Introduction to Text Mining

Part VI: Basics of Machine Learning

Henning Wachsmuth

https://cs.upb.de/css

Basics of Machine Learning: Learning Objectives

Concepts

- Get to know the basic ideas of machine learning.
- Learn what problems can/should be addressed with machine learning.
- Understand the role of machine learning in text mining.

Methods

- Learn about feature representation and pattern generalization.
- Get an idea of supervised and unsupervised learning.
- Get an idea of selected existing machine learning algorithms.
- Understand how to approach a task with machine learning.

Notice

- This part will not explain how machine learning works in detail.
- Rather, it aims to set the basis for applying 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 Research

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 Clustering

VIII. Text Mining using Classification and Regression

IX. Practical Issues

X. Text Mining using Sequence Labeling

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
...	

...

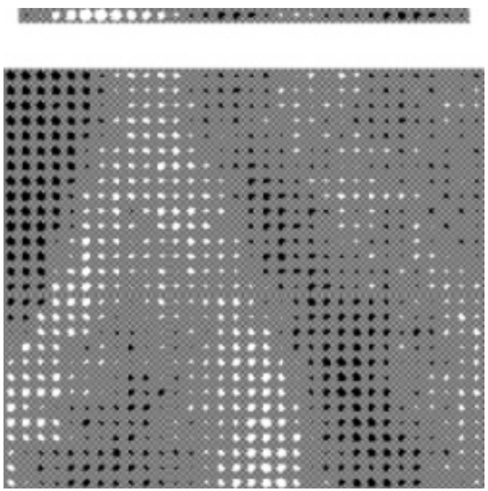
Customer n	
house owner	no
income (p.a.)	55 000 EUR
repayment (p.m.)	1 200 EUR
credit period	8 years
SCHUFA entry	no
age	?
married	yes
...	

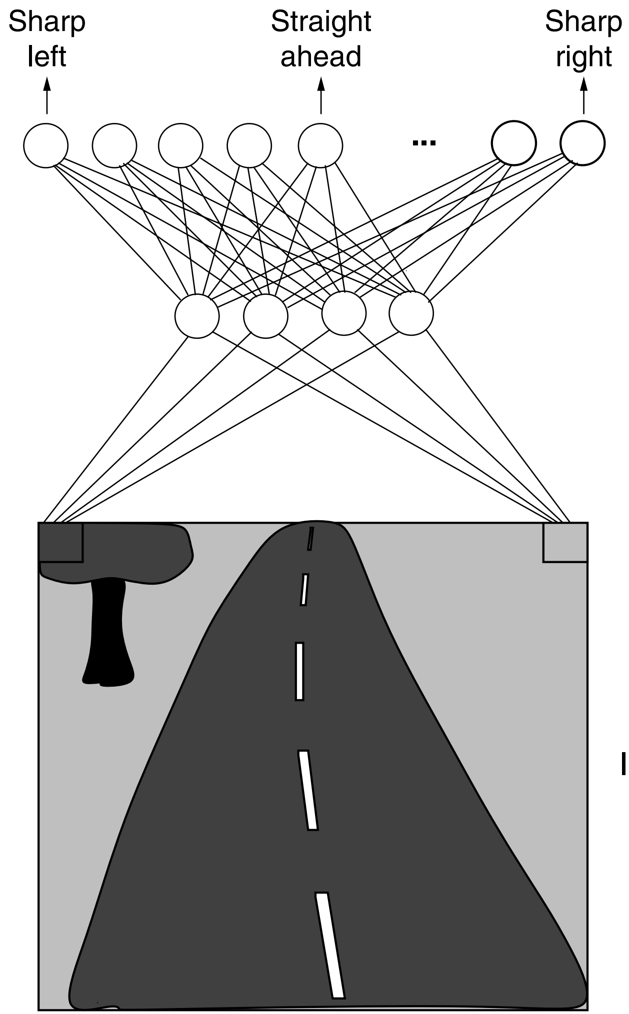
Learned rules

```
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



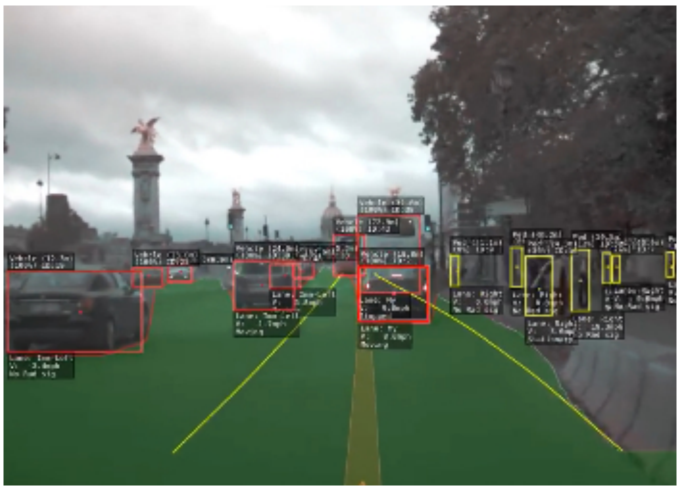


Input retina

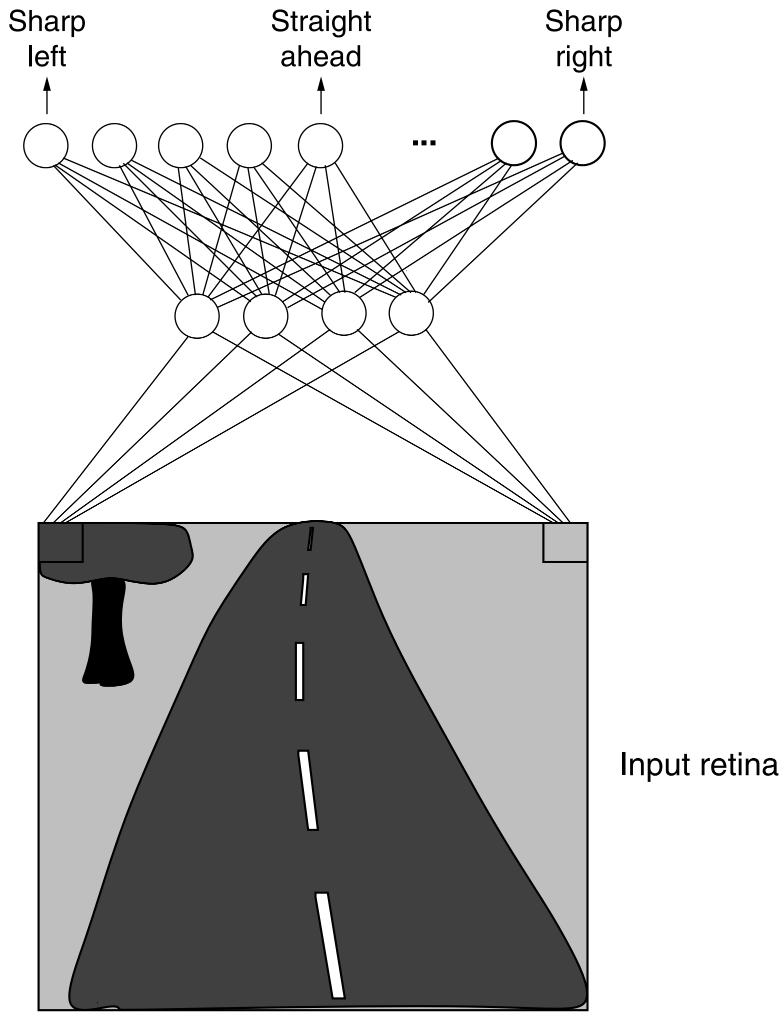
(1992)

Examples of Learning Tasks

Autonomous Driving



(2018)



Examples of Learning Tasks

Named Entity Recognition

Example: Part of a person entity?

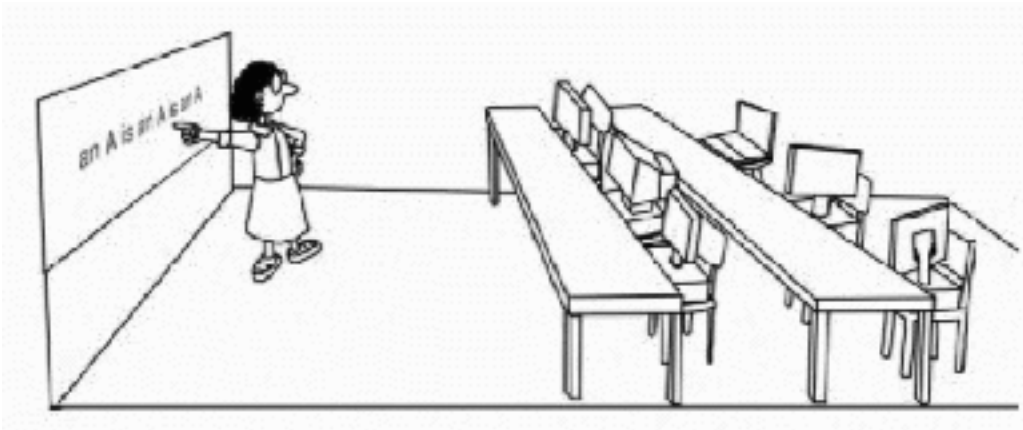
1. "Washington was born into slavery on the farm of James Burroughs."
2. "Blair arrived in Washington D.C. for what may be his last state visit."
3. "In June, Washington passed a primary seatbelt law."

Considered Feature		Candidate 1	Candidate 2	Candidate 3
Previous token	Token	$\perp$	"in"	","
	Capitalization	$\perp$	true	$\perp$
	POS tag	$\perp$	IN	COMMA
	Phrase type	$\perp$	B-PP	$\perp$
	...	...	...	...
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

Machine Learning Parameters

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 not spam. $\rightarrow$ Task

Watching you label emails as spam or not spam. $\rightarrow$ Experience

The fraction of emails correctly classified as spam/not spam. $\rightarrow$ Performance measure

Parameters in the given context

- Experience. Text corpora annotated for some target variable C.
- Task. Any problem, where a target function γ is to be found that maps input texts to output instances of some C.
- Performance measure. Usually capturing effectiveness.

Machine Learning

Target Variables and Function

Target variable

- A target variable C captures the output information sought for in a task.
- The values of C are usually nominal (classes) or real-valued.

Ideal target function

- γ interprets any real-world instance o to "compute" $\gamma(o)$ where $\gamma(o)$ corresponds to a value of some target variable.
- This "computation" can be operationalized by a human or by some other (possibly arcane) mechanism of the real world.

Why learning?

- Machine learning aims at problems where γ is unknown.
- All non-trivial text analysis tasks denote such problems.

  Including those that can be successfully tackled with rule-based approaches.

Machine Learning

Machine Learning in Text Mining

Output information in text mining

- 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.

Target functions in text mining

- Predict the output information for a given text or span of text.

Two-way relationship

- The output information of text analyses serves as the input to machine learning, e.g., to train a text classifier.
- Many text analysis algorithms rely on machine learning to produce output information.

Machine Learning within Data Mining

Data Mining

Data mining

- Aims to infer new output information of specified types from typically huge amounts of input data.
- The input data needs to be given in structured form.
- Approaching this inference can be understood as a *prediction problem*.

Data mining in a nutshell

- Representation. Map data to instances of a defined representation.
- Machine learning. Recognize statistical patterns in the instances that are relevant for the prediction problem (in many cases aka *training*).
- Generalization. Apply the found patterns to infer new information from unseen data (aka *prediction*).

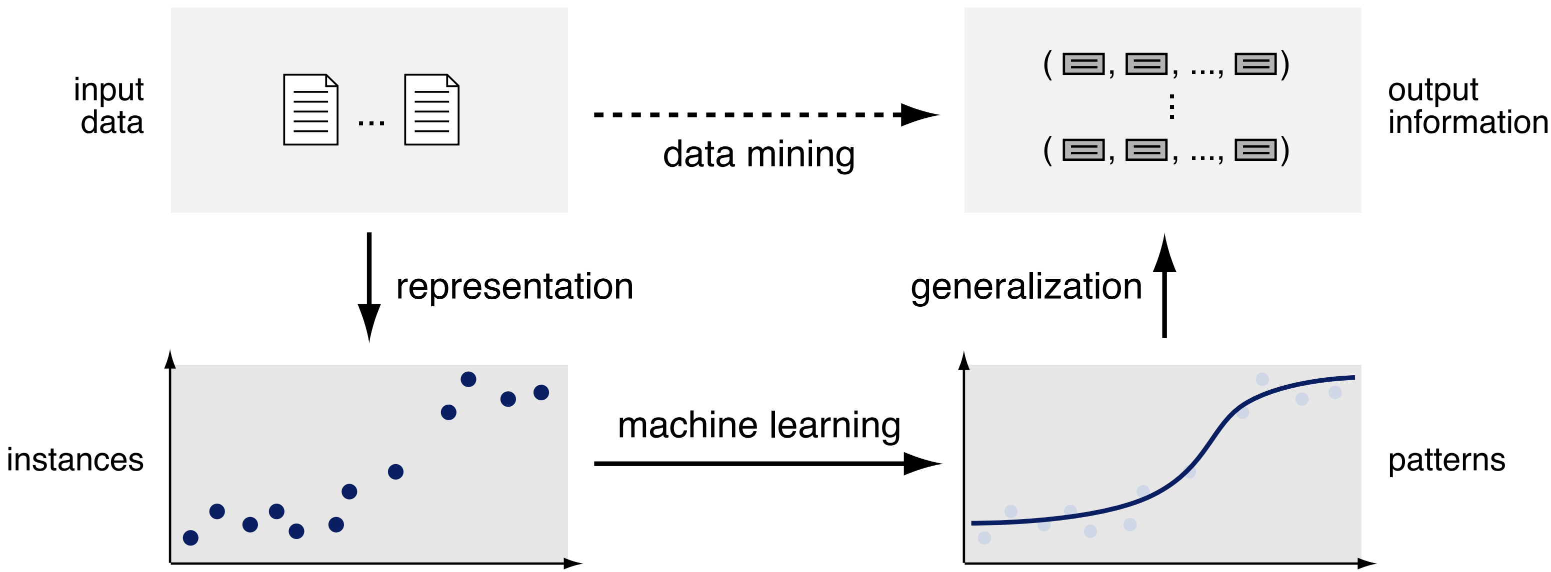
Data mining vs. machine learning

- 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



Concretization for text mining

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

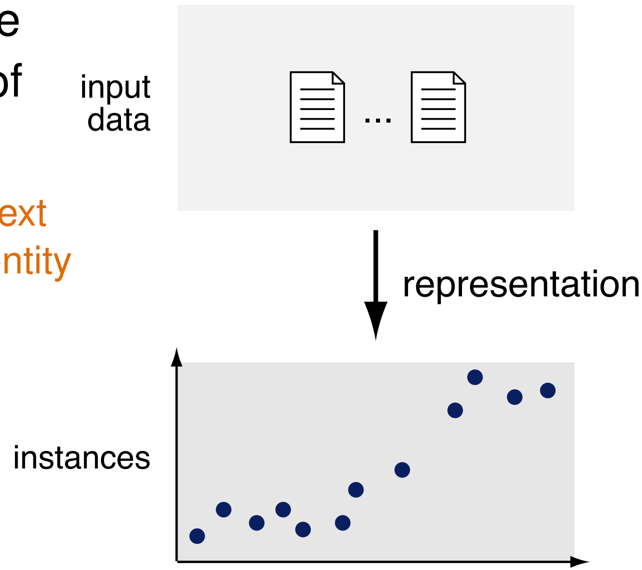
Representation

Feature-based instance representation

- Given a dataset with annotations of some target variable C, each annotated span of text defines one instance o.

  Sentiment analysis: one instance per labeled text
  Entity recognition: one instance per annotated entity

- For many machine learning algorithms, each o is mapped to one *feature vector* via a mapping function α.
- The set of all feature vectors for a given dataset is denoted as X.



Feature-based vs. neural representations

- We restrict our view to feature-based machine learning in this course.
- In neural machine learning, as an alternative, features are not explicitly represented anymore.

Representation

Features

What is a feature?

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

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 $\mathbf{x} = (x_1, \ldots, x_m)$.
- Each feature vector $\mathbf{x}^{(i)}$ contains one value $x_j^{(i)}$ for each feature $x_j \in \mathbf{x}$.
- The number m of features varies depending on the approach, from a handful up to hundreds of thousands.

Representation

Feature Types and Scales

Feature types

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

  Bag-of words. The relative frequency of each considered word.

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

- The concrete features of a type are often found automatically on training data, in order to exclude useless features that would 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”} becomes VP $\rightarrow$ {0,1}, NP $\rightarrow$ {0,1}, PP $\rightarrow$ {0,1}

- Usually, the values of all features are normalized to the same interval, which benefits the optimization process of machine learning.

  Common intervals are [0,1] and [-1, 1], respectively.

Representation

Feature Engineering

Representation in learning

- The representation governs what patterns can be found during learning.
- The mapping α determines the level of abstraction between o and $\mathbf{x}$.
- Engineering features, which predict a given target variable C and which generalize well, is the key step in feature-based learning.

Feature engineering in text mining

- Common feature types such as bag-of-words help in many tasks.
- The most discriminative features usually encode expert knowledge about the task and input.
- Also, some features generalize worse than others towards unseen data.

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.

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) token 1-grams ("bag-of-words") (b) text length in # tokens and # sentences

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

   (a) "the" → 4242, "a" → 2424, . . . , "engineeeering" → 1 (b) not needed

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

   (a) "the", "a", . . . , ~~"engineeeering"~~

How to compute the values for each feature

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

   (a) "the" → 6, "a" → 7, . . . (b) # tokens → 50, # sentences → 10

2. Normalize feature values.

   (a) "the" → 0.12, "a" → 0.14, . . . (b) # tokens → 0.42, # sentences → 0.5

Representation

Feature Value Normalization

What is feature value normalization?

- The range of values of different numeric features may vary drastically.
- Normalization scales the values of all features 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 range is covered by the values of a feature.

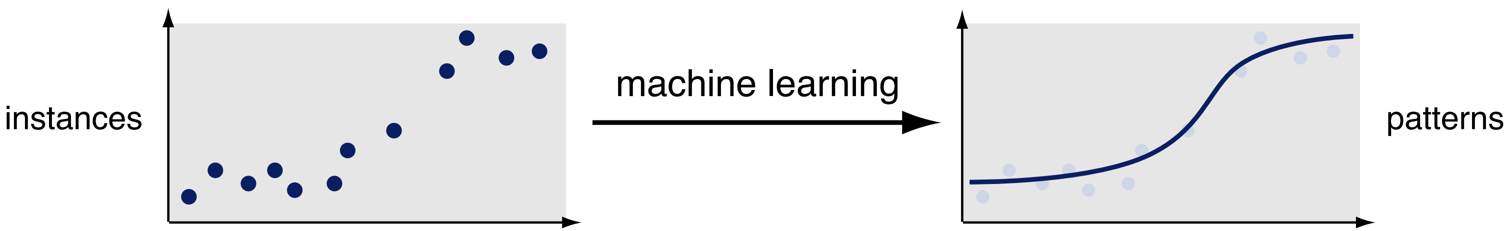
Typical ways to normalize

- 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).

Machine Learning

Machine learning models

- Machine learning recognizes statistical patterns in input data that are relevant for a given prediction problem.
- The patterns are generalized to approximate a target function γ by a model $y : X \to C$ that maps from instances X to a target variable C.
- The goal is to find the y that is optimal wrt. an evaluation measure.



Model vs. target function

- The key difference between γ and y lies in the complexity and the representation of their respective domains.
- α simplifies instances o into a set of measurable features $\mathbf{x} = \alpha(o)$.
- The model $y(\mathbf{x})$ is the abstracted and formalized counterpart of $\gamma(o)$.

Machine Learning

From Target Function to Model (1 out of 2)

Characterization of the real world

- O is a set of instances, C is a target variable.
- $\gamma : O \rightarrow C$ 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 mails, C is "spam" and "no spam".
- γ is a human expert.
- Task. Given a mail, is it spam?

Acquisition of training data

1. Collect real-world instances of the form $(o_i, \gamma(o_i)),\ \ o_i \in O$.
2. Abstract each instance o_i towards 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 world

- 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

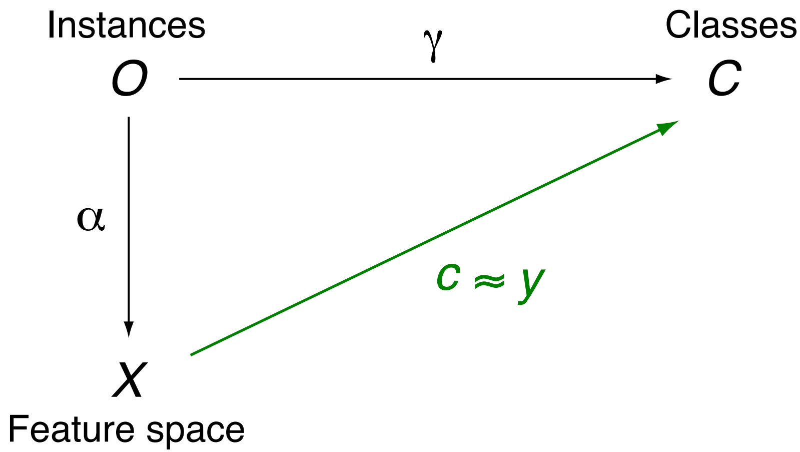
- X represents the frequencies of words.
- C as before: “spam” and “no spam”.
- c is unknown.

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, search, theory, and algorithms from machine learning to optimize the fit between c and y.

Machine Learning

Overview of the Concepts

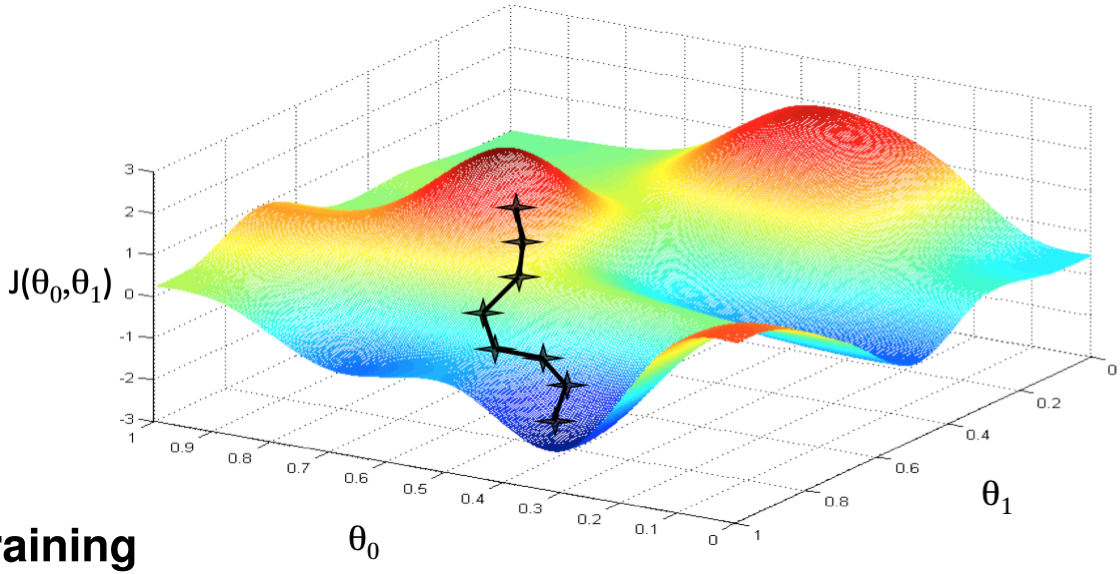


Notation

γ	Ideal classifier (a human) for real-world instances.
α	Feature mapping function.
c	Unknown ideal classifier for vectors from the feature space.
y	Classifier to be learned.
$c \approx y$	c is approximated by y (based on a set of instances).

Machine Learning

Training Process



Optimization during training

- A learning algorithm incrementally explores several candidate models y.
- Each y assigns one weight θ_j to each considered feature x_j.
- y is then evaluated on the training data against some cost function J.
- Based on the result, the weights are adapted to obtain the next model.
- The adaptation relies on an *optimization procedure*.

Hyperparameters

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

Optimization in Machine Learning

Optimization procedures

- Aims to minimize the costs of a model y wrt. a cost function J.
- Learning algorithms usually already integrate a particular procedure.
- The most common procedure is *gradient descent*.

"No free lunch theorem": No optimization procedure is best in all cases.

(Batch) gradient descent

- In each step, y is adapted to all training instances,
- Stepwise heads towards a local minimum of J 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 process 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.

Optimization in Machine Learning

Pseudocode

Signature

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

  η is a small positive constant. The higher k, the more the model will be fit to the data.

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

```
stochasticGradientDescent(List<Instance> X, double η, int k)
    List<double> w ← randomInitialize(-1, 1)
    for int j ← 1 to k do
        for each Instance (x, c(x)) in X do
            double y(x) ← w^T x = θ_0 + θ_1 · x_1 + ... + θ_m · x_m  // Regression
            double error ← c(x) − y(x)  // Cost in terms of error
            double Δw ← η · error · x
            w ← w + Δw  // Adapt weights based on error
    return w
```

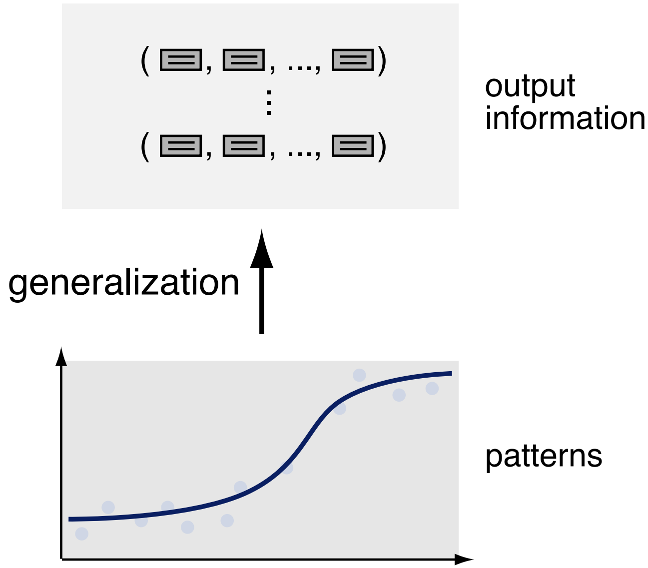
Notice

- No deep understanding of the math behind is needed in this course.

Generalization

Generalization

- Generalization targets the inference of new information from unseen data based on the learned model y.
- How well y generalizes depends on how well it fits the unknown target function γ.
- Strongly connected to the training of machine learning.



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 means here the degree of the underlying polynomial.

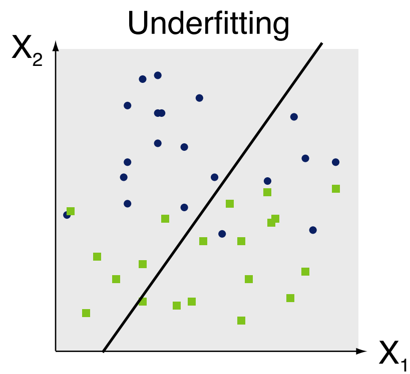
Generalization

Underfitting and Overfitting

Simple vs. complex models

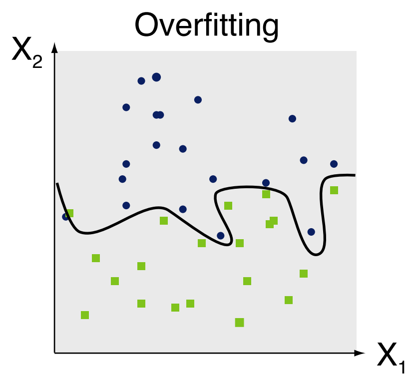
- Simple. Induce high bias to avoid noise; may underfit the input data.
- Complex. Induce low bias to fit the input data well; may capture noise.

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



Underfitting (too high bias)

- The model generalizes too much, not capturing certain relevant properties of the training data.
- It is too simple and will have limited effectiveness.



Overfitting (too high variance)

- The model captures both relevant and irrelevant properties of the input data.
- It is too complex and will thus not generalize well.

Generalization

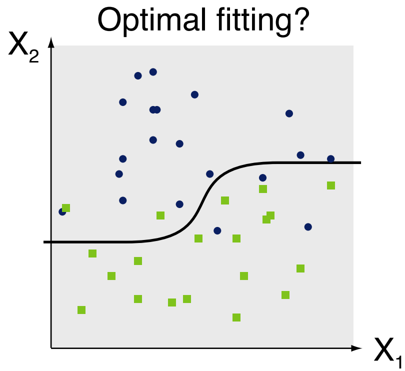
Optimal Fitting and Regularization

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

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



Regularization

- Regularization penalizes the use of high polynomials, 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.

  Details on regularization are beyond the scope of this course.

Learning Types and Algorithms

Learning Types and Algorithms

Learning types

- Machine learning differs wrt. what kind of patterns are learned as well as on what kind of data learning takes place.
- Two major learning 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 follow 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 on the choice and application of suitable algorithms for a task.

  The technical details of the algorithms are beyond the scope of the course.

Supervised Learning

What is supervised (machine) learning?

- A learning algorithm derives a model y from 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 (usually 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

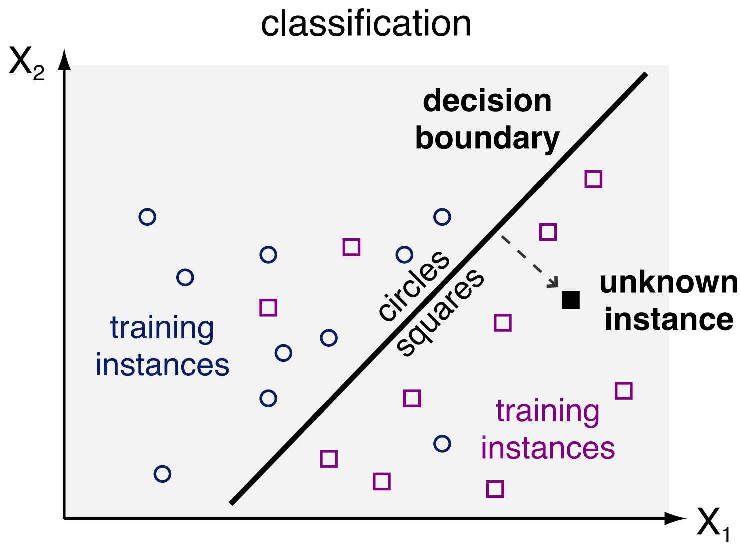
- Classification. Standard technique for any text classification task; also used for extracting relations between entities, ...
- Regression. Used to predict scores, ratings, probabilities, ...

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

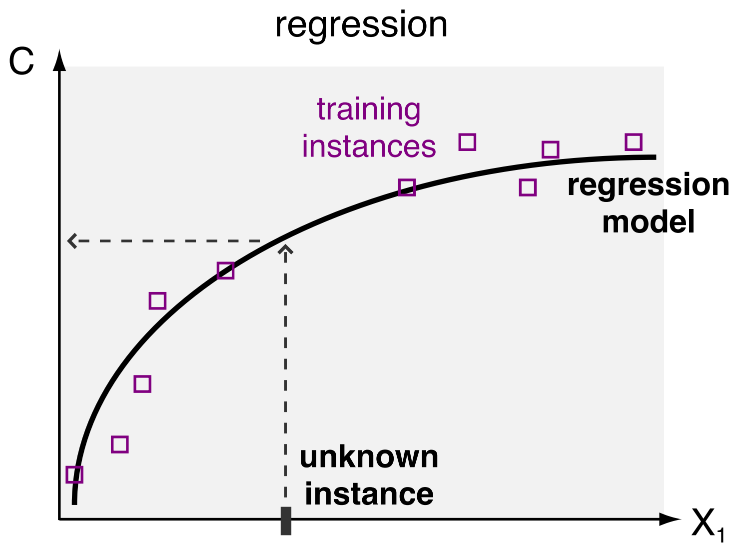
- Binary classifiers separate the instances of two classes.
- Multiple classes are handled through multiple binary classifiers with techniques such as one-versus-all classification.

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.

$$y(\mathbf{x}) = \theta_0 + \theta_1 \cdot x_1 + \ldots + \theta_m \cdot x_m$$

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

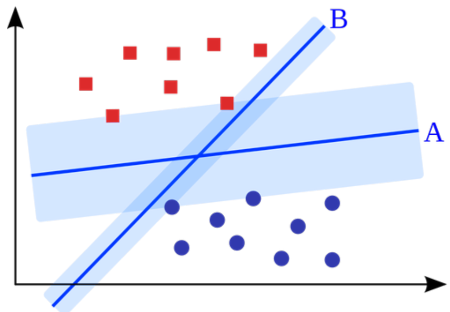
Naïve Bayes

- 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, and scales up well in terms of efficiency.
- Strong feature independence assumption often limits effectiveness.

Support vector machine (SVM)



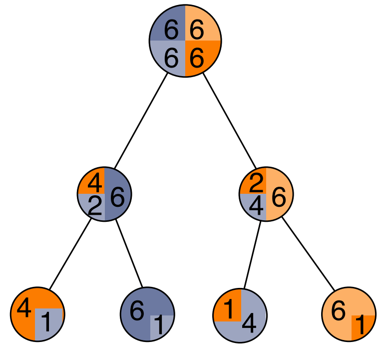
- Maximizes the margin between the decision boundary and the class instances in training.
- Maps instances into a higher-dimensional space where they are linear separable (*kernel trick*).
- SVMs often perform well while not being prone to adapt to noise.
- Hyperparameter tuning tends to be time-intensive.

Supervised Learning

Decision Trees and Random Forest

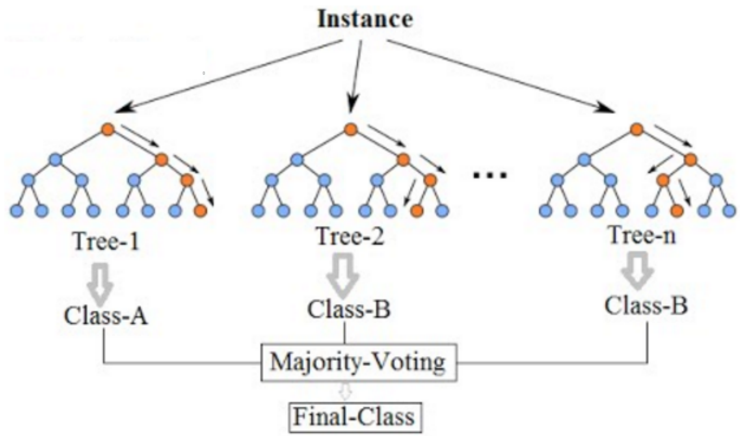
Decision trees

- Each inner node tests one feature x, with a learned threshold. Leafs correspond to classes.
- Nodes are, e.g., ordered based on information gain.
- Low-impact features are 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 strong even without any hyperparameter tuning.
- Performs worse if some features are dominant.

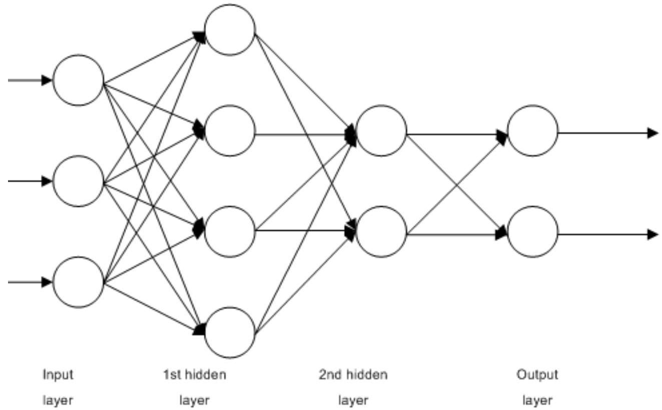


Supervised Learning

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).



- Works well only with much input data; architecture needs to be tuned.

Ensemble methods

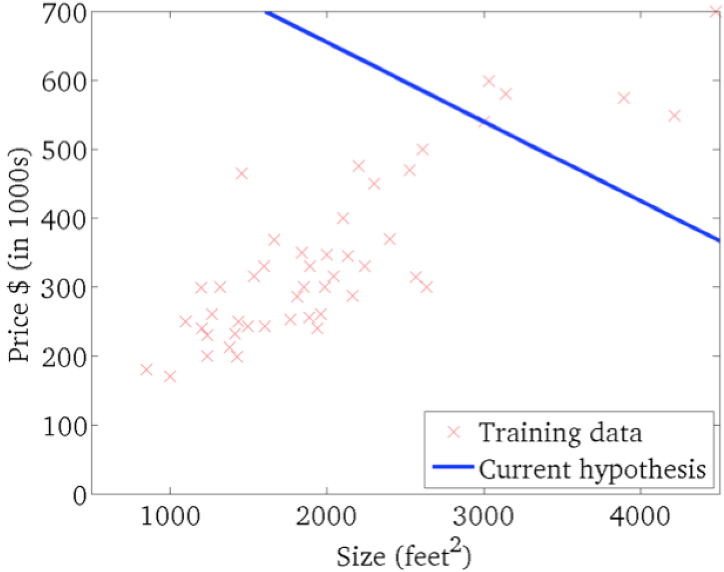
- Combination of multiple algorithms to improve effectiveness (*stacking*).
  Random forest is actually an integrated stacking method.
- Can be based on any learning algorithm (at least for classification).
- Other ensembles aim to decrease variance (*bagging*) or bias (*boosting*).

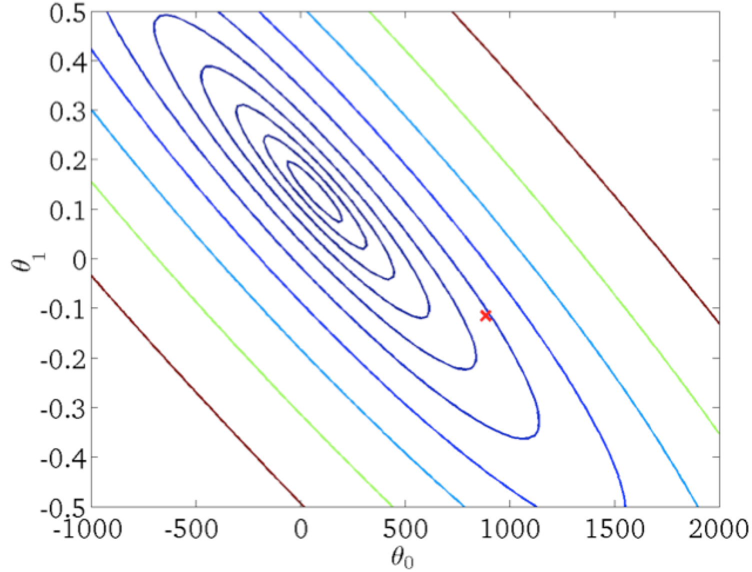
Supervised Learning

Simple Linear Regression (1 out of 9)

Example task

- Predicting the price of a house (y) from the size in square feet (x_1).





Notice

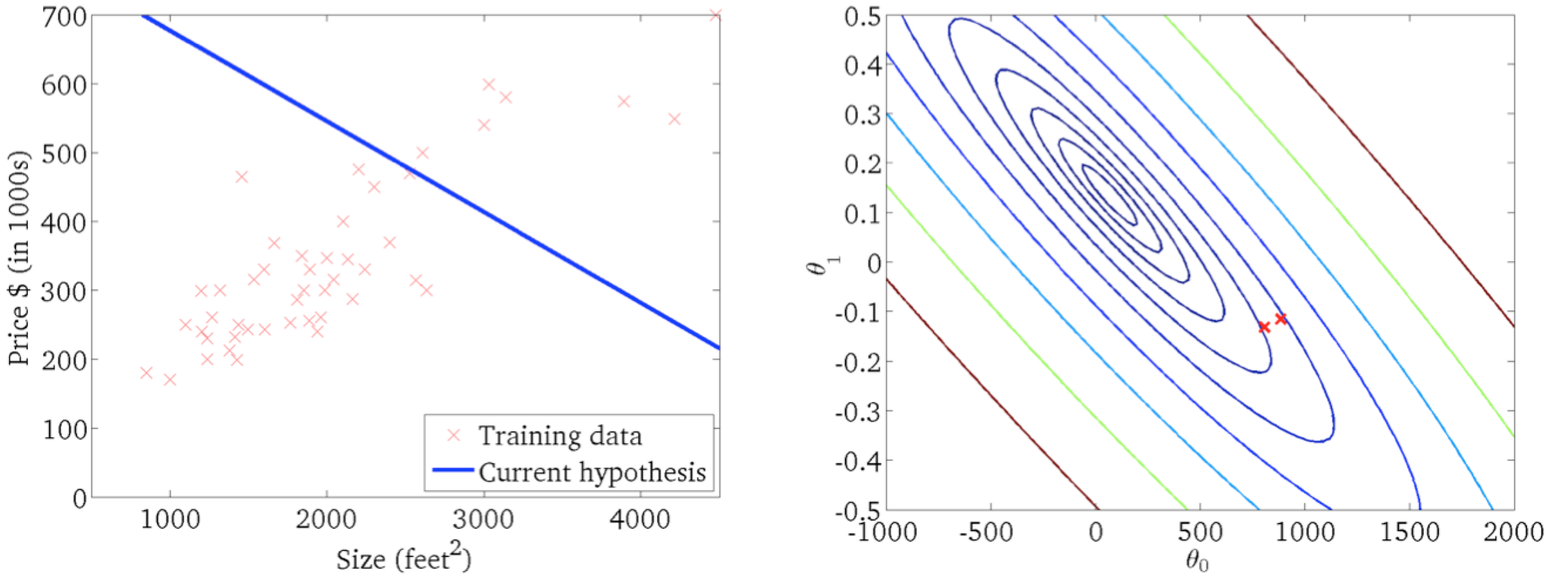
- A regression model $y = \theta_0 + \theta_1 \cdot x_1$ is learned for one single feature x_1.
- Each pair θ_0, θ_1 defines one *hypothesis*, i.e., one candidate model.

Supervised Learning

Simple Linear Regression (2 out of 9)

Example task

- Predicting the price of a house (y) from the size in square feet (x_1).



Notice

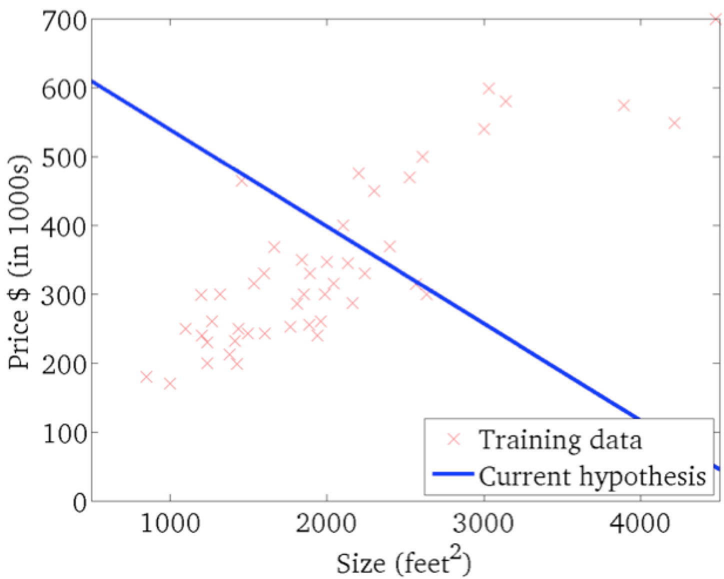
- A regression model $y = \theta_0 + \theta_1 \cdot x_1$ is learned for one single feature x_1.
- Each pair θ_0, θ_1 defines one *hypothesis*, i.e., one candidate model.

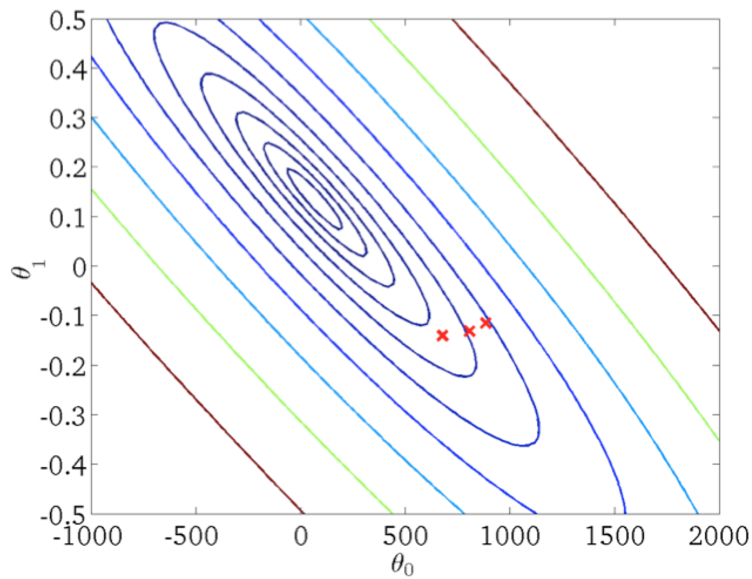
Supervised Learning

Simple Linear Regression (3 out of 9)

Example task

- Predicting the price of a house (y) from the size in square feet (x_1).





Notice

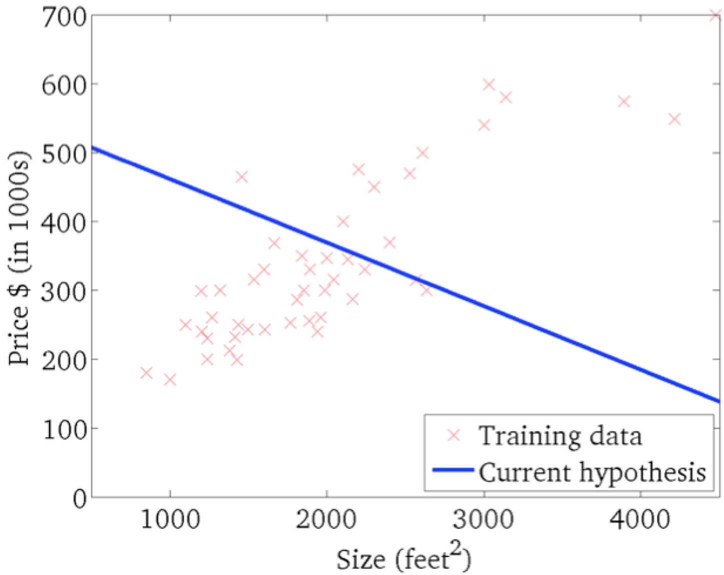
- A regression model $y = \theta_0 + \theta_1 \cdot x_1$ is learned for one single feature x_1.
- Each pair θ_0, θ_1 defines one *hypothesis*, i.e., one candidate model.

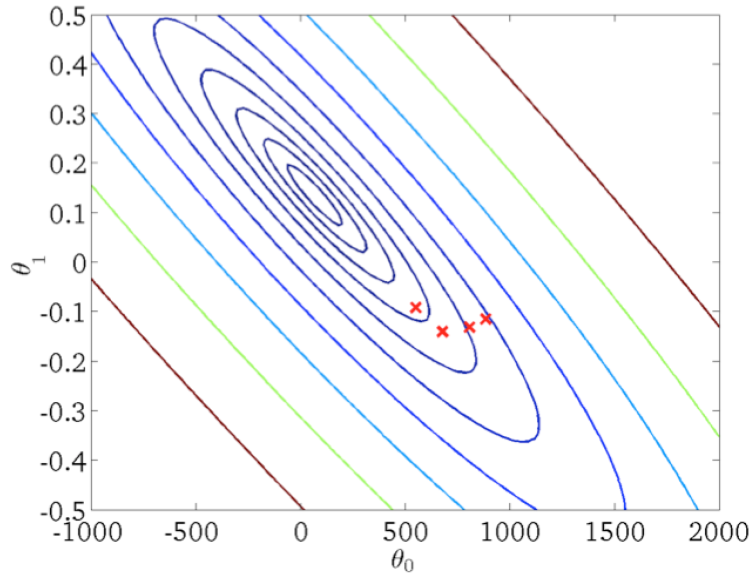
Supervised Learning

Simple Linear Regression (4 out of 9)

Example task

- Predicting the price of a house (y) from the size in square feet (x_1).



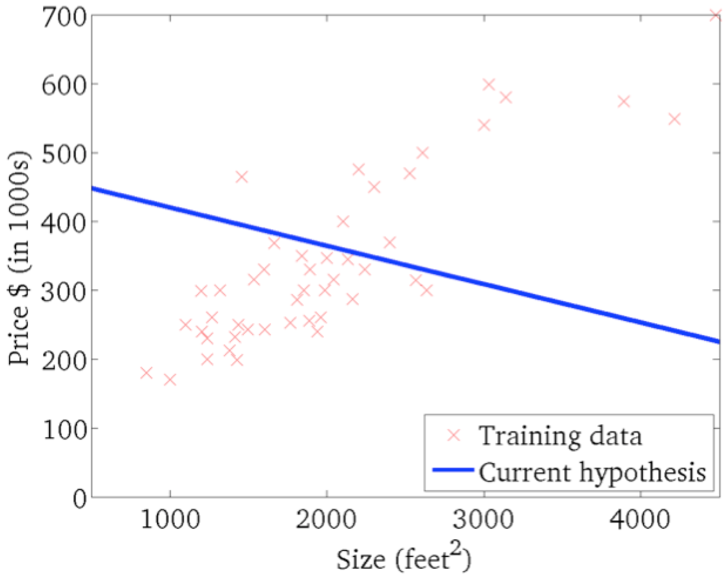


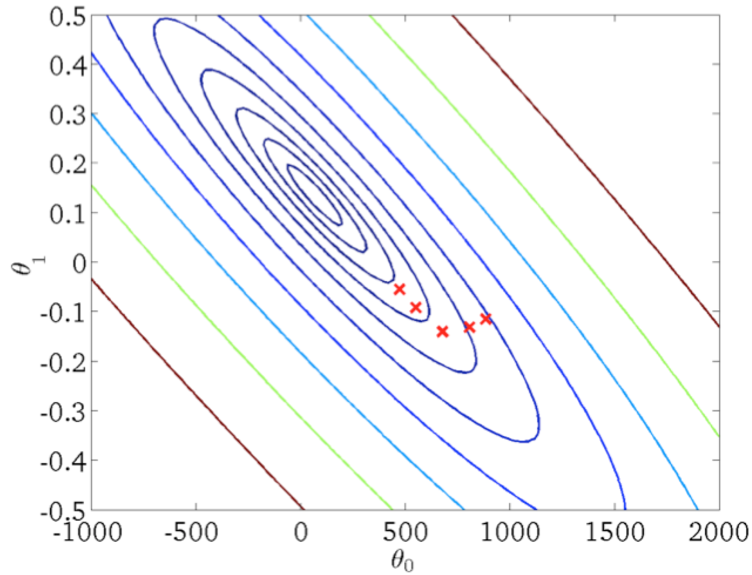
Notice

- A regression model $y = \theta_0 + \theta_1 \cdot x_1$ is learned for one single feature x_1.
- Each pair θ_0, θ_1 defines one *hypothesis*, i.e., one candidate model.

Example task

- Predicting the price of a house (y) from the size in square feet (x_1).



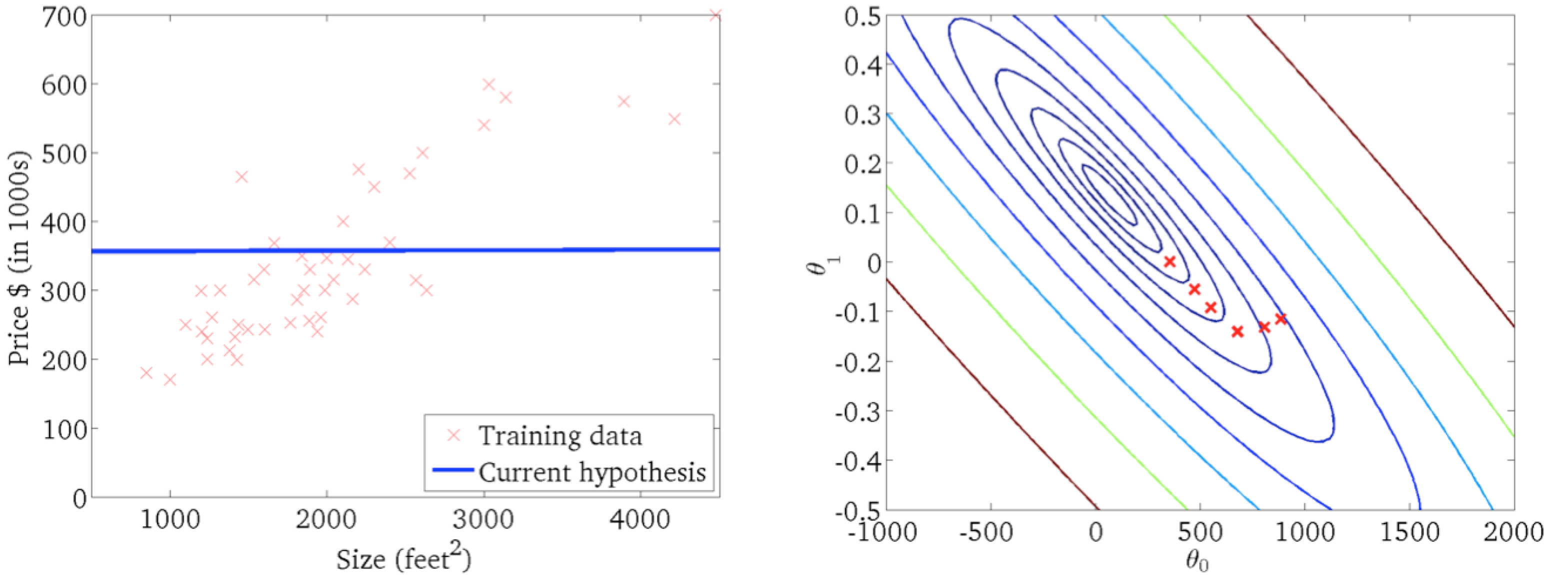


Notice

- A regression model $y = \theta_0 + \theta_1 \cdot x_1$ is learned for one single feature x_1.
- Each pair θ_0, θ_1 defines one *hypothesis*, i.e., one candidate model.

Example task

- Predicting the price of a house (y) from the size in square feet (x_1).



Notice

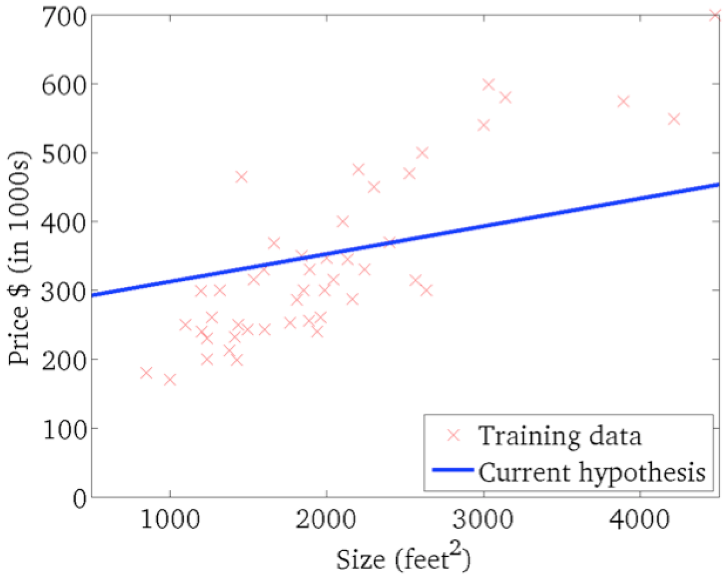
- A regression model $y = \theta_0 + \theta_1 \cdot x_1$ is learned for one single feature x_1.
- Each pair θ_0, θ_1 defines one *hypothesis*, i.e., one candidate model.

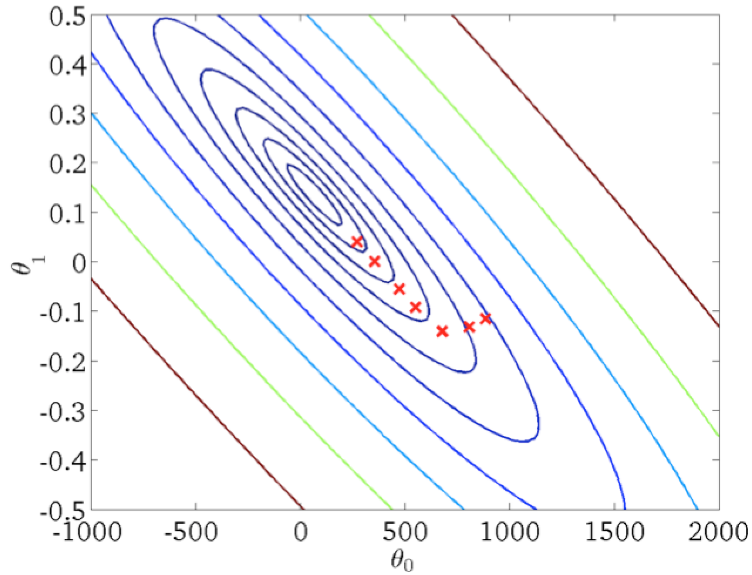
Supervised Learning

Simple Linear Regression (7 out of 9)

Example task

- Predicting the price of a house (y) from the size in square feet (x_1).





Notice

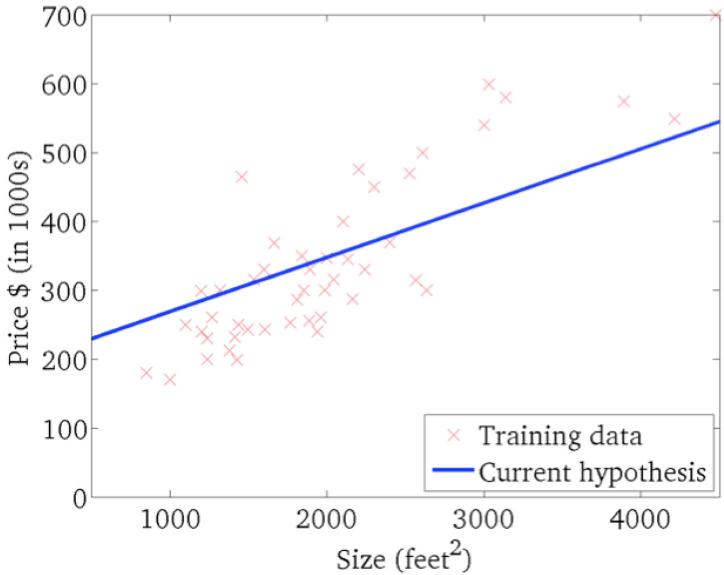
- A regression model $y = \theta_0 + \theta_1 \cdot x_1$ is learned for one single feature x_1.
- Each pair θ_0, θ_1 defines one *hypothesis*, i.e., one candidate model.

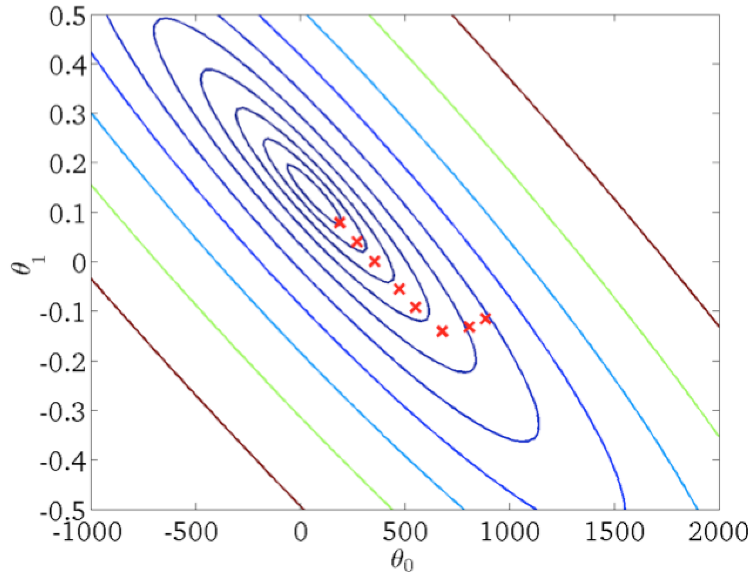
Supervised Learning

Simple Linear Regression (8 out of 9)

Example task

- Predicting the price of a house (y) from the size in square feet (x_1).





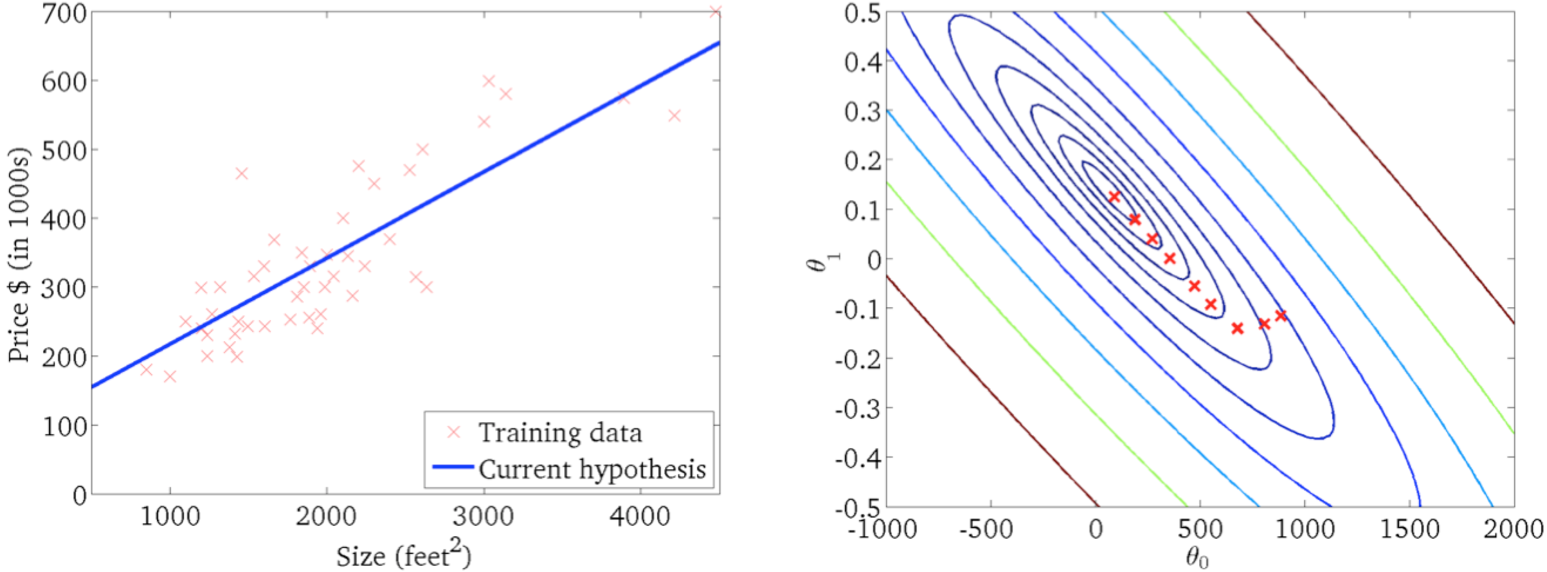
Notice

- A regression model $y = \theta_0 + \theta_1 \cdot x_1$ is learned for one single feature x_1.
- Each pair θ_0, θ_1 defines one *hypothesis*, i.e., one candidate model.

Supervised Learning

Example task

- Predicting the price of a house (y) from the size in square feet (x_1).



Notice

- A regression model $y = \theta_0 + \theta_1 \cdot x_1$ is learned for one single feature x_1.
- Each pair θ_0, θ_1 defines one *hypothesis*, i.e., 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.” $\rightarrow$ <NP> is based in <NP>

Self-supervised learning

- Idea. Generate training data automatically. Works if output information is just accessable, such as time measurements or trivial annotations.

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.
- Main techniques. Clustering, expectation maximization, factor analysis.

  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 vs. soft clustering

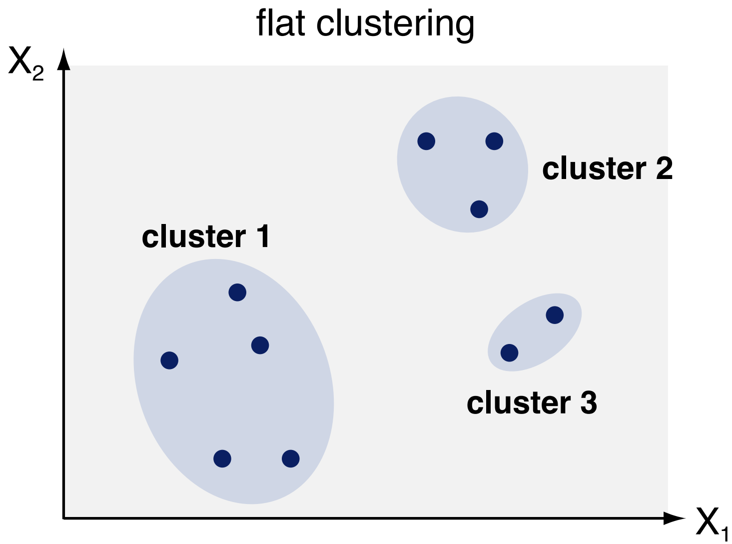
- Hard. Each instance belongs to a single cluster.
- Soft. Instances belong to each cluster with a certain weight.

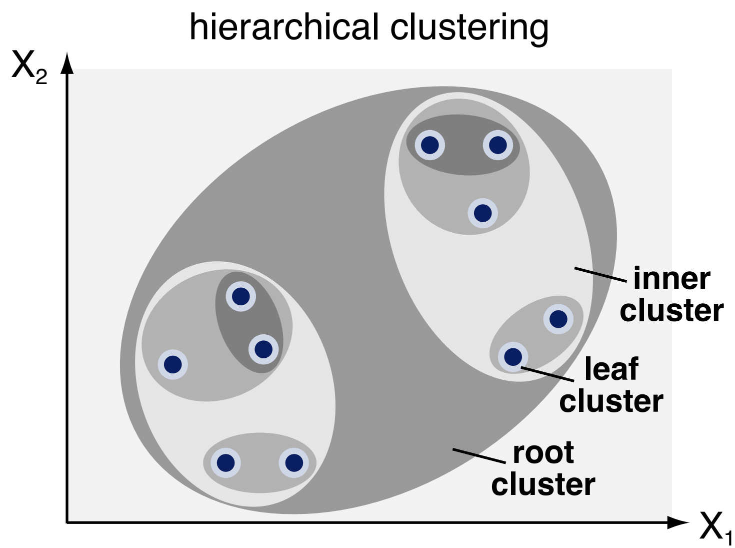
Applications in text mining

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

Unsupervised Learning

Flat vs. Hierarchical Clustering





Types of clustering

- Flat. Group a set of instances into a (possibly predefined) number of clusters, without specifying associations between the clusters.
- 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 groups of instances (*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.
- Clusters can then be evaluated in terms of their *purity*, i.e., the fraction of instances whose class equals the majority class.

Clustering algorithms

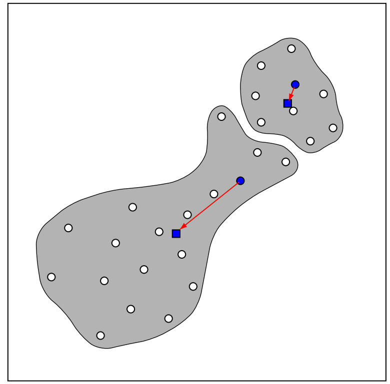
- Iterative clustering, such as k-*means*.
- Density-based clustering, such as *DBSCAN*.
- Hierarchical clustering, either *agglomerative* or *divise*.

  ... among others.

Iterative flat clustering with k-means

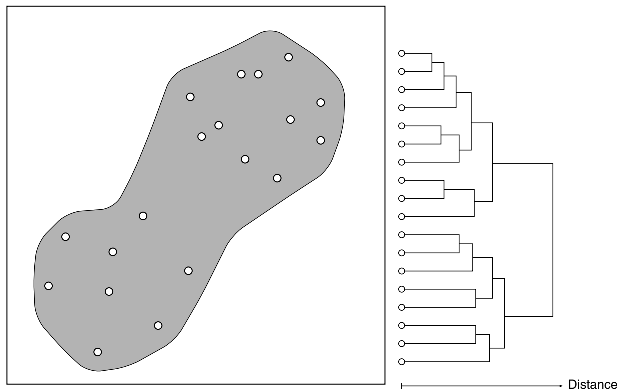
- Partition a set of instances into k clusters.
- Iteratively compute centroids of candidate clusters and re-cluster based on centroids.

  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

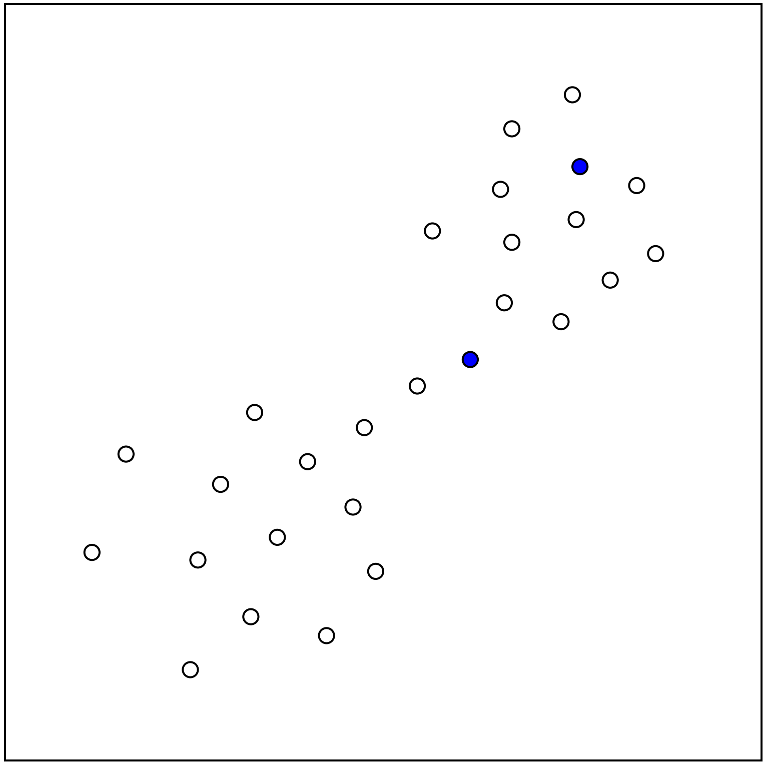
- Incrementally create tree bottom-up, beginning with the single instances.
- Merge clusters based on distances of instances to clusters.
- A flat clustering can be derived from a hierarchical one via cuts in the tree.



Unsupervised Learning

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

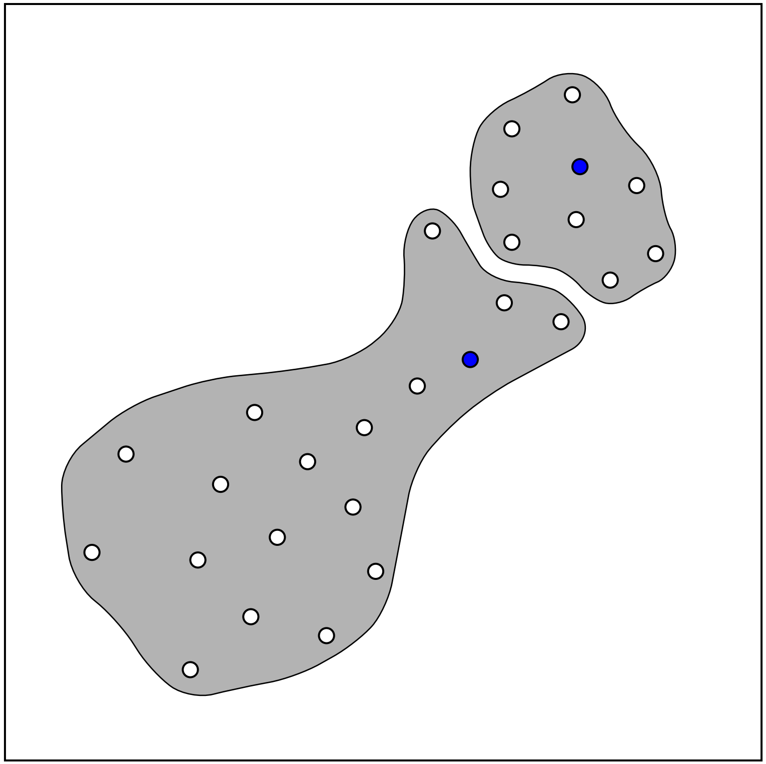
Step 1: Choose k instances randomly



Unsupervised Learning

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

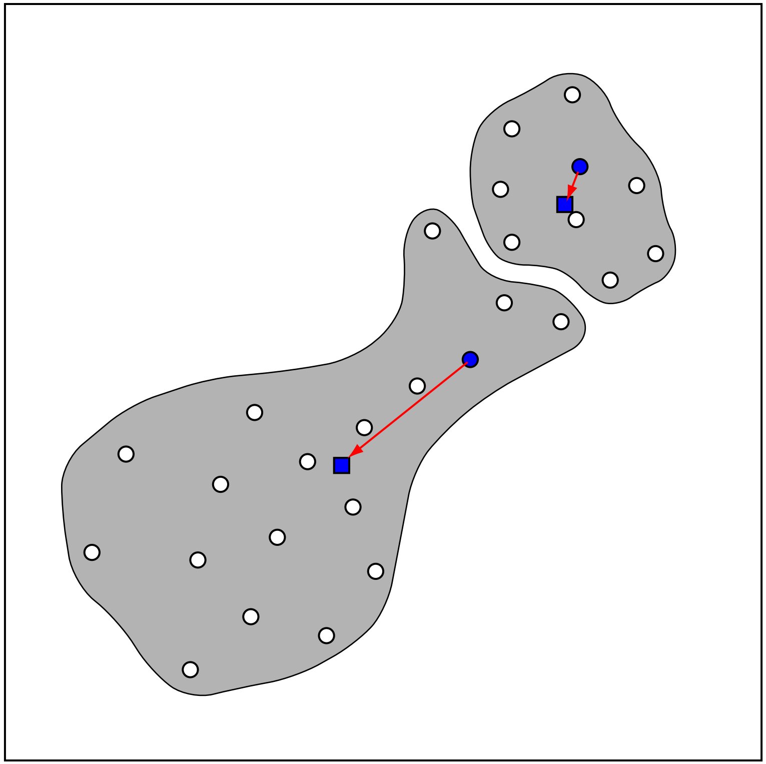
Step 2: Cluster by distance to the k instances



Unsupervised Learning

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

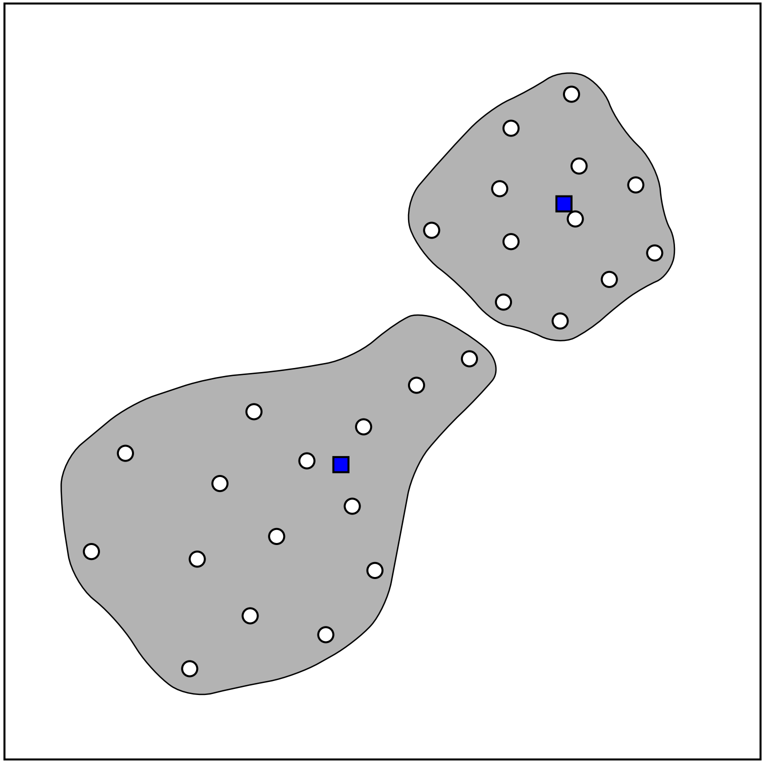
Step 3: Compute centroids of the k clusters



Unsupervised Learning

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

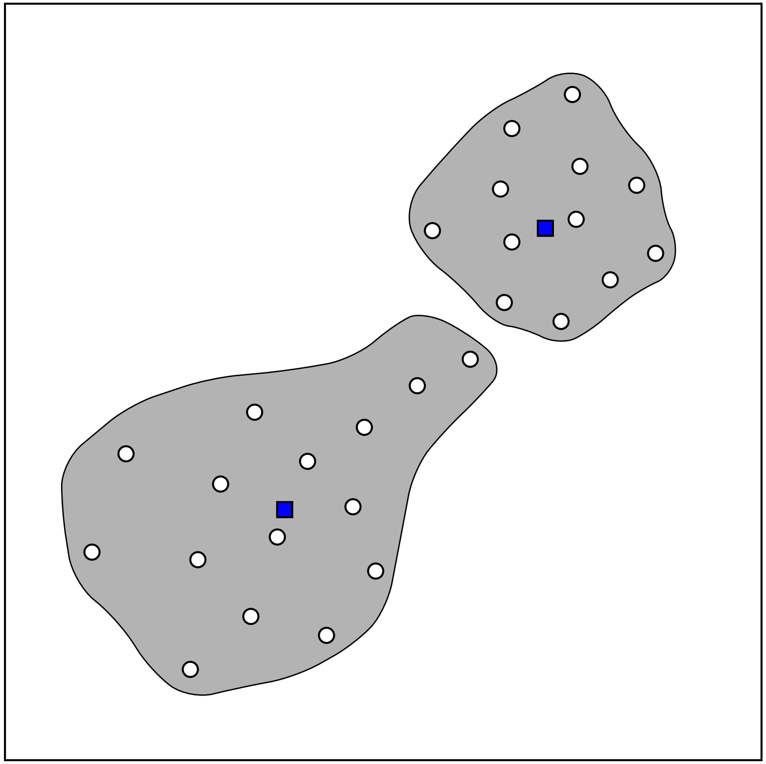
Step 4: Cluster by distance to the k centroids



Unsupervised Learning

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

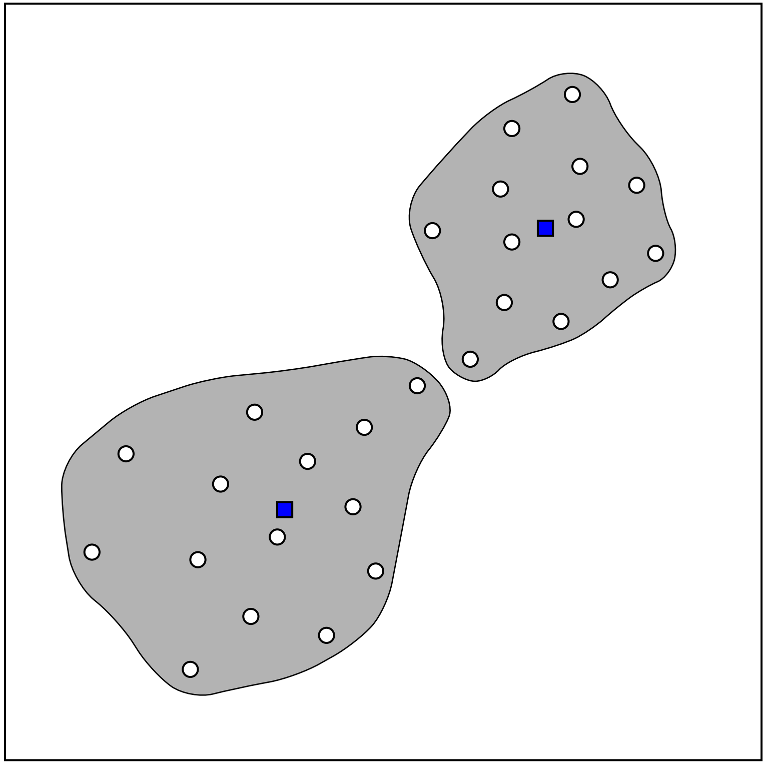
Repeat steps 3–4 until convergence (step 3 again)



Unsupervised Learning

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

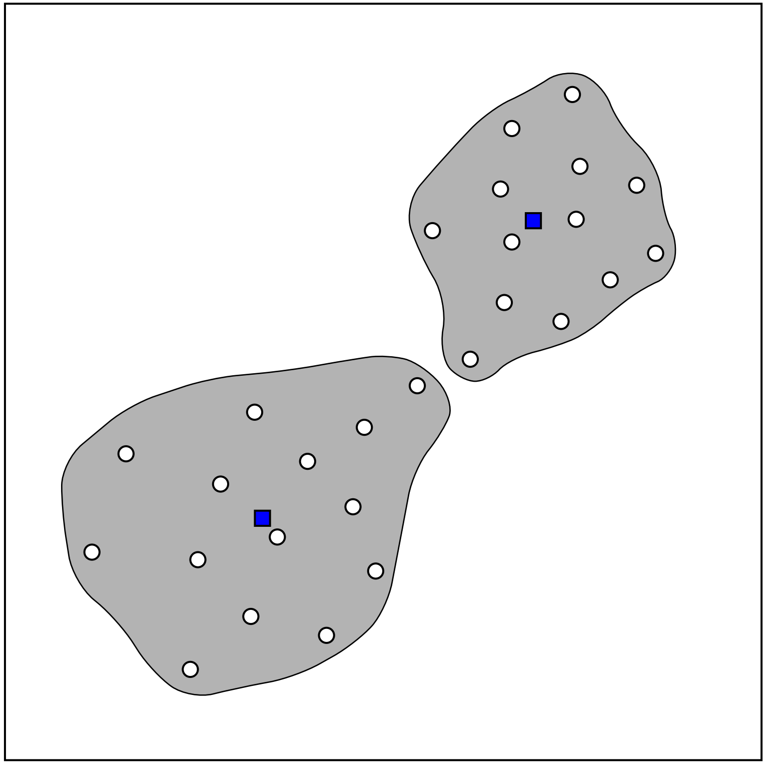
Repeat steps 3–4 until convergence (step 4 again)



Unsupervised Learning

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

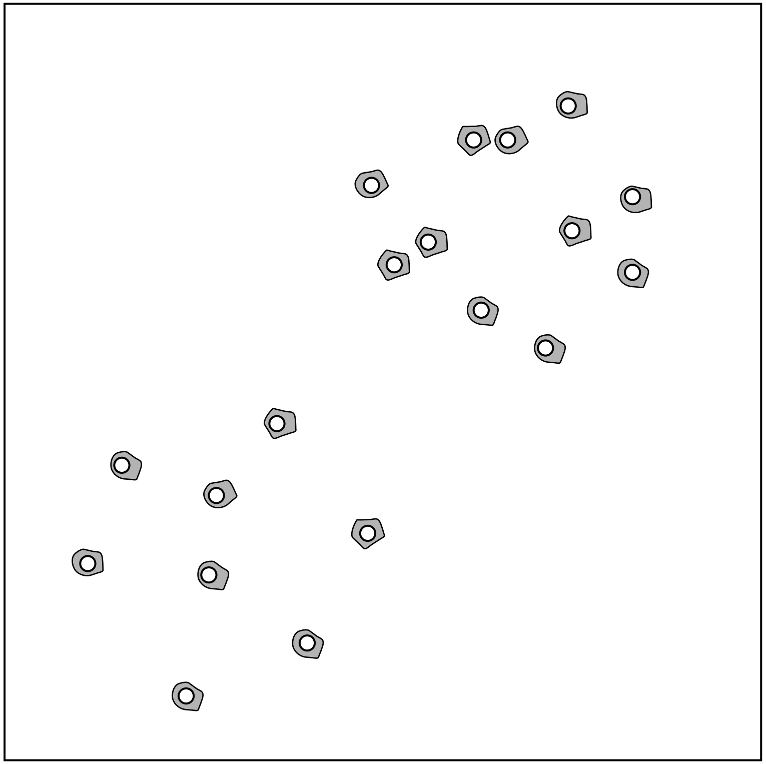
Convergence!



Unsupervised Learning

Example: Agglomerative Hierarchical Clustering

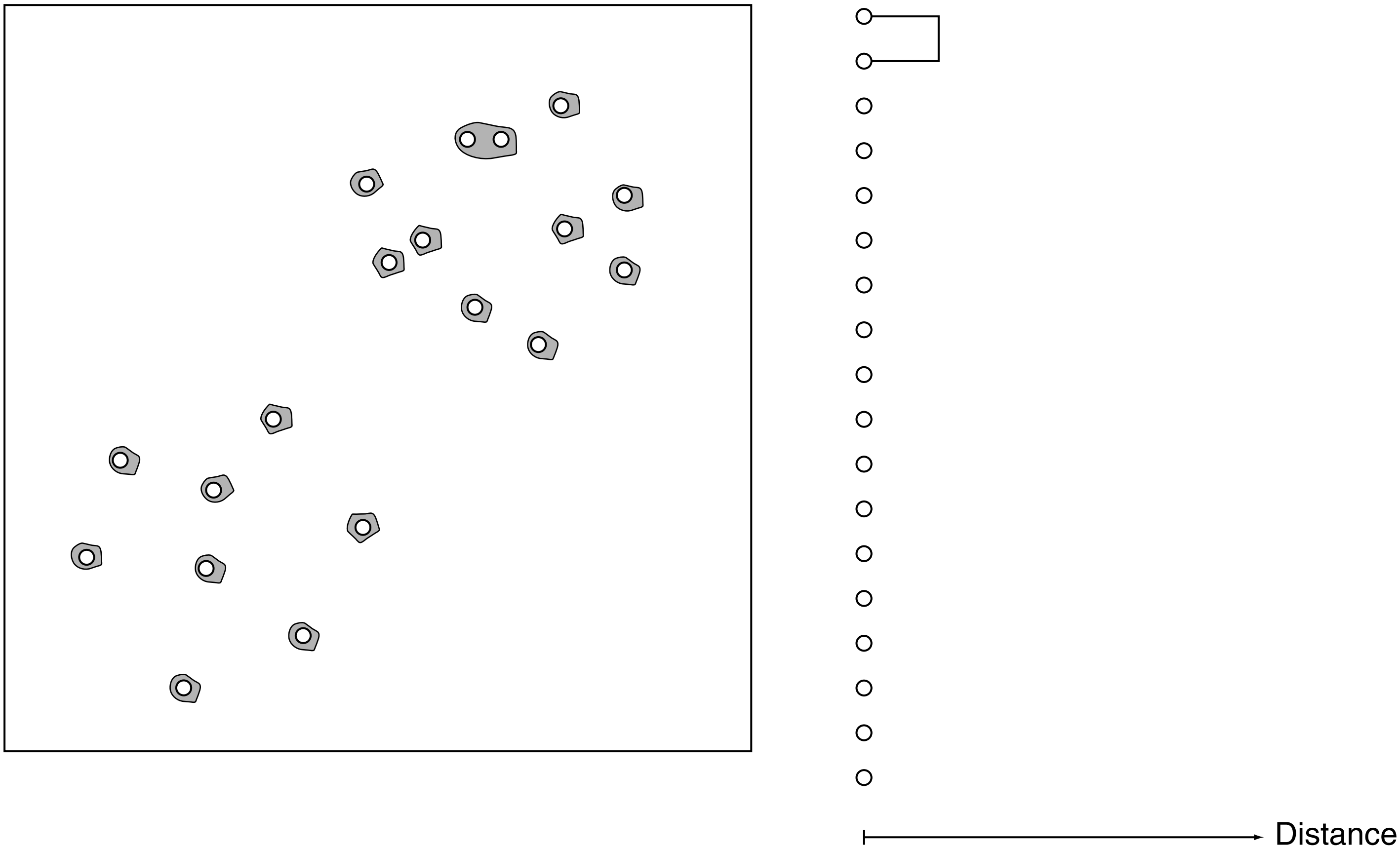
Step 1: Assign each instance to individual cluster



Unsupervised Learning

Example: Agglomerative Hierarchical Clustering

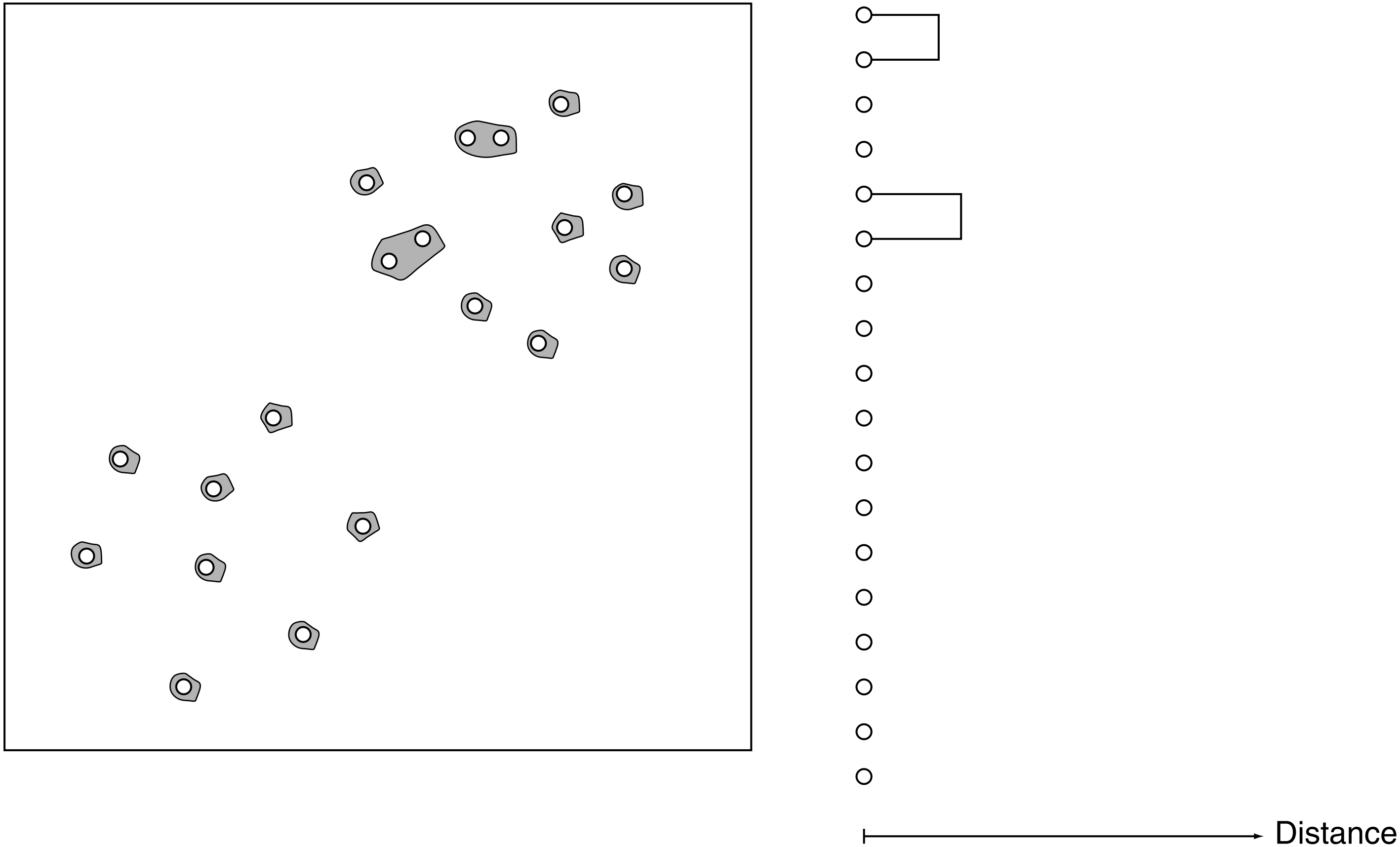
Step 2: Combine closest pair of clusters into one cluster



Unsupervised Learning

Example: Agglomerative Hierarchical Clustering

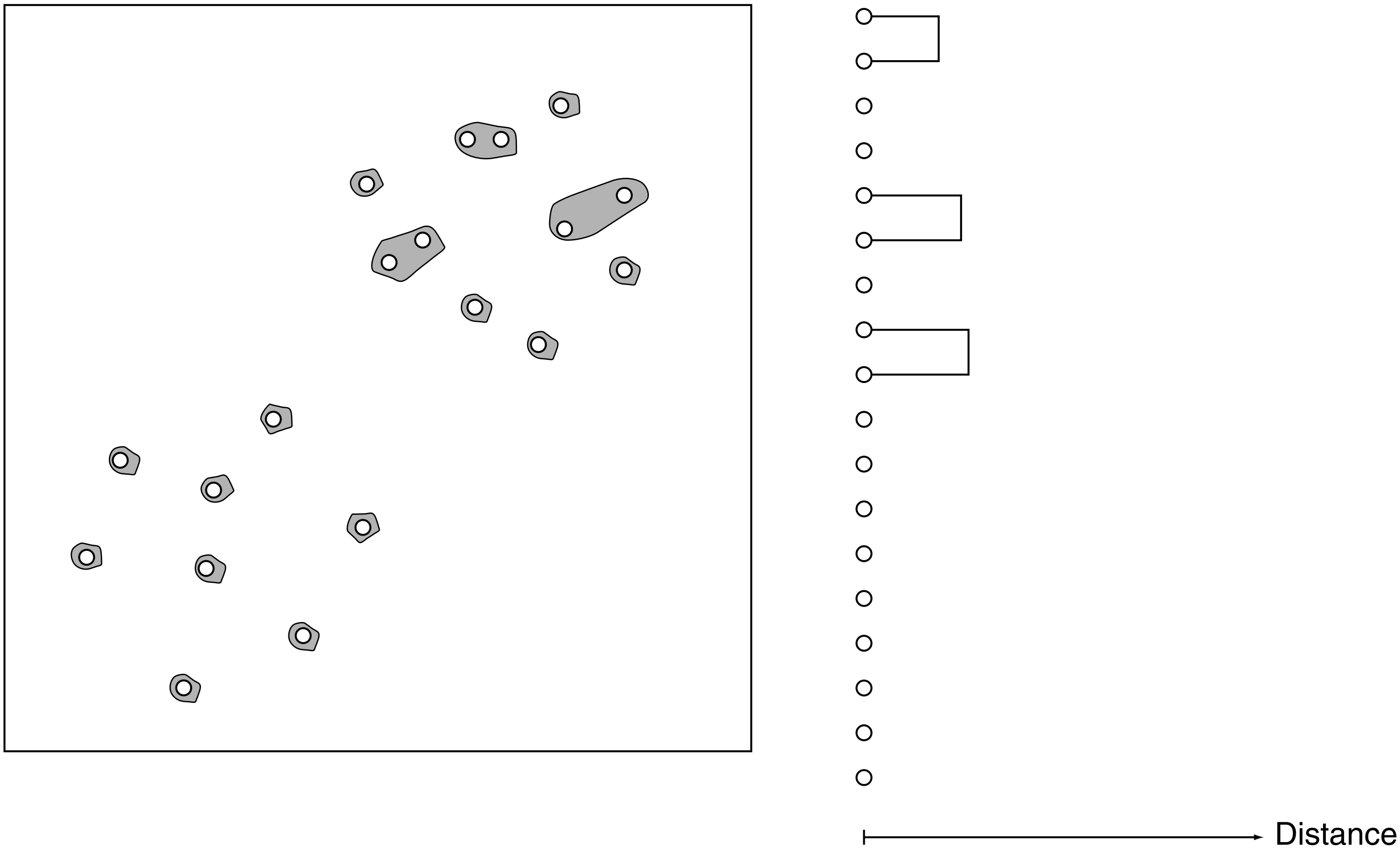
Repeat step 2 until only one cluster remains



Unsupervised Learning

Example: Agglomerative Hierarchical Clustering

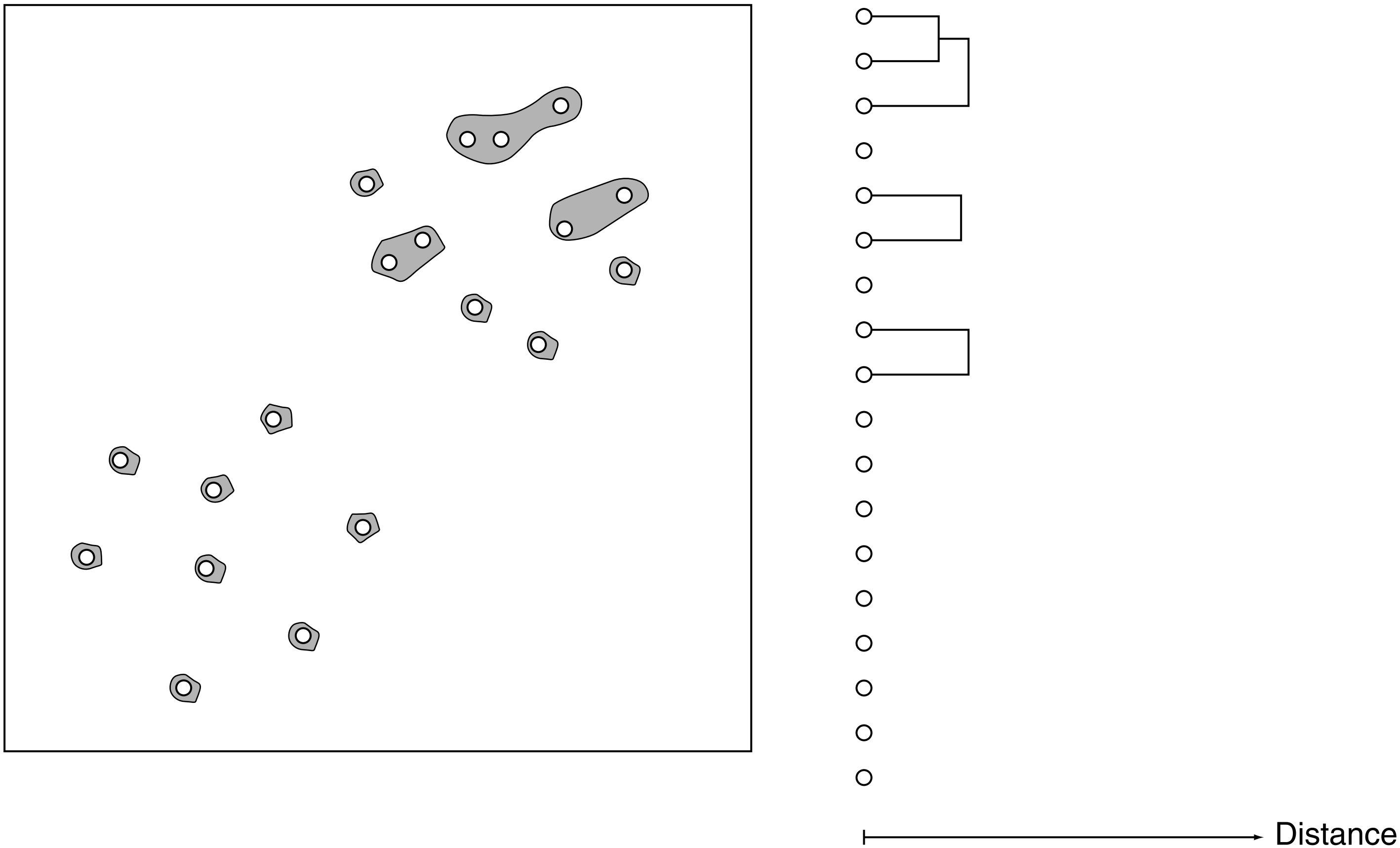
Repeat step 2 until only one cluster remains



Unsupervised Learning

Example: Agglomerative Hierarchical Clustering

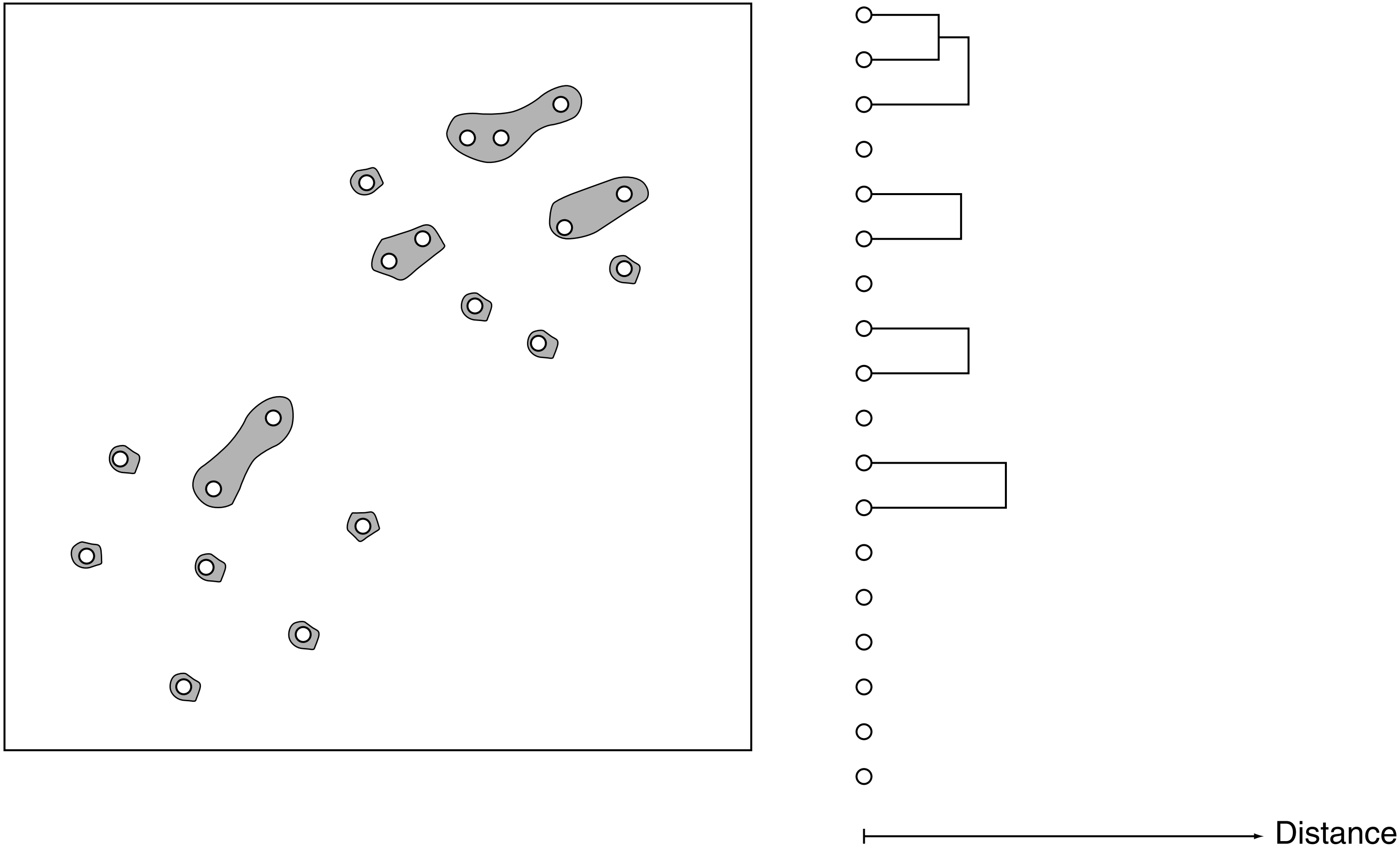
Repeat step 2 until only one cluster remains



Unsupervised Learning

Example: Agglomerative Hierarchical Clustering

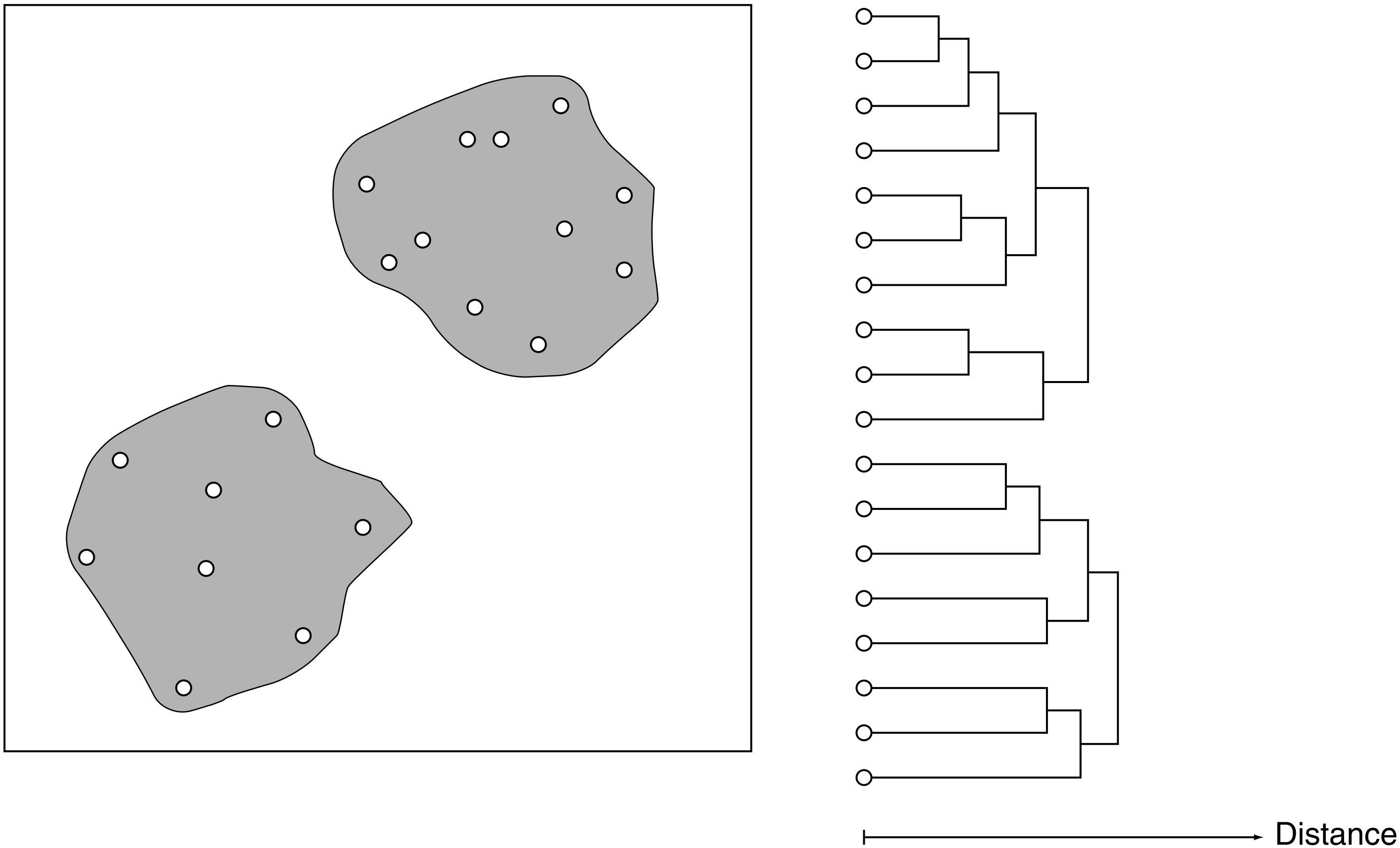
Repeat step 2 until only one cluster remains



Unsupervised Learning

Example: Agglomerative Hierarchical Clustering

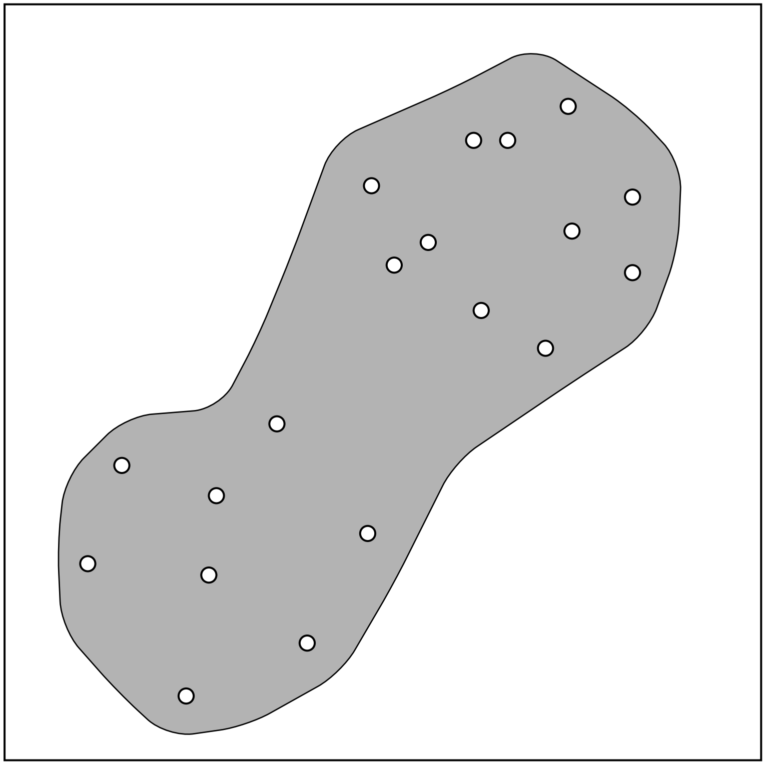
Repeat step 2 until only one cluster remains

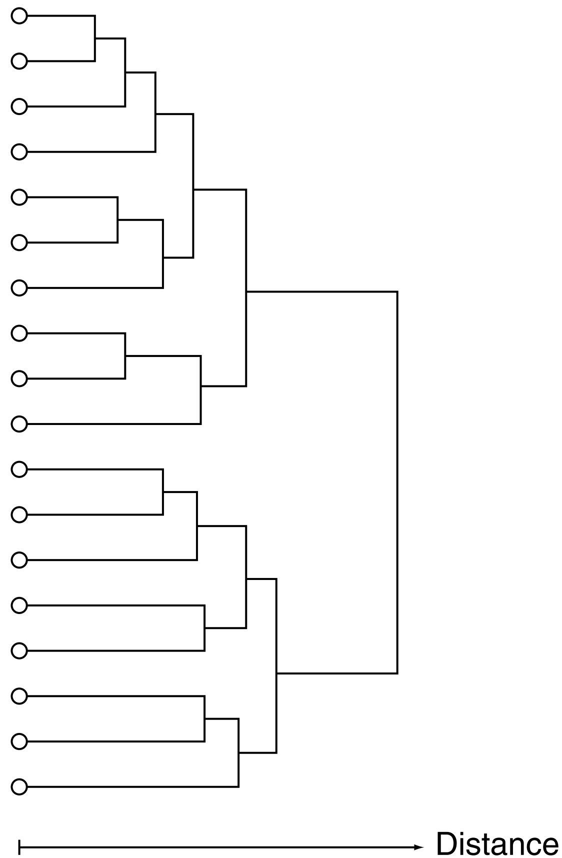


Unsupervised Learning

Example: Agglomerative Hierarchical Clustering

The dendrogram on the right shows the final hierarchical clustering!



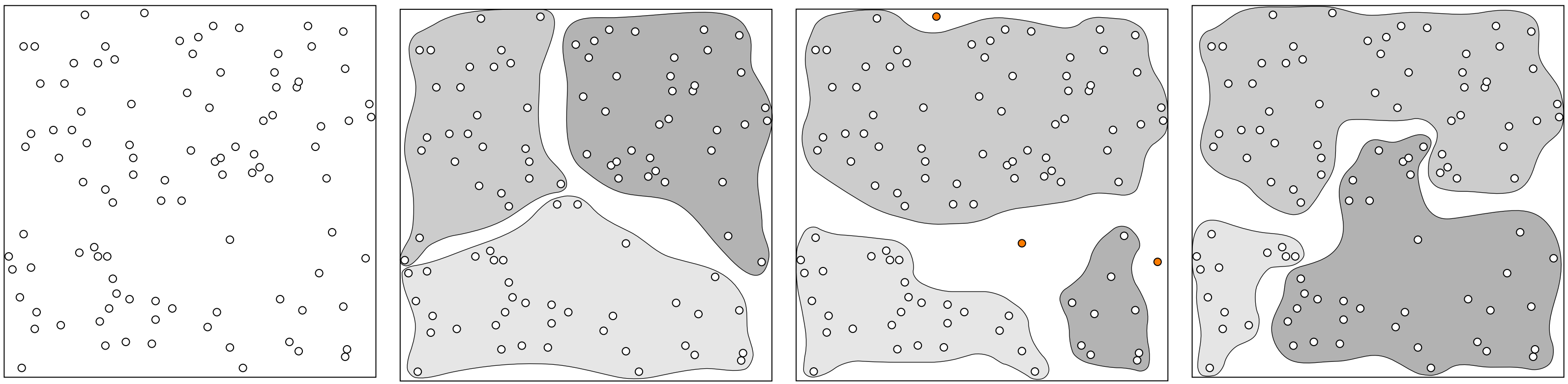


Unsupervised Learning

Evaluation of Clustering

The clustering evaluation problem

> *"The validation of clustering structures is the most difficult and frustrating part of cluster analysis. Without a strong effort in this direction, cluster analysis will remain a black art accessible only to those true believers who have experience and great courage."*
>
> (Jain and Dubes, 1990)



Random points k-means   DBSCAN   Complete link

Unsupervised Learning

Evaluation of Clustering: Goals and Measures

Possible goals of clustering evaluation

- Rank alternative clusterings with regard to their quality.
- Determine the ideal number of clusters.
- Relate found structures to externally provided class information.
- Provide information to choose a suited clustering approach.
- Provide evidence whether data contains non-random structures.

Evaluation measures

- External. Analyze how close is a clustering to an (external) reference, i.e., to a test set with ground-truth information.
- Internal. Analyze intrinsic characteristics of a clustering, such as the average cluster distance.
- Relative. Analyze the sensitivity (of internal measures) during clustering generation.

Learning Types and Algorithms

Example Supervised and Unsupervised Learning Problems

How to treat 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).

Treat both as classification problems.	$\rightarrow$ Not reasonable
Treat #1 as a classification problem, #2 as a regression problem.	$\rightarrow$ Not reasonable
Treat #1 as a regression problem, #2 as a classification problem.	$\rightarrow$ Correct
Treat both as regression problems.	$\rightarrow$ Possible

Which should be treated with unsupervised learning?

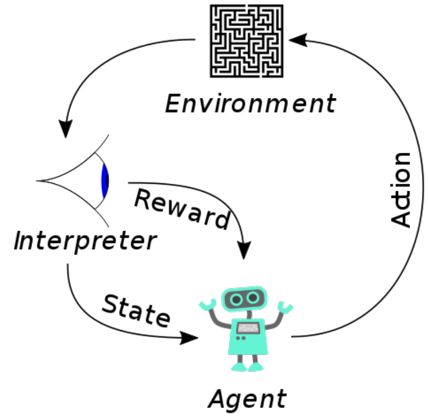
Given mails labeled as "spam" or "not spam", learn a spam filter.	$\rightarrow$ No
Given a set of news articles, group them into sets about the same story.	$\rightarrow$ Yes
Given a database of customer data, automatically discover market segments, and assign new customers to market segments.	$\rightarrow$ Yes
Given a dataset of patients diagnosed as either having diabetes or not, classify diabates for new patients.	$\rightarrow$ No

Learning Types and Algorithms

Select Other Learning Types

Reinforcement learning

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



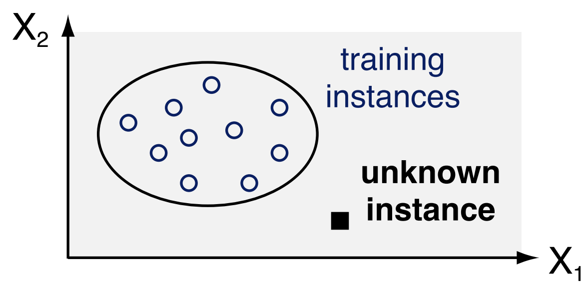
Recommender systems

- 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, v0.20, 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.

Task → Corpus acquisition → Text analysis → Feature engineering → Machine learning → **Approach**

- The process is iterative in general, i.e., steps back are often done.

Application of Machine Learning

Development Process, Steps 1 and 2

Task | Corpus acquisition | Text analysis | Feature engineering | Machine learning | Approach

Corpus acquisition

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

  Usually requires proprietary code.

Text analysis

- Preprocess all corpus texts with existing text analysis algorithms, in order to obtain more information that can be used in features.
- 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

Task | Corpus acquisition | Text analysis | **Feature engineering** | **Machine learning** | **Approach**

Feature engineering

- Identify potentially helpful feature types on training set, implement them.
- Determine concrete features of types on instances from training set.
- Compute feature vectors for each instance from all datasets.

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

Machine learning

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

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

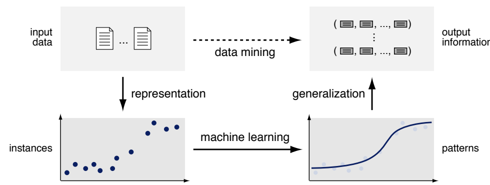
Conclusion

Summary

Machine learning

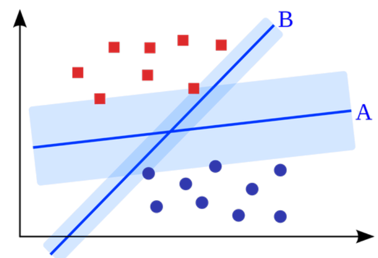
- Aims to learn target functions for prediction problems.
- Infers models from statistical patterns in data.
- Models can be used to approach prediction problems.

Machine learning within data mining



- Instances are usually represented by features.
- Machine learning optimizes feature weightings.
- The learned model is 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.