An Introduction to Semi-Supervised Learning

Foundations of Language Science and Technology

By Michael Wiegand

November 27th, 2009

Outline of Talk

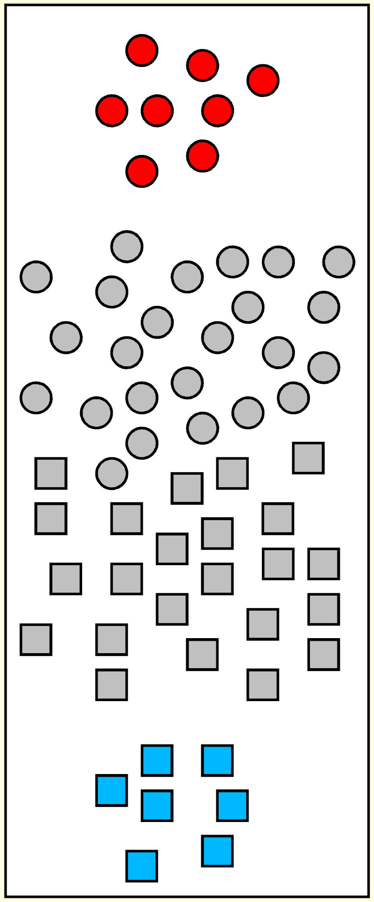
- The Concept of Semi-Supervised Learning
- Bootstrapping
- The Yarowksy Algorithm
- The Expectation Maximization Algorithm
- The Importance of Feature Selection in Semi-Supervised Learning (on Text Classification)

Aknowledgements

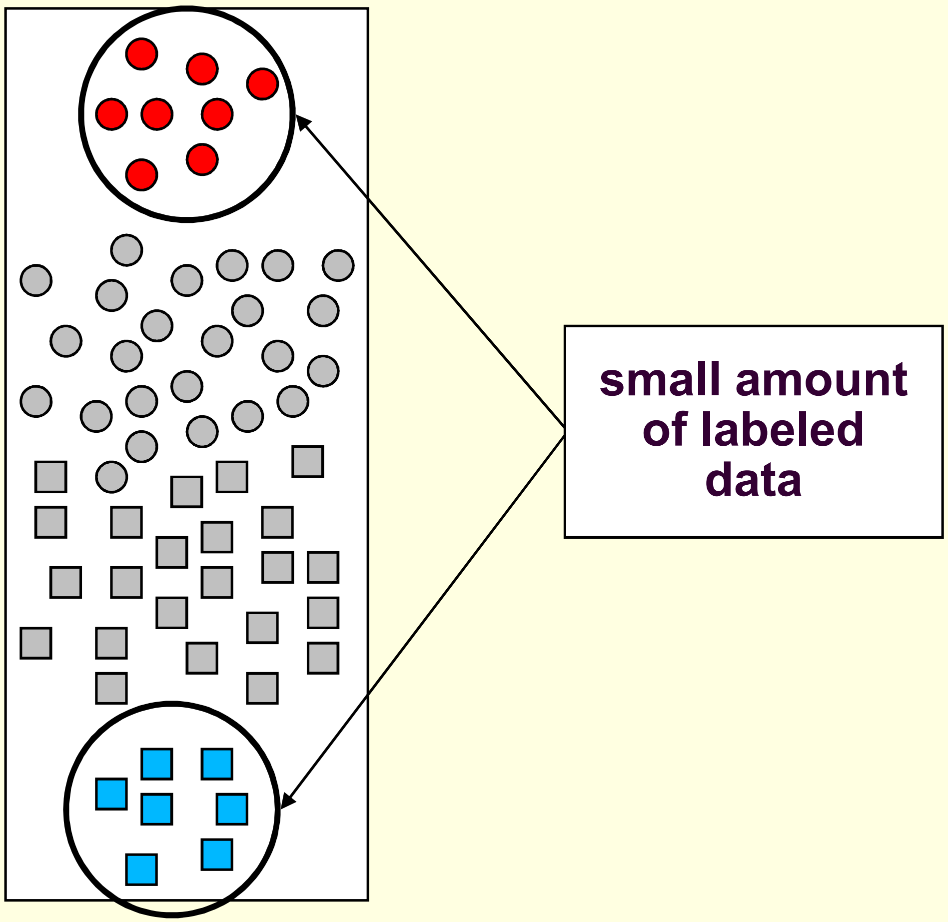
- Dietrich Klakow‘s lecture slides from *„Statistical Natural Language Processing“* (Spring 2008, Saarland University)
- Bing Liu‘s lecture slides from *„Data Mining and Text Mining“* (Spring 2008, University of Illinois at Chicago)
- William Cohen‘s and Tom Mitchell‘s lecture slides from *„Information Extraction“* (Spring 2007, Carnegie Mellon University)

The Concept of Semi-Supervised Learning

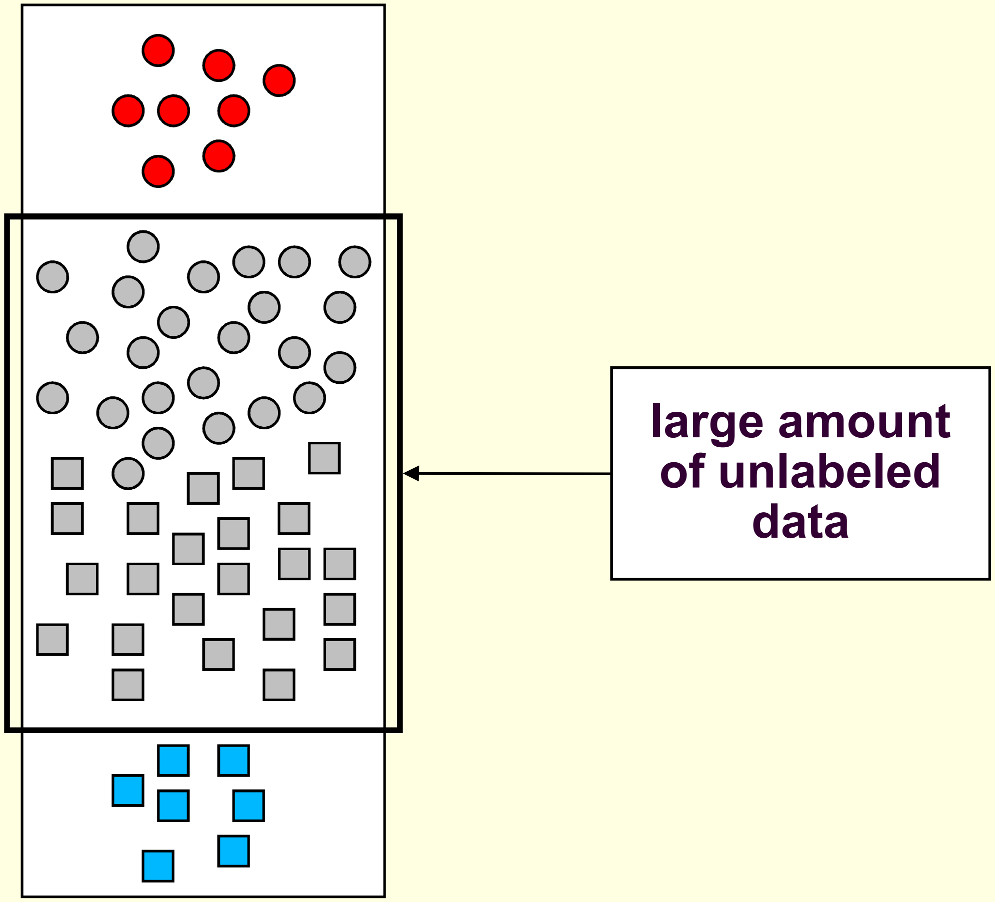
Semi-Supervised Learning - An Illustration



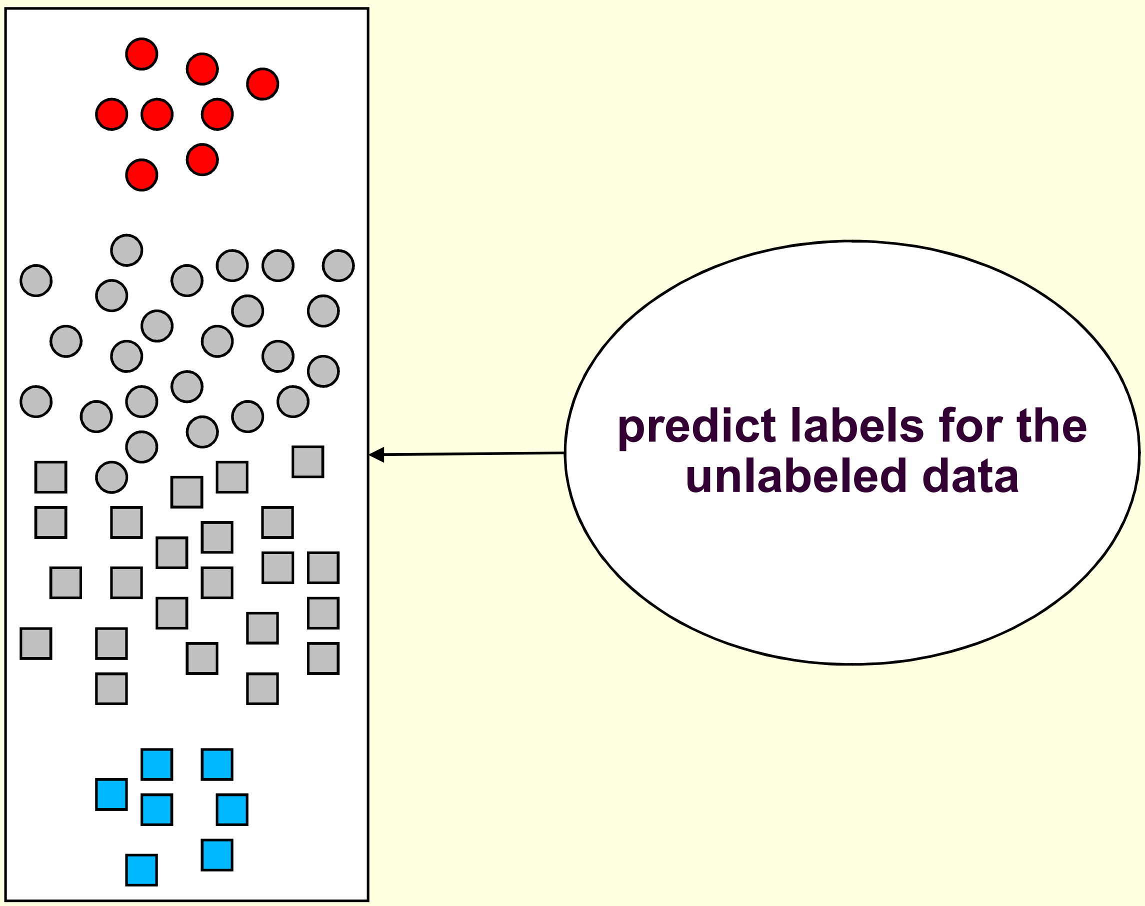
Semi-Supervised Learning - An Illustration



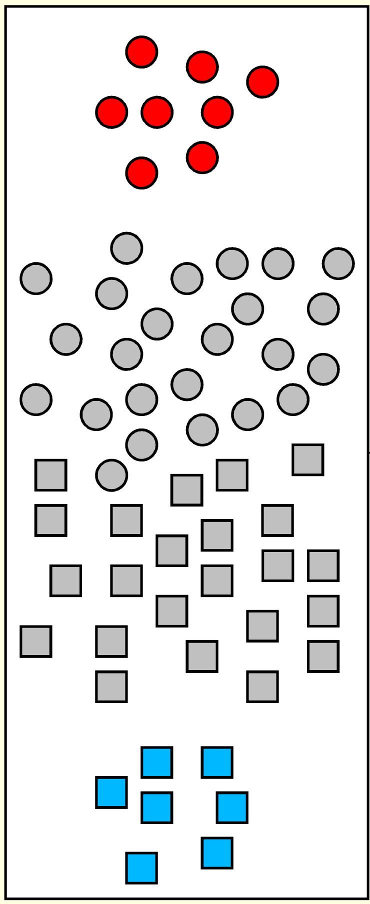
Semi-Supervised Learning - An Illustration



Semi-Supervised Learning - An Illustration

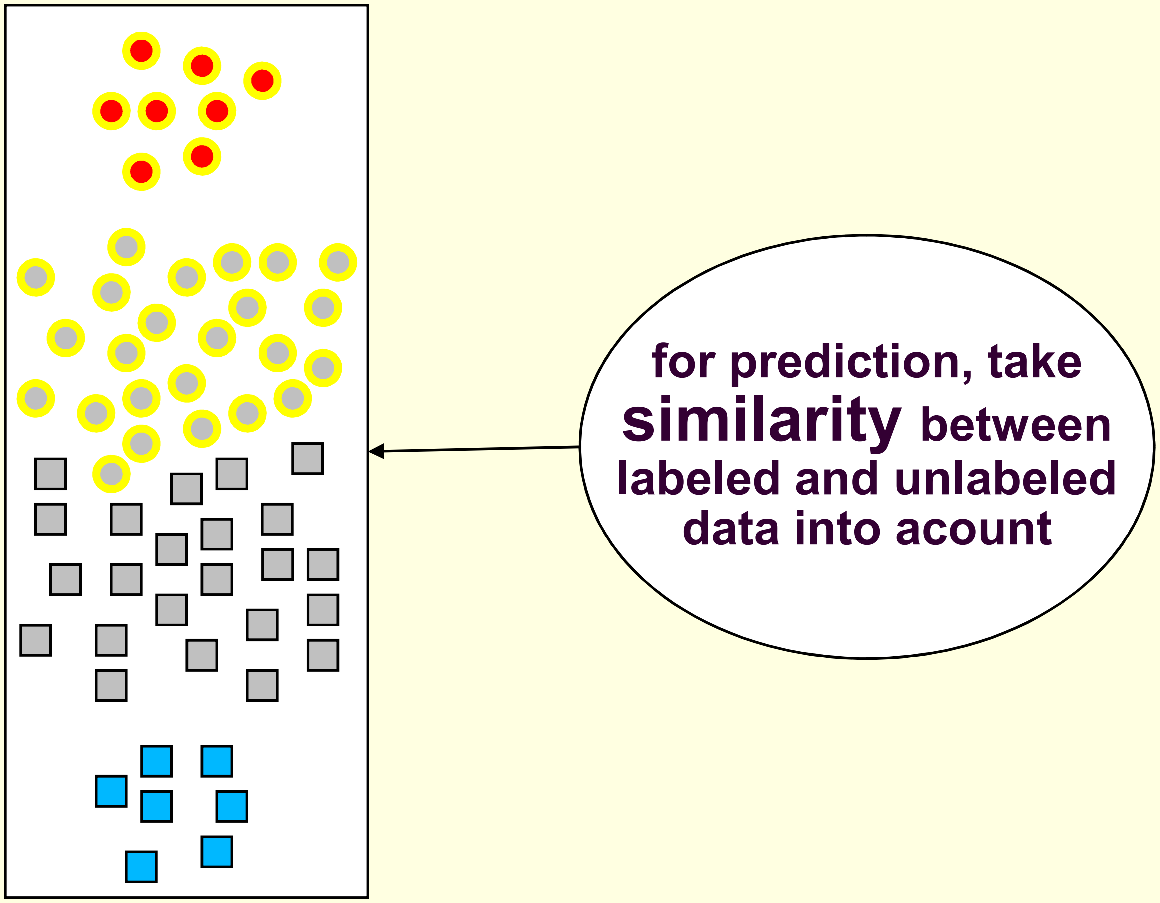


Semi-Supervised Learning - An Illustration

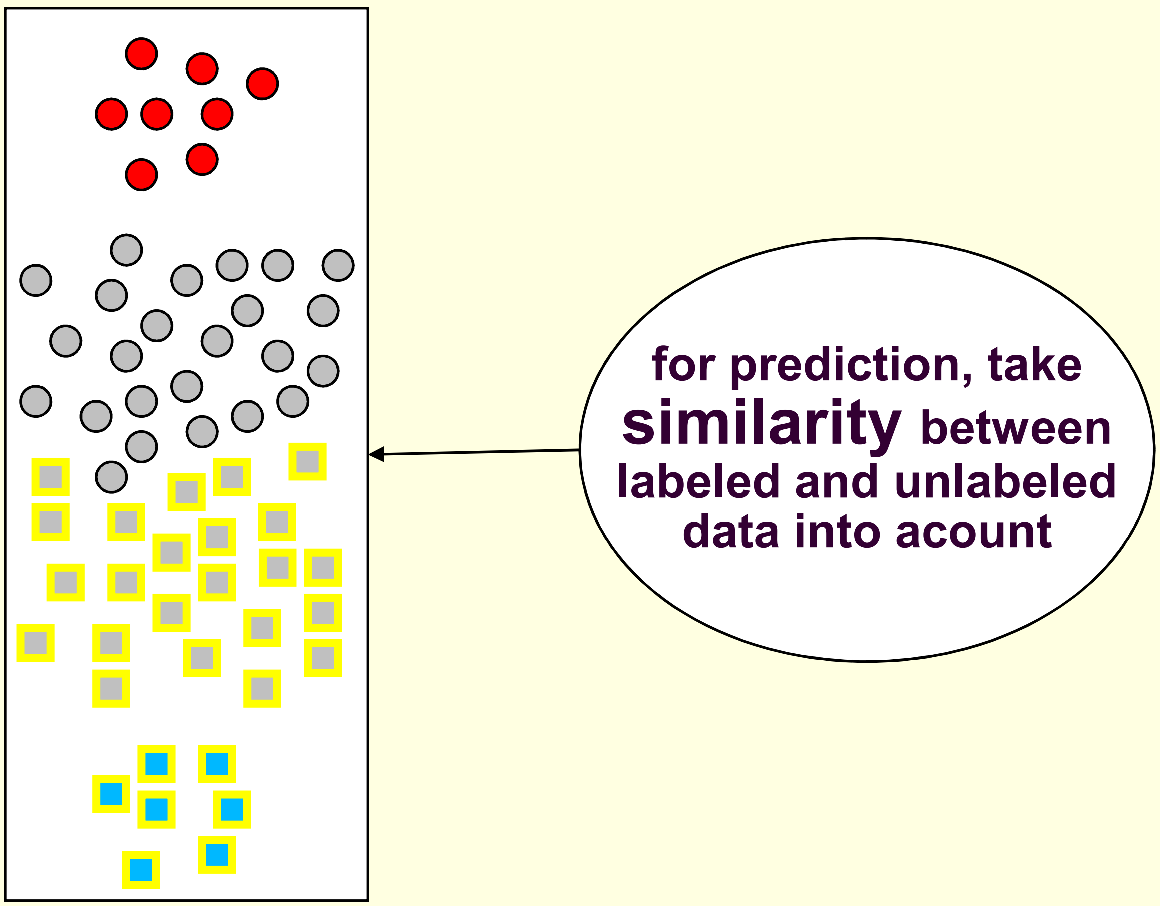


for prediction, take **similarity** between labeled and unlabeled data into acount

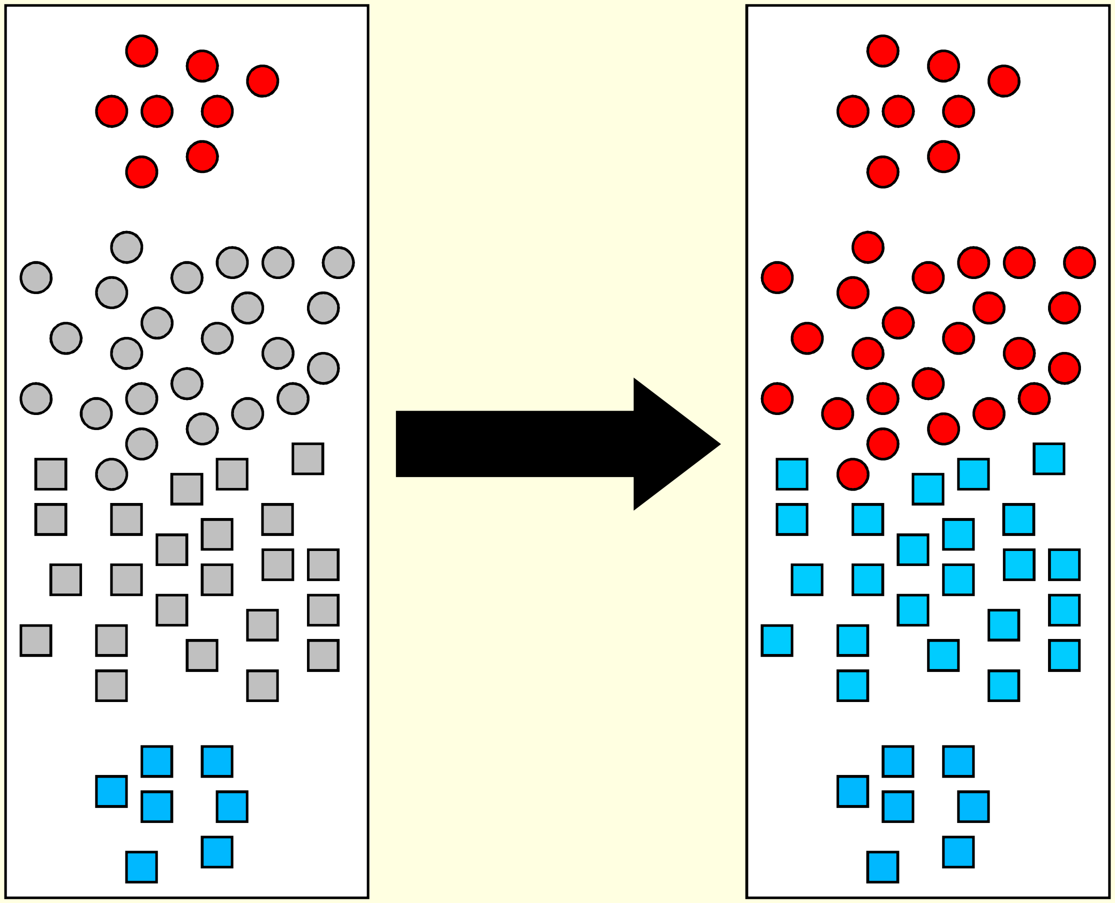
Semi-Supervised Learning - An Illustration



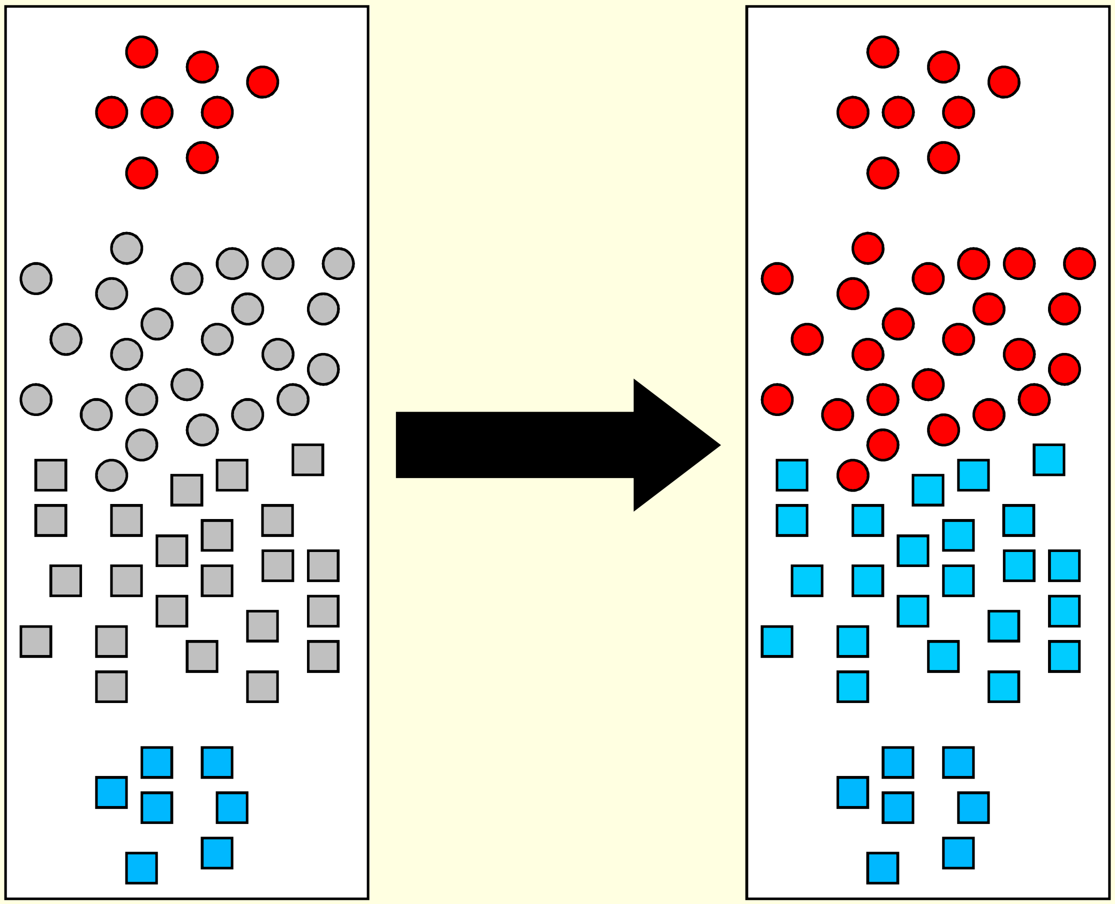
Semi-Supervised Learning - An Illustration



Semi-Supervised Learning - An Illustration

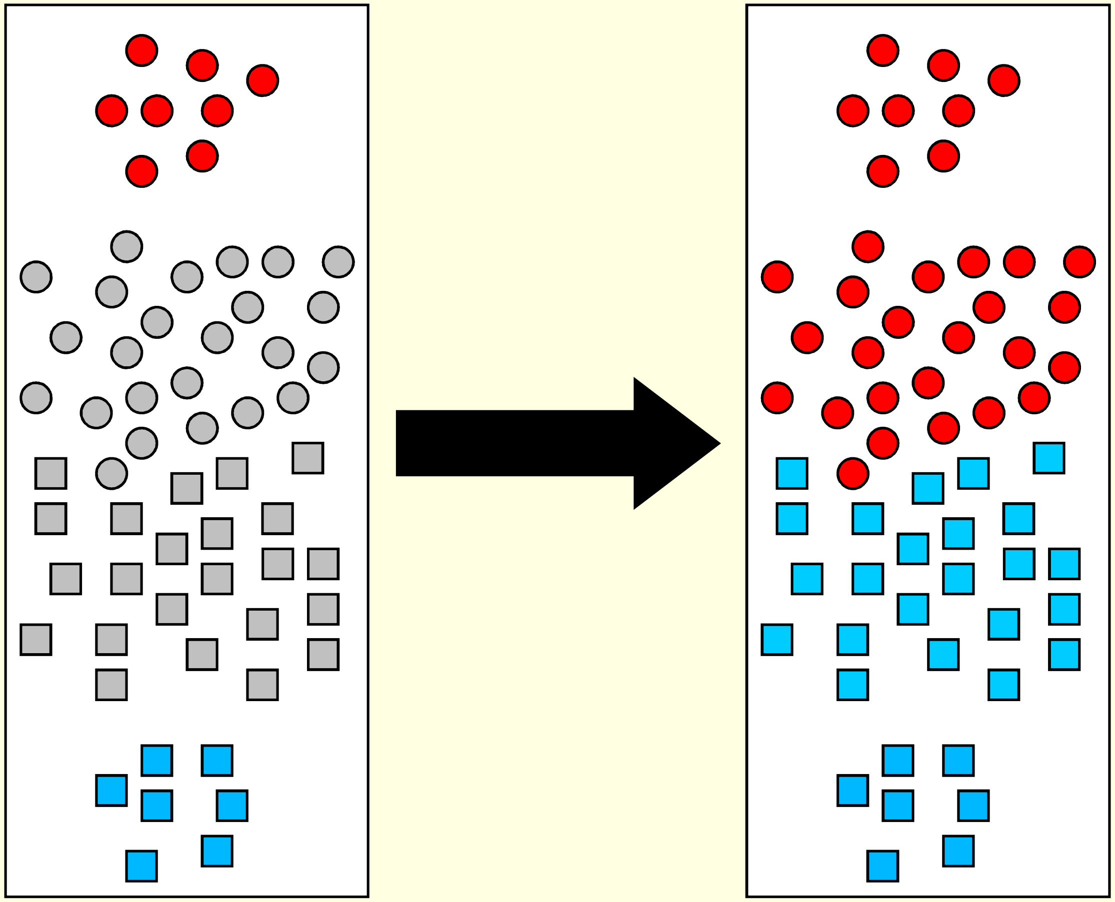


Semi-Supervised Learning - An Illustration



construct a classifier which also uses the predicted labels for the unlabeled data

Semi-Supervised Learning - An Illustration



more labeled data may improve classification accuracy

Why are unlabeled data useful?

- We will use (binary) text classification to study this problem
- Unlabeled data are usually plentiful, labeled data are expensive
- Unlabeled data provide information about the joint probability distribution over words and collocations (in texts)

Why are unlabeled data useful?

- Imagine the following setting:
  - You want to build a classifier which is able to detect text documents about „Machine Learning“
  - We have labeled and unlabeled documents
  - For simplification we denote
    - „+“: label for machine learning documents
    - „-“: label for other documents

Why are unlabeled data useful?

labeled documents

…
… classifier…
…
+

…
… classifier…
…
+

…
… classifier…
…
+

…
… classifier…
…
+

unlabeled documents

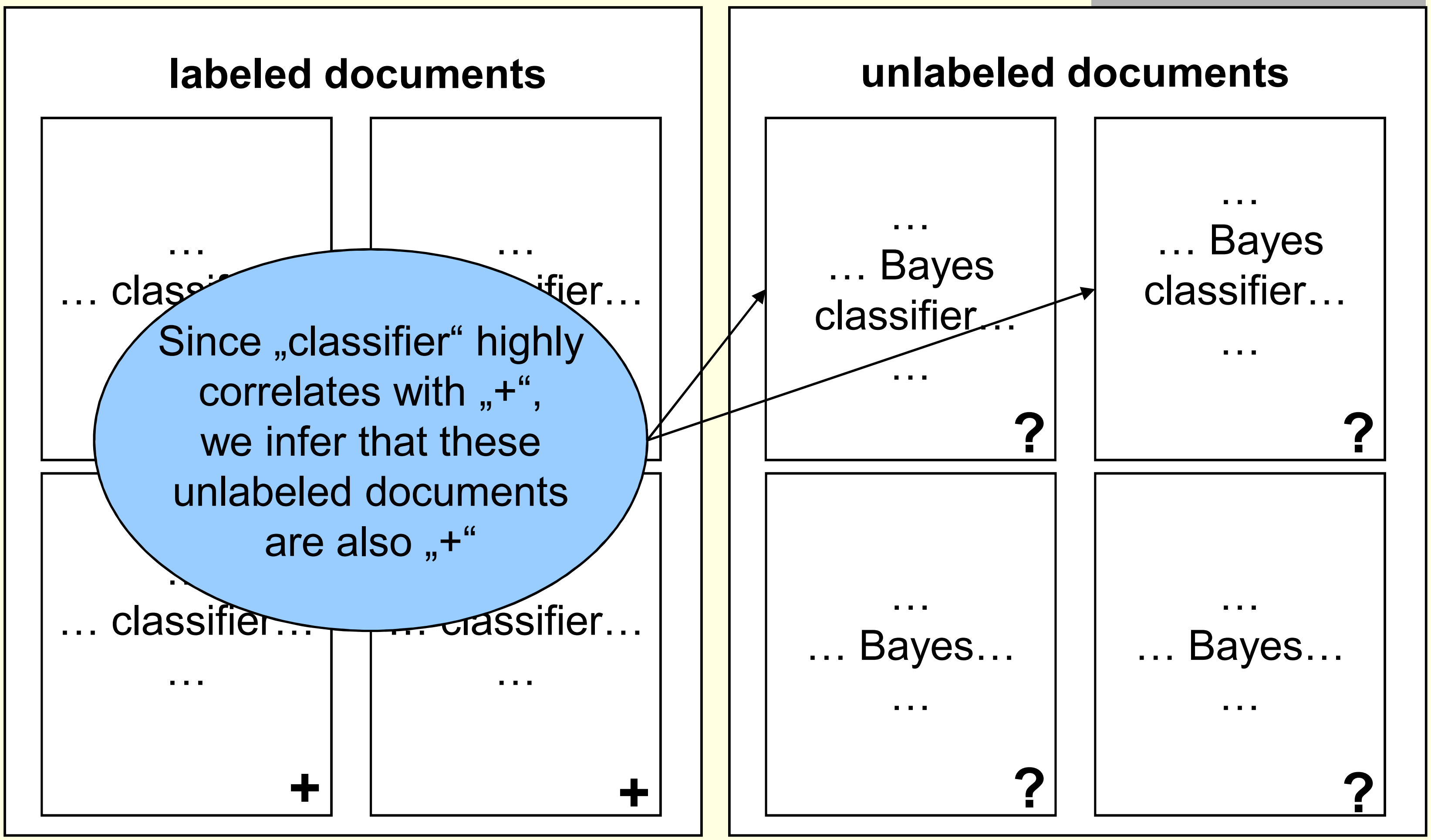
…
… Bayes classifier…
…
?

…
… Bayes classifier…
…
?

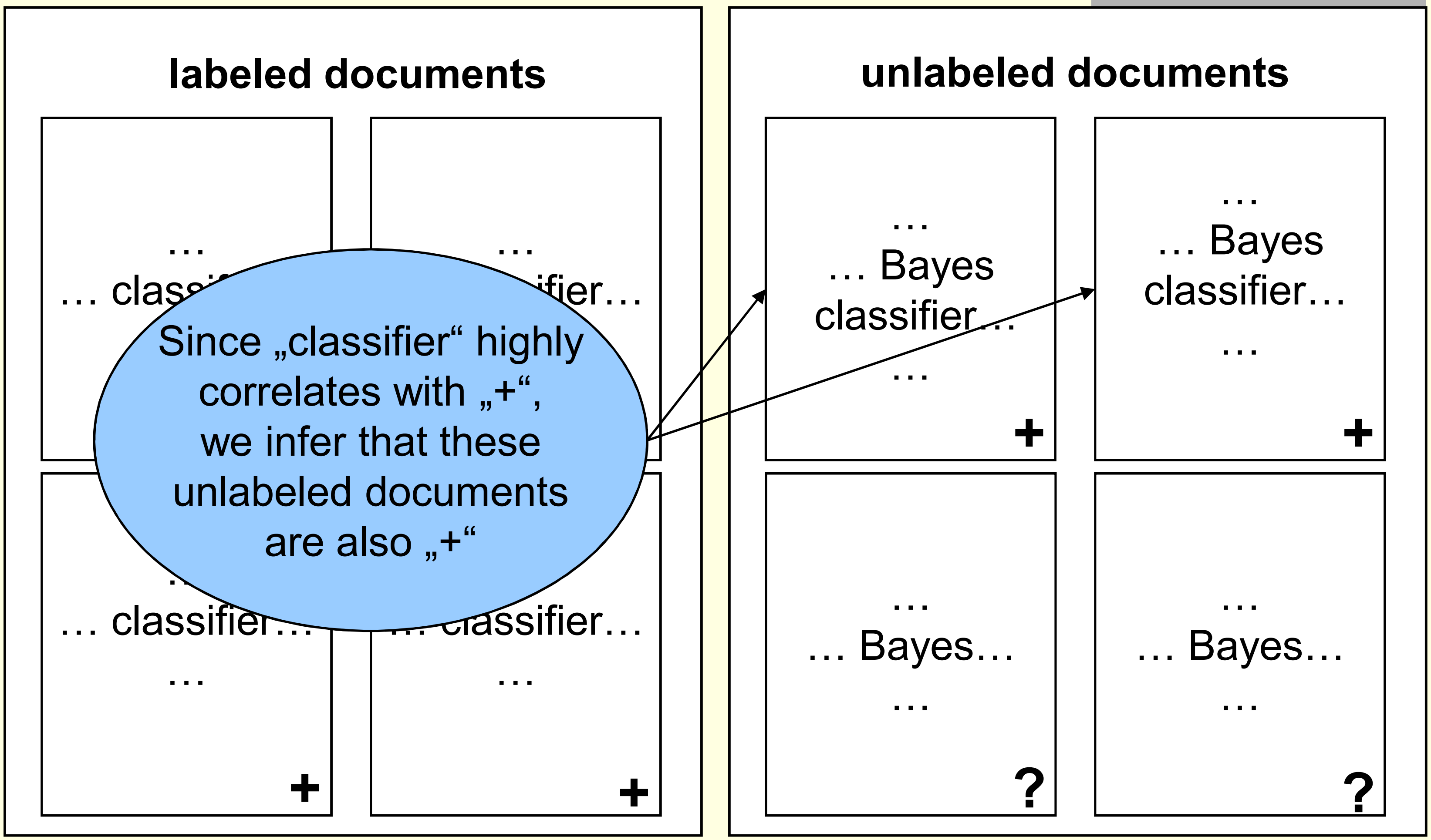
…
… Bayes…
…
?

…
… Bayes…
…
?

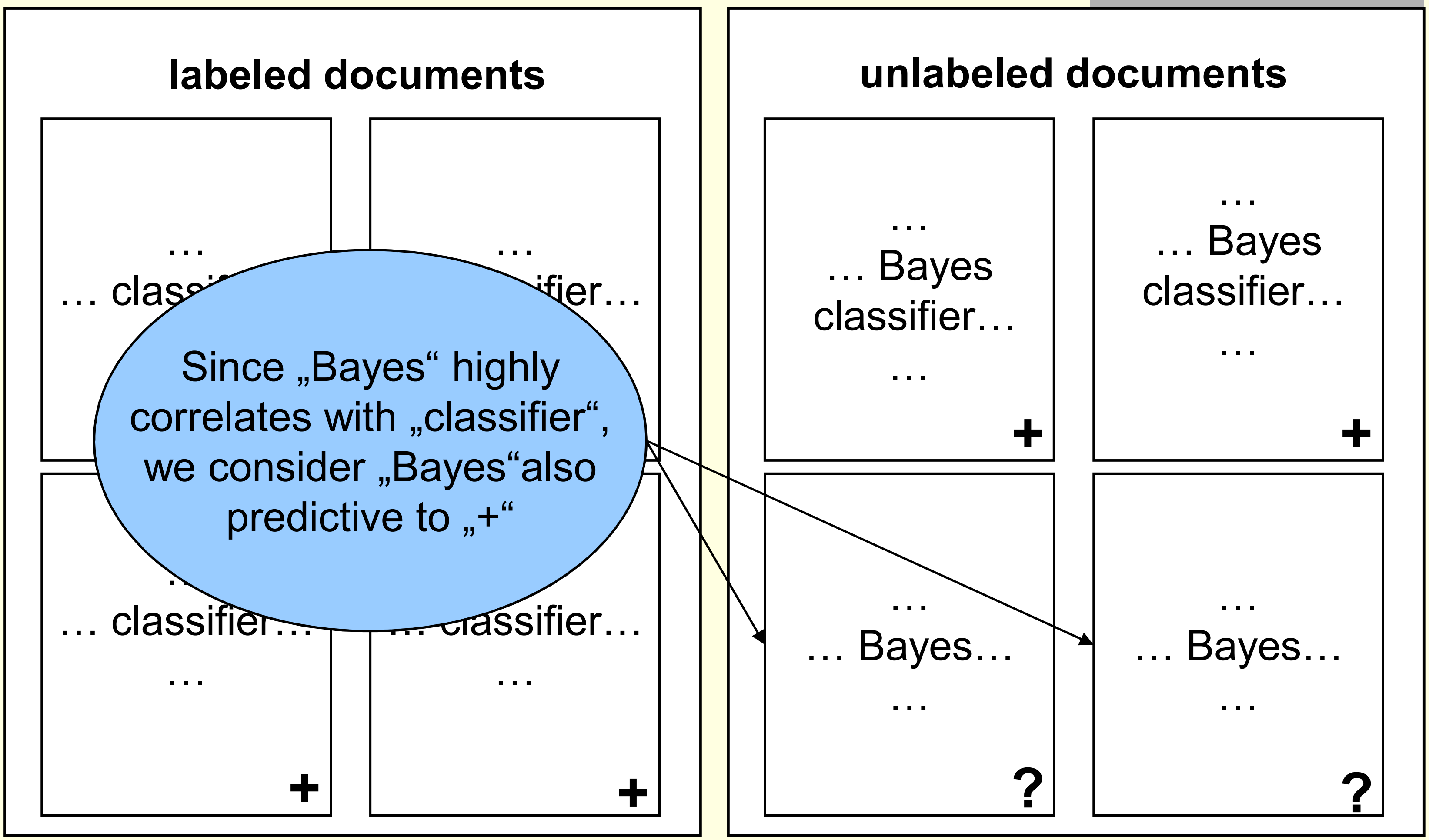
Why are unlabeled data useful?



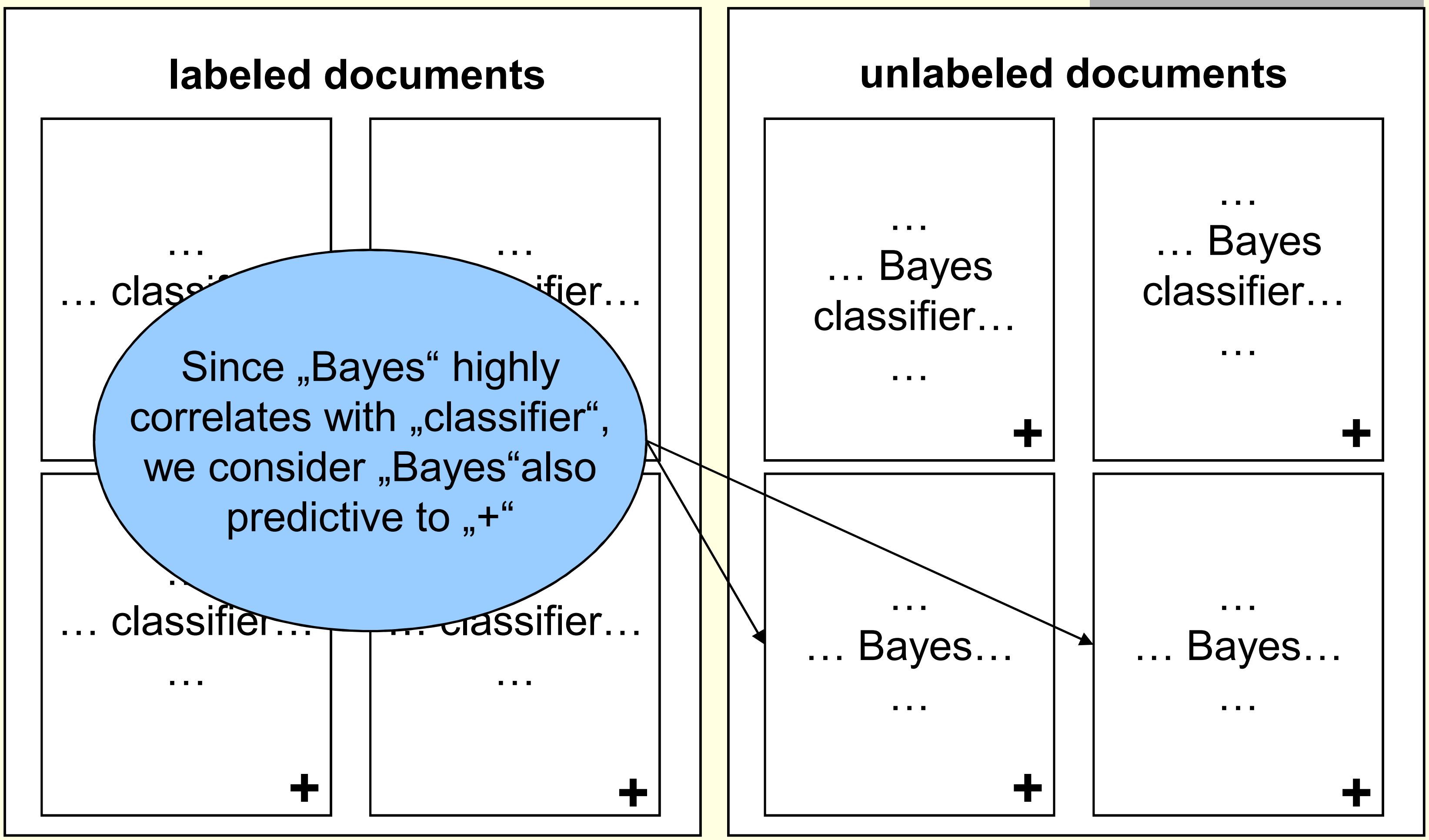
Why are unlabeled data useful?



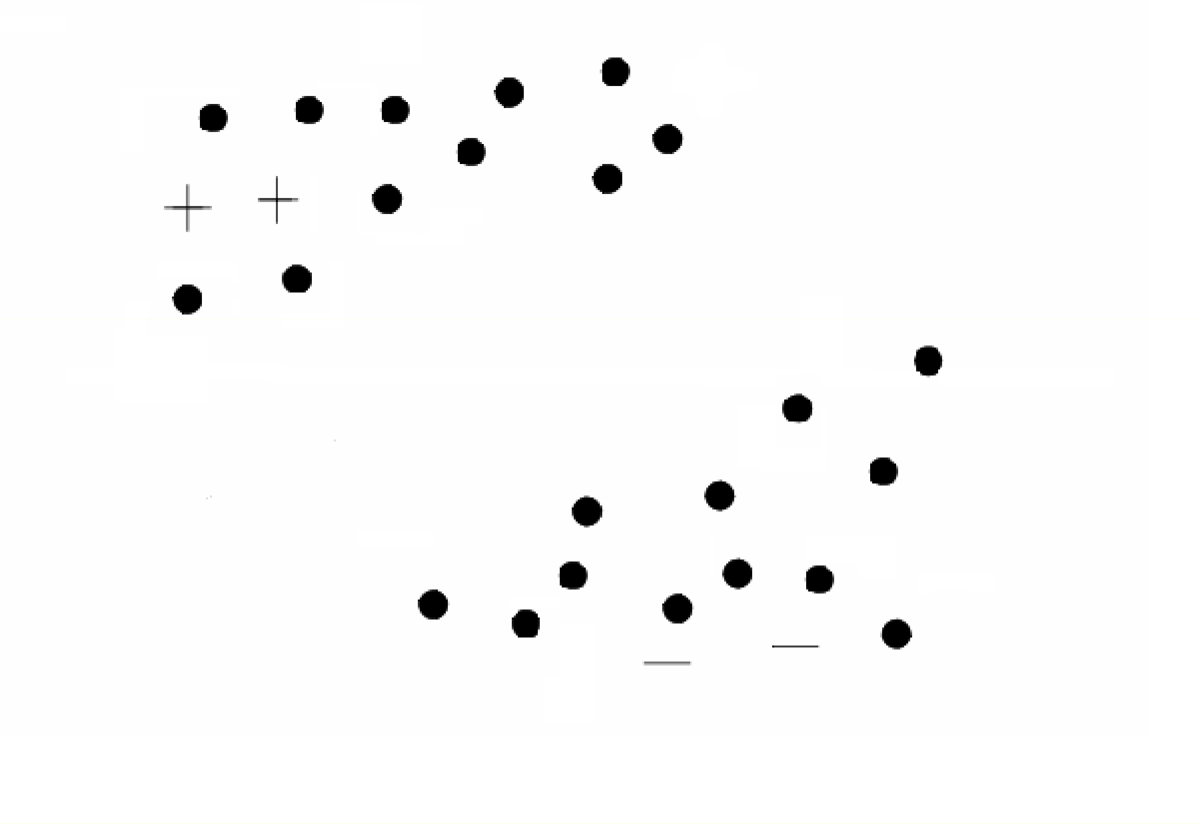
Why are unlabeled data useful?



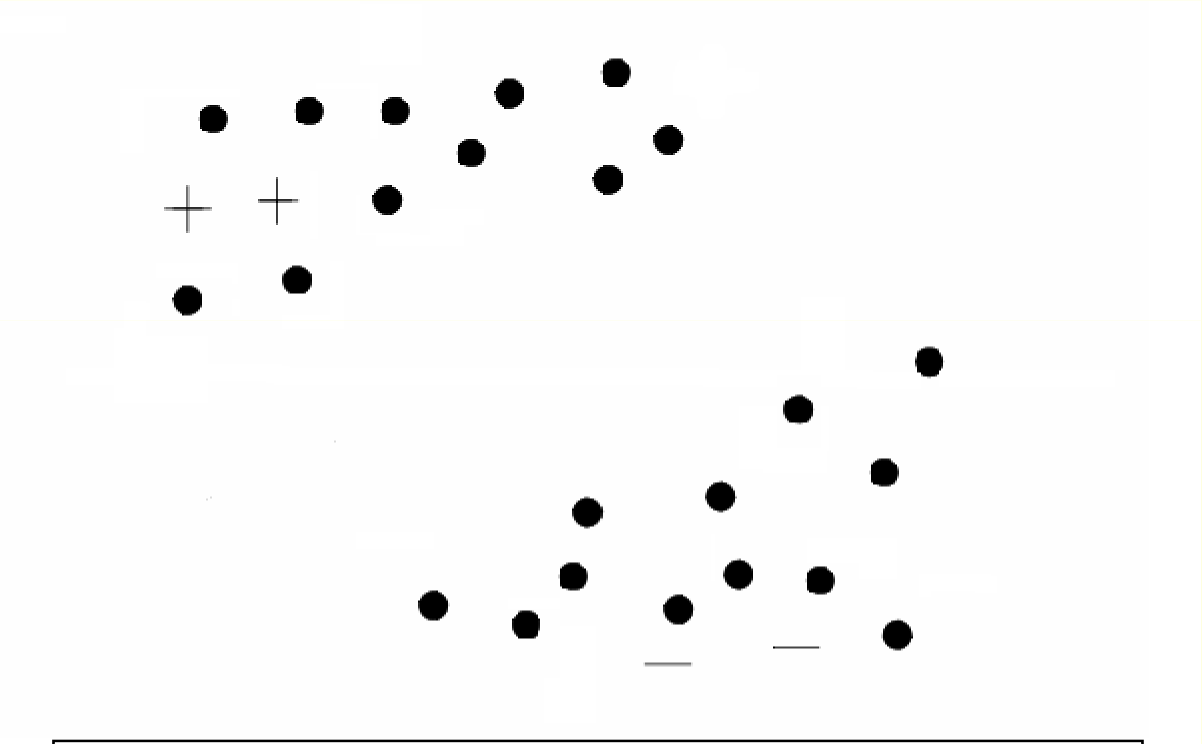
Why are unlabeled data useful?



A Dataset favourable for Semi-Supervised Learning

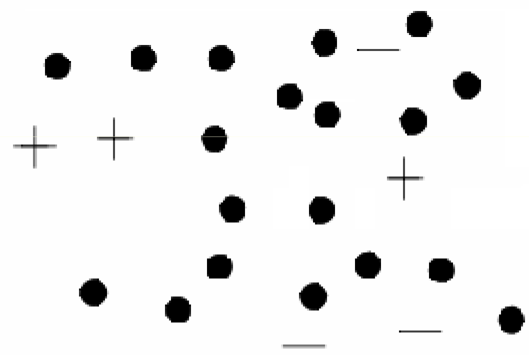


A Dataset favourable for Semi-Supervised Learning

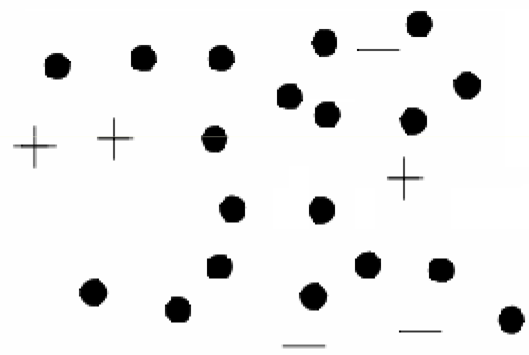


Unlabeled data instances cluster with labeled data instances of their pertaining class

A Dataset unfavourable for Semi-Supervised Learning



A Dataset unfavourable for Semi-Supervised Learning



Unlabeled data instances do **not** cluster with labeled data instances of their pertaining class

Bootstrapping

In computing, **bootstrapping** refers to a process where a simple system activates another more complicated system that serves the same purpose. It is a solution to the *chicken-and-egg problem* of starting a certain system without the system already functioning.

Bootstrapping

In computing, **bootstrapping** refers to a process where a simple system activates another more complicated system that serves the same purpose. It is a solution to the *chicken-and-egg problem* of starting a certain system without the system already functioning.

How does this translate to Semi-Supervised Learning?

Bootstrapping

In computing, **bootstrapping** refers to a process where a _simple system_ *(=supervised classifier using small amounts of labeled data)* activates another more complicated system that serves the same purpose. It is a solution to the *chicken-and-egg problem* of starting a certain system without the system already functioning.

How does this translate to Semi-Supervised Learning?

Bootstrapping

In computing, **bootstrapping** refers to a process where a *simple system* *(=supervised classifier using small amounts of labeled data)* activates *another more complicated system* *(=semi-supervised classifier that uses labeled and unlabeled data)* that serves the same purpose. It is a solution to the *chicken-and-egg problem* of starting a certain system without the system already functioning.

How does this translate to Semi-Supervised Learning?

Bootstrapping – The Origin of the Term

Bootstrapping alludes to a German legend about a **Baron Muench-hausen**, who was able to lift himself out of a swamp by pulling himself up by his own hair *(see picture on the right)*.

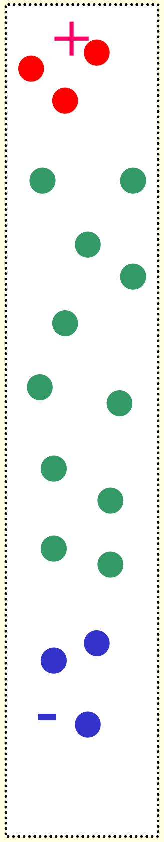
Bootstrapping – The Origin of the Term

In later versions he was using his own **bootstraps** to pull himself out of the sea.

The Yarowsky Algorithm

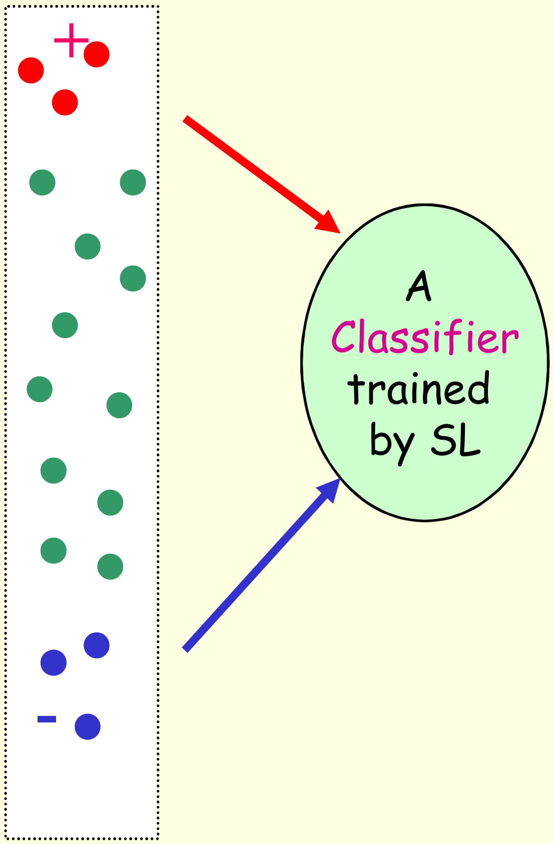
The Yarowsky Algorithm

Iteration: 0



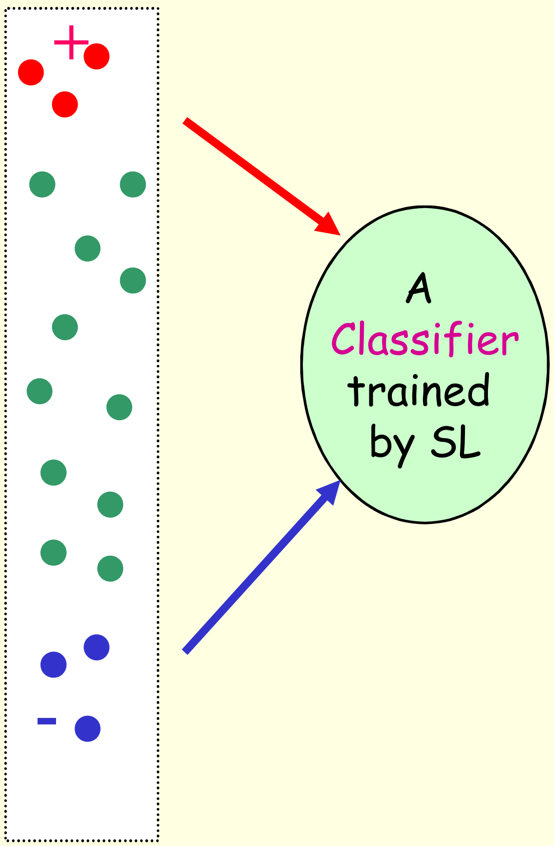
The Yarowsky Algorithm

Iteration: 0



The Yarowsky Algorithm

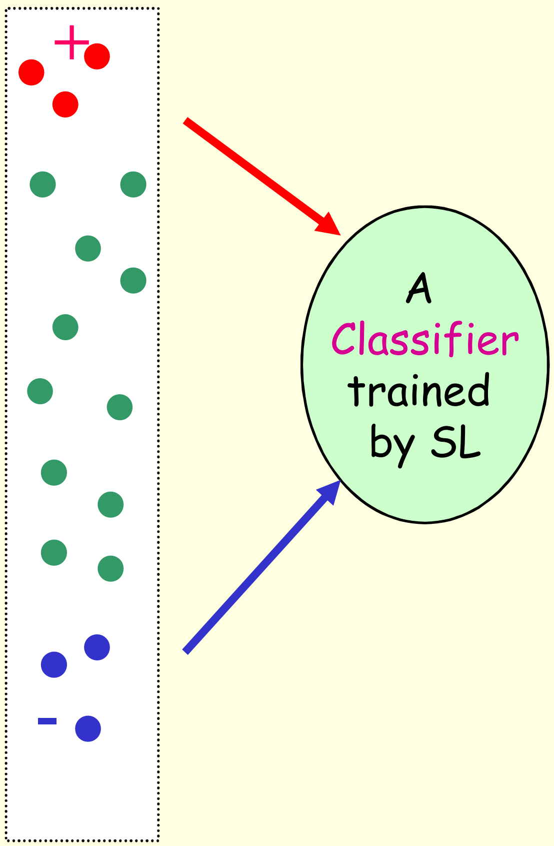
Iteration: 0



Choose instances labeled with high confidence

The Yarowsky Algorithm

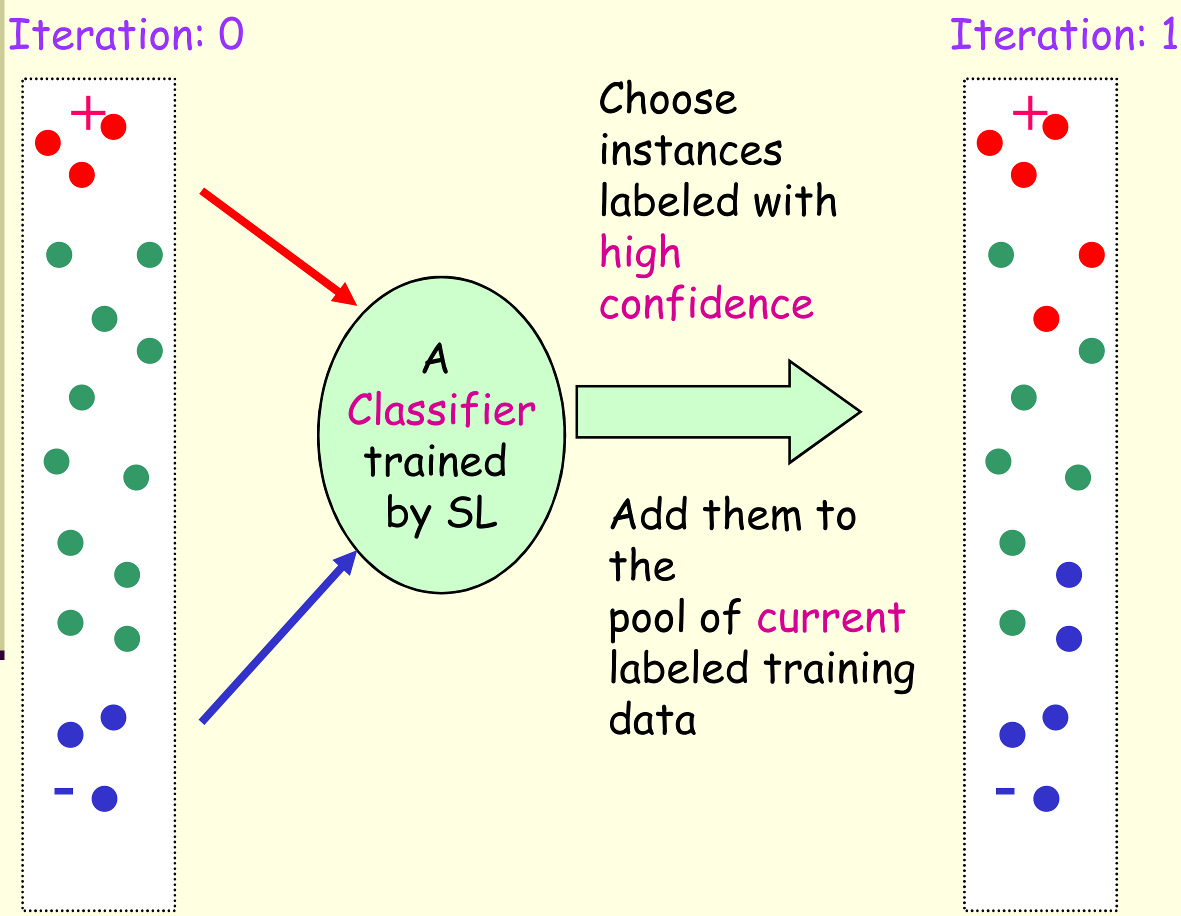
Iteration: 0



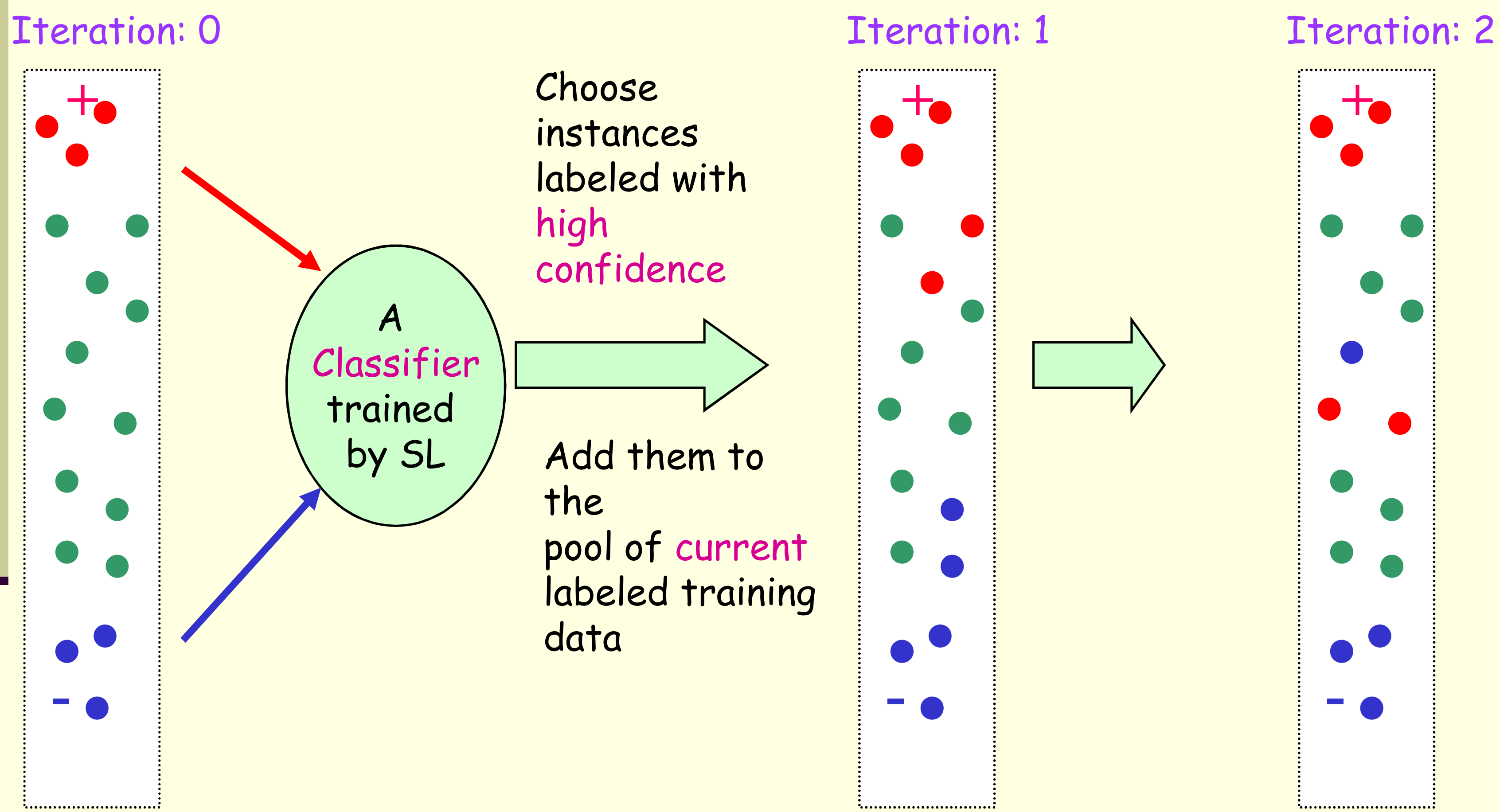
Choose instances labeled with high confidence

Add them to the pool of current labeled training data

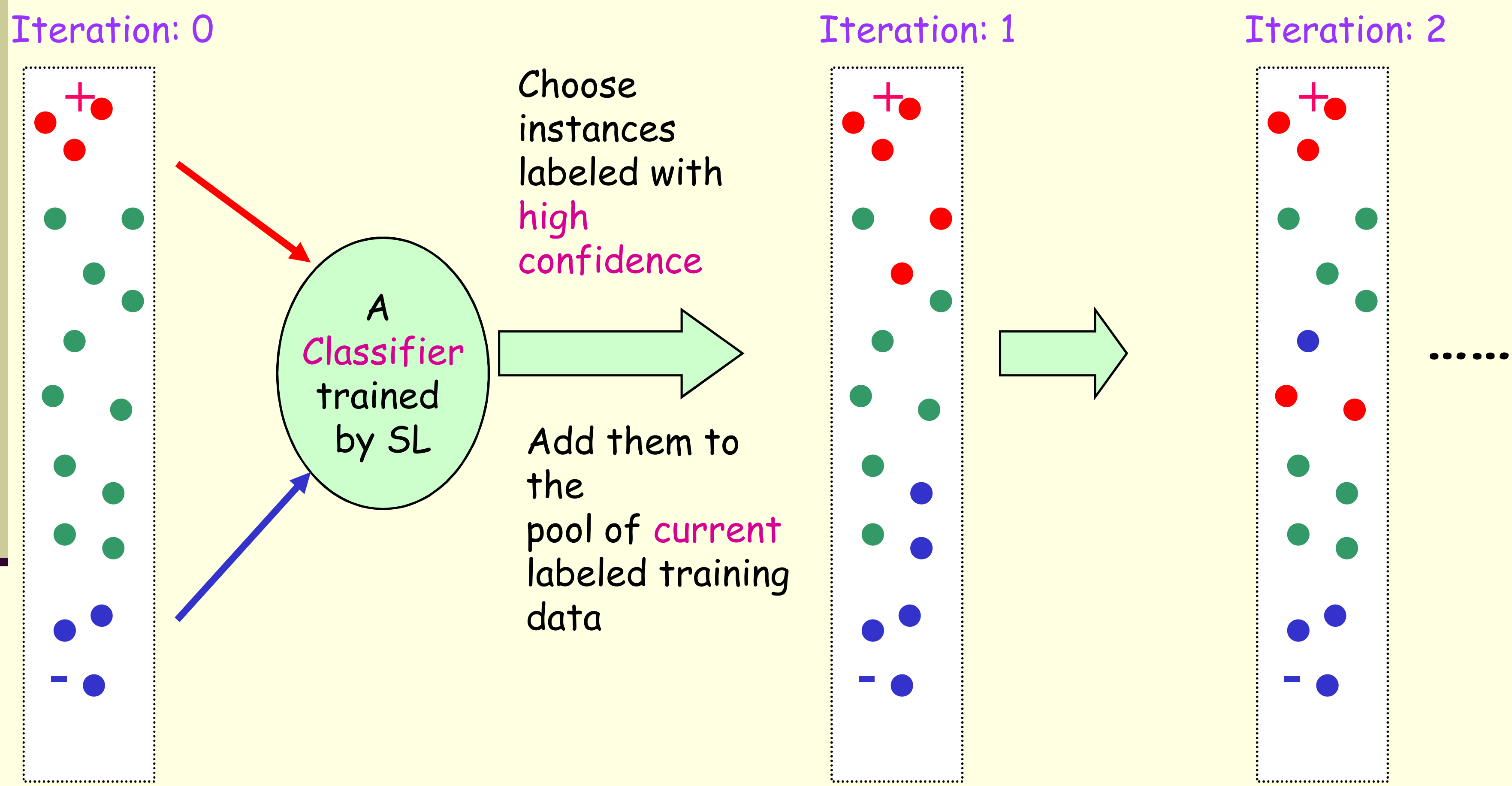
The Yarowsky Algorithm



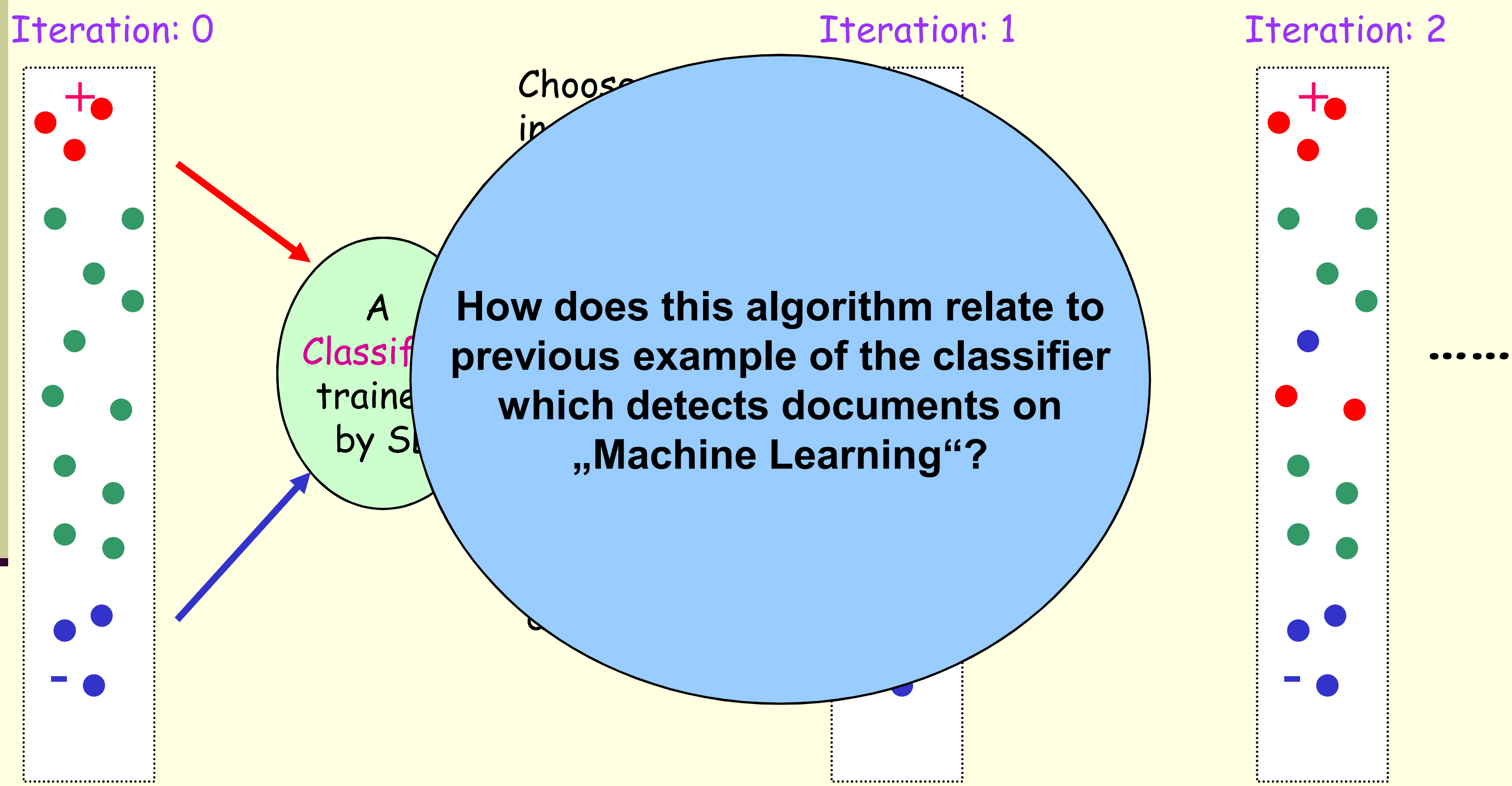
The Yarowsky Algorithm



The Yarowsky Algorithm



The Yarowsky Algorithm



How does this algorithm relate to previous example of the classifier which detects documents on „Machine Learning“?

The Expectation Maximization (EM) Algorithm

Expectation Maximization Algorithm

- The EM algorithm is a **meta algorithm** that can be applied to any probabilistic model which depends on *unobserved/hidden* variables
- We consider the derivation for a ***Multinomial Naive Bayes*** classifier in this lecture
- The standard supervised version was presented last lecture!

Expectation Maximization Algorithm

- Conceptional Idea:
  1. Estimate a model from the labeled data
  2. Label the unlabeled data using current model
  3. Re-estimate the model incl. the newly labeled data from Step 2
  4. Repeat Steps 2-3 until convergence has been reached
- See also (Dempster1977)

Expectation Maximization Algorithm

- Conceptional Idea:
  1. Estimate a model from the labeled data
  2. Label the unlabeled data using current model ***(E-Step)***
  3. Re-estimate the model incl. the newly labeled data from Step 2
  4. Repeat Steps 2-3 until convergence has been reached
- See also (Dempster1977)

Expectation Maximization Algorithm

- Conceptional Idea:
  1. Estimate a model from the labeled data
  2. Label the unlabeled data using current model ***(E-Step)***
  3. Re-estimate the model incl. the newly labeled data from Step 2 ***(M-Step)***
  4. Repeat Steps 2-3 until convergence has been reached
- See also (Dempster1977)

Notation

- The set of classes is C and a specific class is denoted by c_i
- The set of documents is D and a specific document is denoted by d_j
- The set of documents D can be divided into the set of labeled documents D^l and unlabeled documents D^u (specific documents are d^l and d^u, respectively)
- The class of a labeled document d^l is denoted by c_{d^l}
- The vocabulary is V and a specific word is denoted by x_k

Expectation Maximization Algorithm

E-Step: $$P(c_i \mid d_j) = \frac{P(c_i) \cdot P(d_j \mid c_i)}{P(d_j)}$$

Expectation Maximization Algorithm

E-Step: $$P(c_i \mid d_j) = \frac{P(c_i) \cdot P(d_j \mid c_i)}{P(d_j)}$$

Bayes Theorem

Expectation Maximization Algorithm

E-Step:
$$P(c_i \mid d_j) = \frac{P(c_i) \cdot P(d_j \mid c_i)}{P(d_j)} = \frac{P(c_i) \cdot P(d_j \mid c_i)}{\sum_{l=1}^{|C|} P(c_l) \cdot P(d_j \mid c_l)}$$

Expectation Maximization Algorithm

E-Step: $$P(c_i \mid d_j) = \frac{P(c_i) \cdot P(d_j \mid c_i)}{P(d_j)} = \frac{P(c_i) \cdot P(d_j \mid c_i)}{\sum_{l=1}^{|C|} P(c_l) \cdot P(d_j \mid c_l)}$$

Multiplication Rule

Expectation Maximization Algorithm

E-Step:

$$P(c_i \mid d_j) = \frac{P(c_i)\cdot P(d_j \mid c_i)}{P(d_j)} = \frac{P(c_i)\cdot P(d_j \mid c_i)}{\sum_{l=1}^{|C|} P(c_l)\cdot P(d_j \mid c_l)} = \frac{P(c_i)\cdot \prod_{x_k \in d_j} P(x_k \mid c_i)}{\sum_{l=1}^{|C|} P(c_l)\cdot \prod_{x_k \in d_j} P(x_k \mid c_l)}$$

Expectation Maximization Algorithm

E-Step: $$P(c_i \mid d_j) = \frac{P(c_i)\cdot P(d_j \mid c_i)}{P(d_j)} = \frac{P(c_i)\cdot P(d_j \mid c_i)}{\sum_{l=1}^{|C|} P(c_l)\cdot P(d_j \mid c_l)} = \frac{P(c_i)\cdot \prod_{x_k \in d_j} P(x_k \mid c_i)}{\sum_{l=1}^{|C|} P(c_l)\cdot \prod_{x_k \in d_j} P(x_k \mid c_l)}$$

Independence Assumption of Words in a Document

Expectation Maximization Algorithm

E-Step: $$P(c_i \mid d_j) = \frac{P(c_i) \cdot P(d_j \mid c_i)}{P(d_j)} = \frac{P(c_i) \cdot P(d_j \mid c_i)}{\sum_{l=1}^{|C|} P(c_l) \cdot P(d_j \mid c_l)} = \frac{P(c_i) \cdot \prod_{x_k \in d_j} P(x_k \mid c_i)}{\sum_{l=1}^{|C|} P(c_l) \cdot \prod_{x_k \in d_j} P(x_k \mid c_l)}$$

At iteration 0:

- All $P(c_i)$ and $P(x_k|c_i)$ are directly estimated from the labeled data
- **No** information is drawn from the unlabeled data yet
- Initial estimates of $P(x_k|c_i)$ heavily rely on smoothing

Expectation Maximization Algorithm

E-Step: $$P(c_i \mid d_j) = \frac{P(c_i) \cdot P(d_j \mid c_i)}{P(d_j)} = \frac{P(c_i) \cdot P(d_j \mid c_i)}{\sum_{l=1}^{|C|} P(c_l) \cdot P(d_j \mid c_l)} = \frac{P(c_i) \cdot \prod_{x_k \in d_j} P(x_k \mid c_i)}{\sum_{l=1}^{|C|} P(c_l) \cdot \prod_{x_k \in d_j} P(x_k \mid c_l)}$$

M-Step: $$P(x_k \mid c_i) = \frac{\sum_{j=1}^{|D|} N(x_k, d_j) \cdot P(c_i \mid d_j)}{Z(c_i)}$$

Expectation Maximization Algorithm

E-Step:

$$P(c_i \mid d_j) = \frac{P(c_i) \cdot P(d_j \mid c_i)}{P(d_j)} = \frac{P(c_i) \cdot P(d_j \mid c_i)}{\sum_{l=1}^{|C|} P(c_l) \cdot P(d_j \mid c_l)} = \frac{P(c_i) \cdot \prod_{x_k \in d_j} P(x_k \mid c_i)}{\sum_{l=1}^{|C|} P(c_l) \cdot \prod_{x_k \in d_j} P(x_k \mid c_l)}$$

M-Step:

$$P(x_k \mid c_i) = \frac{\sum_{j=1}^{|D|} N(x_k, d_j) \cdot P(c_i \mid d_j)}{Z(c_i)} = \frac{\sum_{j=1}^{|D|} N(x_k, d_j) \cdot P(c_i \mid d_j)}{\sum_{n=1}^{|V|} \sum_{m=1}^{|D|} N(x_n, d_m) \cdot P(c_i \mid d_m)}$$

Expectation Maximization Algorithm

E-Step: $$P(c_i \mid d_j) = \frac{P(c_i)\cdot P(d_j \mid c_i)}{P(d_j)} = \frac{P(c_i)\cdot P(d_j \mid c_i)}{\sum_{l=1}^{|C|} P(c_l)\cdot P(d_j \mid c_l)} = \frac{P(c_i)\cdot \prod_{x_k \in d_j} P(x_k \mid c_i)}{\sum_{l=1}^{|C|} P(c_l)\cdot \prod_{x_k \in d_j} P(x_k \mid c_l)}$$

M-Step: $$P(x_k \mid c_i) = \frac{\sum_{j=1}^{|D|} N(x_k, d_j)\cdot P(c_i \mid d_j)}{Z(c_i)} = \frac{\sum_{j=1}^{|D|} N(x_k, d_j)\cdot P(c_i \mid d_j)}{\sum_{n=1}^{|V|}\sum_{m=1}^{|D|} N(x_n, d_m)\cdot P(c_i \mid d_m)}$$

$$P(c_i) = \frac{\sum_{j=1}^{|D|} P(c_i \mid d_j)}{Z}$$

Expectation Maximization Algorithm

E-Step:
$$P(c_i \mid d_j) = \frac{P(c_i)\cdot P(d_j \mid c_i)}{P(d_j)} = \frac{P(c_i)\cdot P(d_j \mid c_i)}{\sum_{l=1}^{|C|} P(c_l)\cdot P(d_j \mid c_l)} = \frac{P(c_i)\cdot \prod_{x_k \in d_j} P(x_k \mid c_i)}{\sum_{l=1}^{|C|} P(c_l)\cdot \prod_{x_k \in d_j} P(x_k \mid c_l)}$$

M-Step:
$$P(x_k \mid c_i) = \frac{\sum_{j=1}^{|D|} N(x_k, d_j)\cdot P(c_i \mid d_j)}{Z(c_i)} = \frac{\sum_{j=1}^{|D|} N(x_k, d_j)\cdot P(c_i \mid d_j)}{\sum_{n=1}^{|V|}\sum_{m=1}^{|D|} N(x_n, d_m)\cdot P(c_i \mid d_m)}$$

$$P(c_i) = \frac{\sum_{j=1}^{|D|} P(c_i \mid d_j)}{Z} = \frac{\sum_{j=1}^{|D|} P(c_i \mid d_j)}{\sum_{l=1}^{|C|}\sum_{m=1}^{|D|} P(c_l \mid d_m)}$$

Expectation Maximization Algorithm

E-Step:
$$P(c_i \mid d_j) = \frac{P(c_i)\cdot P(d_j \mid c_i)}{P(d_j)} = \frac{P(c_i)\cdot P(d_j \mid c_i)}{\sum_{l=1}^{|C|} P(c_l)\cdot P(d_j \mid c_l)} = \frac{P(c_i)\cdot \prod_{x_k \in d_j} P(x_k \mid c_i)}{\sum_{l=1}^{|C|} P(c_l)\cdot \prod_{x_k \in d_j} P(x_k \mid c_l)}$$

M-Step:
$$P(x_k \mid c_i) = \frac{\sum_{j=1}^{|D|} N(x_k, d_j)\cdot P(c_i \mid d_j)}{Z(c_i)} = \frac{\sum_{j=1}^{|D|} N(x_k, d_j)\cdot P(c_i \mid d_j)}{\sum_{n=1}^{|V|}\sum_{m=1}^{|D|} N(x_n, d_m)\cdot P(c_i \mid d_m)}$$

$$P(c_i) = \frac{\sum_{j=1}^{|D|} P(c_i \mid d_j)}{Z} = \frac{\sum_{j=1}^{|D|} P(c_i \mid d_j)}{\sum_{l=1}^{|C|}\sum_{m=1}^{|D|} P(c_l \mid d_m)} = \frac{\sum_{j=1}^{|D|} P(c_i \mid d_j)}{|D|}$$

Expectation Maximization Algorithm

E-Step:

$$P(c_i \mid d_j) = \frac{P(c_i)\cdot P(d_j \mid c_i)}{P(d_j)} = \frac{P(c_i)\cdot P(d_j \mid c_i)}{\sum_{l=1}^{|C|} P(c_l)\cdot P(d_j \mid c_l)} = \frac{P(c_i)\cdot \prod_{x_k \in d_j} P(x_k \mid c_i)}{\sum_{l=1}^{|C|} P(c_l)\cdot \prod_{x_k \in d_j} P(x_k \mid c_l)}$$

M-Step:

$$P(x_k \mid c_i) = \frac{\sum_{j=1}^{|D|} N(x_k, d_j)\cdot P(c_i \mid d_j)}{\sum_{n=1}^{|V|}\sum_{m=1}^{|D|} N(x_n, d_m)\cdot P(c_i \mid d_m)}$$

Use $P(x_k|c_i)$ and $P(c_i)$ from iteration *n* for the estimation of $P(c_i|d_j)$ at iteration *n+1*

$$P(c_i) = \frac{\sum_{j=1}^{|D|} P(c_i \mid d_j)}{|D|}$$

Expectation Maximization Algorithm

- After each interation compute Likelihood of the entire dataset *L(D)* with current model:

$$L(D) = \prod_{j=1}^{|D^l|} P(c_{d_j^l}) P(d_j^l \mid c_{d_j^l}) \prod_{n=1}^{|D^u|} \sum_{i=1}^{|C|} P(c_i) P(d_n^u \mid c_i)$$

Expectation Maximization Algorithm

- After each interation compute Likelihood of the entire dataset *L(D)* with current model:

$$L(D) = \prod_{j=1}^{|D^l|} P(c_{d_j^l}) P(d_j^l \mid c_{d_j^l}) \prod_{n=1}^{|D^u|} \sum_{i=1}^{|C|} P(c_i) P(d_n^u \mid c_i)$$

For labeled documents only use the actual class the document has been labeled with

Expectation Maximization Algorithm

- After each interation compute Likelihood of the entire dataset *L(D)* with current model:

$$L(D) = \prod_{j=1}^{|D^l|} P(c_{d_j^l}) P(d_j^l \mid c_{d_j^l}) \prod_{n=1}^{|D^u|} \sum_{i=1}^{|C|} P(c_i) P(d_n^u \mid c_i)$$

For unlabeled documents use the weighted sum over all classes

Expectation Maximization Algorithm

- After each interation compute Likelihood of the entire dataset *L(D)* with current model:

$$L(D) = \prod_{j=1}^{|D^l|} P(c_{d_j^l}) P(d_j^l \mid c_{d_j^l}) \prod_{n=1}^{|D^u|} \sum_{i=1}^{|C|} P(c_i) P(d_n^u \mid c_i)$$

- Iterate until Likelihood converges
- Alternatively: fix number of iterations

EM – What actually happens

- Initialization:
  - **Problem 1:** Many words in the vocabulary are not observed in the labeled training set → they are assigned a *low* back-off probability (probability is too low for predictive words!)
  - **Problem 2:** Other words occurring in the labeled training set might have received a too high probability
- Iteration:
  - Solution to Problem 1:
    - Use correlation among features to determine which words only observed in the unlabeled dataset also correlate with the different classes
    - $P(x_j|c_i)$ *(initially estimated with back-off!)* will increase during model re-estimation for these features
  - Solution to Problem 2:
    - Hopefully words which have occurred disproportionately frequently in the labeled data will be less often observed in the unlabeled training set
    - $P(x_j|c_i)$ should gradually decrease

EM - What actually happens

- Experiments on the **WebKB** dataset from (Nigam2000)
- Webpages gathered from computer science departments
- Subset used in these experiments:
  - Classes: *student*, *faculty*, *course*, and *project*
  - Approximately 4200 webpages
- 2500 documents are used as unlabeled data
- Iteration 0 uses only 1 labeled data instance per class

Highest ranked words in class *course* throughout different iterations

Iteration 0	Iteration 1	Iteration 2
intelligence	*DD*	*D*
DD	*D*	*DD*
artificial	lecture	lecture
understanding	cc	cc
*DD*w	*D**	*DD:DD*
dist	*DD:DD*	due
identical	handout	*D**
rus	due	homework
arrange	problem	assignment
games	set	handout
dartmouth	tay	set
natural	*DD*am	hw
cognitive	yurttas	exam
logic	homework	problem
proving	kfoury	*DD*am
prolog	sec	postscript
knowledge	postscript	solution
human	exam	quiz
representation	solution	chapter
field	assaf	ascii

Highest ranked words in class *course* throughout different iterations

Iteration 0	Iteration 1	Iteration 2
intelligence	*DD*	*D*
DD	*D*	*DD*
artificial	lecture	lecture
understanding	cc	cc
*DD*w	*D**	*DD:DD*
dist	*DD:DD*	due
identical	handout	*D**
rus	due	homework
arrange	problem	assignment
games	set	handout
dartmouth	tay	set
natural	*DD*am	hw
cognitive	yurttas	exam
logic	homework	problem
proving	kfoury	*DD*am
prolog	sec	postscript
knowledge	postscript	solution
human	exam	quiz
representation	solution	chapter
field	assaf	ascii

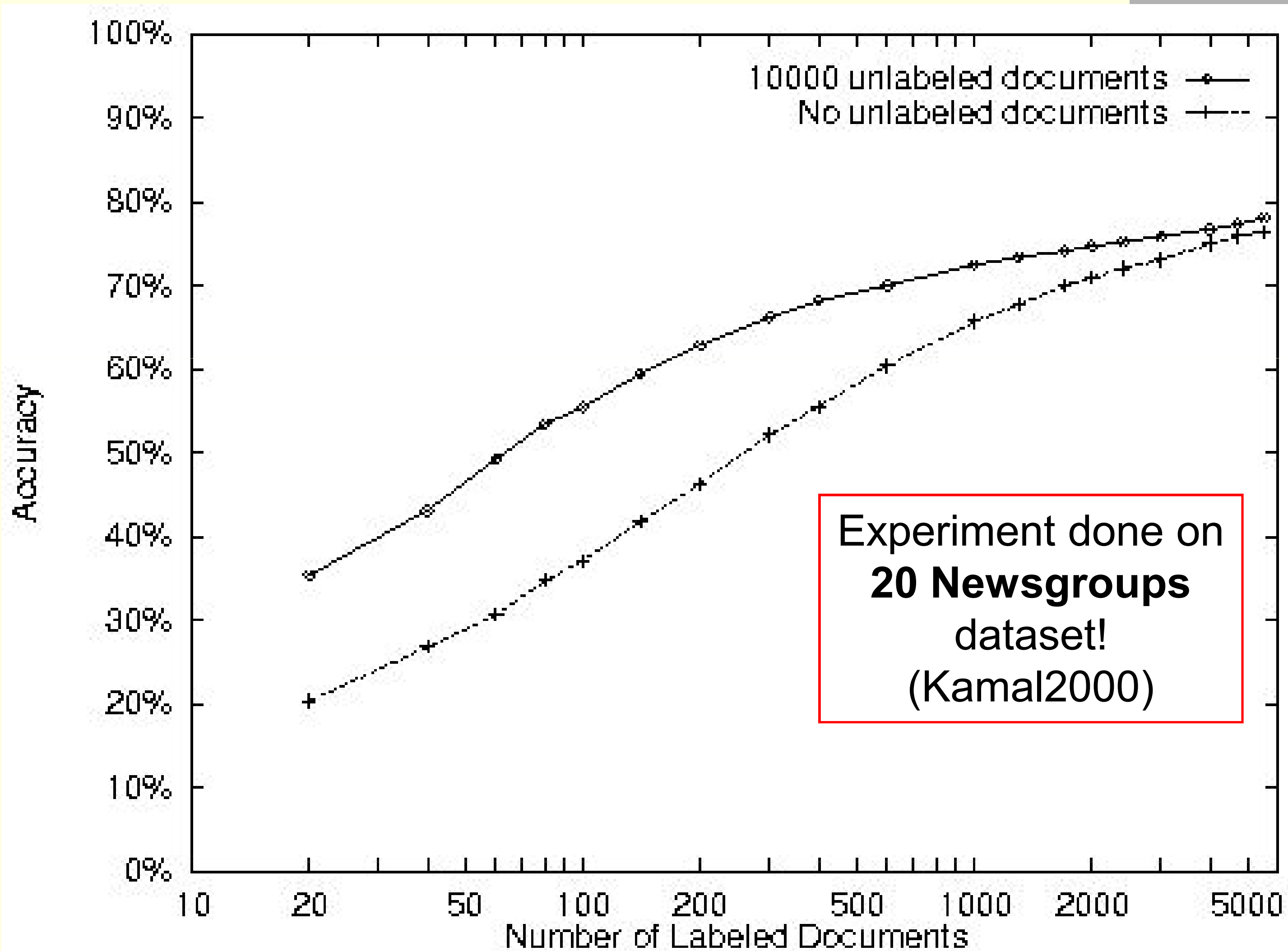
Terms with **no** general significance for the class to be modeled

Highest ranked words in class *course* throughout different iterations

Iteration 0	Iteration 1	Iteration 2
intelligence	DD	D
DD	D	DD
artificial	lecture	lecture
understanding	cc	cc
DDw	D*	$DD{:}DD$
dist	$DD{:}DD$	due
identical	handout	D*
rus	due	homework
arrange	problem	assignment
games	set	handout
dartmouth	tay	set
natural	DDam	hw
cognitive	yurttas	exam
logic	homework	problem
proving	kfoury	DDam
prolog	sec	postscript
knowledge	postscript	solution
human	exam	quiz
representation	solution	chapter
field	assaf	ascii

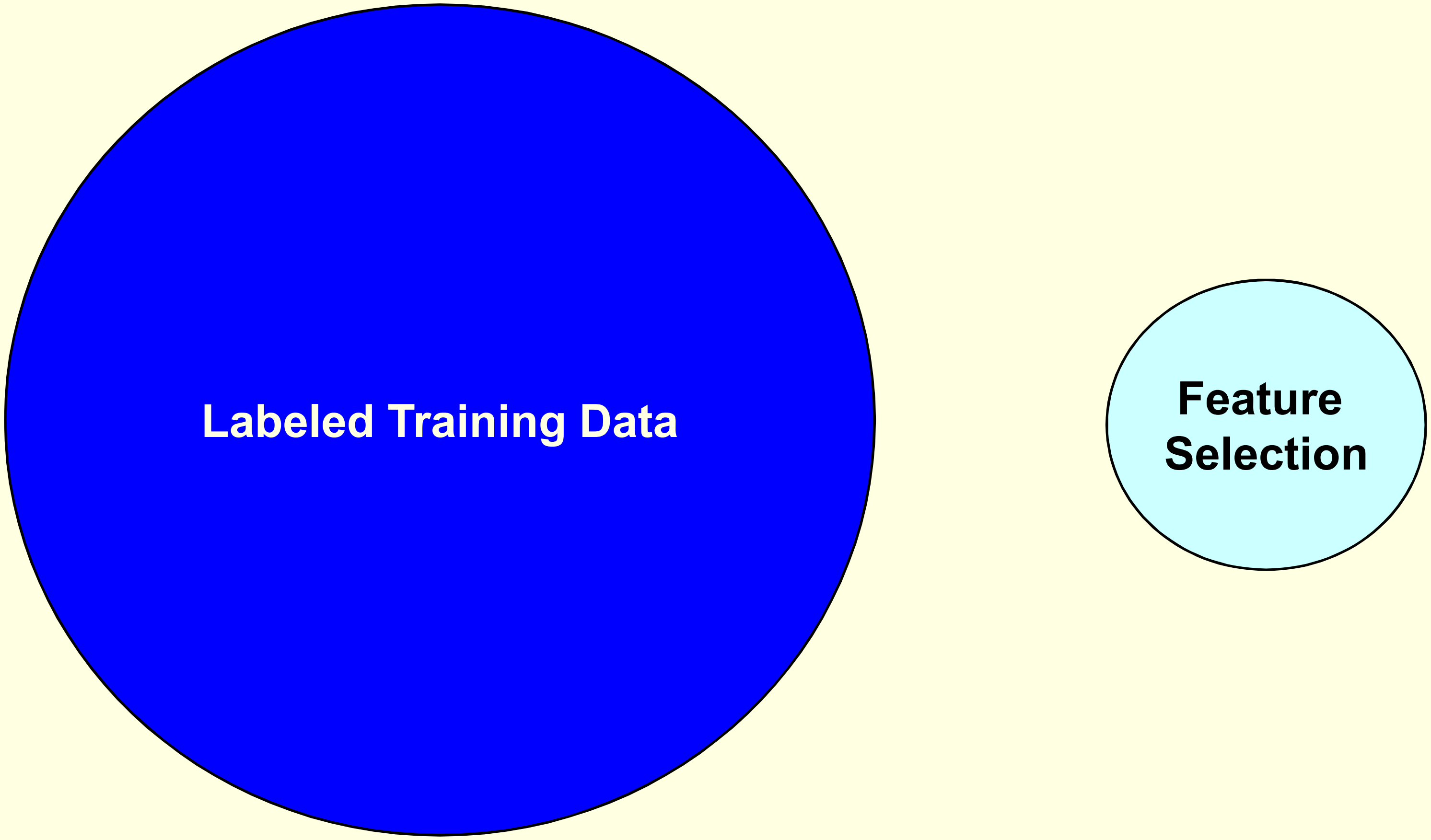
Terms with general significance for the class to be modeled

Improvement of Semi-Supervised Learning Using Different Amounts of Labeled Documents

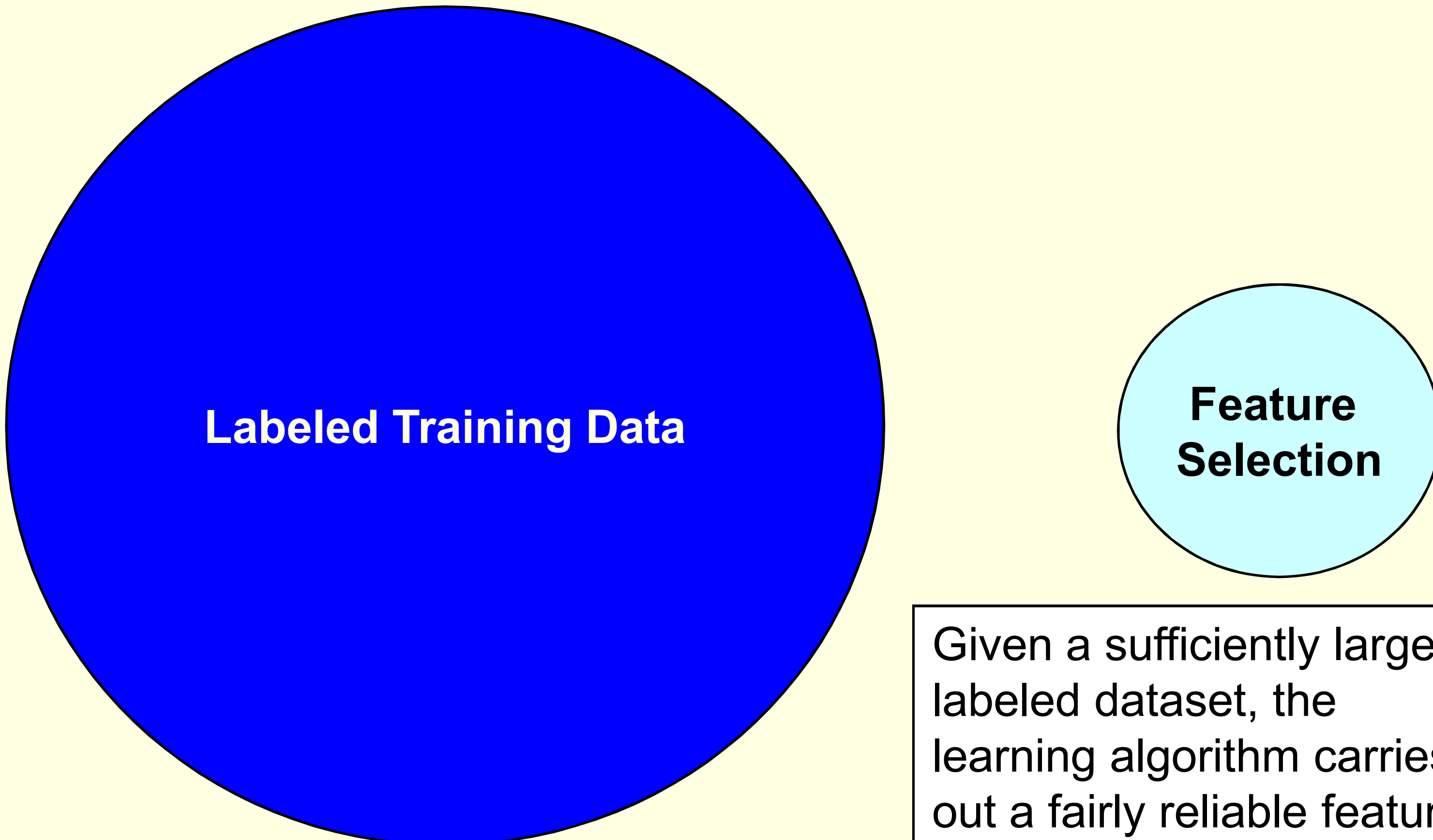


The Importance of Feature Selection in Semi-Supervised Learning (on Text Classification)

The Relation between Labeled Training Data and Feature Selection in **Supervised Learning** on Text Classification

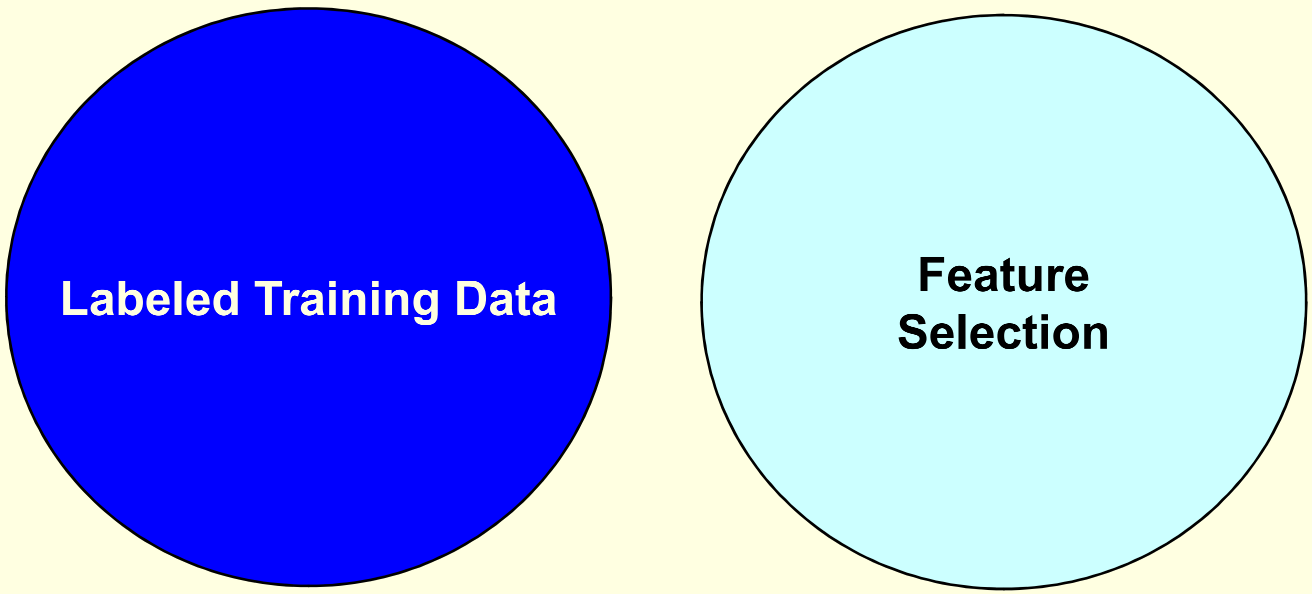


The Relation between Labeled Training Data and Feature Selection in **Supervised Learning** on Text Classification

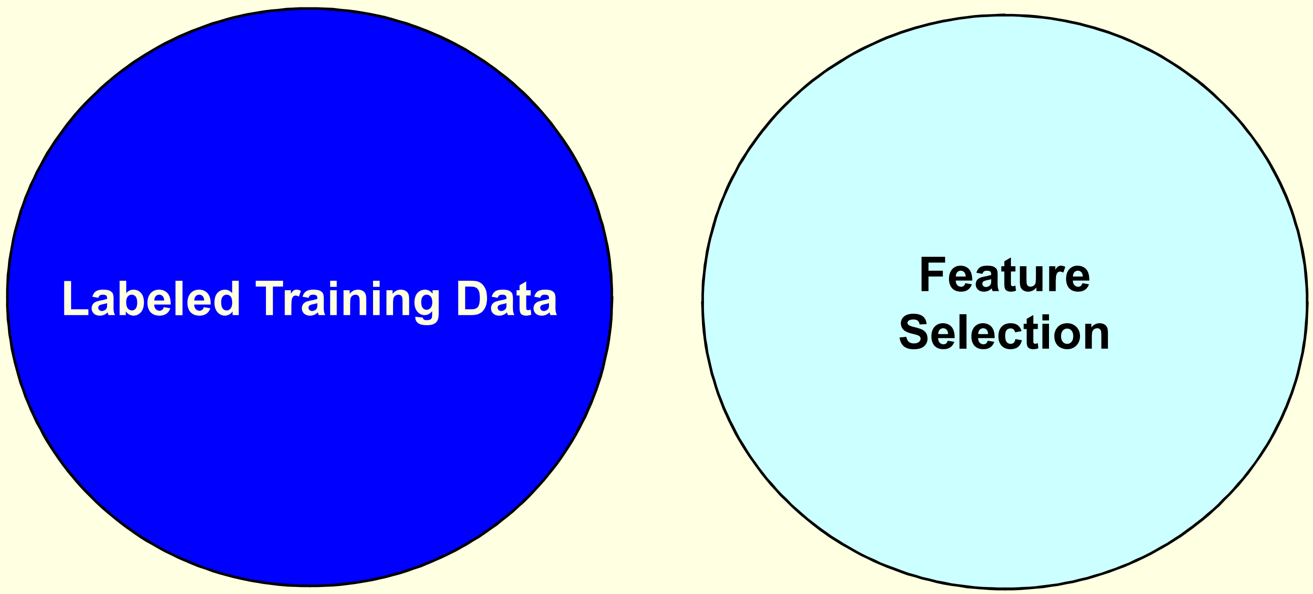


Given a sufficiently large labeled dataset, the learning algorithm carries out a fairly reliable feature selection internally.

The Relation between Labeled Training Data and Feature Selection in **Semi-Supervised Learning** on Text Classification



The Relation between Labeled Training Data and Feature Selection in **Semi-Supervised Learning** on Text Classification



In SSL, the learning algorithm is less robust and a separate feature selection is more important.

A Unified Representation of Machine Learning Classifiers

- Most Machine Learning classifiers learn a function *g* which is a linear combination of weighted features:

$$g(\vec{x}) = x_1 \cdot w_1 + x_2 \cdot w_2 + ... + x_n w_n \; (+b)$$

- *g* is transformed into a binary classifier:

$$if \; g(\vec{x}) > \delta \; then \; c_1 \; else \; c_2$$

A Unified Representation of Machine Learning Classifiers

- Most Machine Learning classifiers learn a function *g* which is a linear combination of weighted features:

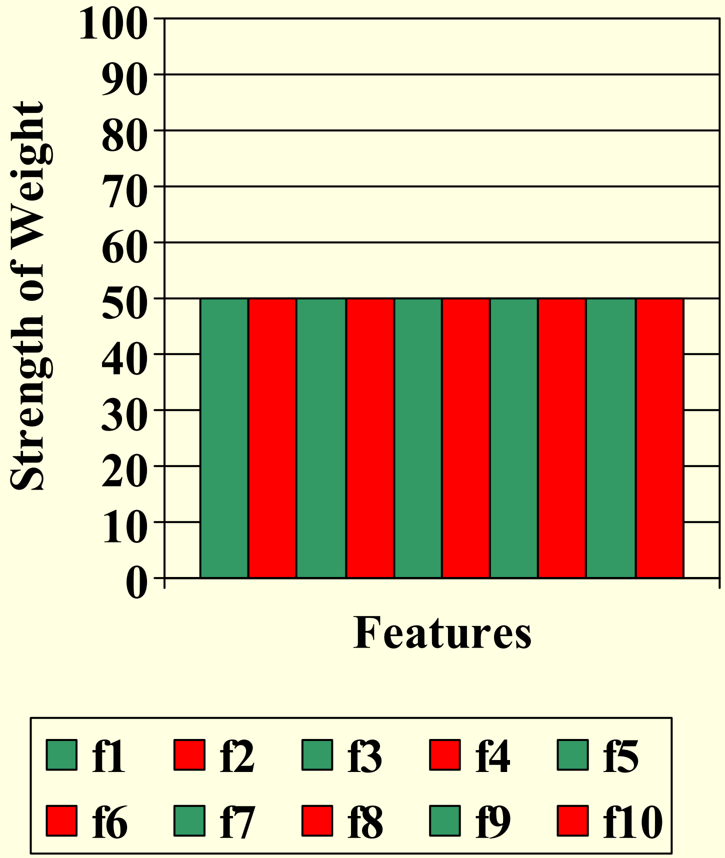
$$g(\vec{x}) = x_1 \cdot w_1 + x_2 \cdot w_2 + \ldots + x_n w_n \ (+b)$$

- *g* is transformed into a binary classifier:

$$if \ g(\vec{x}) > \delta \ then \ c_1 \ else \ c_2$$

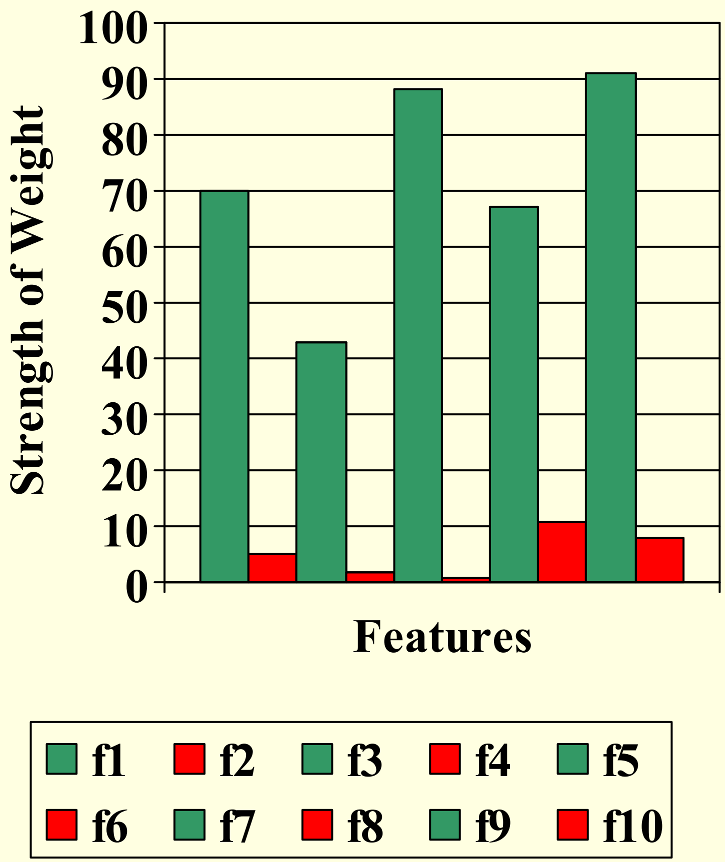
δ is a threshold value

Feature Weights and Feature Selection



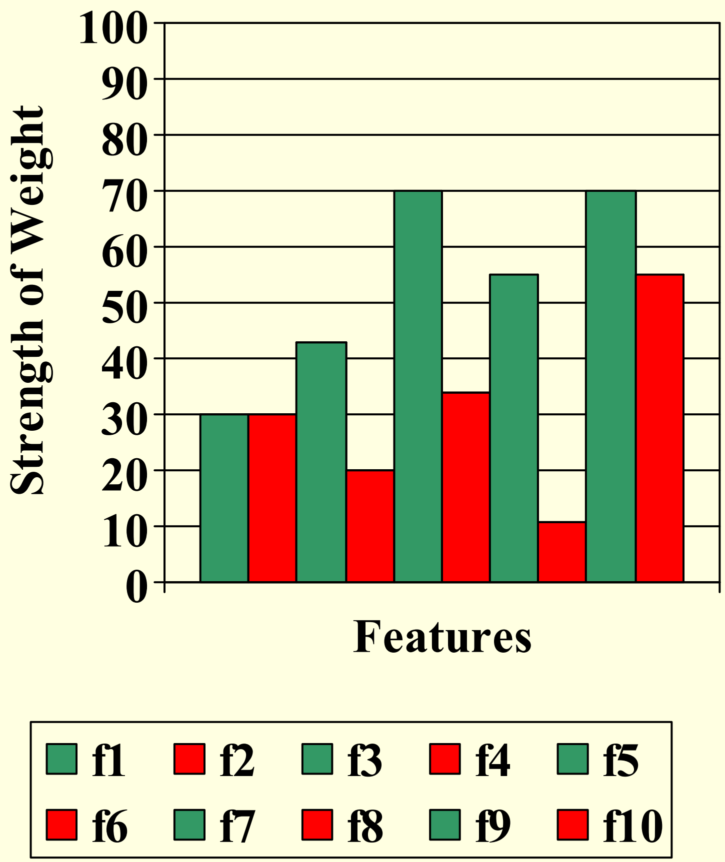
- Figure left displays features
- **Green** features are discriminative (helpful) features
- **Red** features are noisy (obstructive) features

Feature Weights and Feature Selection



- In Supervised Learning there are plenty of labeled data instances
- Feature weights are estimated very reliably
- Discriminative features obtain a high weight
- Noisy features obtain a low weight

Feature Weights and Feature Selection

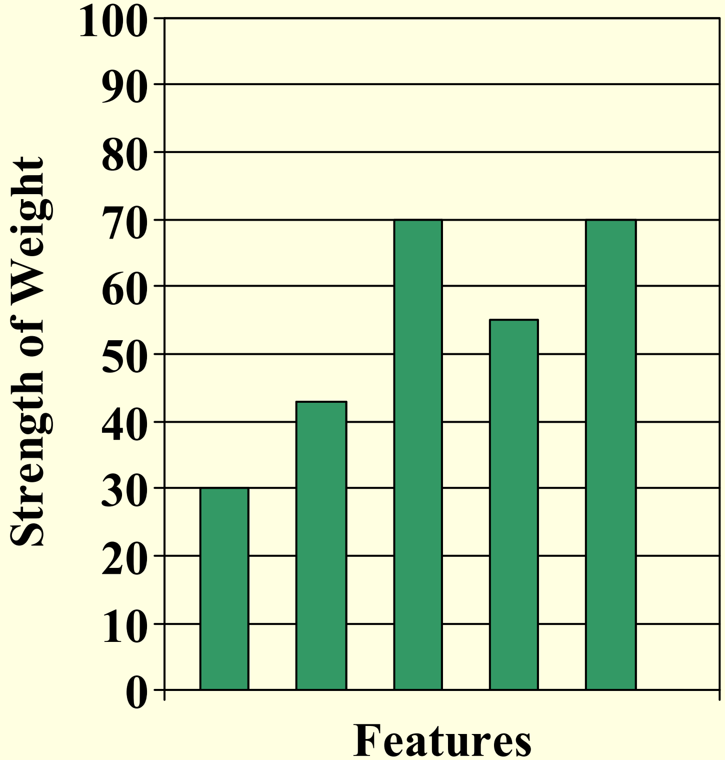


- In Semi-Supervised Learning there are only few labeled data instances available
- Noisy data features may not be properly downweighted
- Noisy features may lead classifier astray during bootstrapping

What does „Leading Astray“ Mean?

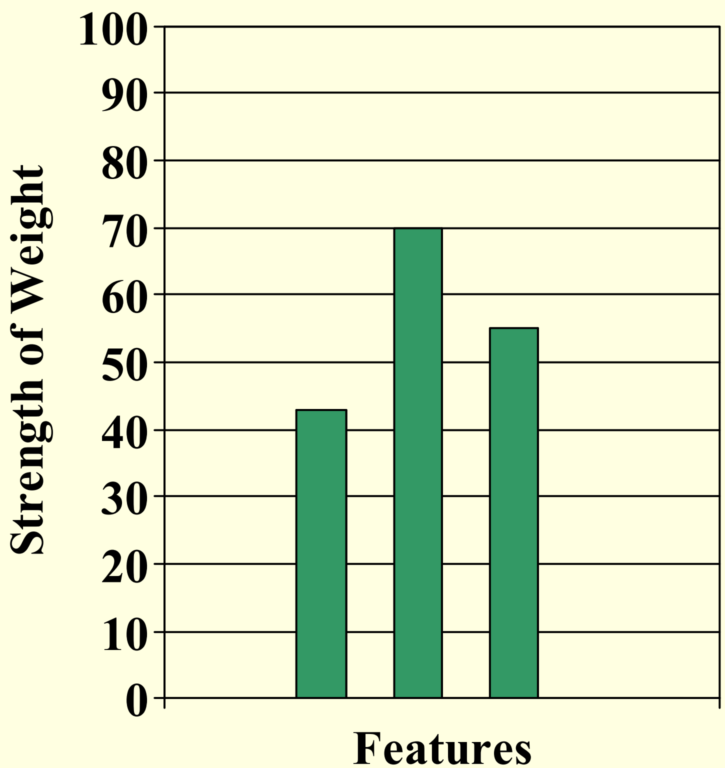
- Imagine a bad feature set applied to EM
- The classifier considers feature x_i a good predictor of class c_j because it is only co-occurring in labeled instances of this class
- However this co-occurrence is *coincidental* (remember the labeled dataset is usually very small in SSL) → feature x_i is a bad feature
- In subsequent iterations other features co-occurring with bad feature x_i will also be inferred to be predictive for c_j, but this is actually wrong and will degrade the performance of the classifier

Feature Weights and Feature Selection



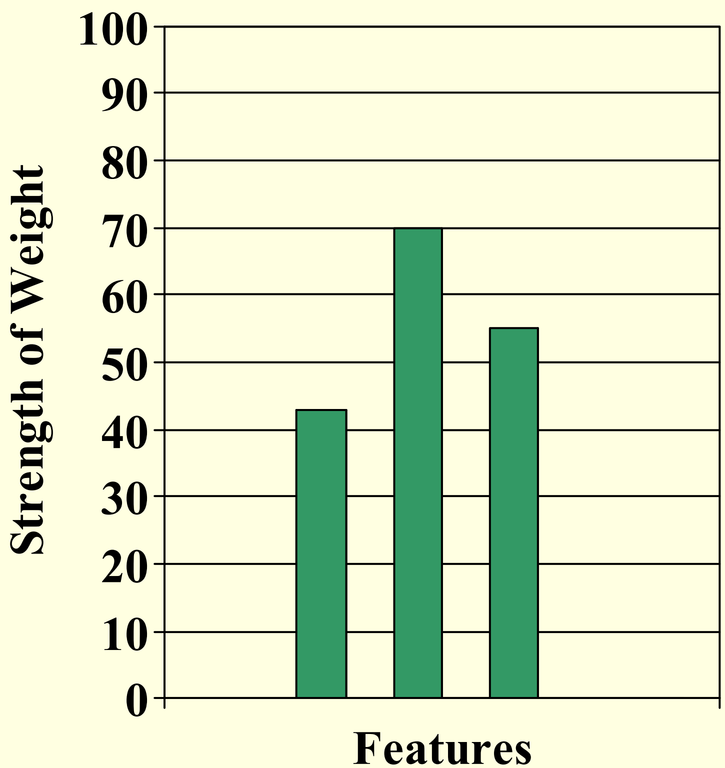
- Solution: use a good feature set, i.e. a feature set with only discriminative features

Feature Weights and Feature Selection



- Solution: use a good feature set, i.e. a feature set with only discriminative features
- Feature selection can be fairly restrictive, so that some discriminative features get lost as well

Feature Weights and Feature Selection



- Solution: use a good feature set, i.e. a feature set with only discriminative features
- Feature selection can be fairly restrictive, so that some discriminative features get lost as well
- But that is still better for SSL than using all features!!!

How can feature selection be done in SSL on text classification

- Correlation-based feature selection methods (e.g. *Point-wise Mutual Information*) do not work well in SSL, since too few labeled instances are available
- Stopword removal may help (i.e. download a list of function words from the web)
- Only consider frequent words in your entire data-set (e.g. Top 2000 words)
- Use your prior knowledge and construct your feature set manually (in case this is cheaper than providing more labeled data instances, otherwise try supervised learning!)

Applications of Semi-Supervised Learning in NLP

- Text Classification
- Part-of-Speech Tagging
- Syntactic Parsing
- Word Sense Disambiguation
- Information Extraction (e.g. Relation Extraction)
- Machine Translation

Other state-of-the-art algorithms

- Extensions to EM (Kamal2000)
  - Lambda-EM (weighting unlabeled and labeled data)
  - M-EM (i.e. with multiple mixture components)
- Co-Training (Blum1998)
- Transductive Support Vector Machines (Joachims1999)
- Label Propagation (Niu2005)
- Spectral Graph Clustering (Joachims2003)

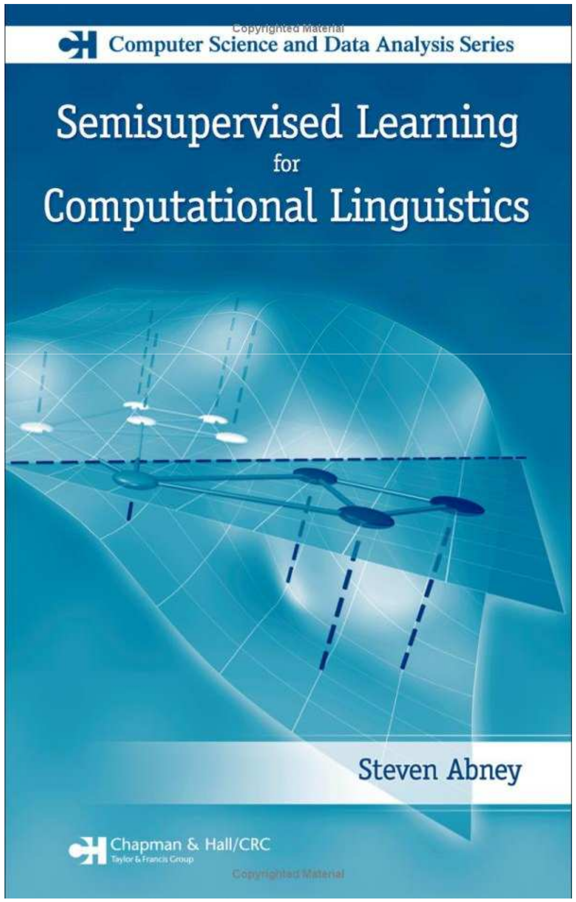
A Word of Warning

- Semi-Supervised Learning does not always work!
  - Classification performance of initial model might be too low (bootstrapping only adds further noise)
  - Classifier from initial (supervised) model might already produce maximal performance
- There are more degrees of freedom that have to be taken into account:
  - Size of the feature set
  - Size of the unlabeled data set
  - Many classifier-specific parameters!

Summary

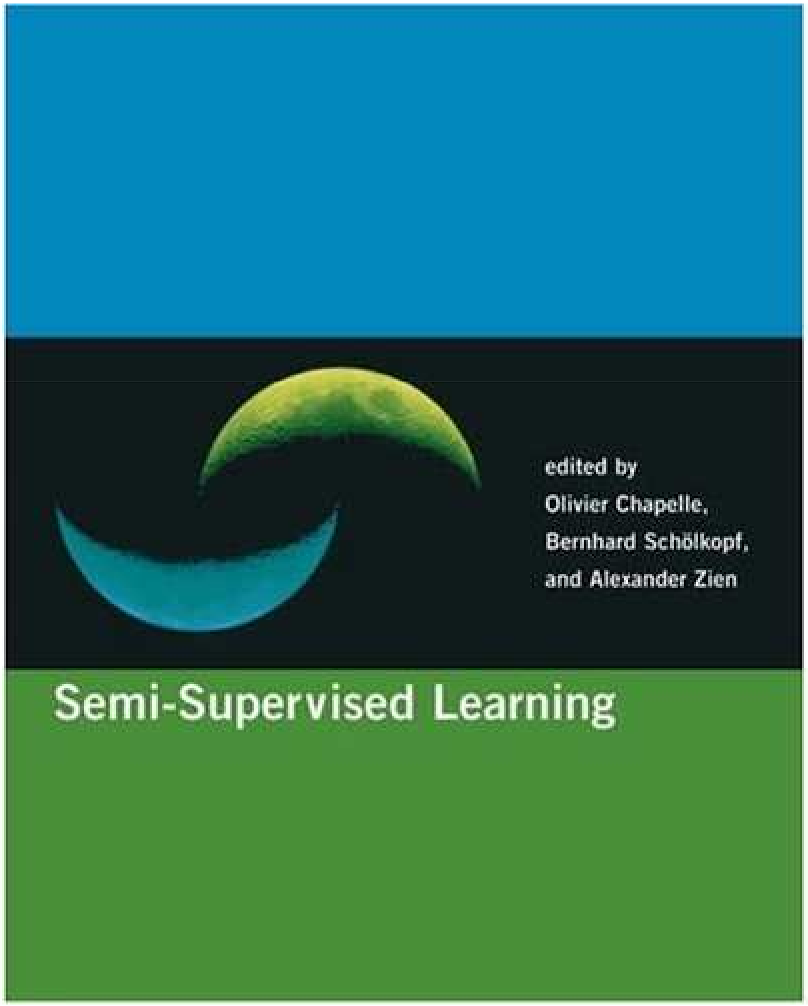
- Semi-Supervised Learning works well when only few labeled data are available
- Most Semi-Supervised Learning algorithms are *bootstrapping algorithms*
- Feature selection is more important in Semi-Supervised Learning than in Supervised Learning (on text classification)
- Bad feature sets may lead classifier astray

Relevant Books



Semisupervised Learning for Computational Linguistics
by *Steven Abney*
Chapman & Hall
2007

Relevant Books



Semi-Supervised Learning
by *Olivier Chapelle, Bernhard Schölkopf, Alexander Zien* (Editors)
MIT Press
2006

References

- ***Maximum Likelihood from Incomplete Data via the EM Algorithm.*** A. Dempster, N. Laird, and D. Rubin. *Journal of the Royal Statistical Society 1977.*
- ***Combining Labeled and Unlabeled Data with Co-Training.*** A. Blum and T. Mitchell, *1998*.
- ***Transductive Inference for Text Classification using Support Vector Machines*****.** T. Joachims. *Proceedings of ICML 1999.*
- ***Text Classification from Labeled and Unlabeled Documents using EM.*** K. Nigam, A. McCallum, S. Thrun, and T. Mitchell. *Machine Learning, 39(2/3),2000.*
- ***Transductive Learning via Spectral Graph Partitioning.*** T. Joachims. *Proceedings of ICML 2003.*
- ***Understanding the Yarowsky Algorithm.*** S. Abney. *Computational Linguistics, vol. 30, 2004.*
- ***Word Sense Disambiguation Using Label Propagation Based Semi-Supervised Learning.*** Z.-Y. Niu, D. Ji, and C. Tan, *Proceedings of ACL 2005.*