Ludwig-Maximilians-Universität München
Institut für Informatik
Lehr- und Forschungseinheit für Datenbanksysteme

Knowledge Discovery in Databases II

Winter Term 2014/2015

Chapter 4: Data Variety

Lectures : PD Dr Matthias Schubert
Tutorials: Markus Mauder, Sebastian Hollizeck

http://www.dbs.ifi.lmu.de/cms/Knowledge_Discovery_in_Databases_II_(KDD_II)

Data Variety

So far: Data is given by a set of vectors (data matrix).

In general: Data is structured and has links.

(compare with general data models: UML, Entity-Relationship..)

Core questions in this chapter:

- Is the structure important for the meaning of data?
- How can we apply data mining algorithms to structured data?
- Can we combine different views on the same data object to get better results?
- Does structured data yield additional data mining tasks?

Types of structures data objects

- Multi-View Data: Data is given by multiple object descriptions. (records, objects in programming)
- Multi-Instance Data: Each object is described as a set of objects from the same domain (arrays, lists, sets in programming)
- Linked Data or Graph data: Objects may reference to other objects. (graph structured data, network data, ..)

Further cases not being discussed in the lecture:

- Sequential Data: multi-instance data having a strict order
- Temporal Data: Describes the same object over a time-period (time series data)
- Tree-Structured Data: acyclic graph data, described hierarchies

Ways to handle data Variety

There are generally three ways to handle data variety:

1. Transform data into a simpler format
   1. Useful information about the structure is preserved in special features
   2. Transformation from more complex to simpler descriptions (e.g. Multi-Instance to vector, Graph to Multi-Instance, ..)
2. Employ distance and similarity measures for structured data
   1. Having a distance/similarity function allows for the use of many data mining algorithms.
   2. Comparing complex descriptions is usually more complex.
3. Employ specialized data mining algorithms
   1. Especially in cases where the task is not standard (e.g. centrality in graphs)
   2. Often solutions involve a workflow of several data mining tasks

Chapter Overview

- Ensemble learning and Multi-View Object Descriptions
- Multi-Instance Data Mining
- Mining Graph-Structured objects
- Graph and Link Mining

Multi-View Data Mining and Ensembles

In general: Having more than one view on the same set of objects can be exploited to learn better results.

⇒ Ensemble theory: Learn better models by combining multiple basic learners.

⇒ Multi-View data mining: In most cases, specialized applications of Ensemble learning

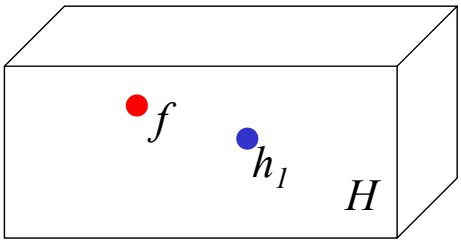
⇒ Well established in application domains such as bio informatics and multi-media retrieval.

1. Introduction and Basic Principles of Ensemble Learning
2. Diversity in Ensemble Learning
3. Methods for Ensemble Construction
4. Mining Multi-View Data

- Given: A *data set containing elements X from data space D. Each element x belong to class* $c_i \in C$ *(set of all classes)*.
- There is a function $f: D \rightarrow C$, describing the connection of x and class c_i.
- The task of classification is to determine f.
- In general, learning algorithms compute an approximation of f which does not hold completely. They learn a hypotheses.

Introduction to Ensemble Learning

- The "true" function f is unknown.
- There is a sample set of tupels $(x, c_i) \in f \subseteq D \times C$ (training set)
- A learning algorithms now determines the hypothese h_i as classifier from the hypotheses space $H \subseteq D \times C$ which fits best to the training set.



- Caution: the "true" f does not need to be contained in H.

Introduction to Ensemble Learning

- A classifier (a learned hypotheses h) can be applied to all $x \in D$ to predict the $c_i = f(x)$
- The accuracy is the relative frequency of correct predictions.

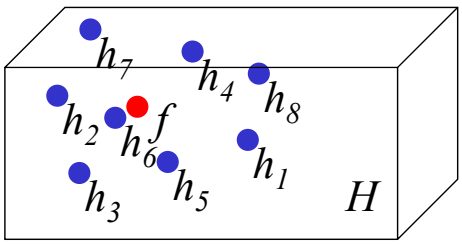
$$Acc(h) = P(h(x)=f(x))$$

- Correspondingly the classification error is the complement:

$$Err(h) = P(h(x)\neq f(x)) = 1 - Acc(h)$$

Introduction to Ensemble Learning

- Idea: Asking multiple „experts“ (classifiers) can avoid mistakes.
- From a mathematical point of view: building the average over multiple functions can smooth the decision surface.



Introduction to Ensemble Learning

- A simple decision rule for two classes $C=\{-1,1\}$:
  - Generate a set of hypotheses $\{h_1,\ldots,h_k\}$ and corresponding weights$\{w_1,\ldots,w_k\}$.
  - An Ensemble-Classifier $\hat{h}$ is defined by the following decision function:

$$\hat{h}(x) = \begin{cases} w_1 h_1(x) + \ldots + w_k h_k \geq 0 \rightarrow 1 \\ w_1 h_1(x) + \ldots + w_k h_k < 0 \rightarrow -1 \end{cases}$$

- often $w_1=\ldots=w_k=1$ (unweighted combination).
- Weights can by used to express the reliability of the classifiers
- More complex decision rules might be used, especially when having more than two classes

  $\rightarrow$ there is a large variety of Ensemble-Methods

$$\hat{h}(x) = \begin{cases} w_1 h_1(x) + \ldots + w_k h_k \geq 0 \rightarrow 1 \\ w_1 h_1(x) + \ldots + w_k h_k < 0 \rightarrow -1 \end{cases}$$

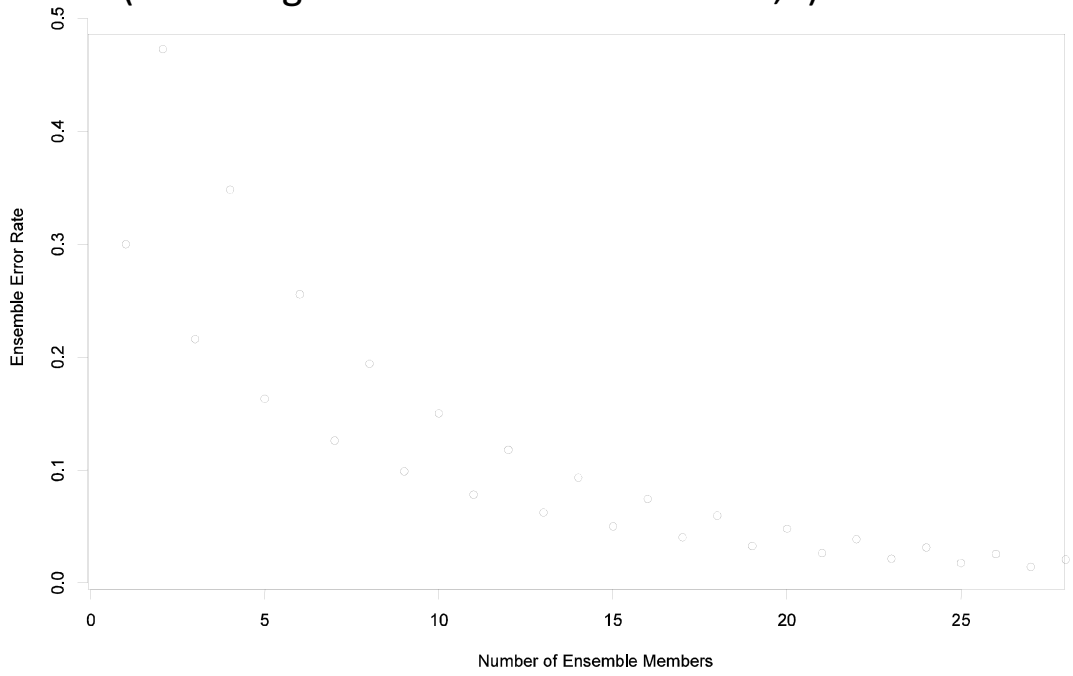
- The error rate of an Ensemble learner depends error-rate of its base classifiers and there amount :

  The Ensemble error w.r.t. to the above is given by the frequency of the cases where at least half of the base classifier are wrong:

$$Err(\hat{h}) = \sum_{i=\lceil \frac{k}{2} \rceil}^{k} \binom{k}{i} e^i (1-e)^{k-i}$$

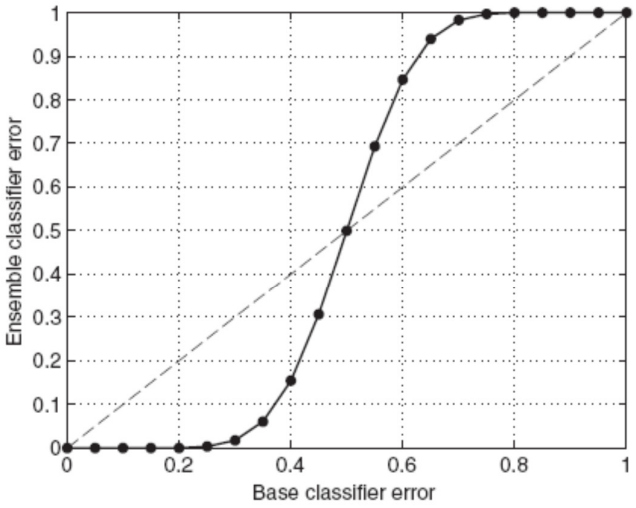
- (Assumption here: $Err(h_1)=\ldots=Err(h_k)=e$)

Dependency of the complete error rate on the number of basis learners (assuming a constant error rate of 0,3):



Introduction to Ensemble Learning

- Error rate of an Ensemble having 25 basis learners for varying error rates of the base classifiers :

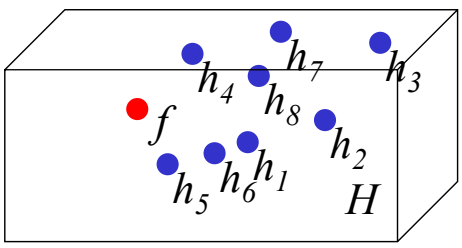


(from: Tan, Steinbach, Kumar: Introduction to Data Mining)

Introduction to Ensemble Learning

- Requirement for an improvement: The errors of each classifier are independent.

$$Err(\hat{h}) = \sum_{i=\left\lceil \frac{k}{2} \right\rceil}^{k} \binom{k}{i} e^{i} (1-e)^{k-i}$$



- if the base classifiers are too similar, they are making the same mistakes => no improvement

- Conclusion:

> Required conditions for an improved accuracy:
> 1. Base classifiers are sufficiently "accurate".
> 2. Base classifiers are "diverse".

- Accuracy: better than random predictions
- Diversity: no or at least no strong correlations between the predictions
- Is it possible to optimize both simultaneaously?

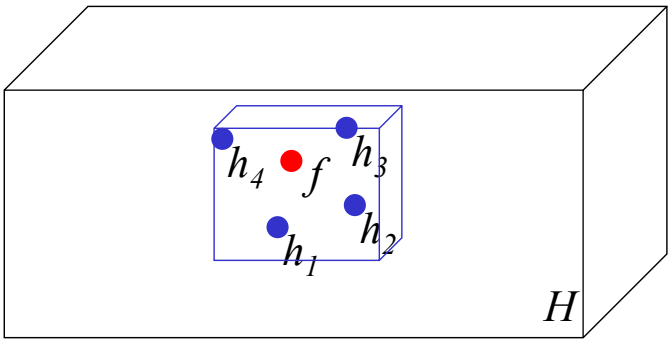
1. Introduction and Basic Principles of Ensemble Learning
2. Diversity in Ensemble Learning
3. Methods for Ensemble Construction
4. Mining Multi-View Data

Aspects of Diversity

- There are several reasons for diverse classifiers:
  - Statistical Variance
  - Computational Variance
  - Representation

Aspects of Diversity

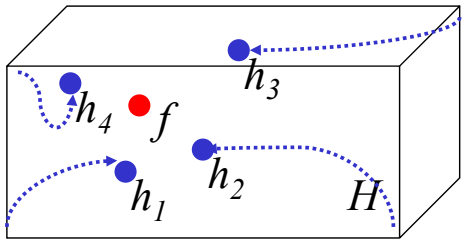
- Statistical Variance:
  - The of potential hypotheses is to big too determine the best one based on a limited sample set.



  - Combining multiple hypotheses reduces the risk to make a large mistake

Aspects of Diversity

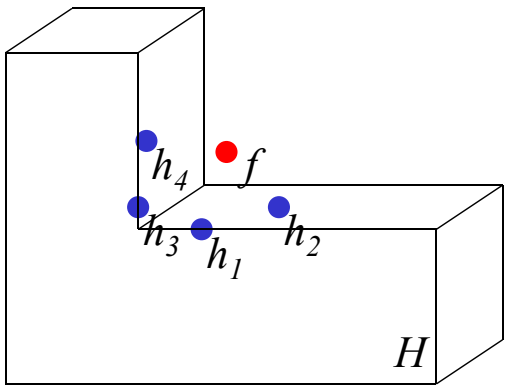
- Computation Variance:
  - Some learning algorithm cannot guarantee, to find the hypotheses fitting best to the training data due to the complexity of the learning algorithm
  - For example, it is common to the use heuristics computing local minima in case computation is too expensive.



  - Combination multiple hypotheses reduces the risk to take the wrong local minimum of an error function.

Aspects of Diversity

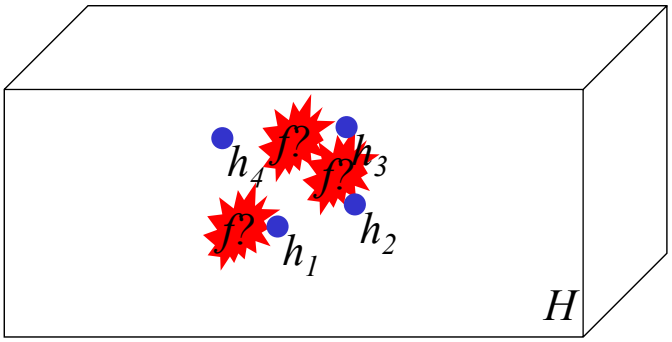
- Representation :
  - The space of representable hypotheses might not contain a good approximation of the “true” function f .



  - Combining multiple hypotheses can extend the space of representable hypotheses.

Aspects of Diversity

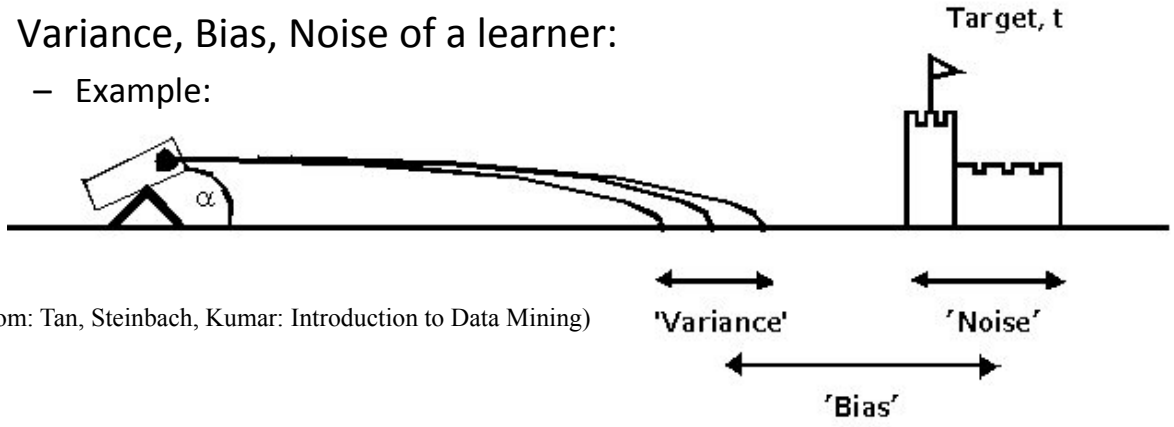
- Fuzzy target functions:
  - The training samples do not allow distinct conclusion about the target function (e.g. training samples might be contradictive).



  - Combining multiple hypotheses reduces the risk to approximate a wrong hypotheses

Aspects of Diversity

- Variance, Bias, Noise of a learner:
  - Example:



(from: Tan, Steinbach, Kumar: Introduction to Data Mining)

  - Variance, Bias and Noiset represent different types of error
    $err = Bias_{\alpha} + Variance_{f} + Noise_{t}$
  - Variance: depends on the employ power *f*
  - Noise: lack of definition of the target
  - Bias: depends on the end bracket (angle of the canon)

Aspects of Diversity

- Variance, Bias, Noise of a classifier:
  - Variance:

    Depends on the variation in the training data or the parameters of of the classifier
  - Noise:

    class for some of the training objects is not clearly defined or ambiguous
  - Bias:

    Represents the assumptions on the data of the used classifier
    (e.g. linear separable, independent attributes, distributions,..).

    "Bias-free learning is futile."

    A core part of learning is to abstract the observation within a model

    => learners are based on this model

Aspects of Diversity

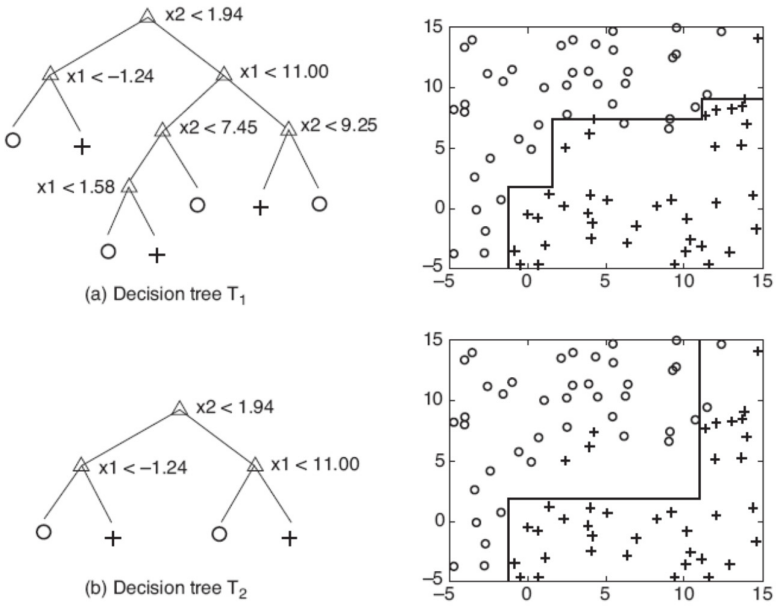


Figure 5.33. Two decision trees with different complexities induced from the same training data.

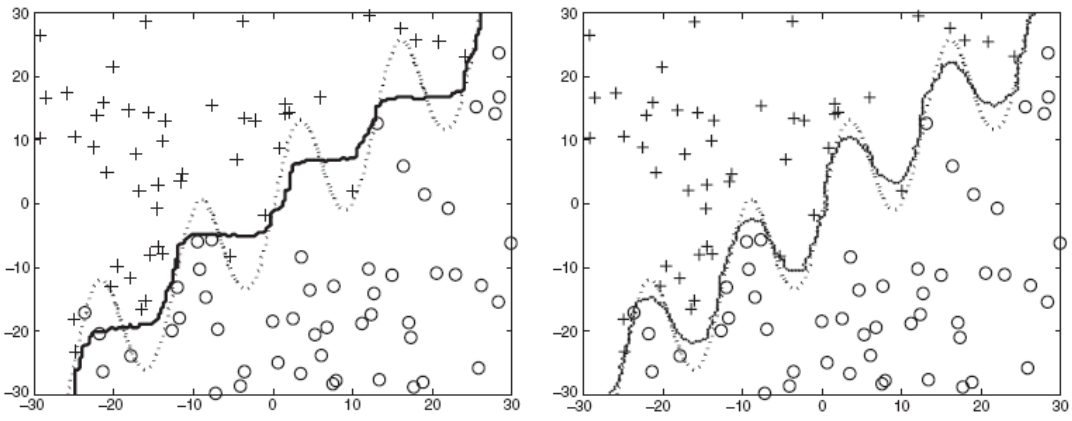
(from: Tan, Steinbach, Kumar: Introduction to Data Mining)

Aspects of Diversity

- Decision Trees as an example for Bias:
  - T_1 and T_2 were trained on the same data
  - T_2 was generated from T_1 by pruning it to the maximal height of two
  - T_2 uses stricter assumptions about class separation (stronge bias)

Aspects of Diversity

- the relative contribution of bias and variance to the complete error depends on the classifier.



(a) Decision boundary for decision tree.

(b) Decision boundary for 1-nearest neighbor.

Figure 5.34. Bias of decision tree and 1-nearest neighbor classifiers.

(from: Tan, Steinbach, Kumar: Introduction to Data Mining)

Aspects of Diversity

- Example:
  - average class borders over 100 classifiers being trained on 100 different training sets containing 1000 objects
  - dashed : true class border being used by the data generator
  - 1-NN classifier has overall smaller distance to the class border
    ➔ less bias
  - the 100 1-NN classifiers display a larger variability
    ➔ larger variance

Ensembles and Multi-View Data

1. Introduction and Basic Principles of Ensemble Learning
2. Diversity in Ensemble Learning
3. Methods for Ensemble Construction
4. Mining Multi-View Data

Methods for Ensemble Construction

- How can we a achieve diversity of base learners?
  - Vary the training sets
    - Methods: Bagging and Boosting
  - Manipulate the Input Features
    - Learn on varying subspaces
    - Use multi-view data
  - Manipulate the class labels
    - various methods for mapping general to two class problems
  - Manipulate the training algorithms
    - introduce randomness
    - employ varying initial models

Varying the training set

- an important property of a learning algorithm is stability
- The more stable a learner is, the less different are the classifiers on different training sets for the same classification task.
- Unstable learners may strongly change even under small modificatons of the training sample.

$\Rightarrow$ Instable learners are more suitable for ensemble learner where diversity is generated by varying the training set

$\Rightarrow$ Examples for instable learners:

- Decision Trees
- Neuronal Network
- rule-based learners

Varying the training set

- Bootstraping:

  sample a training set by drawing n samples with replacement.
  - training set has the same size (n objects) as the original data set
  - the training set contains in average 63% of the objects in the data set (some of them more than one time and ca. 37% are not contained):
    - a particular sample is drawn with the likelihood *1/n* and is not drawn with the likelihood *1-1/n*
    - after making n draws each data object is not contained once with likelihood of $\left(1-\frac{1}{n}\right)^n$
    - for large *n*'s $\left(1-\frac{1}{n}\right)^n \approx e^{-1} \approx 0.368$
  - therefore, the method is also called "0.632 bootstrap"

    (The methods is sometimes used like cross validation for train and test.)

Varying the training set

- Bagging (**B**ootstrap **Agg**regat**ing**):

  build varying training sets by multiple bootstraps
- Bagging aggregates these bootstraps and trains a classifiers on each of them
- Using unstable methods results in multiple varying classifiers
- the final classifier is combined by a simple majority vote

Original Data	1	2	3	4	5	6	7	8	9	10
Bagging (Round 1)	7	8	10	8	2	5	10	10	5	9
Bagging (Round 2)	1	4	9	1	2	3	2	7	3	2
Bagging (Round 3)	1	8	5	10	5	5	9	6	3	7

Varying the training set

- While the 0.632 bootstrap draws from a uniform distribution, b**oosting** employs a weighted distribution.
- Data objects which are hard to classify are weighted higher
- Use of the weighting:
  - higher weights increase the likelihood that the object is drawn in the next bootstrap
    ➔ difficult examples appear more often in the next bootstrap are predominantly used

Original Data	1	2	3	4	5	6	7	8	9	10
Boosting (Round 1)	7	3	2	8	7	9	4	10	6	3
Boosting (Round 2)	5	4	9	4	2	5	1	7	4	2
Boosting (Round 3)	4	4	8	10	4	5	4	6	3	4

  - Some learning algorithms can directly employ instance weights
    ➔ increases the bias of the learned hypotheses

Feature Manipulation

- Manipulate the input features:
  - Learn on varying subspaces or combine features
    - example: Random Forests

      set of decision trees where the training set where diversity is created via:

      a) random selection of features for each split

      b) for each node new features are randomly constructed by combining input features

      c) for each node use one of the F best splits
  - Combine classifier being trained on different views (Multi-View Data Mining)

Manipulating the Class Labels

- in many cases a multi-class problem has to be mapped to a set of two-class problems

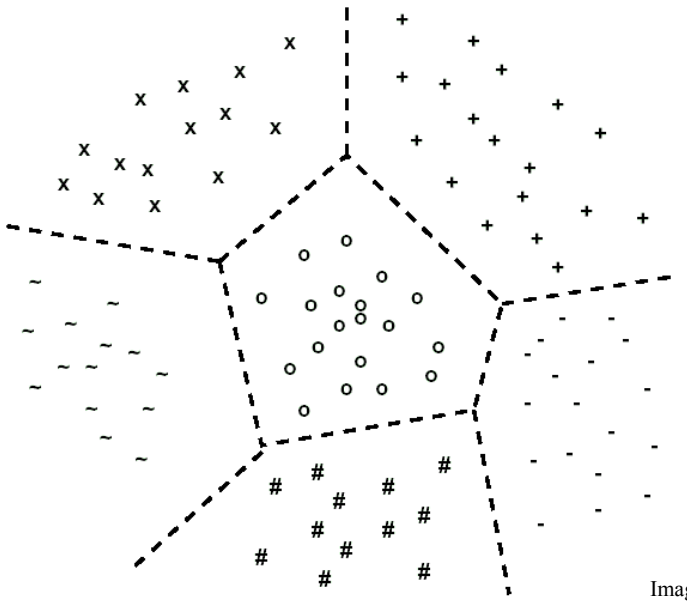


Image from: Fürnkranz 2002

DATABASE SYSTEMS GROUP

Manipulating the Class Labels

LMU

- Train multiple classifiers for two classes an combine the results to a multiclass decision
- this is equivalent to generating diversity by manipulating the class labels
- general methods:
  - one-versus-rest
  - all-pairs
  - error correcting output codes

Manipulating the Class Labels

- *one-versus-rest* (also known as: *one-versus-all*, *one-per-class*):

  For *n* classes, *n* classifiers are trained. Each distinguishes one class from the combination of all other classses

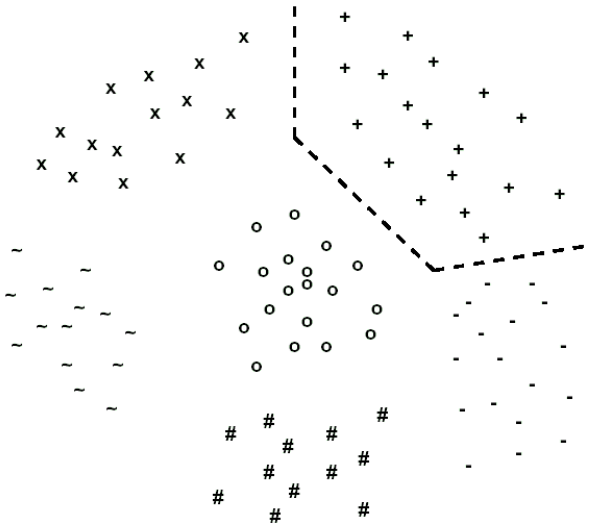


Image from: Fürnkranz 2002

Manipulating the Class Labels

- *all-pairs* (also known as: *all-versus-all*, *one-versus-one*, *round robin*, *pairwise*):

  For each pair of classes a classifier is trained predicting one of both classes.

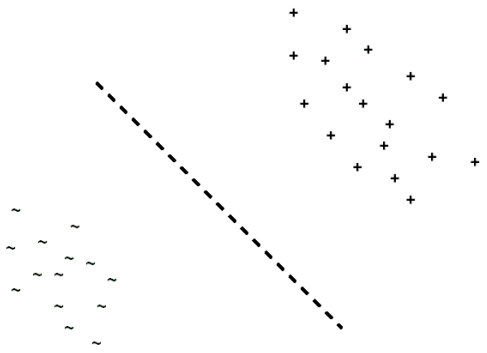


Bild aus: Fürnkranz 2002

Manipulating the Class Labels

- Error Correcting Output Codes (ECOC):
  - Class set *C* is randomly split into two disjunctive *A+B* subsets.
  - Data objects belonging to class set *A* are with *-1*, all other objects belonging to the class set B are labelled with *1*.
  - Train *k* classifiers for the these two class problems
  - If classifier *i* predicts *A*, all classes in A receive a vote
  - The class $c \in C$ receiving the most votes is selected as class prediction of the ensemble

Manipulating the Class Labels

- Example: $C=\{c_1,c_2,c_3,c_4\}$, 7-bit codes

class	code word						
c_1	1	1	1	1	1	1	1
c_2	0	0	0	0	1	1	1
c_3	0	0	1	1	0	0	1
c_4	0	1	0	1	0	1	0

- for each bit in the code word a classifier is trained, 7 classifiers are trained
- Assume the classifiers predict (0,1,1,1,1,1,1) –
  For which class would the ensemble decide?

Manipulating the Class Labels

- the name “Error Correcting Output Codes” reflects the idea that training multiple classifiers introduces redundant class borders
- the “code words” are a binary code which reflect the assignment of the classes for each classifier
- To build a high quality ensemble each of the class borders has to be sufficiently represented:
  - Row Separation: Each pair of code words should display a large Hamming distance (=large amount of mismatching bits).
  - Column Separation: the binary classifiers should be uncorrelated.

Manipulating the Class Labels

class	code word						
c_1	1	1	1	1	1	1	1
c_2	0	0	0	0	1	1	1
c_3	0	0	1	1	0	0	1
c_4	0	1	0	1	0	1	0

- A large Hamming distance between the rows allows a unique class prediction of the ensemble.
- How large is the Hamming distance between the result (0,1,1,1,1,1,1) and the codes for the classes c_1, c_2, c_3 and c_4?

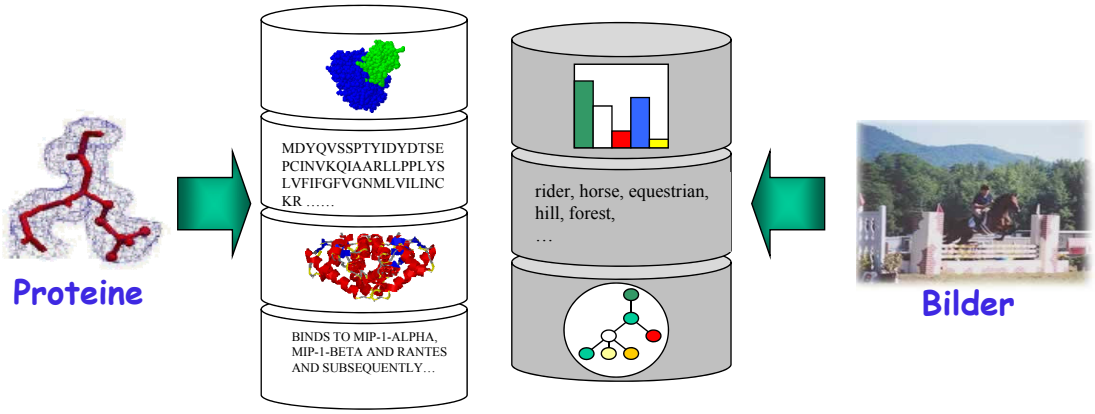
Manipulate the Learner

- Manipulating the learning algorithm by randomization:
  - Start with varying initial models (e.g. starting weight for back propagation in neural networks)
  - Randomized splits in decision trees (compare. random forests)

Ensembles and Multi-View Data

1. Introduction and Basic Principles of Ensemble Learning
2. Diversity in Ensemble Learning
3. Methods for Ensemble Construction
4. Mining Multi-View Data

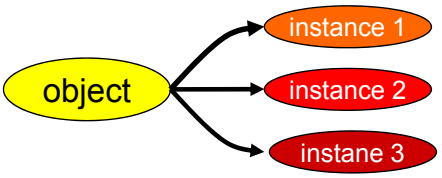
Multiview Data Mining



reason for the existence of multiple views:

- varying feature transformations
- varying measuring techniques
- varying aspect of the same object

=> Multiview Data



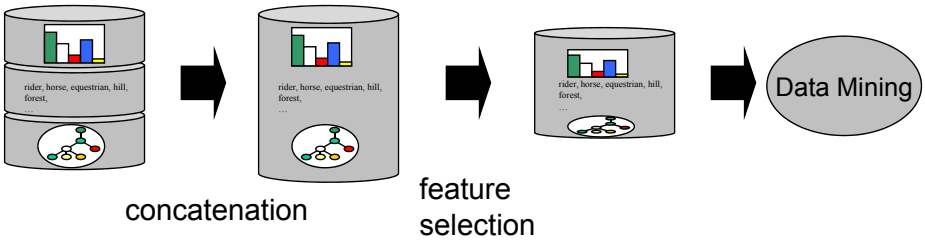
Problem Setting

Basic problems: (compare feature selection)

- all necessary information should be available ot the data mining algorithm
  => employ all available information
- too much unnecessary features might have a negative influence on the result => Use only the necessary features (compare feature selection)

Basic approach:

1. construct a joined features space.
2. Use feature reduction of feature selection
3. employ the data mining algorithm to lower dimensional representation.



Multiview Similarity and Distance Measures

Integrate multiple views on the object comparison level.

Idea: Keep the separated feature spaces and combine the derived similarity values over all views.

Example: weighted linear combination

$d_i(o_1,o_2)$: local metric or kernel in R_i.

$$D_{kombi}(o_1,o_2) = \sum_{R_i \in R} w_i \cdot d_i(o_1,o_2)$$

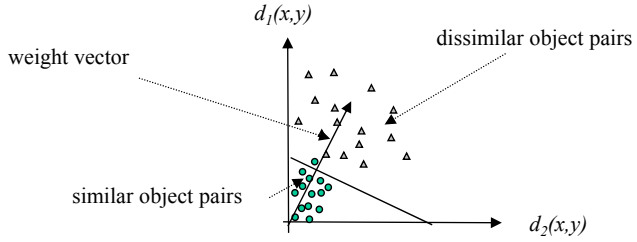
Learning Weights for each View

Formulate the weighting problem as linear classification task:

- use classes {„similar“,“dissimilar“} for pairs of objects (if x.c =y.c then (x,y).c = similar else dissimilar)
- normal vector of the separating hyperplane represents the weights
- training set: set of all pairs of vectors in the training set ($O(n^2)$)
- Feature space: distance vectors having $v_i = d_i(x,y)$ for all views R_i $1 \le i \le n$

Method:

- Determine the distance vectors for all object pairs
- Train a linear classifier
- determine the normal vector of the separating hyperplane as weights (MMH).



Combining Similarity and distance functions

Remarks:

- Caution: linear classifiers must result in positive normal vectors!
- It is possible to use the learned classifier directly as combination function. In this case the class probability for the class dissimilar is used as distance measures
- Combining distance values using more complex combination functions requires that mathematical properties which are required of the data mining algorithm still holds.

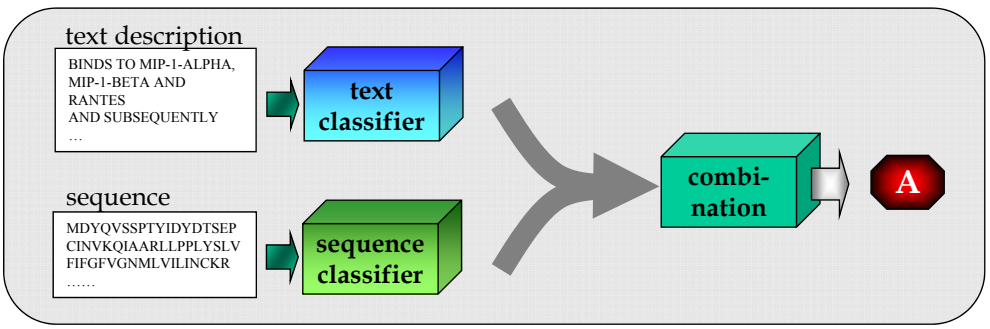
  (symmetry, triangular inequality, positive definiteness)

Multi-View Classifiers

Input: $o \in R_1 \times .. \times R_n$, where R_i *is the feature space of view* i.

Combination multiple classifiers (classifier combination):

1. Train classifier for each view
2. Classify a new objects in each view
3. combine the results to a global class prediction



Classifier Combination

How to combine class predictions in the light of varying prediction qualities?

1. Each classifier returns a confidence value c_A for each class A.
   For the vector confidence values $c^r(x)$: $\sum_{A \in C} c_A^r(x) = 1$

2. Classification by combining confidence values $c^r(x)$

$$pred(X) = \underset{A \in C}{\mathbf{argmax}} \left(\Theta_{r \in R} \left(c_A^r \right) \right)$$

mit $\Theta \in \{\min, \max, \sum, \prod\}$

Combining multiple Classifiers

Example:

Input: 2 views for bitmap images: color histograms(R1) and texture vectors (R2).
classes = {„contains water"=A, „no water"=B}
Bayes classifiers K1 (for R1) and K2 (for R2)

Different ways of combination:

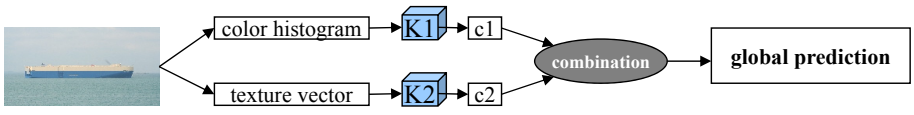
classification of image b:
K1(b)=c1 =(0.45, 0.55); K2(b) = c2 = (0.6, 0.4)

combination by average (sum):
c_{global} = (1.05, 0.95) * ½ = (0.525 , 0.475) and argmax(c_{global}) = A

combination by product:
c_{global} = (0.27, 0.22) and argmax(c_{global}) = A

combination by maximum:
c_{global} = (0.6, 0.55) and argmax(c_{global}) = A

combination by mininum:
c_{global} = (0.45, 0.4) and argmax(c_{global}) = A



Co-Training

Multiple views can be used to extend the set of labeled objects.

Input: 2 views on a object set where only a limited amount of instances is labeled.

Idea: Use a classifier to add additional labeled training samples.

Why does this approach require multi-view data to succeed?

Generating samples using a sinlge view

Experiment:

- Train a classifier ***CL*** on all labeled objects
- classify *k* unlabeled objects and add them to the training set
- Train a new classifier on the extended set

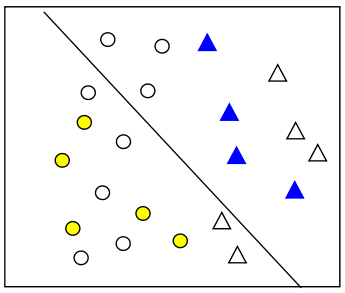
Problem:

- new data is labeled by the classifier CL
- CL is trained on the original samples
- CL is based on the same distribution
- the new labels do not add any diversity

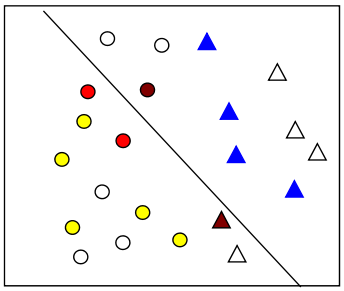
Generating samples using a sinlge view

Examples:

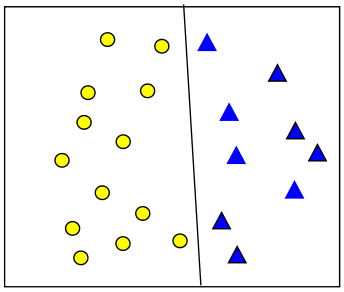
- blue = labelled objects (class triangle)
- yellow = labelled objects (class circle)
- red = labelled objects by CL_1



Training on original data



Training on newly labeled data



optimal solution

Conclusion:

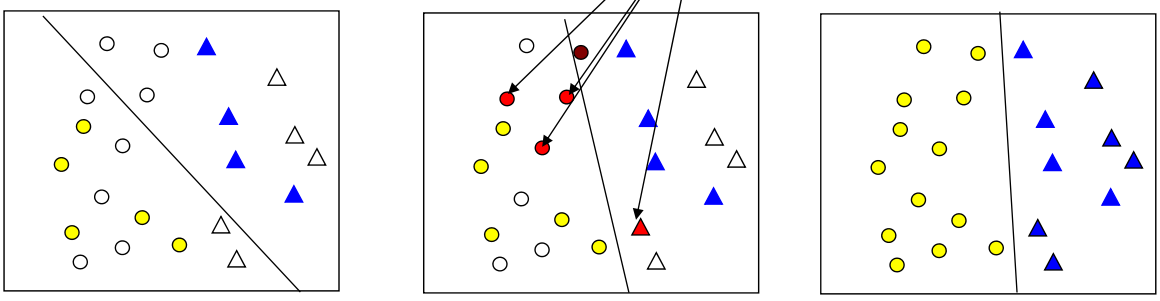
- the red objects only confirm the given assumptions of the classifier
- to add new diversity an additional source information is required

Co-Training

Idea: Employ at least two classifiers being used to label objects being used to train classifiers for other views.

example:

- blue = labelled objects (class triangle)
- yellow = labelled objects (class circle)
- red = labelled objects by CL1



original classifier

classifier based on new labels

optimal classifier

=> classification can be improved by using classifiers being trained on other views

The Co-Training Algorithm

Input: 2 sets of multi-view data

TR = labeled training set, *U* = set of unlabeled samples

Co-Training Algorithm

For *k* times do

For each R_i Do

train CL_i for view *i.*

draw a sample from *U*.

generate new labels using CL_i.

add newly labeled objects to *TR*

Clustering Multi-View Data

Requirements for Clustering multiview data :

- employ all views.
- use specific distance measures
- employ index and data structures for the employed data types
- the effort should only increase linearly with the number of views

Union method

Idea: An object is in a dense region if there are *k* neighbors sufficiently similar over all views. (similarity can be reliably observed from each view)

used for : sparse data

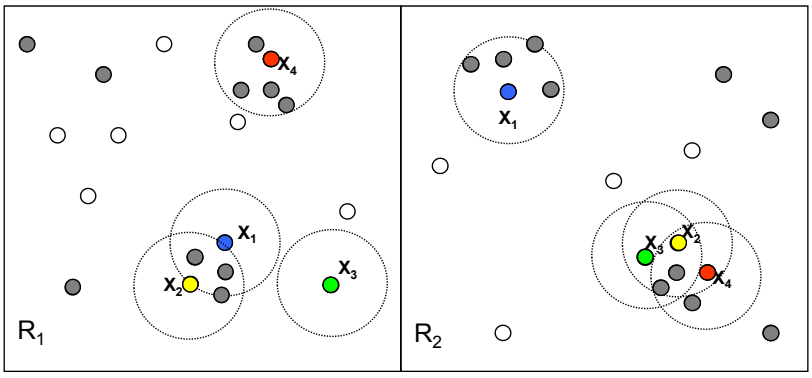
Union Core-Object:
Given: $\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_m \in IR^+$, $MinPts \in N$. $o \in O$ is an union core object if

$$\left| \bigcup_{R_i(o) \in o} N_{\varepsilon_i}^{R_i}(o) \right| \geq MinPts$$ where $N_{\varepsilon_i}^{R_i}(o)$ denotes the local ε-*neighborhood in view i* .

Direct union reachability:
Object $p \in O$ is **direct union reachable** by $q \in O$ w.r.t. $\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_m$
and *MinPts* if *q* is a union core object and $\exists\ i \in \{1, \ldots, m\} : R_i(p) \in N_{\varepsilon_i}^{R_i}(q)$

Union Method

Cluster expansion using the union method



MinPts = 3

Intersection Method

Idea: An object is in a dense region if the ε-neighborhoods in all views are dense.

Used for: dense views and unreliable similarity functions.

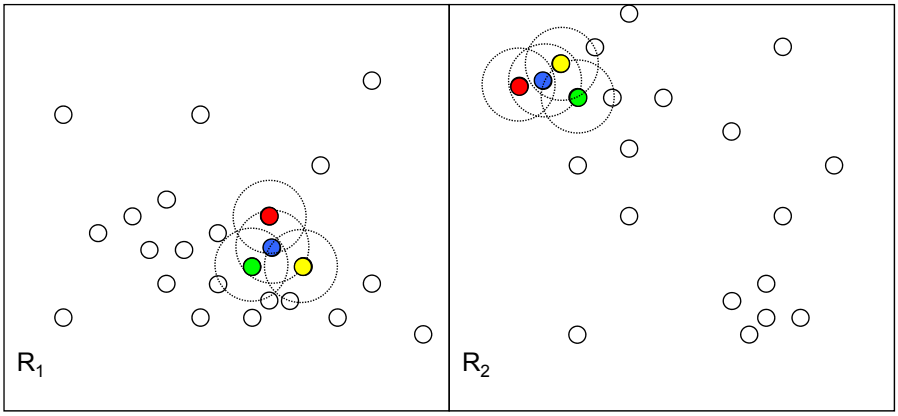
Intersection core object:
Given: $\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_m \in IR^+$, $MinPts \in N$. o ∈ O is an **intersection core** object if

$$\left| \bigcap_{R_i(o) \in o} N_{\varepsilon_i}^{R_i}(o) \right| \geq MinPts \text{ where } N_{\varepsilon_i}^{R_i}(o) \text{ denotes the local } \varepsilon\text{-neighborhood in view } i \text{ .}$$

Direct intersection reachable:
Object $p \in O$ is **direct intersection reachable by** $q \in O$ w.r.t. $\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_m$ *and MinPts if q* is an intersection core object and $\forall \; i \in \{1, \ldots, m\} : R_i(p) \in N_{\varepsilon_i}^{R_i}(q)$

Intersection Method

Cluster expansion using the intersection method



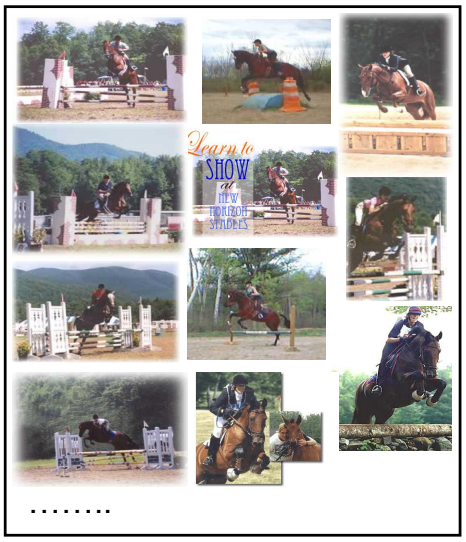
MinPts = 3

Example results

Cluster in single views.

Example images for cluster IC 5 (color histograms)

Example images for cluster IC 5 (segment trees)

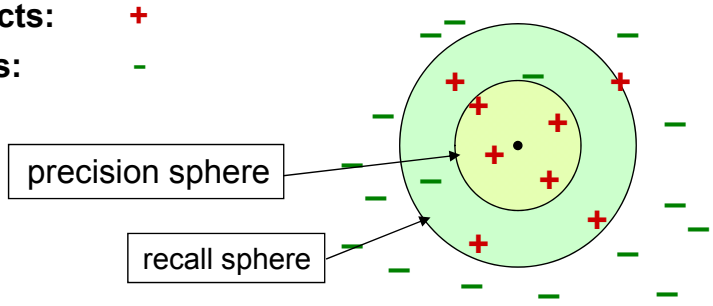


Cluster IC5 based on the intersection method.

Similarity in different views

truly similar objects: +

dissimilar objects: -



- optimally precision sphere = recall sphere
  (one view is enough)
- the intersection methods tries to remove false positive from the recall sphere.
- the union method tries to add false negatives to the precision sphere

Literature

- T. G. Dietterich: **Ensemble methods in machine learning.** In: Multiple Classier Systems, First International Workshop, MCS 2000, Cagliari, Italy, 2000.
- T. G. Dietterich: **Ensemble learning**. In: M. A. Arbib, editor, The Handbook of Brain Theory and Neural Networks. MIT Press 2003.
- J. Fürnkranz: **Round robin classification.** In: Journal of Machine Learning Research, 2:721-747, 2002.
- P.-N.Tan, M. Steinbach, and V. Kumar: **Introduction to Data Mining**, Addison-Wesley, 2006, Kapitel 5.6+5.8.
- G. Valentini and F. Masulli: **Ensembles of learning machines**. In: Neural Nets WIRN Vietri 2002.
- Kailing K., Kriegel H.-P.,Pryakhin A., Schubert M.: ***Clustering Multi-Represented Objects with Noise*** Proc. 8th Pacific-Asia Conference on Knowledge Discovery and Data Mining **(PAKDD'04)**, 2004.
- Blum. A, Mitchell T.: ***Combining Labeled and Unlabeled Data with Co-Training***, Workshop on Computational Learning Theory (COLT 98),1998