# Introduction to Text Mining

## Part VI: Basics of Machine Learning

Henning Wachsmuth

https://cs.upb.de/css

# Basics of Machine Learning: Learning Objectives

## Concepts

- Basic concepts of machine learning
- What problems can/should be addressed with machine learning
- The role of machine learning in text mining

## Methods

- Feature representation and pattern generalization
- Supervised and unsupervised learning
- Selected existing machine learning algorithms
- How to approach a task with machine learning

## Notice

- This part will not explain how machine learning works in detail.
- Rather, it teaches how to apply machine learning to approach text mining tasks.

# Outline of the Course

I. Overview

II. Basics of Linguistics

III. Text Mining using Rules

IV. Basics of Empirical Methods

V. Text Mining using Grammars

VI. Basics of Machine Learning

- What Is Machine Learning?
- Machine Learning within Data Mining
- Learning Types and Algorithms
- Application of Machine Learning

VII. Text Mining using Similarities and Clustering

VIII. Text Mining using Classification and Regression

IX. Text Mining using Sequence Labeling

X. Practical Issues

# What Is Machine Learning?

# Examples of Learning Tasks

## Car Purchase Decision Making *

?

**Question**

- What criteria form the basis of a decision?

# Examples of Learning Tasks

## Credit Approval *

| Customer 1 | |
|---|---|
| house owner | yes |
| income (p.a.) | 51 000 EUR |
| repayment (p.m.) | 1 000 EUR |
| credit period | 7 years |
| SCHUFA entry | no |
| age | 37 |
| married | yes |
| ... | |

...

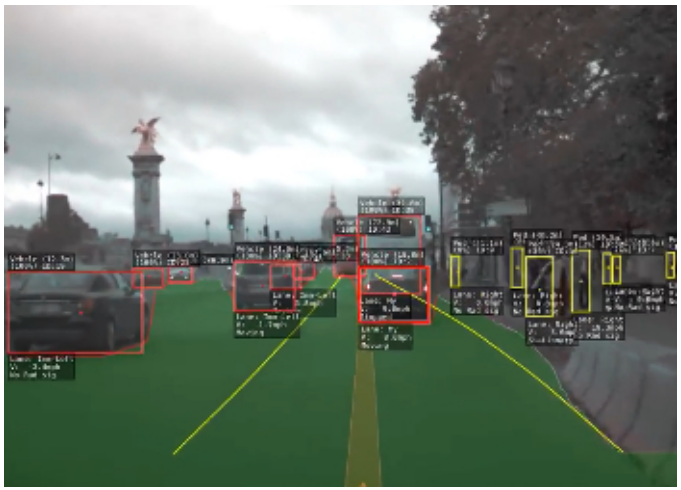
| Customer n | |
|---|---|
| house owner | no |
| income (p.a.) | 55 000 EUR |
| repayment (p.m.) | 1 200 EUR |
| credit period | 8 years |
| SCHUFA entry | no |
| age | ? |
| married | yes |
| ... | |

## Learned rules

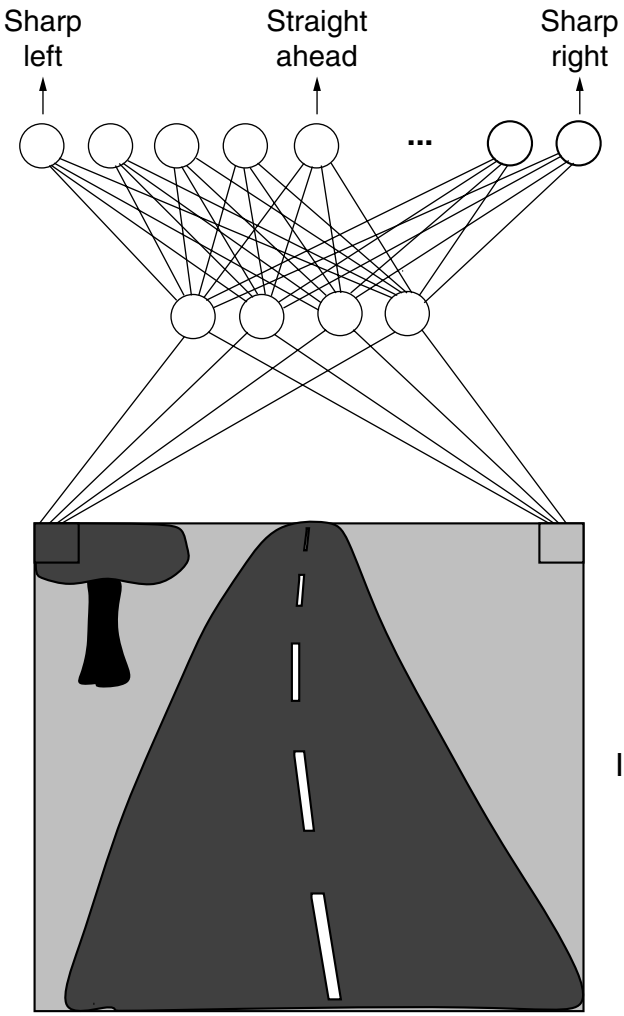
```
if   (income > 40 000 and credit_period < 3)  or  house_owner = yes
then credit_approval = yes

if   SCHUFA_entry = yes  or  (income < 20 000 and repayment > 800)
then credit_approval = no

...
```

# Examples of Learning Tasks

## Autonomous Driving *



(2018)



# Examples of Learning Tasks

## Named Entity Recognition

### Example: Part of a person entity?

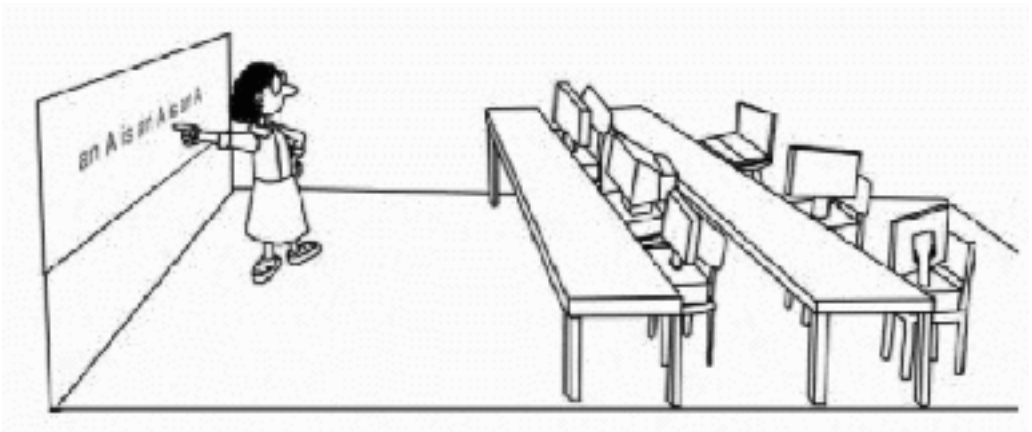
1. "Washington was born into slavery on the farm of James Burroughs."
2. "Blair arrived in Washington D.C. for what may be his last state visit."
3. "In June, Washington passed a primary seatbelt law."

| Considered Feature | | Candidate 1 | Candidate 2 | Candidate 3 |
|---|---|---|---|---|
| Previous token | Token | ⊥ | "in" | "," |
| | Capitalization | ⊥ | true | ⊥ |
| | POS tag | ⊥ | IN | COMMA |
| | Phrase type | ⊥ | B-PP | ⊥ |
| | ... | ... | ... | ... |
| This token | Token | "Washington" | "Washington" | "Washington" |
| | Capitalization | true | true | true |
| | POS tag | NP | NP | NP |
| | Phrase tag | B-NP | I-PP | B-NP |
| | ... | ... | ... | ... |
| Next token | Token | "was" | "D.C." | "passed" |
| | Capitalization | false | true | false |
| | POS tag | VBD | NP | VBD |
| | Phrase type | B-VP | PP | I-VP |
| | ... | ... | ... | ... |

# Machine Learning

**What is machine learning?** (Samuel, 1959)

- Machine learning describes the ability of an algorithm to learn without being explicitly programmed.



**An algorithm is said to learn...** (Mitchell, 1997)

- ... from *experience*
- ... with respect to a given *task*
- ... and some *performance measure*,
- ... if its performance on the task increases with the experience.

# Machine Learning

## Prediction Tasks

### What is a (prediction) task?

- A real-world problem that can be solved by a target function $\gamma : O \to C$.
- Input. Instances $o_1, o_2, \ldots$ of some real-world concept $O$.
- Output. Instances $c_1, c_2, \ldots$ of some target variable $C$.

### Target variables and functions

- Target variable. Captures the output information sought for in a task.
  The values of $C$ are either all nominal or all real-valued.
- (Ideal) Target function. Interprets any instance $o \in O$ to infer $\gamma(o)$, where $\gamma(o)$ corresponds to some $c \in C$.
  Operationalized by a human or some other (possibly unknown) real-world mechanism.

### Why learning/prediction?

- Machine learning aims at problems where the $\gamma$ is unknown.
- All non-trivial text mining tasks denote such prediction tasks.
  Including those that can be successfully tackled with rule-based approaches.

# Machine Learning

## Machine Learning in Text Mining

### Machine learning problems in text mining

- Task. Predict the output information for a given text (or span of text).
- Experience. Texts, possibly annotated for some target variable $C$.
- Performance measure. Usually, an effectiveness metric.

### Output information

- Text labels, such as topic, genre, and sentiment.
- Span annotations, such as tokens and entities.
- Span classifications, such as entity types and part-of-speech tags.
- Relations between annotations, such as entity relations.

  Combinations of these types are possible.

### Two-way relationship

- The output of text mining serves as the input to machine learning, e.g., to train a text classifier.
- Text mining often uses machine learning to produce output information.

# Machine Learning

## Example

### Example: Spam detection

- Suppose your mail tool learns to detect spam based on monitoring which mails you do or do not mark as spam.
- Experience, task, and performance measure?


| | |
|---|---|
| Classifying emails as spam or no spam. | → Task |
| Watching you label emails as spam or no spam. | → Experience |
| Fraction of emails correctly classified as spam/no spam. | → Performance measure |

### Spam detection with machine learning

- Infinitely many ways exist how spam (and no-spam) may look like.
- But many spam (and no-spam) emails have common features.
- This is where machine learning may be useful.

# Machine Learning within Data Mining

# Data Mining

## Data mining

- The inference of new (or “hidden”) output information of specified types from typically huge amounts of input data.
- Data mining hence deals with prediction tasks.

## Data mining in a nutshell

- Representation. Map data to instances of a defined representation.
  Data mining approaches mostly require input data in some *structured* form.
- Machine learning. Find statistical patterns in the instances that are relevant to the prediction task (aka *training*).
- Generalization. Apply the found patterns to infer new information from unseen data (aka *prediction*).

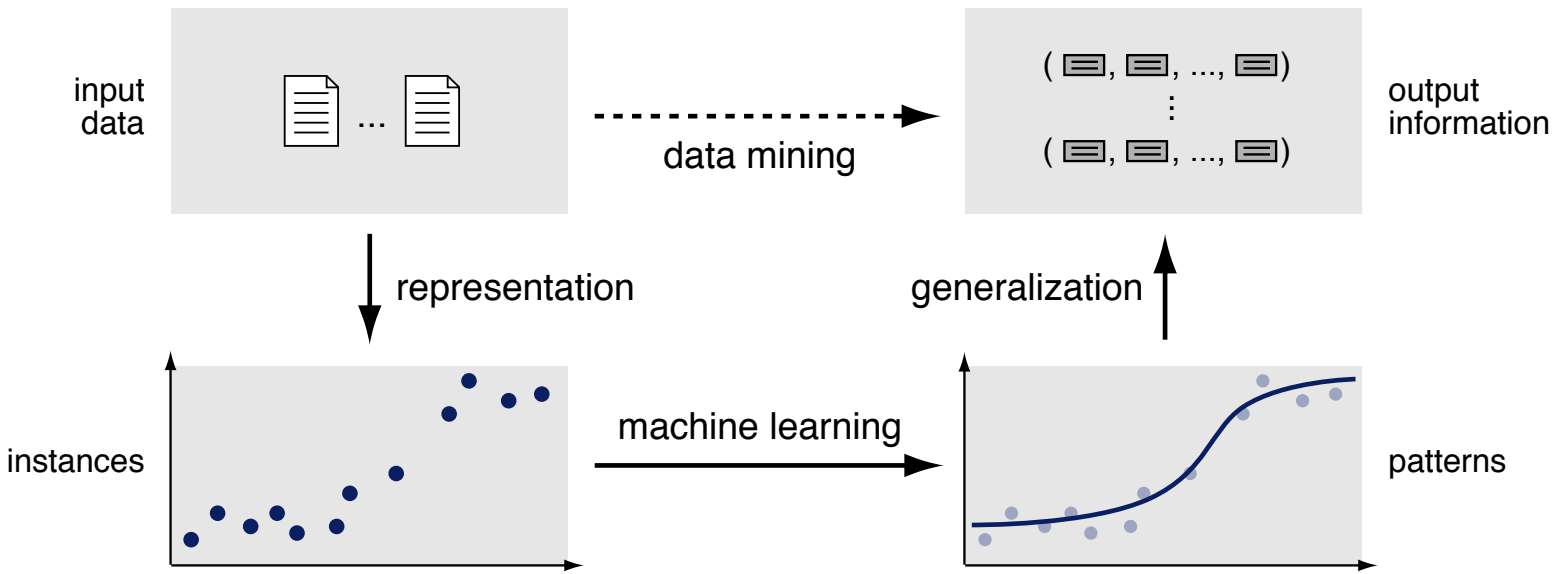
## Data mining vs. machine learning

- Data mining puts the output into the view, machine learning the method.
- Machine learning is the technical basis of data mining applications.

# Data Mining

## Data Mining Process

### The data mining process




### Text mining is a specific type of data mining

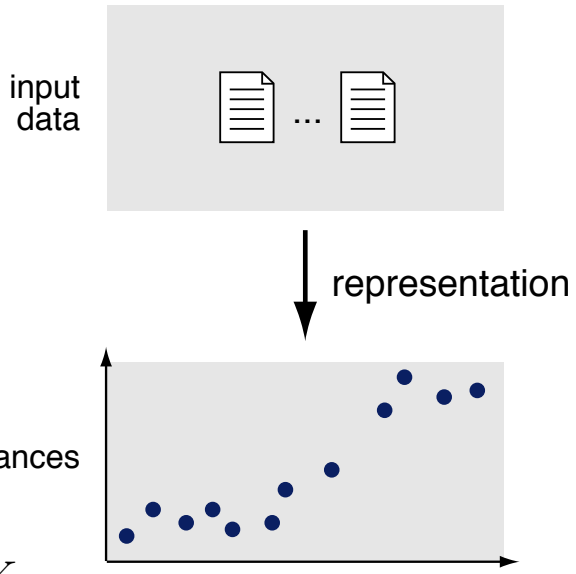
- Input data. A text corpus, i.e., a collection of texts to be processed.
- Output information. Annotations of (spans of) the texts.

# Representation

## Feature-based instance representation

- Given a task to infer instances of some $C$, each candidate text (span) to be annotated defines one instance $o_i$.

  Sentiment analysis: one instance per text
  Entity recognition: one instance per candidate entity




- In feature-based machine learning, $o_i$ is mapped to one *feature vector* $\mathbf{x}^{(i)}$.
- The mapping is given by a function $\alpha : O \to X$ where $X$ is the set of all feature vectors.

## Other instance representations

- Kernels. Model instance differences based on a similarity function.
- Neural models. Represent features implicitly using complex functions.

We restrict our view to feature-based machine learning in this course.

# Representation

## Features

### What is a feature?

- A feature $x$ denotes any measurable property of an input.
- An instance $o_i$ has one value $x_j^{(i)}$ for each considered feature $x_j$.
- What features to consider is a design decision.

### Example features in text mining

- The relative frequency of a particular word, e.g., "the" $\rightarrow$ [0,1]
- The shape of a word, e.g., Shape $\rightarrow$ {CAPS, CamelCase, ...}
- The existence of an entity type in a sentence, e.g., Organization $\rightarrow$ {0,1}

  ... among zillions of other features

### Feature vector

- An ordered set of $m \geq 1$ features of the form $\mathbf{x} = (x_1, \ldots, x_m)$.

  $m$ varies depending on the approach, from a handful to also hundreds of thousands.
- Each feature vector $\mathbf{x}^{(i)}$ contains one value $x_j^{(i)}$ for each feature $x_j \in \mathbf{x}$.

# Representation

## Feature Types and Scales

### Feature types

- A feature type is a set of features that conceptually belong together.

  Bag-of words. Relative frequency of each considered word.

  POS 3-grams. Relative frequency of each possible part-of-speech 3-gram.

- The features of a type are often found automatically on training data, in order to exclude useless features that might introduce noise.

  Bag-of words. All words with a training set occurrence $> \tau$.

### Scales of features

- We consider only features here with values from a real-valued scale.
- Nominal, boolean, and similar features can be transformed.

  phrase type $\rightarrow$ {"VP", "NP", "PP"} $\mapsto$ VP $\rightarrow$ {0,1}, NP $\rightarrow$ {0,1}, PP $\rightarrow$ {0,1}

- Usually, the values of all features are *normalized* to the same interval.

# Representation

## Feature Value Normalization

### What is feature value normalization?

- Value ranges of features may vary drastically.
- Normalization scales all values to a uniform range, typically $[0, 1]$ (used here) or $[-1, 1]$.

| | # "the" | US? | ... |
|---|---|---|---|
| $o_1$ | 1 | 1 | |
| $o_2$ | 102 | 1 | |
| $o_3$ | 42 | 1 | |
| $o_4$ | 0 | 0 | |
| ... | ... | | |

### Why normalize?

- Machine learning works better with uniform values, due to the interplay with weights, learning rates, and similar (see below).
- At best, the whole (normalized) value range is covered for a feature.

### Typical ways to normalize

- Divide by the length of the given text (e.g., in # tokens).
- Divide by the maximum value found in a training set, and cut at 1.0.
- Divide by a manually defined maximum based on expert knowledge.

  The best way depends on the feature (and partly on the application).

# Representation

## Feature Determination and Computation

### How to determine the set of features in a vector

1. Specify using expert knowledge which feature types to consider.

   (a) Bag-of-words (b) text length in # tokens

2. Where needed, process training set to get counts of candidate features.

   (a) “the” $\rightarrow$ 4242, “a” $\rightarrow$ 2424, . . . , “woooodchuck” $\rightarrow$ 1 (b) max tokens = 120

3. Keep only features whose counts lie within some defined thresholds.

   (a) “the”, “a”, . . . , ~~“woooodchuck”~~ (b) n/a

### How to compute the values for each feature

1. Compute value of each feature in a vector for a given input text.

   (a) “the” $\rightarrow$ 6, “a” $\rightarrow$ 7, . . . (b) # tokens $\rightarrow$ 50

2. Normalize feature values.

   (a) “the” $\rightarrow$ 0.12, “a” $\rightarrow$ 0.14, . . . (b) # tokens $\rightarrow$ 0.417

# Representation

## Feature Engineering

### The importance of feature engineering

- The representation governs what patterns can be found during learning.
  The function $\alpha$ determines the level of abstraction between $o$ and $\mathbf{x}$.
- Some features generalize worse than others towards unseen data.
- Engineering features that predict a target variable $C$ and generalize well is key to effective (feature-based) machine learning.

### Feature engineering in text mining

- Standard features. Some types help in many tasks, e.g., bag-of-words.
- Specific features. The most discriminative types usually encode expert knowledge about the task and input.

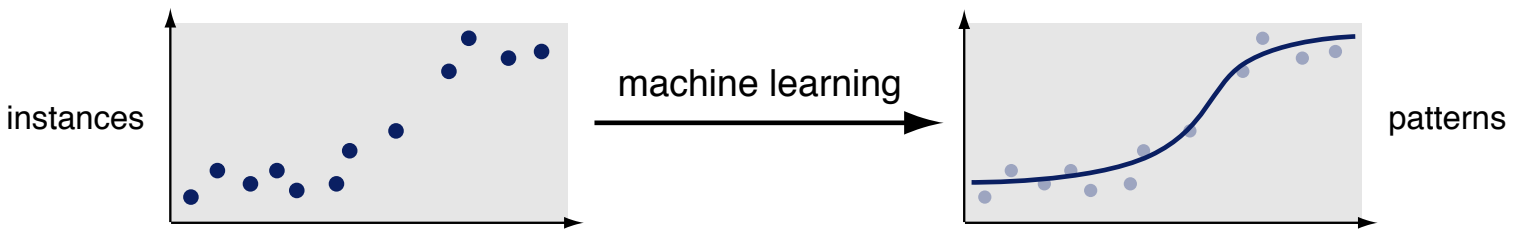
### Feature selection and dimensionality reduction

- Techniques that aim to reduce the set of considered features to improve generalizability and training efficiency.
  Not covered in this course.

# Machine Learning

## Machine learning models

- Machine learning recognizes statistical patterns in instances $X$ that are relevant to infer a target variable $C$.
- A model $y : X \to C$ generalizes the patterns to approximate the ideal target function $\gamma$.
- Machine learning seeks for the optimal $y$ wrt. a performance measure.




## Model vs. target function

- $\gamma$ and $y$ differ in the complexity and representation of their domain.
- Complexity. $\alpha$ abstracts instances $o$ into feature vectors $\mathbf{x} = \alpha(o)$.
- Representation. $y(\mathbf{x})$ is the formalized counterpart of $\gamma(o)$.

# Machine Learning

## From Target Function to Model (1 out of 2)

### Characterization of the real-world domain

- $O$ is a set of instances, $C$ is a target variable.
- $\gamma : O \to C$ is the ideal target function for $O$.
- Task. Given some $o \in O$, determine its class or value $\gamma(o) \in C$.

### Example: Spam detection

- $O$ is a set of emails, $C = \{$“spam”, “no spam”$\}$.
- $\gamma$ is a human expert.
- Task. Given an email, is it spam or not?


### Acquisition of training data

1. Collect real-world instances of the form $(o_i, \gamma(o_i))$ with $o_i \in O, \gamma(o_i) \in C$.
2. Abstract each instance $o_i$ to a feature vector $\mathbf{x}^{(i)} = \alpha(o_i)$.
3. Construct instances $(\mathbf{x}^{(i)}, c(\mathbf{x}^{(i)})$, where $c(\mathbf{x}^{(i)}) \equiv \gamma(o_i)$ is the ground truth.

# Machine Learning

## From Target Function to Model (2 out of 2)

### Characterization of the model domain

- $X$ is a set of feature vectors (the *feature space*), $C$ as before.
- $c : X \to C$ is the ideal predictor for $X$.
- $\{(\mathbf{x}^{(1)}, c(\mathbf{x}^{(1)})), \ldots, (\mathbf{x}^{(n)}, c(\mathbf{x}^{(n)}))\} \subseteq X \times C$ is a set of instances.

### Example: Spam detection

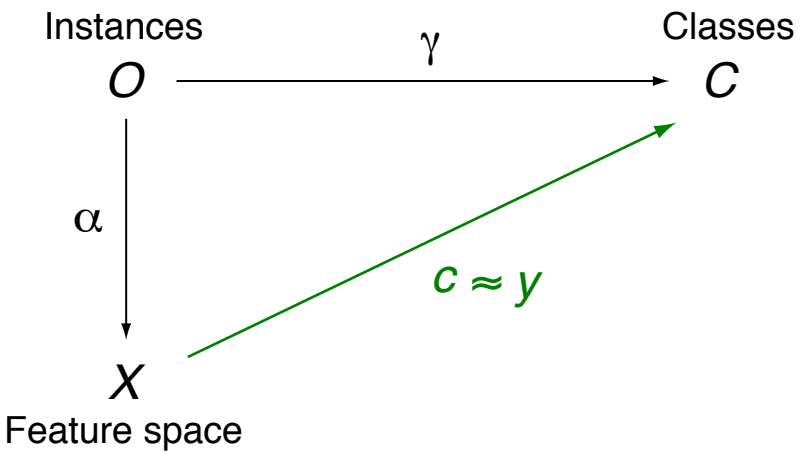

- $X$ represents the frequencies of words.
- $C$ as before: “spam” and “no spam”.
- $c$ is unknown.

### Training in machine learning

- Approximate the ideal classifier $c$ (implicitly following from all instances) by a model $y : X \to C,\ \mathbf{x} \mapsto y(\mathbf{x})$.
- Apply statistics and learning algorithms to optimize the fit between $c$ and $y$ wrt. to a given performance measure.

# Machine Learning

## Overview of the Concepts




### Notation

| | |
|---|---|
| $\gamma$ | Ideal target function for real-world instances. |
| $\alpha$ | Feature mapping function. |
| $c$ | Unknown ideal predictor for vectors from the feature space. |
| $y$ | Machine learning model to be learned. |
| $c \approx y$ | $c$ is approximated by $y$ (based on a set of instances). |

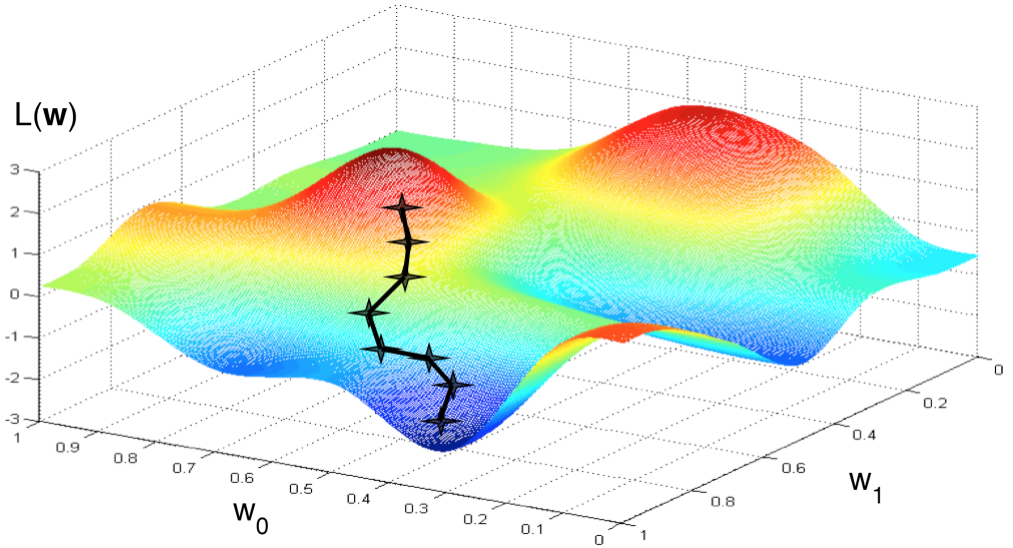
# Machine Learning

## Optimization Process

### Training

- Machine learning incrementally creates candidate models $y$.
- Each $y$ assigns one weight $w_j$ to every feature $x_j$.

  Not all learning algorithms assign weights explicitly.




- On training data, the weights $\mathbf{w}$ are tested against a cost function $L$.

  The cost function, also called *loss function*, reflects the performance measure.

- Based on $L(\mathbf{w})$, $\mathbf{w}$ is adapted to create the next model $y'$.
- The adaptation relies on an *optimization procedure*.

### Hyperparameter optimization

- Many learning algorithms have *hyperparameters* that are not optimized in training. They need to be optimized against a validation set.

# Machine Learning

## Optimization Procedures

### Optimization procedures

- Aim to minimize the costs of a model wrt. a function $L$.
- The most common procedure is *gradient descent*.

  Learning algorithms usually integrate a particular procedure.

**No free lunch theorem**

No optimization procedure is best in all cases.

### (Batch) Gradient descent

- In each step, the model is adapted to all training instances.
- Stepwise heads towards a local minimum of $L$ until convergence.
- Guarantees to find the local minimum.

  For convex models spaces, guarantees to find the *global* minimum.

### Stochastic gradient descent (SGD)

- Variant that adapts the model subsequently to each single instance.
- The adaptation is repeated for some number of iterations $k$.
- Does not guarantee to find a local minimum, but is much faster.

  Particularly used in large-scale scenarios.

# Machine Learning

## Pseudocode of Stochastic Gradient Descent

### Signature

- Input. Training instances $X$ of the form $(\mathbf{x}, c(\mathbf{x}))$, a learning rate $\eta$, and a number of iterations $k$.

  $\eta$ is a small positive constant. The higher $k$, the more the model will adapt to the data.

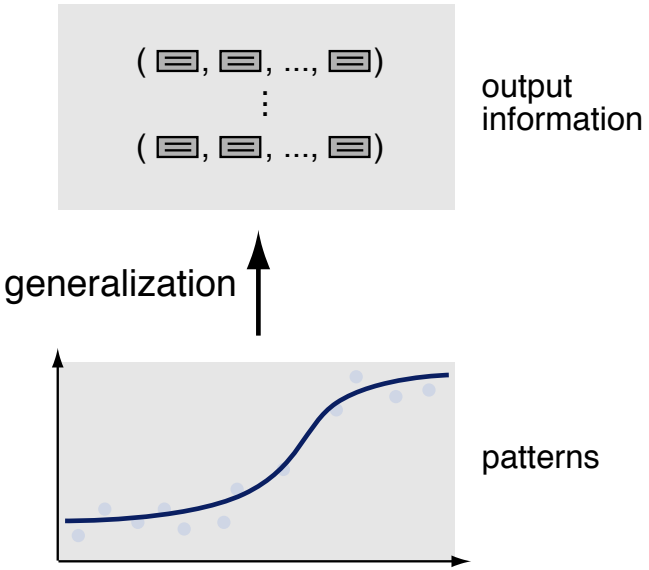
- Output. A vector $\mathbf{w}$ with one weight $w_i$ for each feature $x_i \in \mathbf{x}$, $1 \leq i \leq n$.

```
stochasticGradientDescent(List<Instance> X, double η, int k)
1.    List<double> w ← randomInitialize(-1, 1)
2.    for int j ← 1 to k do
3.        for each Instance (x, c(x)) in X do // ∀x ∈ X : x|_{x_0} ≡ 1
4.            double y(x) ← w^T x = w_0 + w_1 · x_1 + ... + w_m · x_m // Regression
5.            double error ← c(x) − y(x) // Cost in terms of error
6.            double Δw ← η · error · x
7.            w ← w + Δw // Adapt weights based on error
8.    return w
```

# Generalization

## Generalization

- Application of the learned model $y$ to unseen data, in order to infer new information.
- How well $y$ generalizes depends on how well it fits the unknown target function $\gamma$.
- Generalization is mainly decided by the training process (see above).




## Bias in training

- The training process explores a large space of models, in which several parameters are optimized.
- An important training decision is how much to bias the process wrt. the complexity of the model to be learned.

  Complexity here refers to the shape of the underlying polynomial.

# Generalization

## Model Complexity: Underfitting and Overfitting

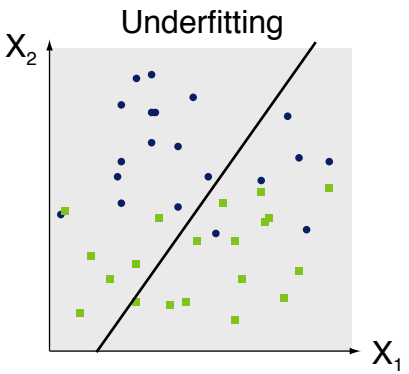
### Simple vs. complex models

- Simple. Induce high bias to avoid noise; may underfit the input data.
- Complex. Induce low bias to fit the input data well; may capture noise.

  Simple models may, e.g., be linear functions, complex models high polynomials.

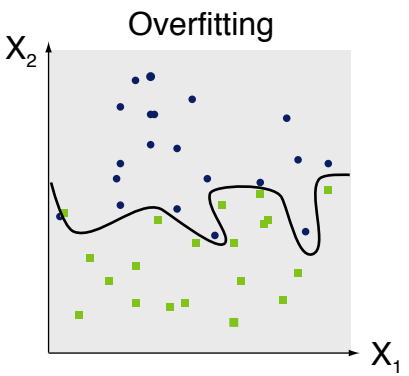
### Underfitting (too high bias)

- A model $y$ generalizes too much, not capturing all relevant properties of the training data.
- $y$ is too simple and will have limited effectiveness.




### Overfitting (too high variance)

- A model $y$ captures both relevant and irrelevant properties of the training data.
- $y$ is too complex and will thus not generalize well.



# Generalization

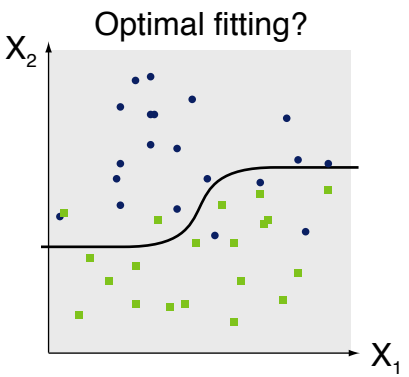
## Model Complexity: Optimal Fitting and Regularization

### Avoiding underfitting and overfitting

- The best way to avoid both is to achieve an *optimal fitting*.
- Overfitting can also be countered through *regularization*.

### Optimal fitting

- A model $y$ perfectly approximates the complexity of the target function based on the training data.
- In general, the right complexity is unknown.




### Regularization

- Regularization prevents the use of complex functions, unless they significantly reduce the prediction error.
- This is done by adding a term to the cost function that forces the feature weights to be small.

More on regularization in a later part of this course.

# Learning Types and Algorithms

# Learning Types and Algorithms

## Learning types

- Learning algorithms differ wrt. what kind of patterns are learned as well as to what kind of data they are applied to.
- Two major types. *Supervised* and *unsupervised* learning.

  They are most important for text mining and in the focus of this course.

## Supervised vs. unsupervised learning

- Supervised. Derive a model from patterns found in annotated training data (where the ground truth is known).
- Unsupervised. Find patterns in unannotated data (without ground truth).

  More details below.

## Learning algorithms

- Different algorithms of one type differ wrt. how they combine features, how they optimize, how complex the learned patterns are, ...
- We focus here on selecting and applying algorithms suitable for a task.

  Some algorithms will be discussed in detail in the following lecture parts.

# Supervised Learning

## What is supervised (machine) learning?

- A learning algorithm builds a model $y$ on *known* training data, i.e., pairs of instances $\mathbf{x}^{(i)}$ and the associated output information $c(\mathbf{x}^{(i)})$.
- $y$ can then be used to predict output information for unknown data.

## Supervised classification vs. regression

- Classification. Assign a nominal class to an instance.
- Regression. Predict a numeric value for an instance.

## Why "supervised"?

- The learning process is guided by instances of correct predictions.

## Manifold applications in text mining

- Classification. Standard technique for any text classification task, for extracting relations between entities, and similar.
- Regression. Used to predict scores, ratings, probabilities, ...

# Supervised Learning

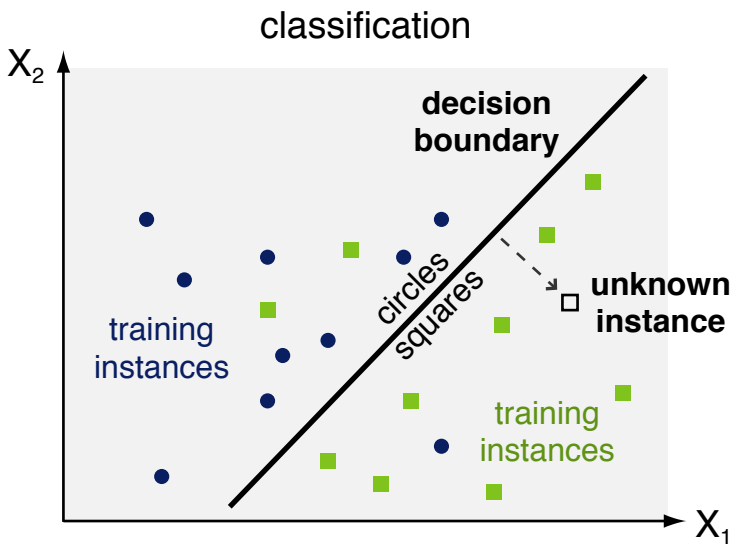
## Classification

### What is classification?

- The task to assign an instance to the most likely of a set of two or more predefined discrete classes.

### Supervised classification

- An optimal decision boundary $y$ is sought for on training instances $X$ with known classes $C$.
- The boundary decides the class of unknown instances.




### Binary vs. multiple-class classification

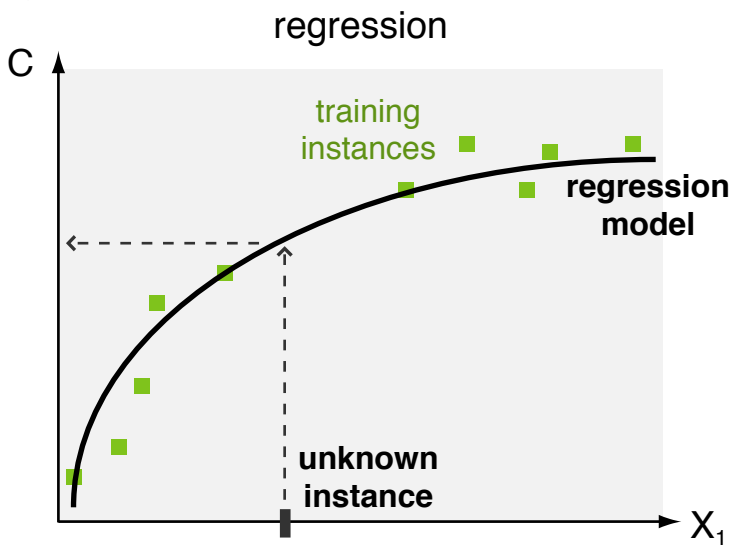
- Binary classifiers separate the instances of two classes.
- Multiple classes are handled through multiple binary classifiers with techniques such as one-versus-all classification.

# Supervised Learning

## Regression

### What is regression?

- The task is to assign a given instance to the most likely value of a real-valued, continuous target variable.




### Supervised regression

- A regression function $y$ is sought for on training instances $X$ with known values $C$.
- The function decides the value of unknown instances.

### Linear regression models

- Only constants and parameters multiplied by independent variables.

$$y(\mathbf{x}) = w_0 + w_1 \cdot x_1 + \ldots + w_m \cdot x_m$$

# Supervised Learning

## Overview of Supervised Learning Algorithms

### Common algorithms for classification

- Naïve Bayes
- Support vector machines
- Decision trees
- Random forest
- (Artificial) Neural networks

... among many others

### Common algorithms for regression

- Simple linear regression
- Support vector machines
- (Artificial) Neural networks

... among many others

### Ensemble methods

- Meta-algorithms that combine multiple classifiers/regressors.

# Supervised Learning

## Naïve Bayes and Support Vector Machines

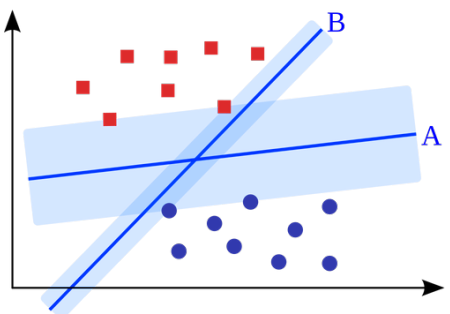
### Naïve Bayes

- Predicts the most likely class $c$ using Bayes' Theorem on conditional probabilities.

$$P(c|x) = \frac{P(x|c) \cdot P(c)}{P(x)} \qquad P(c|\mathbf{x}) \propto P(x_1|c) \cdot \ldots \cdot P(x_m|c) \cdot P(c)$$

- Effective with few training data, scales well in terms of efficiency.
- Strong feature independence assumption often limits effectiveness.

### Support vector machine (SVM)



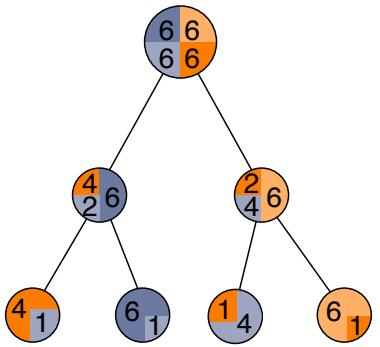

- Maximizes the margin between the decision boundary and class instances in training.
- Maps instances into a higher-dimensional space where they are linear separable (*kernel trick*).
- Often effective, while not being prone to adapt to noise.
- Hyperparameter tuning tends to be time-intensive.

# Supervised Learning

## Decision Trees and Random Forest

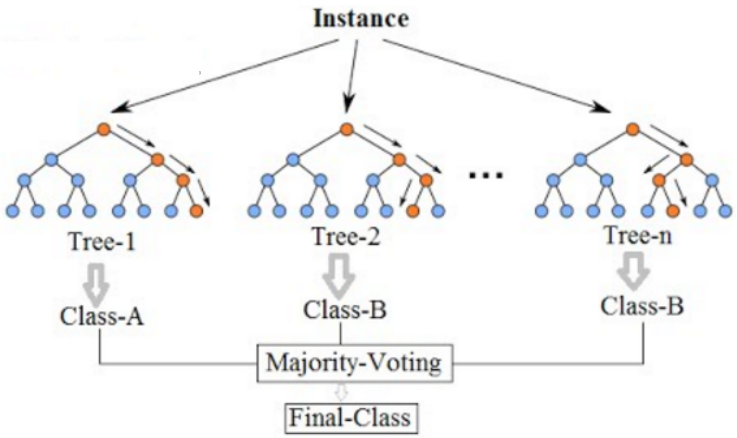
### Decision trees

- Each inner node tests one feature $x$, with a learned threshold. Leafs correspond to classes.
- Node ordering learned, e.g., from information gain.
- Low-impact features pruned for generalization.
- Effective when classes are just decided by feature-value combinations (without weightings).




### Random forest

- Builds several decision trees (e.g., 100) for small feature subsets.
- Majority voting to decide class $c$.
- Often very effective without any hyperparameter tuning.
- Less effective if some features are dominant.

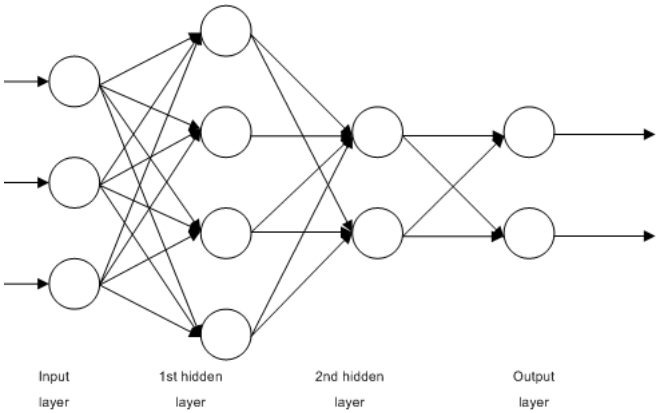


# Supervised Learning

## Neural Networks and Ensemble Methods

### Artificial neural network (ANN)

- Weights connections between nodes, mimicking the human brain.
- Learns complex features automatically often on “raw” input (e.g., tokens).
- Conceptually, the strongest algorithm type (and the basis of deep learning).




- Effective only with much input data, and architecture needs to be tuned.

### Ensemble methods

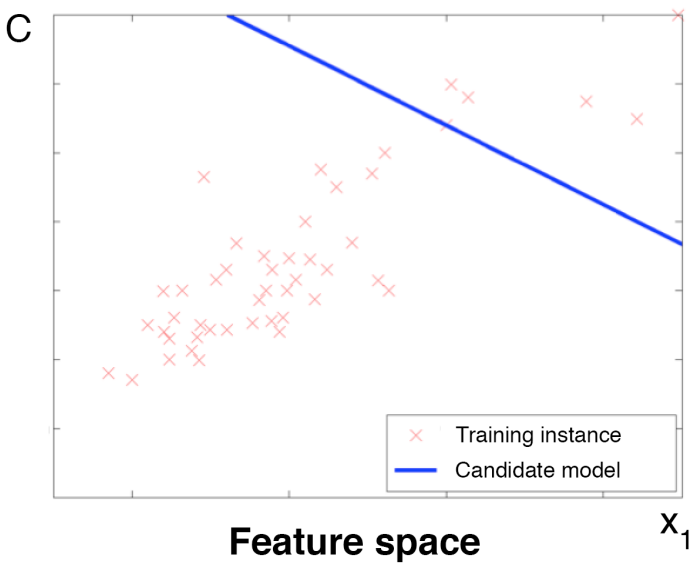
- Combination of multiple algorithms to improve effectiveness (*stacking*).
  Random forest is actually an integrated stacking method.
- Can be based on any learning algorithm (at least for classification).
- Other ensembles aim to decrease variance (*bagging*) or bias (*boosting*).

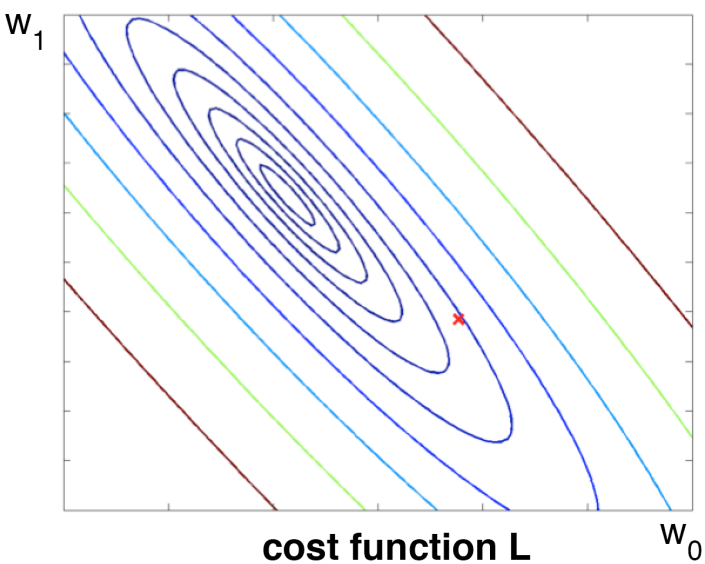
# Supervised Learning

## Example (1 out of 9)

### Learning Simple Linear Regression

- Feature space (left) and cost function (right) during training.




### Notice

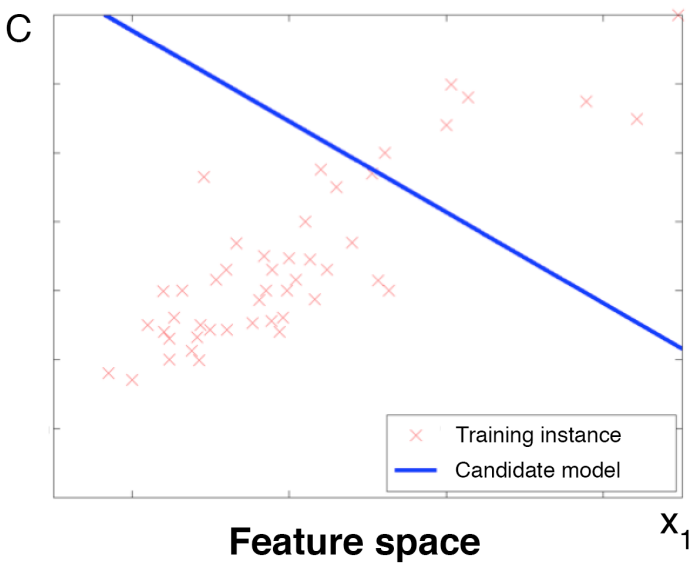
- A regression model $y = w_0 + w_1 \cdot x_1$ is learned for one single feature $x_1$.
- Each pair $w_0, w_1$ defines one candidate model.

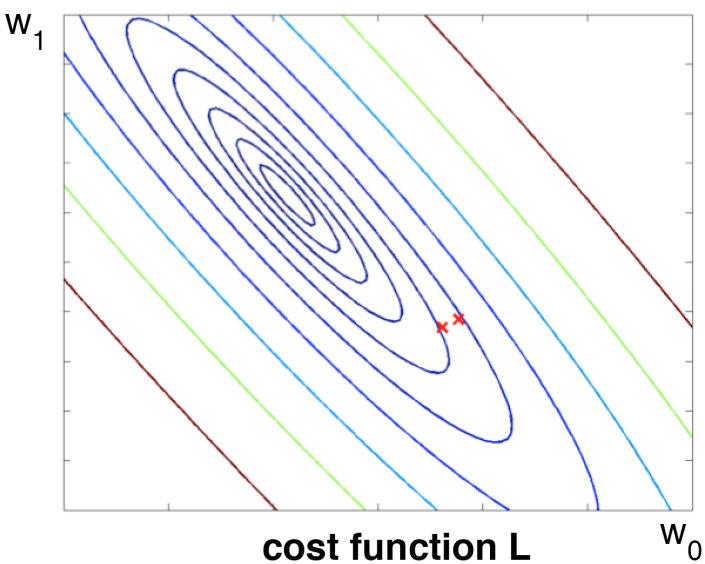
# Supervised Learning

## Example (2 out of 9)

### Learning Simple Linear Regression

- Feature space (left) and cost function (right) during training.




### Notice

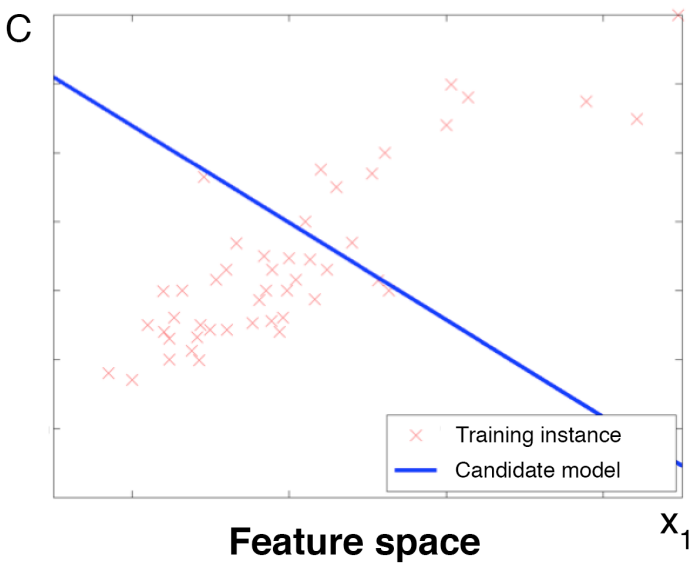
- A regression model $y = w_0 + w_1 \cdot x_1$ is learned for one single feature $x_1$.
- Each pair $w_0, w_1$ defines one candidate model.

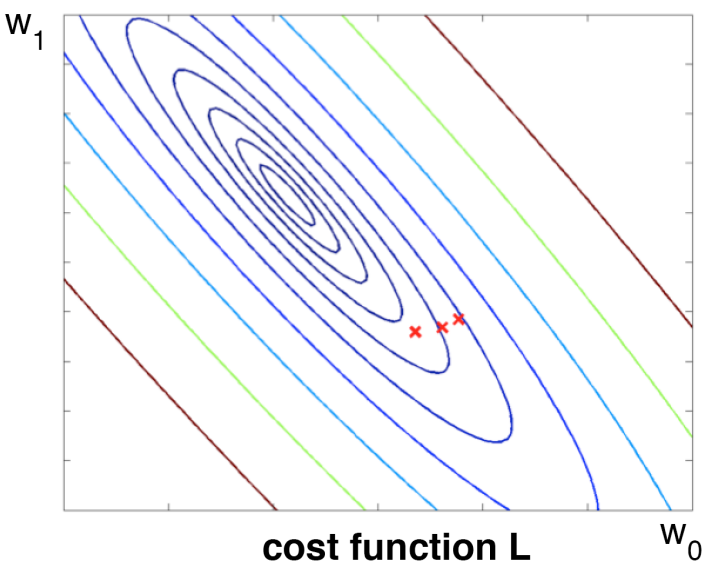
# Supervised Learning

## Example (3 out of 9)

### Learning Simple Linear Regression

- Feature space (left) and cost function (right) during training.




### Notice

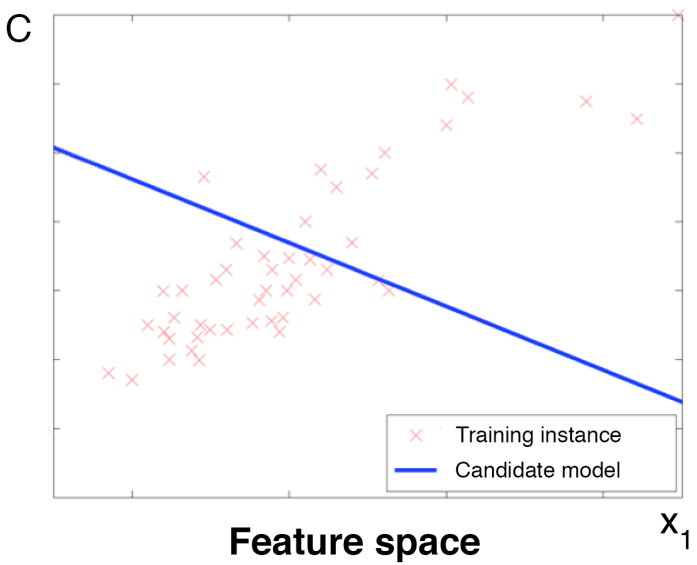
- A regression model $y = w_0 + w_1 \cdot x_1$ is learned for one single feature $x_1$.
- Each pair $w_0, w_1$ defines one candidate model.

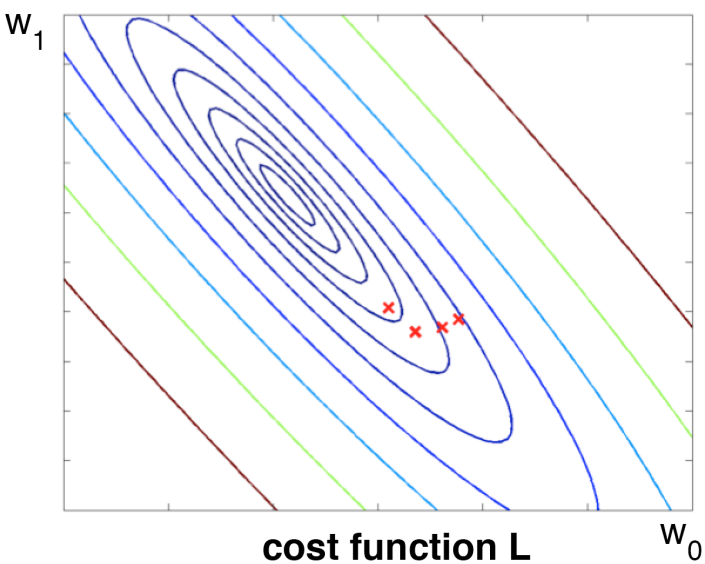
# Supervised Learning

## Example (4 out of 9)

### Learning Simple Linear Regression

- Feature space (left) and cost function (right) during training.




### Notice

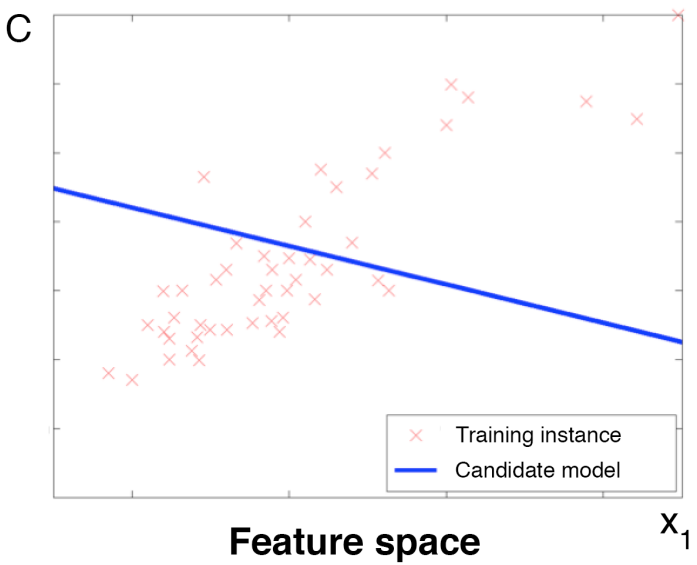
- A regression model $y = w_0 + w_1 \cdot x_1$ is learned for one single feature $x_1$.
- Each pair $w_0, w_1$ defines one candidate model.

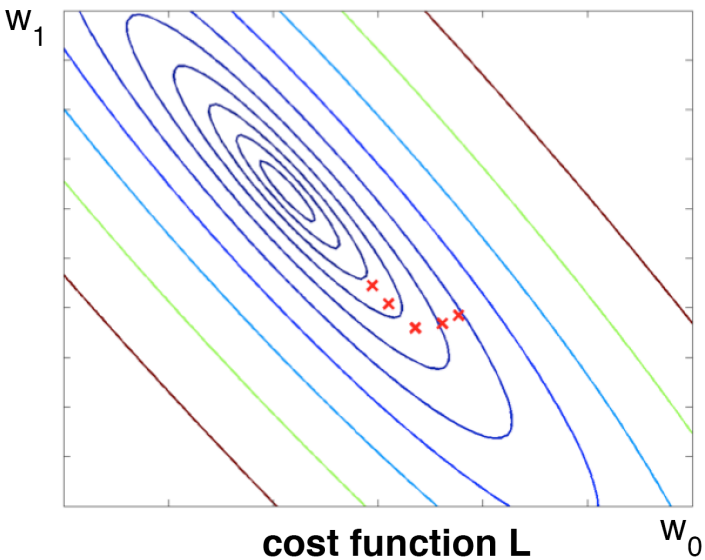
# Supervised Learning

## Example (5 out of 9)

### Learning Simple Linear Regression

- Feature space (left) and cost function (right) during training.




### Notice

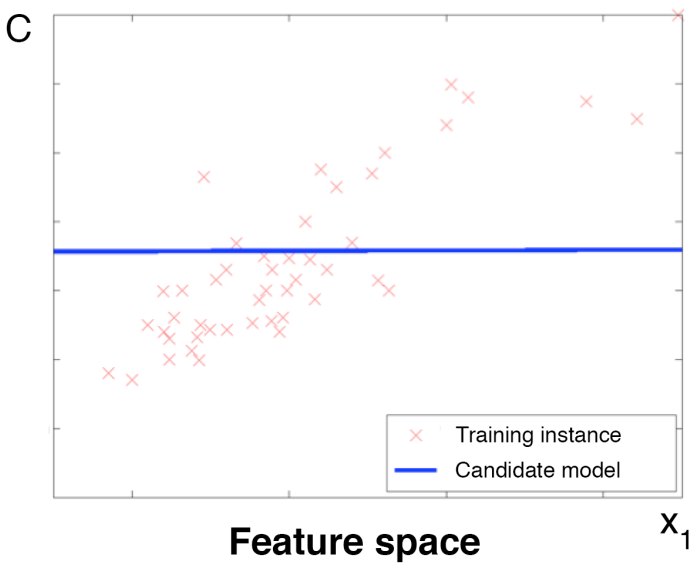
- A regression model $y = w_0 + w_1 \cdot x_1$ is learned for one single feature $x_1$.
- Each pair $w_0, w_1$ defines one candidate model.

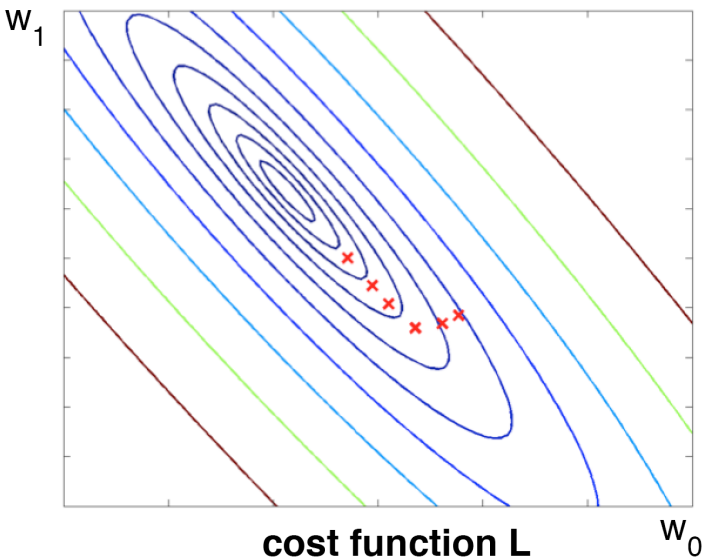
# Supervised Learning

## Example (6 out of 9)

### Learning Simple Linear Regression

- Feature space (left) and cost function (right) during training.




### Notice

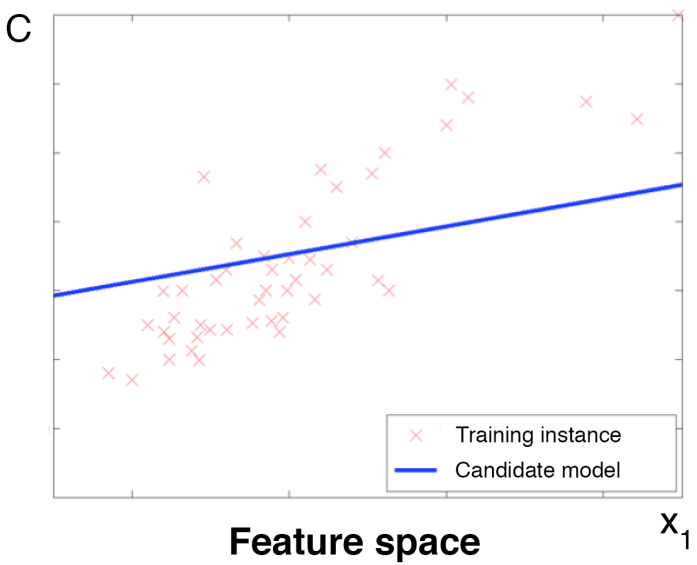
- A regression model $y = w_0 + w_1 \cdot x_1$ is learned for one single feature $x_1$.
- Each pair $w_0, w_1$ defines one candidate model.

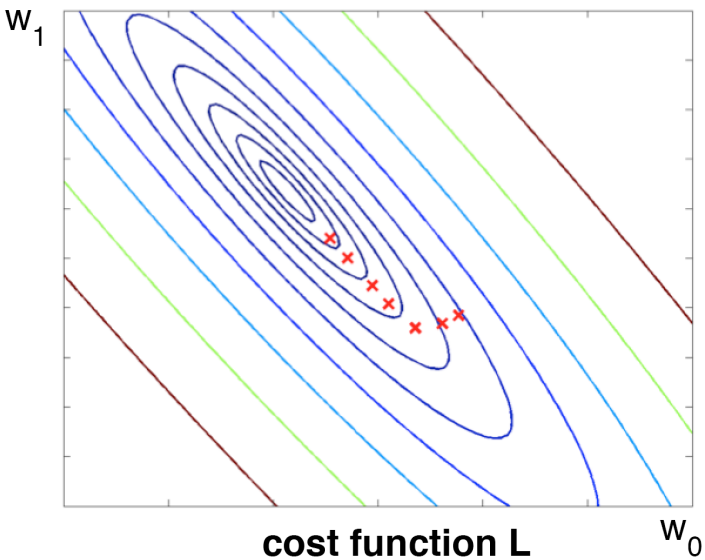
# Supervised Learning

## Example (7 out of 9)

### Learning Simple Linear Regression

- Feature space (left) and cost function (right) during training.




### Notice

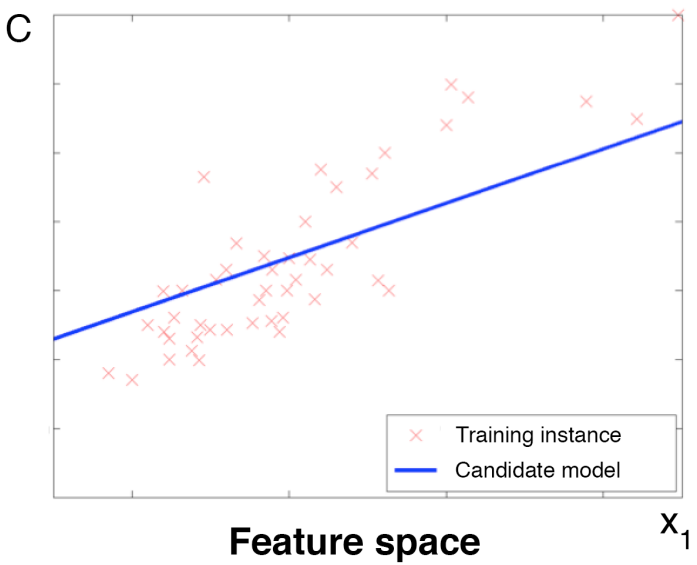
- A regression model $y = w_0 + w_1 \cdot x_1$ is learned for one single feature $x_1$.
- Each pair $w_0, w_1$ defines one candidate model.

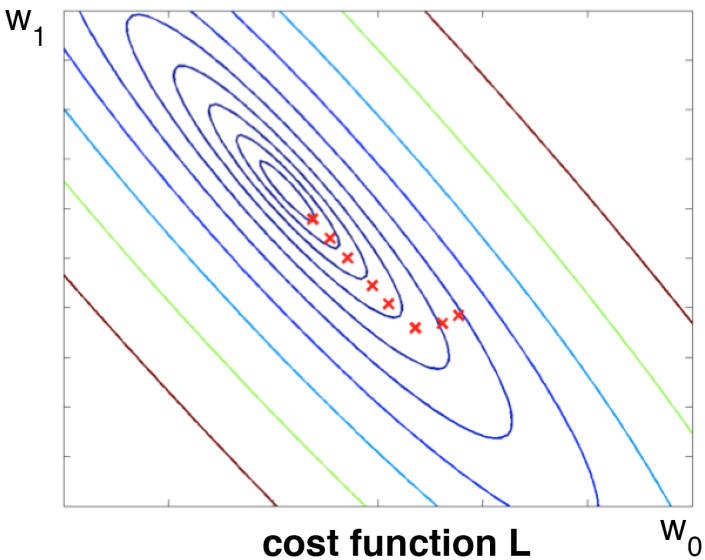
# Supervised Learning

## Example (8 out of 9)

### Learning Simple Linear Regression

- Feature space (left) and cost function (right) during training.




### Notice

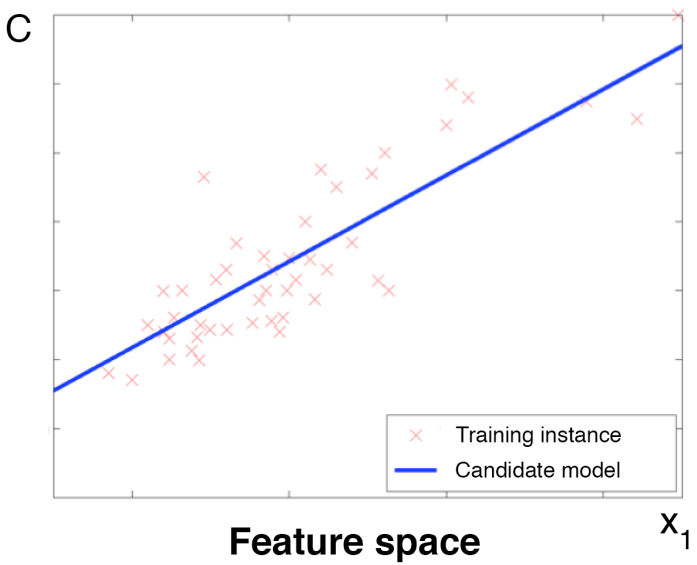
- A regression model $y = w_0 + w_1 \cdot x_1$ is learned for one single feature $x_1$.
- Each pair $w_0, w_1$ defines one candidate model.

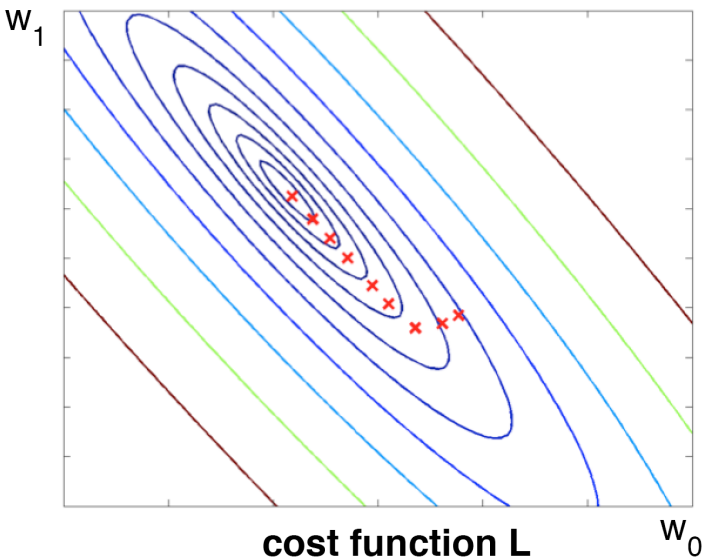
# Supervised Learning

## Example (9 out of 9)

### Learning Simple Linear Regression

- Feature space (left) and cost function (right) during training.




### Notice

- A regression model $y = w_0 + w_1 \cdot x_1$ is learned for one single feature $x_1$.
- Each pair $w_0, w_1$ defines one candidate model.

# Supervised Learning

## Variations of Supervised Learning *

### Sequence labeling

- Classifies each instance in a sequence of instances.
- The goal is to find the optimal sequence of classes.
- Idea. Exploit information about dependencies between instances.

"The"/DT "play"/NN "was"/VBD "great"/JJ "."/.

### Semi-supervised learning

- Targets tasks where much data is available, but little training data.
- Idea. Derive patterns from training data, then find similar patterns in unannotated data to get more training data.

"Microsoft is based in Seattle.", "Apple is based in Cupertino." → <NP> is based in <NP>

### Self-supervised learning

- Idea. Generate training data automatically. Works if output information can be computed, such as time measurements.

# Unsupervised Learning

## What is unsupervised (machine) learning?

- A model $y$ is derived from instances only, without output information.
- The model reveals the organization and association of input data.
- Techniques. Clustering, autoencoders, principal component analysis, self-organizing maps, ...

  The focus is on clustering here.

## What is clustering?

- The grouping of a set of instances into a possibly but not necessarily predefined number of classes (aka *clusters*).

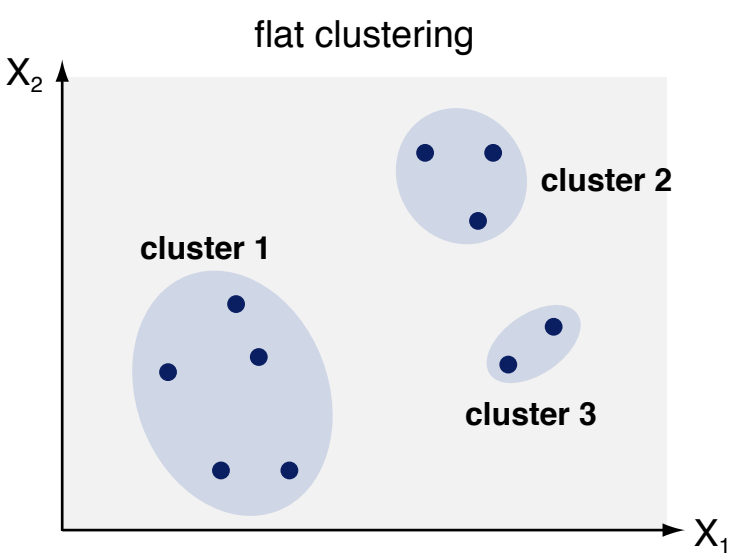
  The meaning of a class is usually unknown in advance.
- Hard clustering. Each instance belongs to a single cluster.
- Soft clustering. Instances belong to each cluster with some weight.

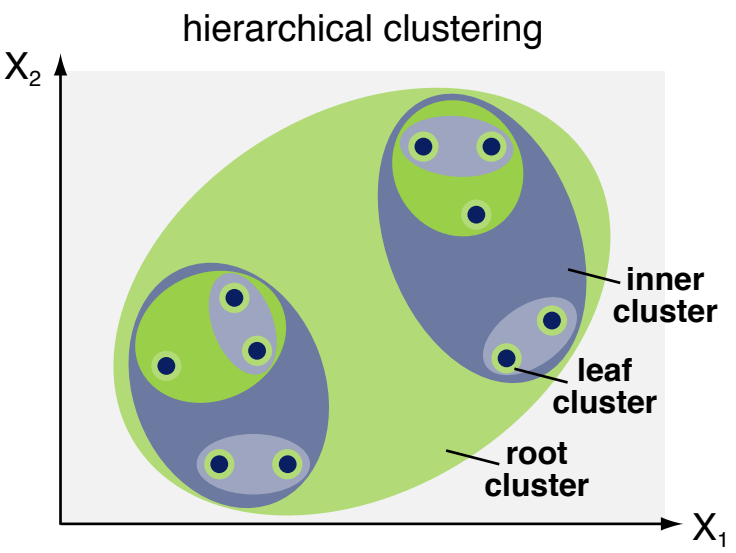
## Applications in text mining

- Detection of texts with similar properties, mining of topics, ...

# Unsupervised Learning

## Flat vs. Hierarchical Clustering




### Types of clustering

- Flat. Group instances into a (possibly predefined) number of clusters. No associations between the clusters are specified.
- Hierarchical. Create a binary tree over all instances. Each tree node represents a cluster of a certain size.

  The root covers all instances and each leaf refers to a single instance.
- Both types have certain advantages wrt. efficiency and cluster quality.

# Unsupervised Learning

## Clustering Algorithms

### Unsupervised clustering

- Patterns in the instances are learned based on similarity measures.
- The resulting clusters correspond to classes.
- The resulting model can assign arbitrary instances to the clusters.

### Supervised clustering?

- Sometimes, it makes sense to cluster instances with known classes.
  For instance, when classes shall be subdivided — or just for evaluation.
- Clusters can then be evaluated in terms of their *purity*, i.e., the fraction of instances whose class equals the majority class.

### Clustering algorithms

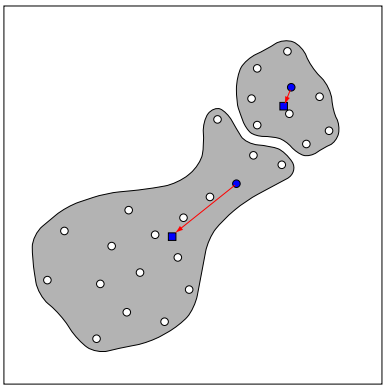
- Iterative clustering, such as $k$-*means*.
- Density-based clustering, such as *DBSCAN*.
- Hierarchical clustering, either *agglomerative* or *divise*.

  ... among others.

# Unsupervised Learning

## $k$-means and Agglomerative Clustering

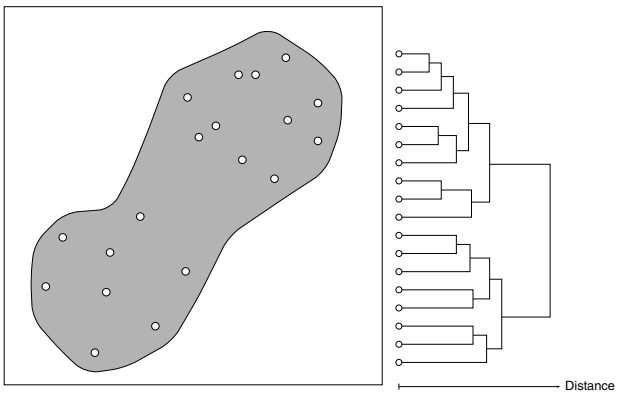
### Iterative flat clustering with k-means



- Partition instances into $k$ clusters.
- Iteratively compute *centroids* of candidate clusters and re-cluster based on centroids.

  The centroid is the average of all instances in the cluster.
- $k$ is chosen based on domain knowledge or through intrinsic cluster evaluation.

### Agglomerative hierarchical clustering

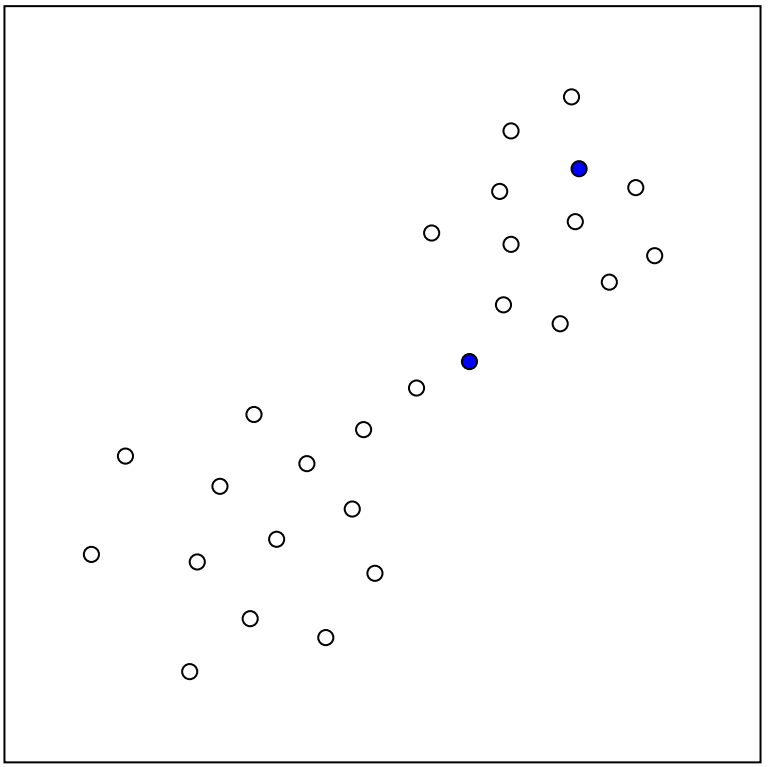



- Incrementally create tree bottom-up, beginning with single instances.
- Merge clusters based on distances of instances to clusters.
- A flat clustering can be derived from a hierarchical one via cuts in the tree.

# Unsupervised Learning

## Example: $k$-means Flat Clustering with $k = 2$

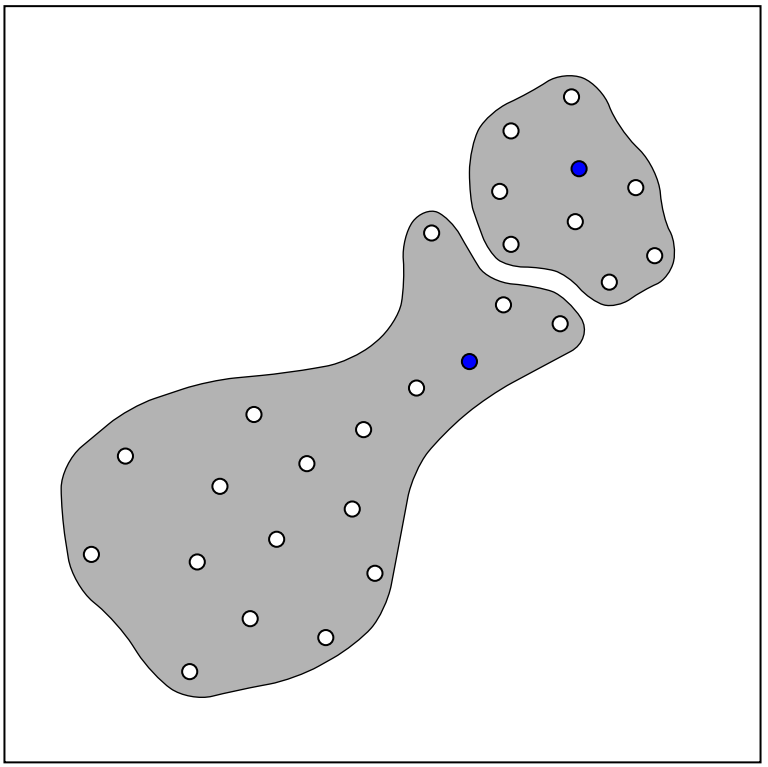
### Step 1: Choose $k$ instances randomly



# Unsupervised Learning

## Example: $k$-means Flat Clustering with $k = 2$

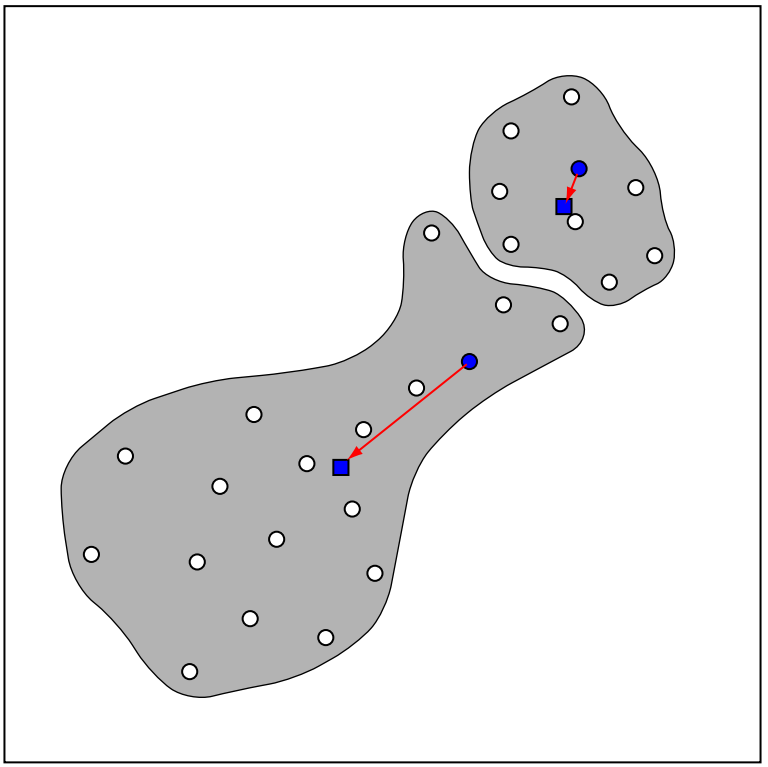
### Step 2: Cluster by distance to the $k$ instances



# Unsupervised Learning

## Example: $k$-means Flat Clustering with $k = 2$

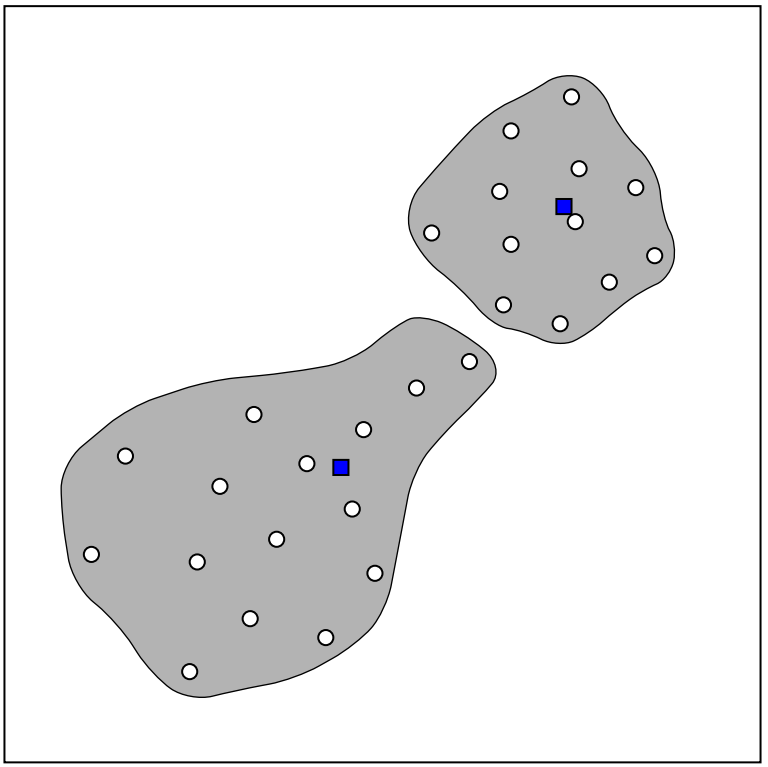
### Step 3: Compute centroids of the $k$ clusters



# Unsupervised Learning

## Example: $k$-means Flat Clustering with $k = 2$

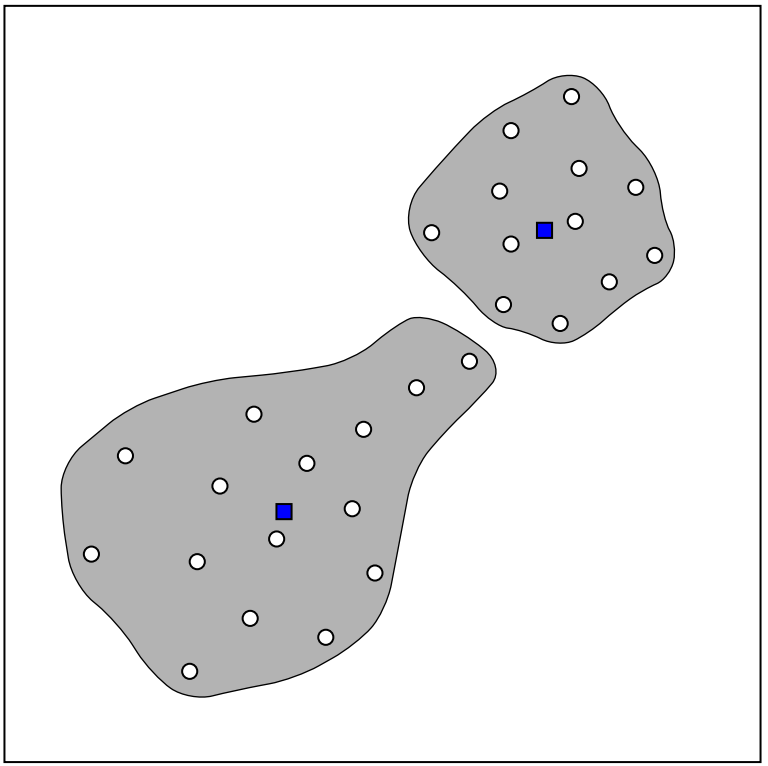
### Step 4: Cluster by distance to the $k$ centroids



# Unsupervised Learning

## Example: $k$-means Flat Clustering with $k = 2$

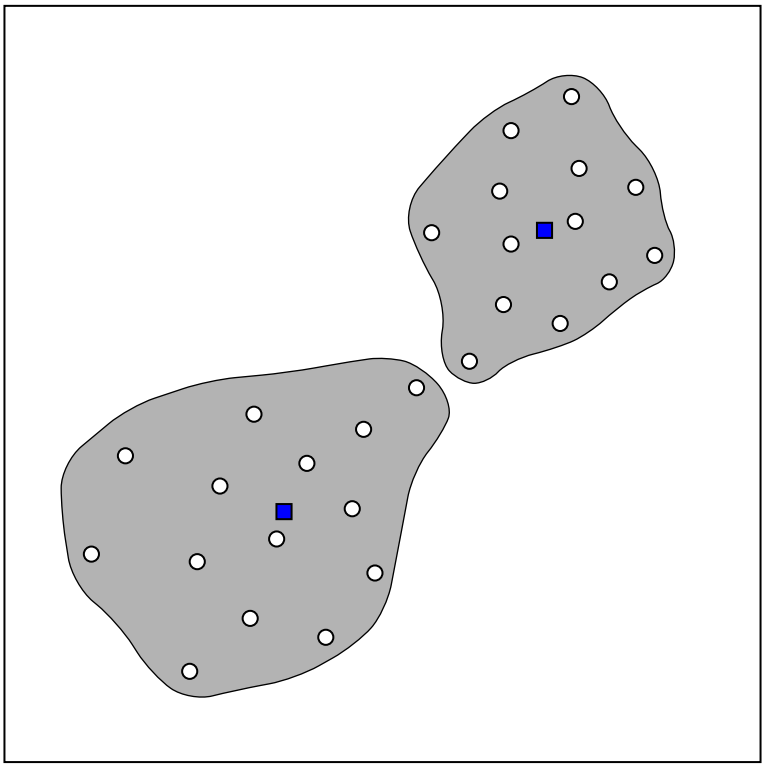
**Repeat steps 3–4 until convergence (step 3 again)**



# Unsupervised Learning

## Example: $k$-means Flat Clustering with $k = 2$

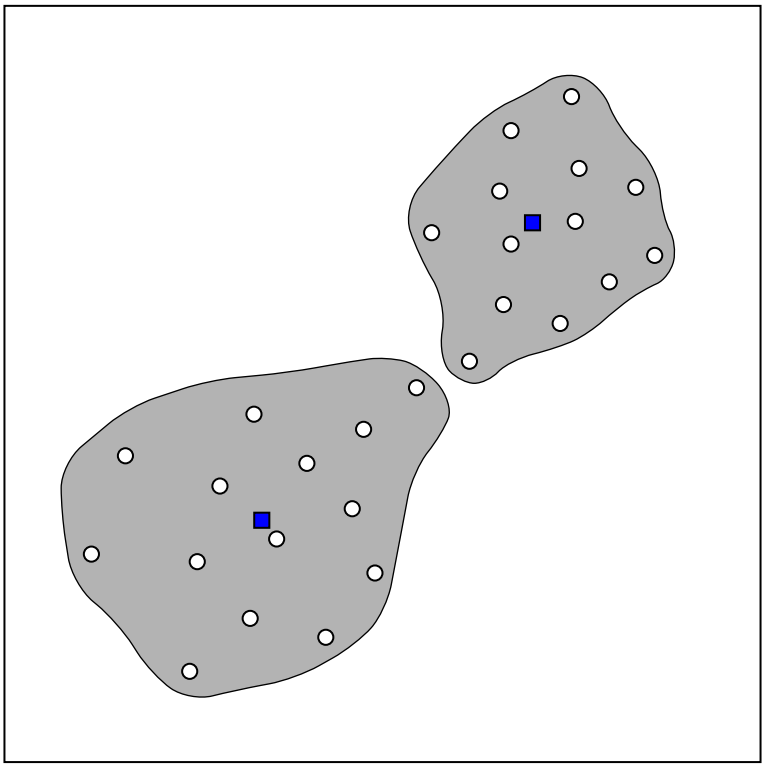
**Repeat steps 3–4 until convergence (step 4 again)**



# Unsupervised Learning

## Example: $k$-means Flat Clustering with $k = 2$

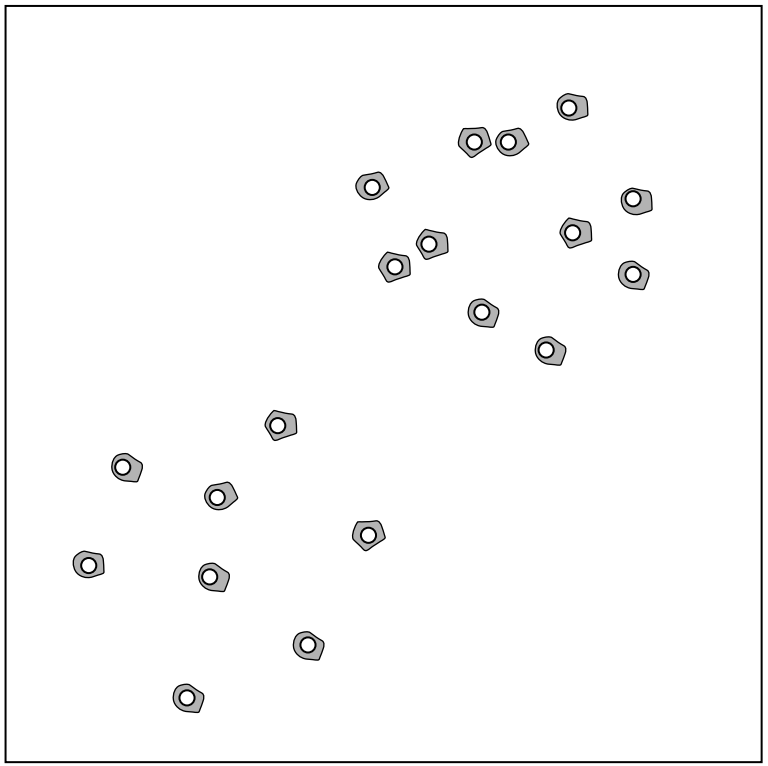
**Convergence!**



# Unsupervised Learning

## Example: Agglomerative Hierarchical Clustering

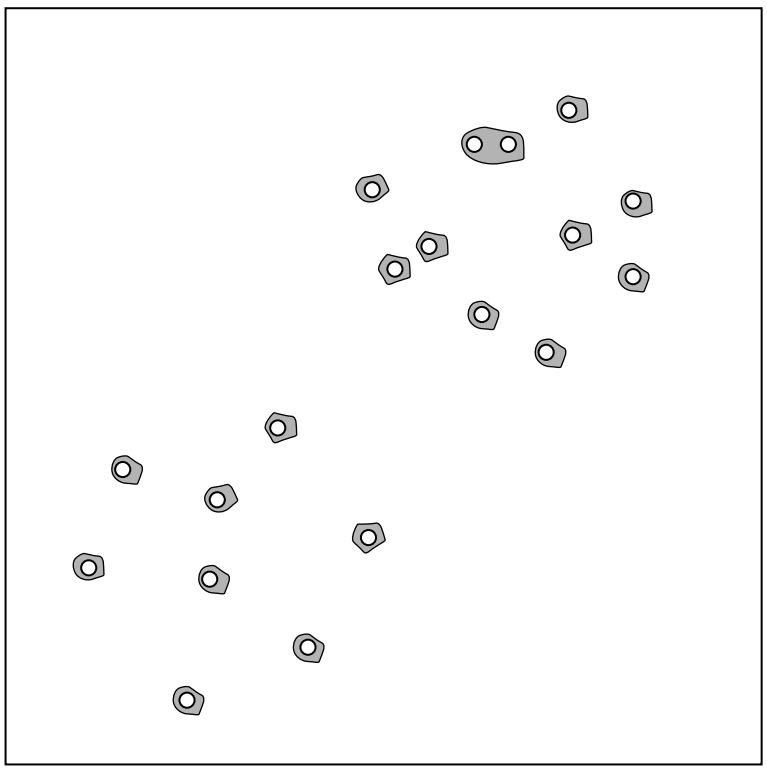
### Step 1: Assign each instance to individual cluster

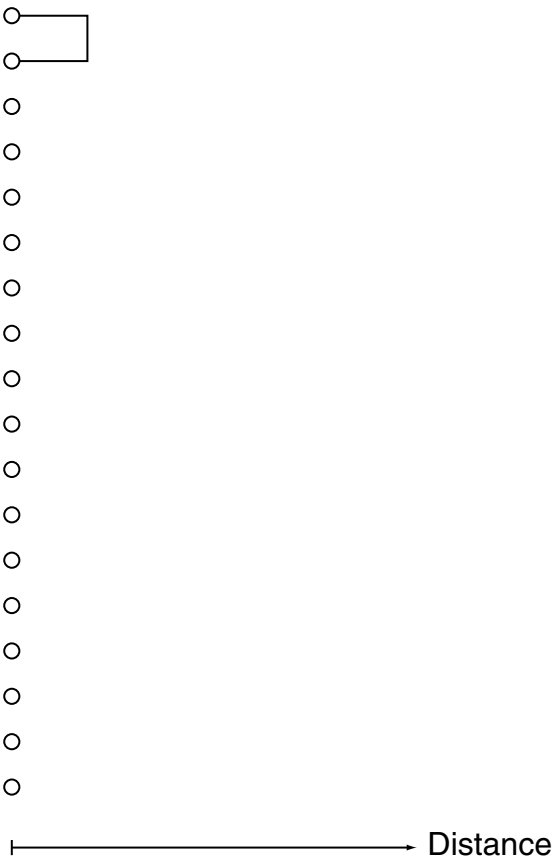


# Unsupervised Learning

## Example: Agglomerative Hierarchical Clustering

### Step 2: Combine closest pair of clusters into one cluster

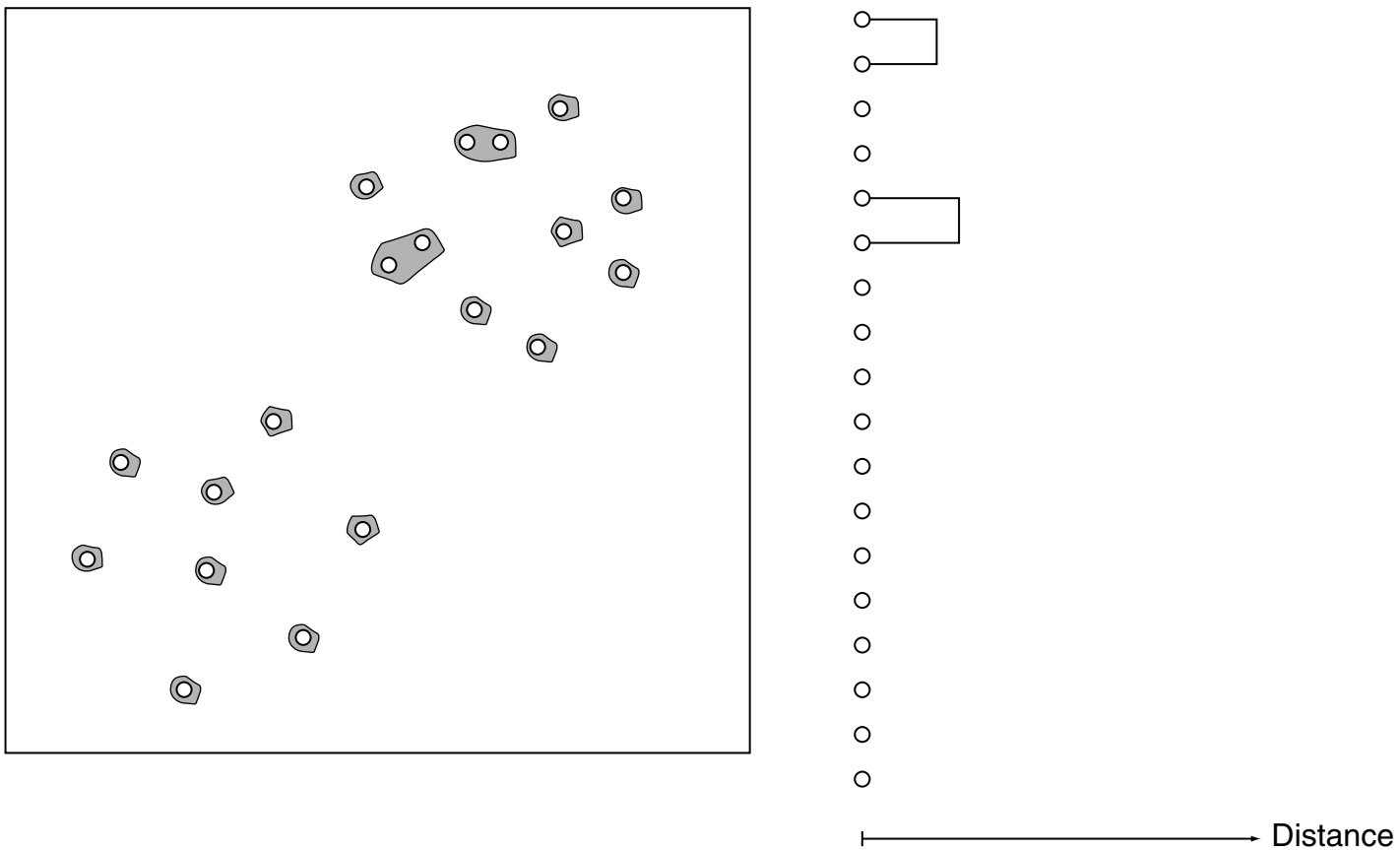




# Unsupervised Learning

## Example: Agglomerative Hierarchical Clustering

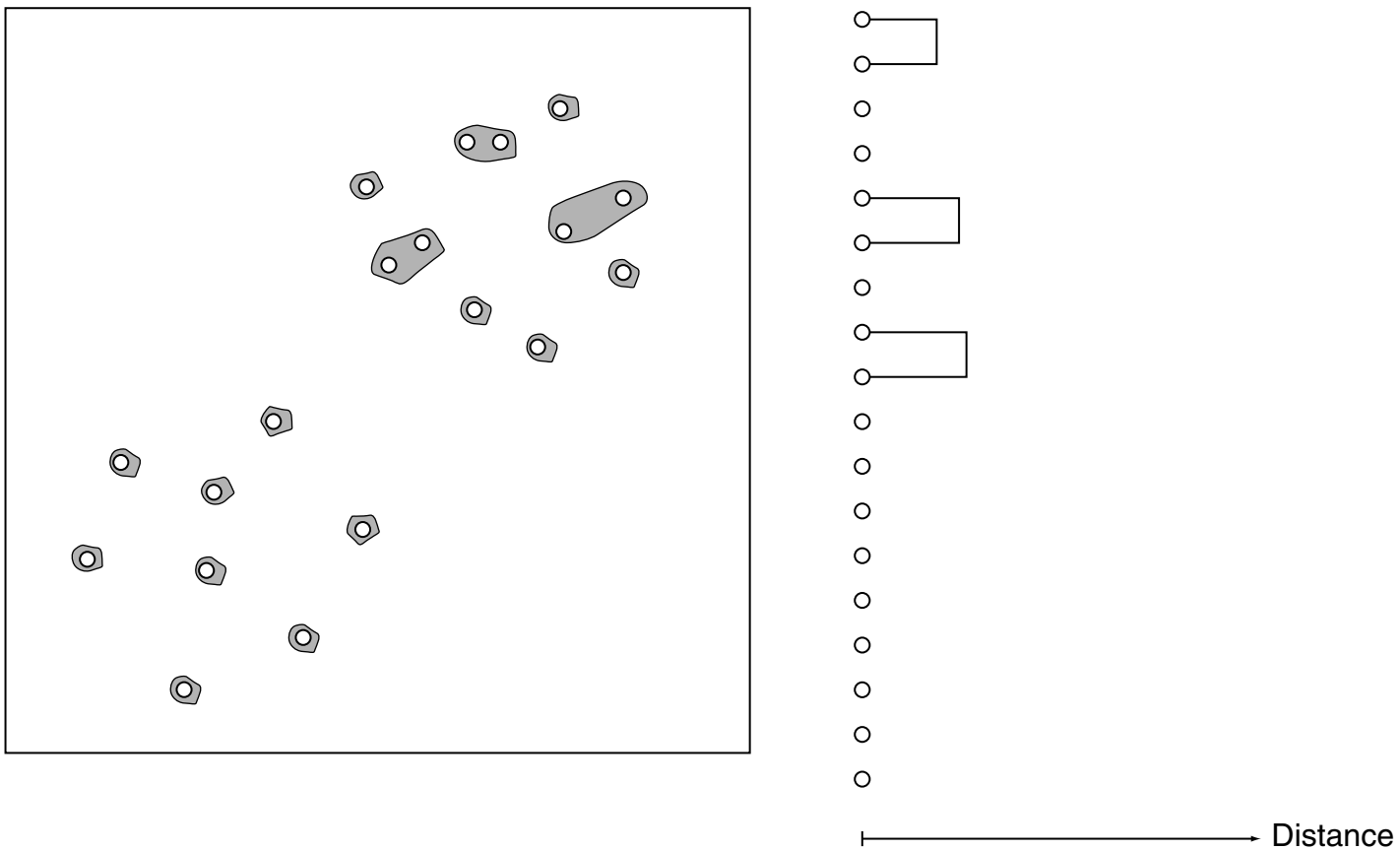
**Repeat step 2 until only one cluster remains**



# Unsupervised Learning

## Example: Agglomerative Hierarchical Clustering

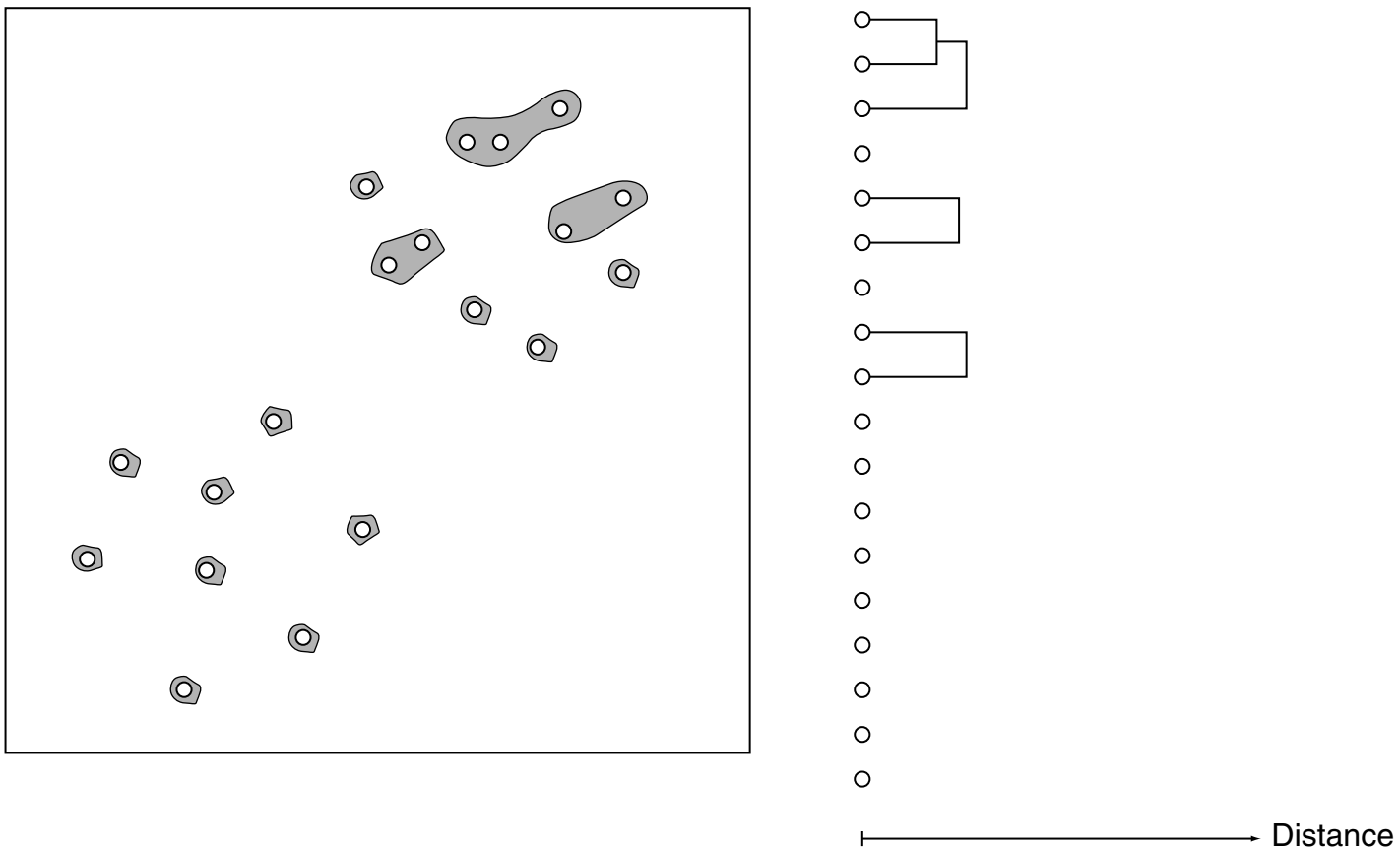
**Repeat step 2 until only one cluster remains**



# Unsupervised Learning

## Example: Agglomerative Hierarchical Clustering

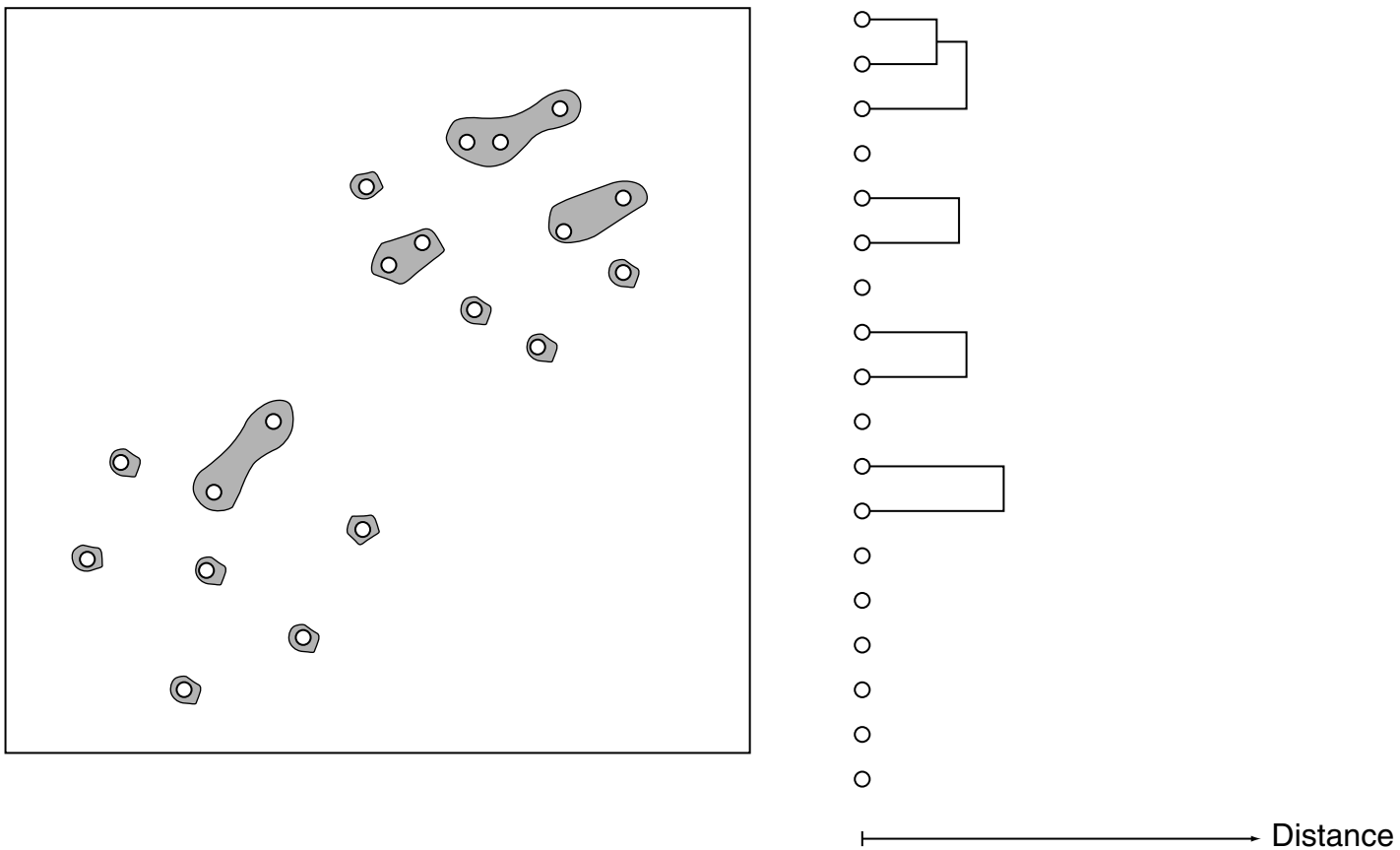
**Repeat step 2 until only one cluster remains**



# Unsupervised Learning

## Example: Agglomerative Hierarchical Clustering

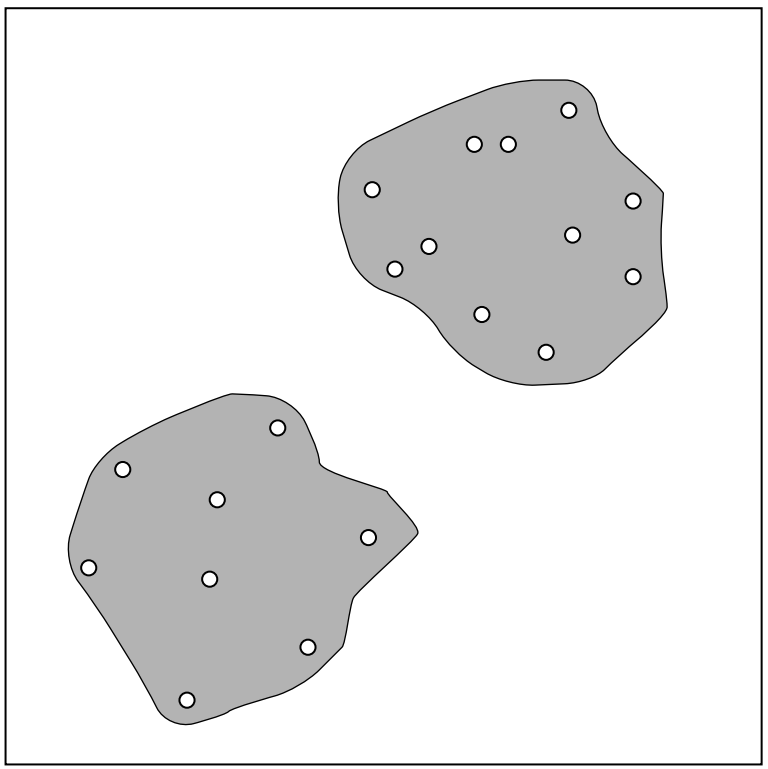
**Repeat step 2 until only one cluster remains**

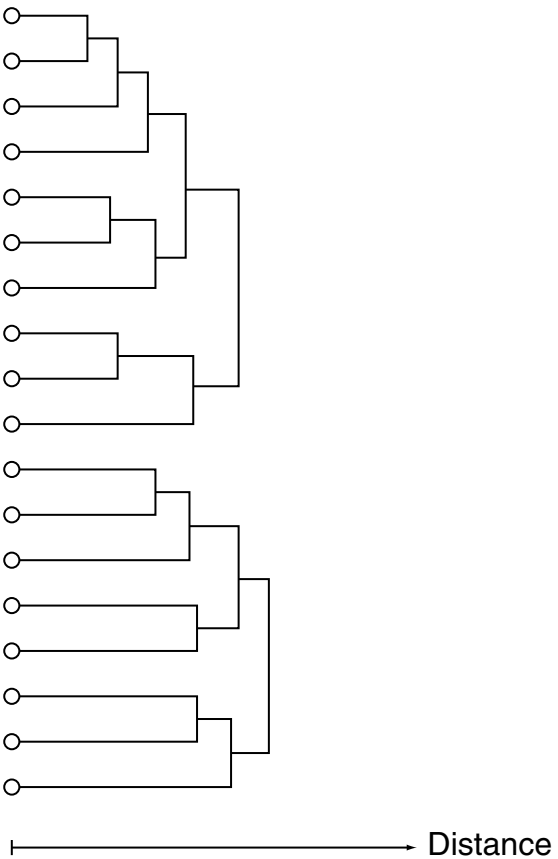


# Unsupervised Learning

## Example: Agglomerative Hierarchical Clustering

**Repeat step 2 until only one cluster remains**

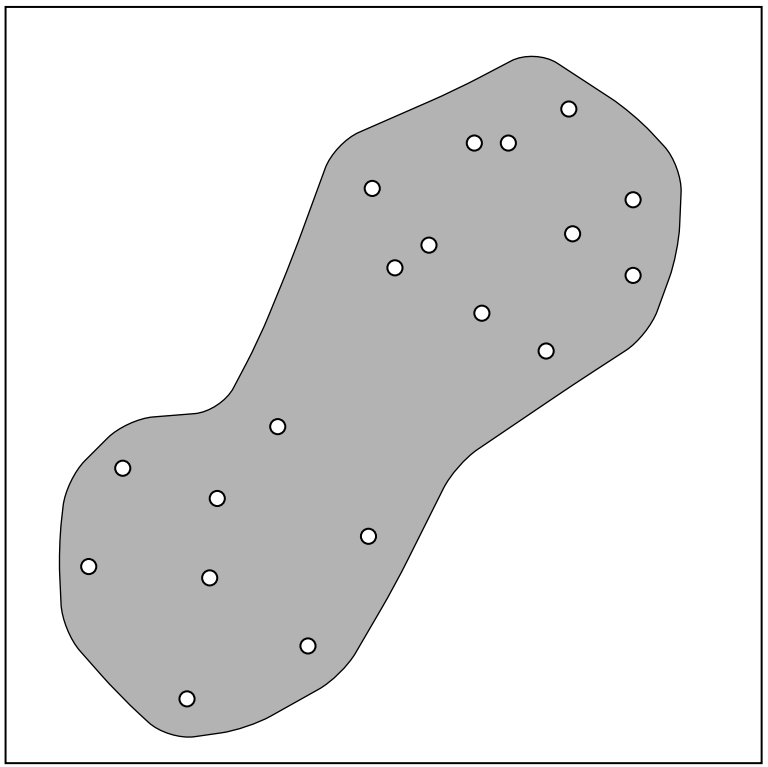


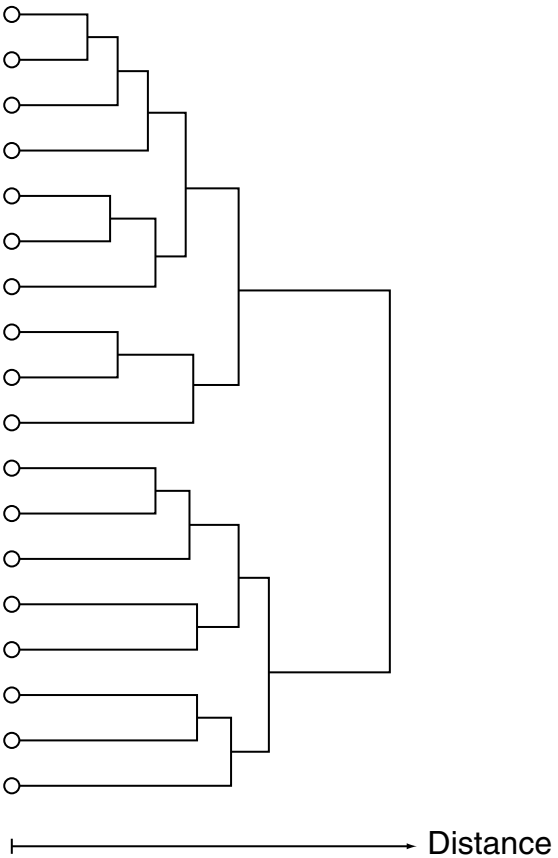


# Unsupervised Learning

## Example: Agglomerative Hierarchical Clustering

**The dendrogram on the right shows the final hierarchical clustering!**





# Learning Types and Algorithms

## Machine Learning: Comparison

### Learning processes

- Supervised. The process is guided by an external “teacher”.
- Unsupervised. The process is self-organized (no “teacher”).

### Optimization criterion

- Supervised. The criterion relates to the target function of a task/domain. The goal is to approximate the function best possible.
- Unsupervised. The criterion is usually task- and domain-independent.

### Learning Techniques

- Many prediction tasks can be tackled with different techniques.
  E.g., classification $\sim$ regression + thresholding.
- Still, for most prediction tasks, there is a most reasonable “first choice”.
  Mostly dependent on prediction goal and available data.

| Technique | Prediction Goal |
|---|---|
| Clustering | Groups |
| Classification | Group membership |
| Regression | “Missing” values |

# Learning Types and Algorithms

## Example Supervised and Unsupervised Learning Problems

### How to tackle these problems?

1. Predict the number of iPhones that will be sold over the next year.
2. Decide for each mail account whether it has been hacked (1) or not (0).

(a) Both as classification problems. → Not reasonable

(b) #1 as a classification problem, #2 as a regression problem. → Not reasonable

(c) #1 as a regression problem, #2 as a classification problem. → Correct

(d) Both as regression problems. → Possible

### Which should be tackled with unsupervised learning?

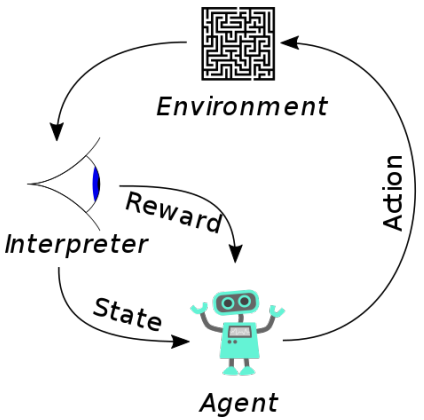
1. Given files in two folders (“public”, “private”), learn to distinguish them. → No
2. Given a set of twitter posts, group them into sets about the same event. → Yes
3. Given a database of user profiles, discover market segments, and assign new users to market segments. → Yes
4. Given political texts labeled to have a left, neutral, or right ideology, classify ideology for new texts. → No

# Learning Types and Algorithms

## Select Other Learning Types

### Reinforcement learning

- Learn, adapt, or optimize a behavior to maximize own benefit, based on feedback provided by the environment.
- Example. Agents negotiating in online auctions.



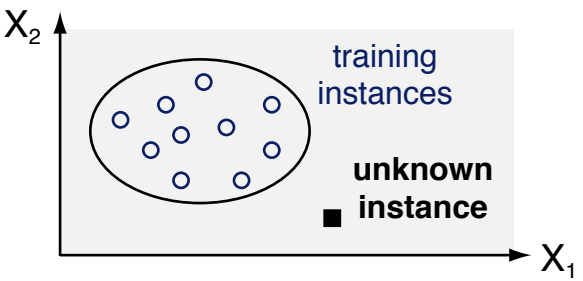

### Recommender systems

- Predict missing values of entities based on values of similar entities.
- Example. Suggesting products, movies, ...

| User | Item 1 | Item 2 | Item 3 |
|---|---|---|---|
| Max | 1 | 1 | ? |
| Linda | ? | 0 | 0 |
| Tim | 1 | 0 | ? |
| Sue | 1 | 1 | 1 |

### One-class classification and outlier detection

- Learn to classify without having an anyhow representative sample of one class.
- Example. Detecting fraud or anomalies.



# Application of Machine Learning

# Application of Machine Learning

## Machine learning in text mining

- Machine learning serves as a technique to approach a given task.
- Learning algorithms are rarely implemented newly.
- Rather, a suitable algorithm from an existing library is applied.

## Selected libraries and toolkits

- Java. Weka Machine Learning Project, v3.8, cs.waikato.ac.nz/ml/weka
- Python. scikit-learn, v0.22, http://scikit-learn.org/stable/

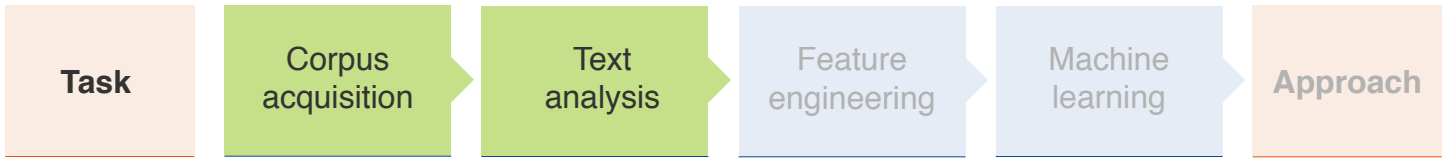
## Development of a machine learning approach

- The following high-level development process is usually carried out, both conceptually and operationally.

  The process is iterative in general, i.e., steps back may be done.

# Application of Machine Learning

## Development Process, Steps 1 and 2




### Corpus acquisition

1. Acquire or build a corpus suitable to study the task.
2. If needed, convert the corpus to a reasonable format.
3. If not given, split the corpus into training and test datasets.

   Step 2 and possibly step 3 require proprietary code.

### Text analysis

1. Preprocess all corpus texts with existing text analysis algorithms, in order to obtain more information that can be used in features.
2. Derive instances from the data; sometimes the definition of negative instances is not trivial.

   Different text analysis frameworks exist. The derivation may require proprietary code.

# Application of Machine Learning

## Development Process, Steps 3 and 4


### Feature engineering

1. Identify and implement potentially helpful feature types on training set.
2. Determine concrete features of types on training set.
3. Compute feature vectors for each instance from all datasets.

   Usually requires a lot of proprietary (but widely reusable) code.

### Machine learning

1. Choose a learning algorithm suitable for the data and task.
2. Automatically train the algorithm on the training set.
3. Evaluate algorithm against a validation set, optimize hyperparameters.
4. In the very last run, evaluate against a test set.

   Can be done from code using libraries, or in stand-alone tools.

# Conclusion

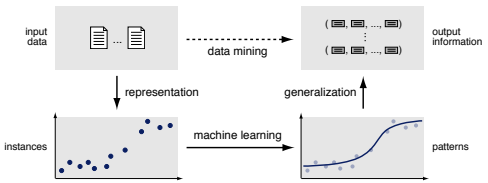
# Summary

## Machine learning

- Aims to learn target functions for prediction tasks.
- Infers models from statistical patterns in data.
- Models can be used to approach prediction tasks.

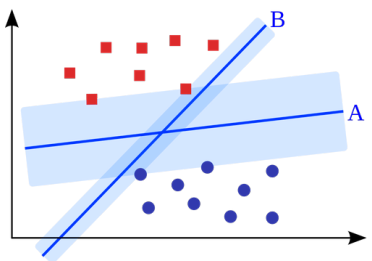
## Machine learning within data mining

- Instances are usually represented by features.
- Machine learning optimizes feature weights.
- Learned models are generalized to new data.




## Text mining using machine learning

- Focus on supervised and unsupervised learning.
- Existing algorithms are applied to approach tasks.
- The decisive step is feature engineering.



# References

## Much content and many examples taken from


- Andrew Ng (2018). Machine Learning. Lecture slides from the Stanford Coursera course. https://www.coursera.org/learn/machine-learning.
- Benno Stein and Theodor Lettmann (2010). Machine Learning. Lecture Slides. https://webis.de/lecturenotes/slides.html#machine-learning
- Henning Wachsmuth (2015): Text Analysis Pipelines — Towards Ad-hoc Large-scale Text Mining. LNCS 9383, Springer.
- Ian H. Witten and Eibe Frank (2005): Data Mining: Practical Machine Learning Tools and Techniques. Morgan Kaufmann Publishers, San Francisco, CA, 2nd edition.