcal{X} \times \mathcal{Y}$ on the input space $\mathcal{X}$ and the output space $\mathcal{Y}$.
- Pairs (X, Y) are random variables distributed according to the unknown distribution D.
- The observed data points we denote by $(\mathbf{x_i}, y_i)$ and say that they are independently and identically distributed according to D.
- The goal is to construct a hypothesis h such that for any instance from the input space $\mathcal{X}$ it predicts its label from the output space $\mathcal{Y}$, i.e. $h : \mathcal{X} \rightarrow \mathcal{Y}$.

Main learning notions (4)

- Formally, let observations (training data) (X, Y) be defined as $(X, Y) \in \mathcal{X} \times \mathcal{Y}$ on the input space $\mathcal{X}$ and the output space $\mathcal{Y}$.
- Pairs (X, Y) are random variables distributed according to the unknown distribution D.
- The observed data points we denote by $(\mathbf{x_i}, y_i)$ and say that they are independently and identically distributed according to D.
- The goal is to construct a hypothesis h such that for any instance from the input space $\mathcal{X}$ it predicts its label from the output space $\mathcal{Y}$, i.e. $h : \mathcal{X} \to \mathcal{Y}$.

Main learning notions (4)

- Formally, let observations (training data) (X, Y) be defined as $(X, Y) \in \mathcal{X} \times \mathcal{Y}$ on the input space $\mathcal{X}$ and the output space $\mathcal{Y}$.
- Pairs (X, Y) are random variables distributed according to the unknown distribution D.
- The observed data points we denote by $(\mathbf{x_i}, y_i)$ and say that they are independently and identically distributed according to D.
- The goal is to construct a hypothesis h such that for any instance from the input space $\mathcal{X}$ it predicts its label from the output space $\mathcal{Y}$, i.e. $h : \mathcal{X} \rightarrow \mathcal{Y}$.

Main learning notions (5)

- Let also every example $\mathbf{x_i} \in \mathcal{X}, i = 1, \ldots, n$ be represented by a fixed number of features, $\mathbf{x_i} = (x_{i1}, \ldots, x_{ik})$.
- For instance, for the task *'is a given word a noun?'*, $\mathbf{X}$ is a collection of words, $\mathbf{Y} = \{0, 1\}$, and training examples are of the form $\{w, y\}$, where $w \in \mathbf{X}$ and $y \in \mathbf{Y}$, as in $\{Utrecht, 1\}$, $\{in, 0\}$, ...
- For PoS tagging, $|\mathbf{Y}| > 2$ and is represented by tags NNS, IN, NN, and others.

Main learning notions (5)

- Let also every example $\mathbf{x_i} \in \mathcal{X}, i = 1, \ldots, n$ be represented by a fixed number of features, $\mathbf{x_i} = (x_{i1}, \ldots, x_{ik})$.
- For instance, for the task *'is a given word a noun?'*, $\mathbf{X}$ is a collection of words, $\mathbf{Y} = \{0, 1\}$, and training examples are of the form $\{w, y\}$, where $w \in \mathbf{X}$ and $y \in \mathbf{Y}$, as in $\{Utrecht, 1\}$, $\{in, 0\}$, ...
- For PoS tagging, $|\mathbf{Y}| > 2$ and is represented by tags NNS, IN, NN, and others.

Main learning notions (5)

- Let also every example $\mathbf{x_i} \in \mathcal{X}, i = 1, \ldots, n$ be represented by a fixed number of features, $\mathbf{x_i} = (x_{i1}, \ldots, x_{ik})$.
- For instance, for the task *'is a given word a noun?'*, $\mathbf{X}$ is a collection of words, $\mathbf{Y} = \{0, 1\}$, and training examples are of the form $\{w, y\}$, where $w \in \mathbf{X}$ and $y \in \mathbf{Y}$, as in $\{Utrecht, 1\}$, $\{in, 0\}$, ...
- For PoS tagging, $|\mathbf{Y}| > 2$ and is represented by tags NNS, IN, NN, and others.

Main learning notions (6)

- Classification: For $h : \mathcal{X} \rightarrow \mathcal{Y}$, if $\mathcal{Y}$ is discrete (set of categories). If $\mathcal{Y} = \{+1, -1\}$, then it is a binary classification task
- Regression: if output is continuous (a real number).

Main learning notions (6)

- Classification: For $h : \mathcal{X} \rightarrow \mathcal{Y}$, if $\mathcal{Y}$ is discrete (set of categories). If $\mathcal{Y} = \{+1, -1\}$, then it is a binary classification task
- Regression: if output is continuous (a real number).

Main learning notions (7)

- *Supervised* learning requires a training set (as described above), which is used by an algorithm to produce a function (hypothesis).
- *Unsupervised* learning uses no labeled data, and its goal is to reveal hidden structure in data.
- *Semi-supervised* learning takes as input both labeled (small amount) and unlabeled data.
- In the *active* learning scenario, a learning algorithm is querying a human expert for true labels of the examples it selects according to some criteria (i.e., an example the algorithm is not certain about).

Main learning notions (7)

- *Supervised* learning requires a training set (as described above), which is used by an algorithm to produce a function (hypothesis).
- *Unsupervised* learning uses no labeled data, and its goal is to reveal hidden structure in data.
- *Semi-supervised* learning takes as input both labeled (small amount) and unlabeled data.
- In the *active* learning scenario, a learning algorithm is querying a human expert for true labels of the examples it selects according to some criteria (i.e., an example the algorithm is not certain about).

Main learning notions (7)

- *Supervised* learning requires a training set (as described above), which is used by an algorithm to produce a function (hypothesis).
- *Unsupervised* learning uses no labeled data, and its goal is to reveal hidden structure in data.
- *Semi-supervised* learning takes as input both labeled (small amount) and unlabeled data.
- In the *active* learning scenario, a learning algorithm is querying a human expert for true labels of the examples it selects according to some criteria (i.e., an example the algorithm is not certain about).

Main learning notions (7)

- *Supervised* learning requires a training set (as described above), which is used by an algorithm to produce a function (hypothesis).
- *Unsupervised* learning uses no labeled data, and its goal is to reveal hidden structure in data.
- *Semi-supervised* learning takes as input both labeled (small amount) and unlabeled data.
- In the *active* learning scenario, a learning algorithm is querying a human expert for true labels of the examples it selects according to some criteria (i.e., an example the algorithm is not certain about).

Main learning notions (8): examples

How does this relate to natural language processing?

- Most research in NLP (at least initially) has concerned supervised learning: parsing (treebanks for training available), named entity recognition systems, text categorization, others.
- It has shifted to semi-supervised learning because of the cost of human labour (e.g., for parsing Steedman'02).
- Unsupervised learning is used when clustering words/documents based on their similarity.
- Active learning is less studied, but is becoming more popular in the NLP community (e.g., text annotation by Tomanek et al.'09).

Main learning notions (8): examples

How does this relate to natural language processing?

- Most research in NLP (at least initially) has concerned supervised learning: parsing (treebanks for training available), named entity recognition systems, text categorization, others.
- It has shifted to semi-supervised learning because of the cost of human labour (e.g., for parsing Steedman'02).
- Unsupervised learning is used when clustering words/documents based on their similarity.
- Active learning is less studied, but is becoming more popular in the NLP community (e.g., text annotation by Tomanek et al.'09).

Main learning notions (8): examples

How does this relate to natural language processing?

- Most research in NLP (at least initially) has concerned supervised learning: parsing (treebanks for training available), named entity recognition systems, text categorization, others.
- It has shifted to semi-supervised learning because of the cost of human labour (e.g., for parsing Steedman'02).
- Unsupervised learning is used when clustering words/documents based on their similarity.
- Active learning is less studied, but is becoming more popular in the NLP community (e.g., text annotation by Tomanek et al.'09).

Main learning notions (8): examples

How does this relate to natural language processing?

- Most research in NLP (at least initially) has concerned supervised learning: parsing (treebanks for training available), named entity recognition systems, text categorization, others.
- It has shifted to semi-supervised learning because of the cost of human labour (e.g., for parsing Steedman'02).
- Unsupervised learning is used when clustering words/documents based on their similarity.
- Active learning is less studied, but is becoming more popular in the NLP community (e.g., text annotation by Tomanek et al.'09).

Main learning notions (9)

- *Empirical risk*: Since the underlying distribution is unknown, the quality of h is usually measured by the empirical error in Eq. 1.

$$R_n(h) = \frac{1}{n} \sum_{i=1}^{n} l(h(\mathbf{x_i}), y_i) \tag{1}$$

- *Zero-one loss* Several loss functions have been proposed in the literature so far, the best known of which is the zero-one loss (Eq. 2). This loss is a function that outputs 1 any time a method errs on a data point ($h(\mathbf{x_i}) \neq y_i$) and 0 otherwise.

$$l(h(\mathbf{x_i}), y_i) = \mathbb{I}_{h(\mathbf{x_i}) \neq y_i} \tag{2}$$

Main learning notions (9)

- *Empirical risk*: Since the underlying distribution is unknown, the quality of h is usually measured by the empirical error in Eq. 1.

$$R_n(h) = \frac{1}{n}\sum_{i=1}^{n} l(h(\mathbf{x_i}), y_i) \tag{1}$$

- *Zero-one loss* Several loss functions have been proposed in the literature so far, the best known of which is the zero-one loss (Eq. 2). This loss is a function that outputs 1 any time a method errs on a data point ($h(\mathbf{x_i}) \neq y_i$) and 0 otherwise.

$$l(h(\mathbf{x_i}), y_i) = \mathbb{I}_{h(\mathbf{x_i}) \neq y_i} \tag{2}$$

Main learning notions (10)

- At first glance the goal of any learning algorithm should be to minimize empirical error $R_n(h)$, which is often referred to as empirical risk minimization.
- This turns out to be not sufficient as some methods can perform well on the training set but be not as accurate on the new data points.
- In structural risk minimization (Eq. 3) not only the empirical error is taken into account, but the complexity (capacity) of h as well. In Eq. 3, $pen(h)$ stands for a penalty that reflects complexity of a hypothesis.

$$g_n = \arg\min_{h \in H} R_n(h) + pen(h) \tag{3}$$

Main learning notions (10)

- At first glance the goal of any learning algorithm should be to minimize empirical error $R_n(h)$, which is often referred to as empirical risk minimization.
- This turns out to be not sufficient as some methods can perform well on the training set but be not as accurate on the new data points.
- In structural risk minimization (Eq. 3) not only the empirical error is taken into account, but the complexity (capacity) of h as well. In Eq. 3, $pen(h)$ stands for a penalty that reflects complexity of a hypothesis.

$$g_n = \arg\min_{h \in H} R_n(h) + pen(h) \tag{3}$$

Main learning notions (10)

- At first glance the goal of any learning algorithm should be to minimize empirical error $R_n(h)$, which is often referred to as empirical risk minimization.
- This turns out to be not sufficient as some methods can perform well on the training set but be not as accurate on the new data points.
- In structural risk minimization (Eq. 3) not only the empirical error is taken into account, but the complexity (capacity) of h as well. In Eq. 3, $pen(h)$ stands for a penalty that reflects complexity of a hypothesis.

$$g_n = \arg\min_{h \in H} R_n(h) + pen(h) \tag{3}$$

Main learning notions (11)

How to evaluation NLP tasks/systems? Evaluation:

- **manual vs. automatic**: manual by experts, automatic by comparing results against gold standard.
- **extrinsic vs. intrinsic**: extrinsic against the gold standard, intrinsic - in use (e.g., how important is PoS tagging for machine translation?).
- black-box vs. glass-box

Main learning notions (11)

How to evaluation NLP tasks/systems? Evaluation:

- **manual vs. automatic**: manual by experts, automatic by comparing results against gold standard.
- **extrinsic vs. intrinsic**: extrinsic against the gold standard, intrinsic - in use (e.g., how important is PoS tagging for machine translation?).
- black-box vs. glass-box

Main learning notions (11)

How to evaluation NLP tasks/systems? Evaluation:

- **manual vs. automatic**: manual by experts, automatic by comparing results against gold standard.
- **extrinsic vs. intrinsic**: extrinsic against the gold standard, intrinsic - in use (e.g., how important is PoS tagging for machine translation?).
- **black-box vs. glass-box**

Main learning notions (12)

Consider for instance binary classification where each example has to be classified either as positive or as negative.

- Positive examples on which the method errs are referred to as *false negatives* (FN) and negative examples which it misclassifies are called *false positives* (FP).
- Those examples that are classified correctly are either *true positives* (TP) or *true negatives* (TN).

Main learning notions (12)

Consider for instance binary classification where each example has to be classified either as positive or as negative.

- Positive examples on which the method errs are referred to as *false negatives* (FN) and negative examples which it misclassifies are called *false positives* (FP).
- Those examples that are classified correctly are either *true positives* (TP) or *true negatives* (TN).

Main learning notions (13)

- *Accuracy* is defined as the fraction of all examples that were classified correctly (Eq. 4). Accuracy is often used when the data set is balanced (i.e., a number of true positives and true negatives is the same).

$$Acc = \frac{TP + TN}{TP + TN + FP + FN} \tag{4}$$

- *Precision* reflects how many examples in the data set that were classified as positive really belong to true positives, Eq. 5.

$$precision = \frac{TP}{TP + FP} \tag{5}$$

Main learning notions (13)

- *Accuracy* is defined as the fraction of all examples that were classified correctly (Eq. 4). Accuracy is often used when the data set is balanced (i.e., a number of true positives and true negatives is the same).

$$Acc = \frac{TP + TN}{TP + TN + FP + FN} \tag{4}$$

- *Precision* reflects how many examples in the data set that were classified as positive really belong to true positives, Eq. 5.

$$precision = \frac{TP}{TP + FP} \tag{5}$$

Main learning notions (14)

- *Recall* shows what fraction of the true positives were found by the method (Eq. 6).

$$recall = \frac{TP}{TP + FN} \tag{6}$$

- The F_1 *score* is defined as the harmonic mean between precision and recall (Eq. 9).

$$F_1 = \frac{2 * precision * recall}{precision + recall} \tag{7}$$

More on F-score

- The F_1 *score* is defined as the harmonic mean between precision and recall (Eq. 8).

$$F = \frac{(\beta^2 + 1) * precision * recall}{\beta^2 * precision + recall} \tag{8}$$

- commonly used F_1 measure ($\beta = 1$):

$$F_1 = \frac{2 * precision * recall}{precision + recall} \tag{9}$$

- if $\beta > 1$, it favours precision (recall otherwise).

Other measures

- Recall is also referred to as true positives (tp) rate:

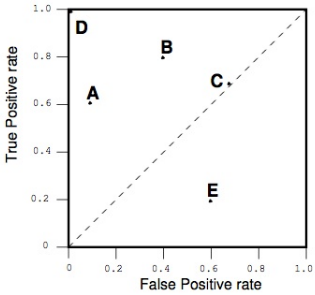
$$tp = \frac{TP}{TP + FN} \tag{10}$$

- There is also false positives (fp) rate:

$$fp = \frac{FP}{FP + TN} \tag{11}$$

ROC space

- Receiver operating characteristics (ROC) graphs (Fawcett'2004) are two-dimensional graphs in which TP rate is plotted on the Y axis and FP rate is plotted on the X axis.
- An ROC graph depicts relative trade-offs between benefits (true positives) and costs (false positives).



Performance

Many challenges have been organized over the years (on morphological analysis, parsing, translation, etc.)
The best (official) results for some of the tasks:

- **Morpheme analysis:**
  - **extrinsic** English (67.40%), Finnish (62.52%), German (50.85%), Turkish (65.31%) from Morpho Challenge 2010.
  - **intrinsic** on machine translation and information retrieval
- **Chunking:** English (94.13% of F-score) from CoNLL'200.

Performance

Many challenges have been organized over the years (on morphological analysis, parsing, translation, etc.)
The best (official) results for some of the tasks:

- **Named entity recognition:** English (88.76%), German (72.41%) from CoNLL'2003, Spanish (81.39%) and Dutch (77.05%) from CoNLL'2002 (F-score).
- **Parsing:** Arabic (66.91%), Chinese (89.96%), Czech (80.18%), Dutch (79.19%), Turkish (65.68%) (labeled attachment, alltogether 12 languages in CoNLL'2006).

Back to learning and language

We focus on statistical learning of language:

- It exploits statistical properties in language.
- It can be ported to languages other than English.
- In practice, it requires large amount of data: the larger, the better.

But how are words distributed?

Back to learning and language

We focus on statistical learning of language:

- It exploits statistical properties in language.
- It can be ported to languages other than English.
- In practice, it requires large amount of data: the larger, the better.

But how are words distributed?

Back to learning and language

We focus on statistical learning of language:

- It exploits statistical properties in language.
- It can be ported to languages other than English.
- In practice, it requires large amount of data: the larger, the better.

But how are words distributed?

Back to learning and language

We focus on statistical learning of language:

- It exploits statistical properties in language.
- It can be ported to languages other than English.
- In practice, it requires large amount of data: the larger, the better.

But how are words distributed?

Zipf's law

Zipf's law

- named after G. K. Zipf (1902-1950).
- "... the observation that frequency of occurrence of some event ($\mathcal{A}$), as a function of the rank (i) when the rank is determined by the above frequency of occurrence, is a power-law function $\approx \frac{1}{i^{\alpha}}$ with the exponent α close to unity (1)."

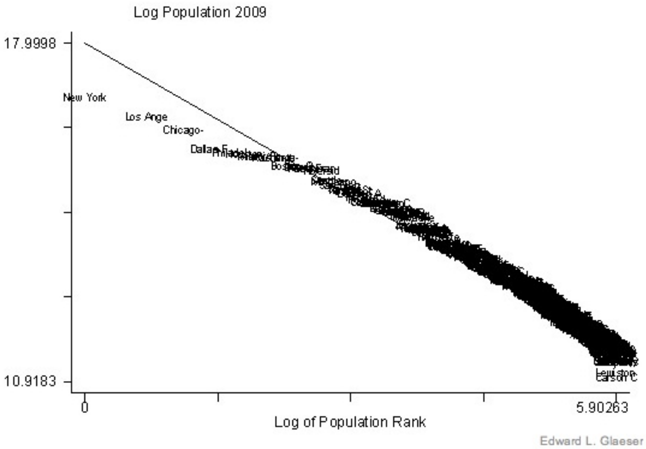
Zipf's law

Examples of Zipf's law

- Population: there are a few very populous cities, while numerous cities have a small population.
- Economics: income or revenue of a company is a function of the rank.
- Language: English words follows an exponential distribution. The most common words tend to be short and appear often.

Zipf's law

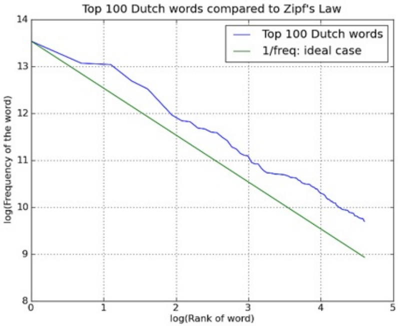
Source: E. Glaeser. *A Tale of Many Cities*. `http://economix.blogs.nytimes.com/2010/04/20/a-tale-of-many-cities/`



Zipf's law

Le Quan Ha et al.'02

Emre Sevinc



Zipf's law and top 100 Dutch words

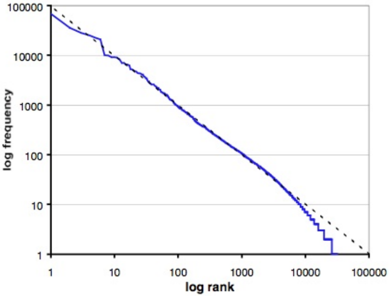


Figure 2 Zipf curve for the unigrams extracted from the 1 million words of the Brown corpus

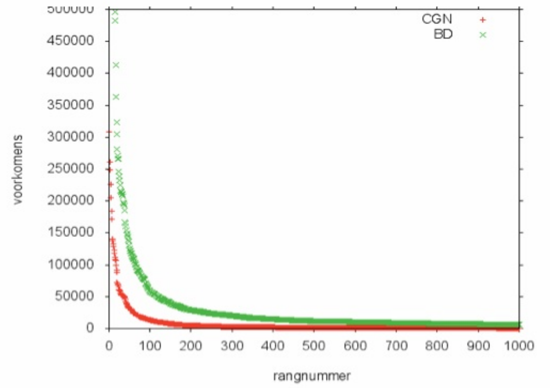
Zipf's law in Dutch (1)

Spoken: CGN

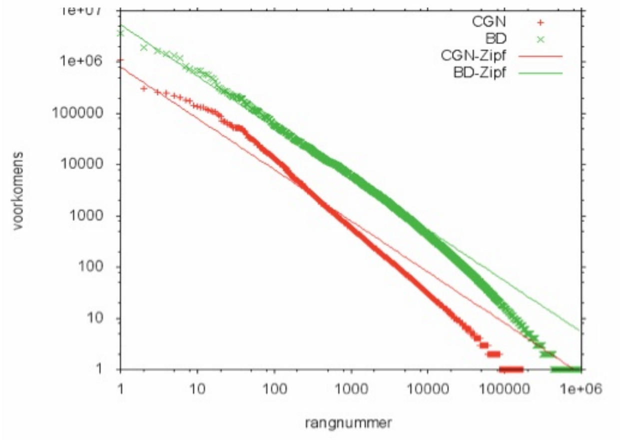
- 8.9 million words
- "Current" Dutch
- Spontaneous speech
- Adults

Written: BD

- 78,9 million words
- Newspaper articles
- Formal, dense
- Adults



Zipf's law in Dutch (2)



Zipf's law in Dutch (3)

- Spoken (CGN)

Rang	woord	Zipf	echt
5	en	160.995	226.106
50	naar	16.100	32.564
500	twaalf	1.610	1.424
5.000	vertrouwd	161	76
50.000	verhelderd	16	3

- Written (BD)

Rang	woord	Zipf	echt
5	in	1.077.832	1.333.512
50	moet	107.783	131.190
500	men	10.778	11.660
5.000	uitermate	1.078	1.014
50.000	summum	108	47

Statistics and language

So, how is this all related to NLP?

- First, probabilities in language need to be estimated, given some text or speech sample.
- Second, these estimates are often based on *relative frequency*, which is the number of times an outcome v occurs given n trials, $\frac{freq(v)}{n}$.
- Third, there are two ways in which modeling can be done:
  - parametric (given assumptions about the data distribution)
  - non-parametric, or distribution-free (no assumptions about the underlying distribution)

Terminology

Bayes' rule (or theorem)

$$P(\mathcal{B}|\mathcal{A}) = \frac{P(\mathcal{A}|\mathcal{B})P(\mathcal{B})}{P(\mathcal{A})} \tag{12}$$

If one is interested in finding out whether $P(\mathcal{B}|\mathcal{A})$ or $P(\mathcal{B}'|\mathcal{A})$ is more likely given $\mathcal{A}$, the denominator is ignored. Consequently, we get

$$\arg\max_{\mathcal{B}} P(\mathcal{A}|\mathcal{B})P(\mathcal{B}) \tag{13}$$

Naive Bayesian classifier: an example

Naive Bayes: document classification (1)

$$P(c|d) = P(c) \prod_{1 \le k \le n_d} P(t_k|c) \tag{14}$$

- $P(t_k|c)$ is the conditional probability of term t_k occurring in a document of class c.
- $P(c)$ is the prior probability of a document occurring in class c.
- $< t_1, \ldots, t_{n_d} >$ tokens in a document d.

Naive Bayes: document classification (2)

The goal becomes to find *maximum a posteriori* class (MAP):

$$c_{map} = \arg\max_{c \in C} \hat{P}(c|d) = \arg\max_{c \in C} \hat{P}(c) \prod_{1 \leq k \leq n_d} \hat{P}(t_k|c) \qquad (15)$$

Naive Bayes: document classification (3)

Using *maximum likelihood estimates*:

$$\hat{P}(c) = \frac{N_c}{N} \tag{16}$$

where N_c is the number of documents in class and N is the total number of documents.

$$\hat{P}(t_k|c) = \frac{T_{ct}}{\sum_{t' \in V} T_{ct'}} \tag{17}$$

T_{ct} is the number of occurrences of t in training documents from class c, including multiple occurrences of a term in a document.

Naive Bayes: document classification (4)

Data (*attribute-value representation, bag-of-words model*), doc_1 - doc_4 for training, doc_5 - a test example:

DocID	disk	monitor	ball	gate	class
doc_1	4	6	1	2	IT
doc_2	1	1	6	10	sport
doc_3	1	2	4	2	sport
doc_4	1	1	4	6	sport
doc_5	2	4	0	2	?

Naive Bayes: document classification (5)

Priors: $\hat{P}(sport) = 3/4$, $\hat{P}(IT) = 1/4$

Conditional probabilities:

$\hat{P}(\text{disk}|\text{IT}) = 4/13$
$\hat{P}(\text{disk}|\text{sport}) = 3/39$
$\hat{P}(\text{monitor}|\text{IT}) = 6/13$
$\hat{P}(\text{monitor}|\text{sport}) = 4/39$
$\hat{P}(\text{gate}|\text{IT}) = 2/13$
$\hat{P}(\text{gate}|\text{sport}) = 18/39$

$\hat{P}(sport|doc_5) = 3/4 * (3/39)^2 * (4/39)^4 * (2/39)^2 = 1.291 * 10^{-9}$
$\hat{P}(IT|doc_5) = 1/4 * (4/13)^2 * (6/13)^4 * (2/13)^2 = 1.652 * 10^{-5}$

Naive Bayes: document classification (6)

What are the problems while using Naive Bayes?

- sparse data → *smoothing* has to be used.
- the assumption on feature independence does not always hold in reality.

but

- the first and second place in KDD-CUP 97 competition, among 16 (then) state of the art algorithms
- computationally inexpensive
- Zhang (2004): the behaviour of NB is influenced by the dependence distribution rather than feature distribution

Naive Bayes: document classification (6)

What are the problems while using Naive Bayes?

- sparse data → *smoothing* has to be used.
- the assumption on feature independence does not always hold in reality.

but

- the first and second place in KDD-CUP 97 competition, among 16 (then) state of the art algorithms
- computationally inexpensive
- Zhang (2004): the behaviour of NB is influenced by the dependence distribution rather than feature distribution

Decision trees

Decision tree

From T. Mitchell. *Machine learning*.

Tree representation

- each internal node tests an attribute
- each branch (edge) corresponds to an attribute value
- each leaf node assigns a classification

Decision trees can be used when

- data is in attribute-value representation
- disjunctive hypotheses may be used

Decision tree

From T. Mitchell. *Machine learning*.

Tree representation

- each internal node tests an attribute
- each branch (edge) corresponds to an attribute value
- each leaf node assigns a classification

Decision trees can be used when

- data is in attribute-value representation
- disjunctive hypotheses may be used

Decision tree: induction

1. $A \leftarrow$ the "best" decision attribute for next *node*
2. Assign A as decision attribute for *node*
3. For each value of A, create new descendant of *node*
4. Sort training examples to leaf nodes
5. If training examples perfectly classified, Then STOP, Else iterate over new leaf nodes

Which attribute is best?

Decision tree: induction

From T. Mitchell. *Machine learning*.

- S is a training set
- p_+ is the proportion of positive examples in S
- p_- is the proportion of negative examples in S
- Entropy (= impurity) of S (the number of bits needed to encode class ($+$ or $-$) of randomly drawn examples of S)

$$Entropy(S) = -p_+ \log p_+ - p_- \log p_- \tag{18}$$

Decision tree: induction

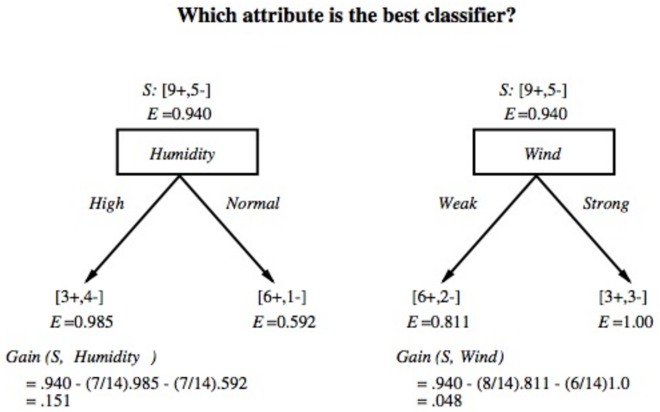
$Gain(S, A)$ = expected reduction in entropy due to sorting on A

$$Gain(S, A) \equiv Entropy(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} Entropy(S_v)$$

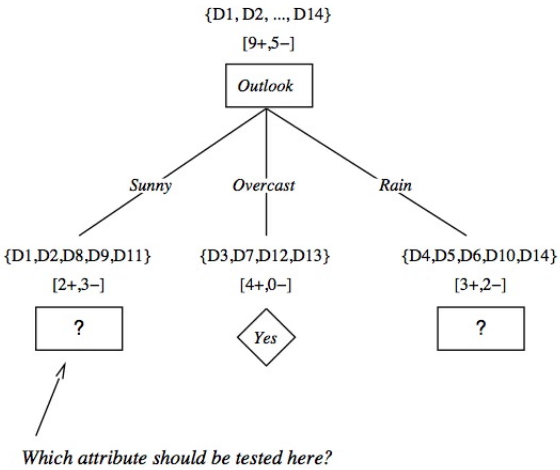
Example

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

Example


Which attribute is the best classifier?
S: [9+,5-]
E =0.940
Humidity
High
Normal
[3+,4-]
E =0.985
[6+,1-]
E =0.592
Gain (S, Humidity)
= .940 - (7/14).985 - (7/14).592
= .151
S: [9+,5-]
E =0.940
Wind
Weak
Strong
[6+,2-]
E =0.811
[3+,3-]
E =1.00
Gain (S, Wind)
= .940 - (8/14).811 - (6/14)1.0
= .048

Example


{D1, D2, ..., D14}
[9+,5−]
Outlook
Sunny
Overcast
Rain
{D1,D2,D8,D9,D11}
[2+,3−]
{D3,D7,D12,D13}
[4+,0−]
{D4,D5,D6,D10,D14}
[3+,2−]
?
Yes
?
Which attribute should be tested here?

Example

$$S_{sunny} = \{D1, D2, D8, D9, D11\}$$

$\mathrm{Gain}(\mathrm{S}_{sunny}, Humidity) = .970 - (3/5)0.0 - (2/5)0.0 = .970$
$\mathrm{Gain}\ (\mathrm{S}_{sunny}, Temperature) = .970 - (2/5)0 - (2/5)1.0 - (1/5)0 = .57$
$\mathrm{Gain}\ (\mathrm{S}_{sunny}, Wind) = .970 - (2/5)1.0 - (3/5).918 = .019$

Decision tree: induction

From T. Mitchell. *Machine learning*.

- inductive bias: shorter trees are preferred as well as those with high gain attributes near the root
- Okham's (Occam's) razor: prefer the shortest hypothesis that fit data

why shorter?

- + a short hypothesis that fits data is unlikely to be coincidental
- + a long hypothesis that fits data might be coincidence
- - there are many ways to define small sets of hypotheses

Decision tree: overfitting

Consider error of hypothesis h over

- training data: $error_{train}(h)$
- entire distribution $\mathcal{D}$ of data: $error_{\mathcal{D}}(h)$

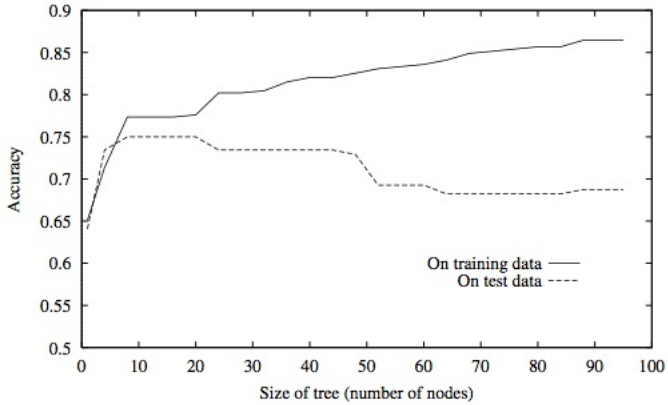
Hypothesis $h \in H$ **overfits** training data if there is an alternative hypothesis $h' \in H$ such that

$$error_{train}(h) < error_{train}(h')$$

and

$$error_{\mathcal{D}}(h) > error_{\mathcal{D}}(h')$$

Decision tree: overfitting



Decision tree: overfitting

How to avoid overfitting?

- stop growing a tree when a data split is not statistically significant
- form a full tree and then *post-prune* it

how to select the "best" tree?

- measure performance on training data
- measure performance on validation data
- use Minimum Description Length (MDL): minimize size(tree)+ size(misclassifications(tree))

Memory-based learning

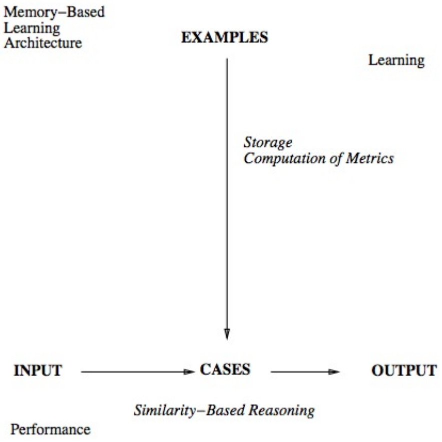
Basic idea

[thanks to A. van den Bosch]
Memory-based learning = *lazy* learning = *k-nearest neighbour* approach ML approaches can be divided into

- Greedy (e.g., Naive Bayes, decision trees, regression)
  - Learning (abstract model from data)
  - Classification (apply abstracted model to new data)
- Lazy (e.g., k-nearest neighbour)
  - Learning (store data in memory)
  - Classification (compare new data to data in memory)

Basic idea

[from Timbl guidelines]



Basic idea

- Learning component
  - memory-based (= training instances are added to memory, the instance base or case base; no abstraction or restructuring involved here.
  - fast!
- Performance component
  - an unseen example x_{n+1} is compared against **all** stored examples $x_1, \ldots, x_n$ using a distance metric $\delta(x_{n+1}, x_i), i = 1, \ldots, n$.
  - k-nearest neighbours are selected and x_{n+1} is assigned the most frequent category (label) that k-nearest neighbours have.
  - can be time-consuming!

The nearest neighbour has the largest similarity or the smallest distance!

There are many apects to consider

- Metrics
  - Overlap (Hamming distance): the number of mismatching attributes
  - Modified value difference metric
- Weighting schemes (if some attributes are more informative): Information gain, gain ratio

Example

Overlap measure: {Sunny, Hot, High, Unknown, ?}

Day	Outlook		Temperature	Humidity	Wind	PlayTennis
D1	Sunny	1	Hot	High	Weak	No
D2	Sunny	1	Hot	High	Strong	No
D3	Overcast	2	Hot	High	Weak	Yes
D4	Rain	3	Mild	High	Weak	Yes
D5	Rain	4	Cool	Normal	Weak	Yes
D6	Rain	4	Cool	Normal	Strong	No
D7	Overcast	4	Cool	Normal	Strong	Yes
D8	Sunny	2	Mild	High	Weak	No
D9	Sunny	3	Cool	Normal	Weak	Yes
D10	Rain	4	Mild	Normal	Weak	Yes
D11	Sunny	3	Mild	Normal	Strong	Yes
D12	Overcast	3	Mild	High	Strong	Yes
D13	Overcast	3	Hot	Normal	Weak	Yes
D14	Rain	3	Mild	High	Strong	No

Example

Overlap measure: {Sunny, Hot, High, Unknown, ?}

Day	Outlook		Temperature	Humidity	Wind	PlayTennis
D1	Sunny	1	Hot	High	Weak	No
D2	Sunny	1	Hot	High	Strong	No
D3	Overcast	2	Hot	High	Weak	Yes
D4	Rain	3	Mild	High	Weak	Yes
D5	Rain	4	Cool	Normal	Weak	Yes
D6	Rain	4	Cool	Normal	Strong	No
D7	Overcast	4	Cool	Normal	Strong	Yes
D8	Sunny	2	Mild	High	Weak	No
D9	Sunny	3	Cool	Normal	Weak	Yes
D10	Rain	4	Mild	Normal	Weak	Yes
D11	Sunny	3	Mild	Normal	Strong	Yes
D12	Overcast	3	Mild	High	Strong	Yes
D13	Overcast	3	Hot	Normal	Weak	Yes
D14	Rain	3	Mild	High	Strong	No

Learn more

- Memory-based learning is part of several learning toolkits, including WEKA.
- Software focused exclusively on MBL is Timbl (Tilburg Memory-Based Learner):

  `http://ilk.uvt.nl/downloads/pub/papers/Timbl_6.3_Manual.pdf`

- Walter Daelemans, Antal van den Bosch (2005). *Memory-Based Language Processing*. Studies in Natural Language Processing, Cambridge University Press.

Example

Morphological analysis of Dutch (from den Bosch & Daelemans. *Memory-based morphological analysis*.)

instance number	left context					focus letter	right context					TASK
1	_	_	_	_	_	a	b	n	o	r	m	A+Da
2	_	_	_	_	a	b	n	o	r	m	a	0
3	_	_	_	a	b	n	o	r	m	a	l	0
4	_	_	a	b	n	o	r	m	a	l	i	0
5	_	a	b	n	o	r	m	a	l	i	t	0
6	a	b	n	o	r	m	a	l	i	t	e	0
7	b	n	o	r	m	a	l	i	t	e	i	0
8	n	o	r	m	a	l	i	t	e	i	t	0
9	o	r	m	a	l	i	t	e	i	t	e	N_A*
10	r	m	a	l	i	t	e	i	t	e	n	0
11	m	a	l	i	t	e	i	t	e	n	_	0
12	a	l	i	t	e	i	t	e	n	_	_	0
13	l	i	t	e	i	t	e	n	_	_	_	0
14	i	t	e	i	t	e	n	_	_	_	_	m
15	t	e	i	t	e	n	_	_	_	_	_	0