Knowledge Discovery in Databases II

Winter Term 2015/2016

Lecture 2:
Volume: High-Dimensional Data: Feature Selection

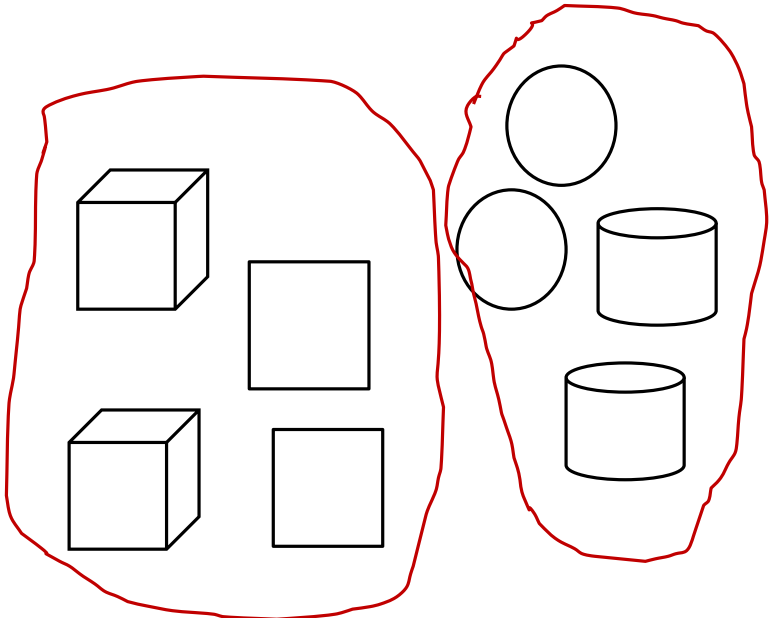
Lectures : Dr Eirini Ntoutsi, PD Dr Matthias Schubert
Tutorials: PD Dr Matthias Schubert

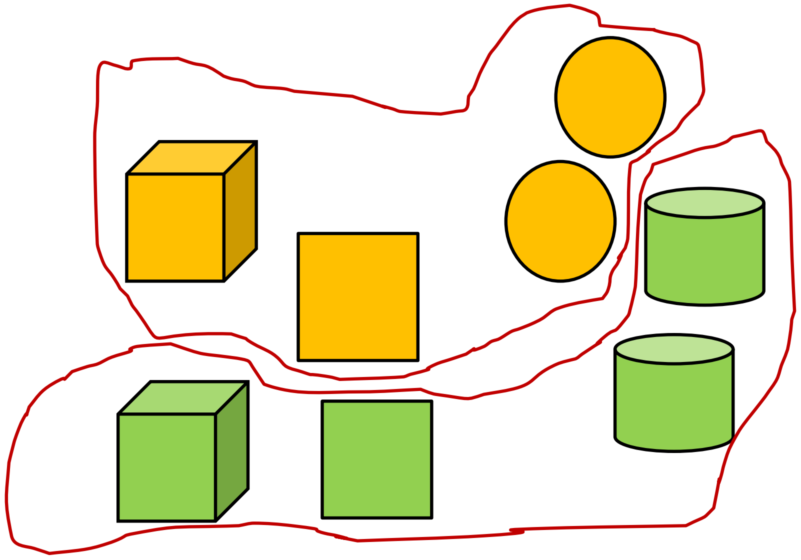
http://www.dbs.ifi.lmu.de/cms/Knowledge_Discovery_in_Databases_II_(KDD_II)

Outline

The more the merrier or more is less?

- Baby shapes game example

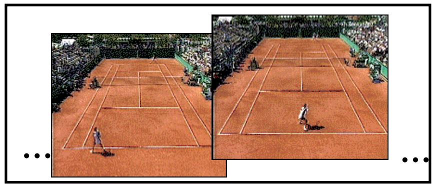

Based on shape grouping


Based on color grouping

What about grouping based on both shape and color?

Examples of High-Dimensional Data 1/2

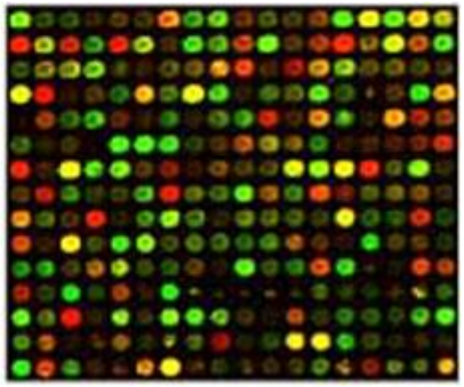
- **Image data**
  - low-level image descriptors (color histograms, textures, shape information ...)
  - If each pixel a feature, a 64x64 image → 4,096 features
  - Regional descriptors
  - Between 16 and 1,000 features



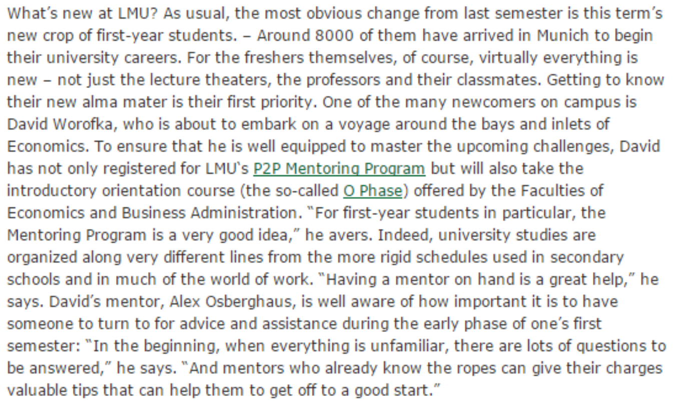
- **Metabolome data**
  - feature = concentration of one metabolite
  - The term metabolite usually restricted to small molecules, that are intermediates and products of metabolism.
  - The Human Metabolome Database contains 41,993 metabolite entries
  - Bavaria newborn screening (For each newborn in Bavaria, the blood concentrations of 43 metabolites are measured in the first 48 hours after birth)
  - between 50 and 2,000 features

Examples of High-Dimensional Data 2/2

- **Microarray data**
  - Features correspond to genes
  - Thousands or tens of thousands of genes in a single experiment
  - Up to 20,000 features
  - Dimensionality is much higher than the sample size



- **Text data**
  - Features correspond to words/terms
  - Different documents have different words
  - between 5,000 and 20,000 features
  - Very often, esp. in social media,
    - Abbreviations (e.g., Dr)
    - colloquial language (e.g., luv)
    - Special words (e.g, hashtags, @TwitterUser)



What's new at LMU? As usual, the most obvious change from last semester is this term's new crop of first-year students. – Around 8000 of them have arrived in Munich to begin their university careers. For the freshers themselves, of course, virtually everything is new – not just the lecture theaters, the professors and their classmates. Getting to know their new alma mater is their first priority. One of the many newcomers on campus is David Worofka, who is about to embark on a voyage around the bays and inlets of Economics. To ensure that he is well equipped to master the upcoming challenges, David has not only registered for LMU's P2P Mentoring Program but will also take the introductory orientation course (the so-called O Phase) offered by the Faculties of Economics and Business Administration. "For first-year students in particular, the Mentoring Program is a very good idea," he avers. Indeed, university studies are organized along very different lines from the more rigid schedules used in secondary schools and in much of the world of work. "Having a mentor on hand is a great help," he says. David's mentor, Alex Osberghaus, is well aware of how important it is to have someone to turn to for advice and assistance during the early phase of one's first semester: "In the beginning, when everything is unfamiliar, there are lots of questions to be answered," he says. "And mentors who already know the ropes can give their charges valuable tips that can help them to get off to a good start."

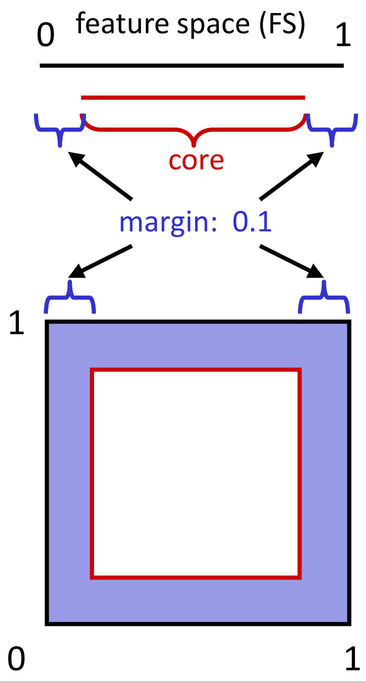
Excerpt from LMU website: http://tinyurl.com/qhq6byz

Challenges due to high dimensionality 1/5

- **Curse of Dimensionality**
  - Distance to the nearest and the farthest neighbor converge

$$\frac{\text{nearestNeighborDist}}{\text{farthestNeighborDist}} \approx 1$$

  - The likelihood that a data object is located on the margin of the data space exponentially increases with the dimensionality



1D: $P_{FS} = 1^1 = 1$

$P_{core} = 0.8^1 = 0.8$

$P_{margin} = 1 – 0.8 = 0.2$

2D: $P_{FS} = 1^2 = 1$

$P_{core} = 0.8^2 = 0.64$

$P_{margin} = 1 – 0.64 = 0.36$

3D: $P_{FS} = 1^3 = 1$

$P_{core} = 0.8^3 = 0.512$

$P_{margin} = 1 – 0.512 = 0.488$

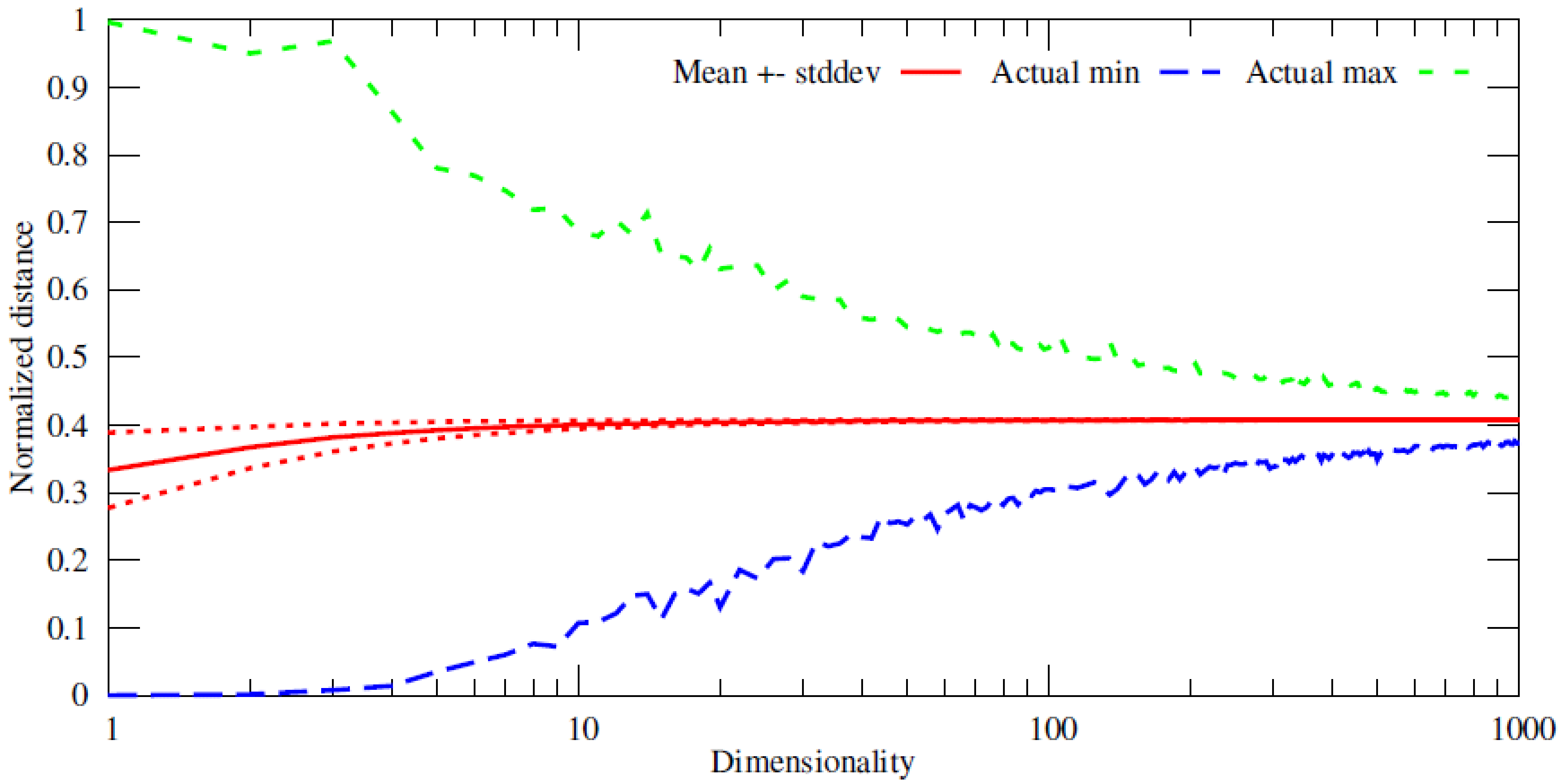
10D: $P_{FS} = 1^{10} = 1$

$P_{core} = 0.8^{10} = 0.107$

$P_{margin} = 1\text{-}0.107\text{=}0.893$

Challenges due to high dimensionality 2/5

- Pairwise distances example: sample of 10^5 instances drawn from a uniform [0, 1] distribution, normalized (1/ sqrt(d)).



Source: Tutorial on Outlier Detection in High-Dimensional Data, Zimek et al, ICDM 2012

Challenges due to high dimensionality 3/5

Further explanation of the *Curse of Dimensionality*:

- Consider the feature space of *d relevant* features for a given application

  => truly similar objects display small distances in most features

- Now add *d*x* additional features being *independent* of the initial feature space

- With increasing *x* the distance in the independent subspace will dominate the distance in the complete feature space

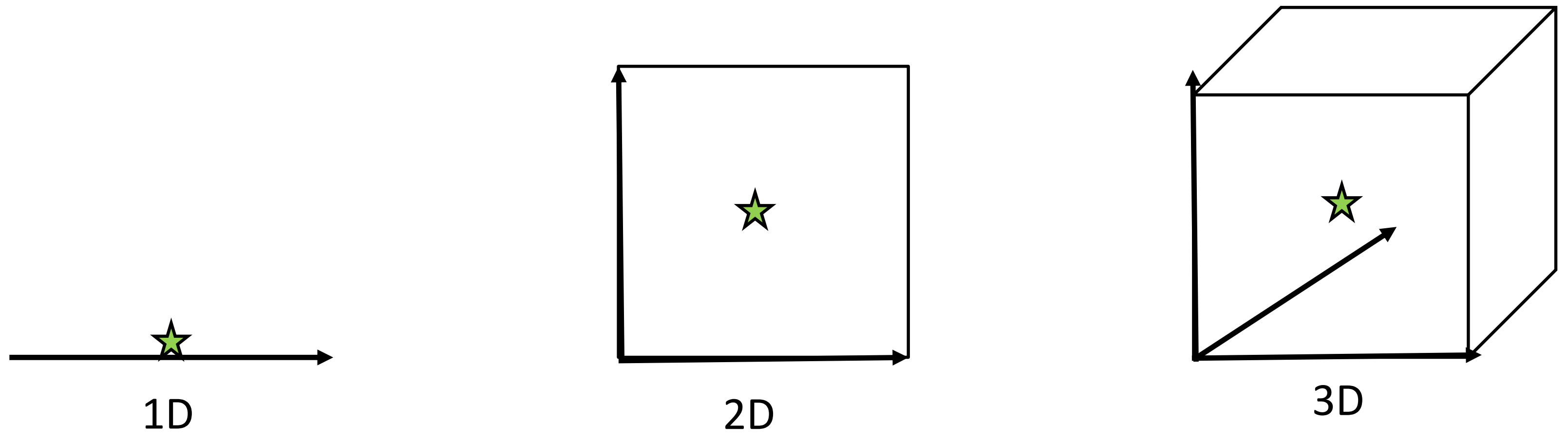
$\Rightarrow$ How many relevant features must by similar to indicate object similarity?

$\Rightarrow$ How many relevant features must be dissimilar to indicate dissimilarity?

$\Rightarrow$ With increasing dimensionality the likelihood that two objects are similar in every respect gets smaller.

Challenges due to high dimensionality 4/5

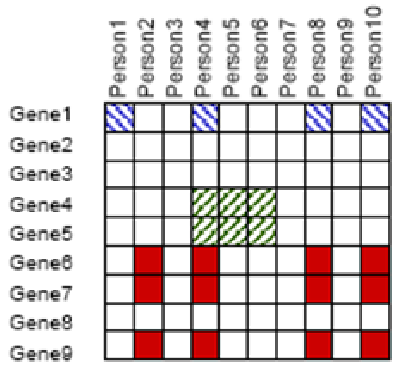
- The more features, the larger the *hypothesis space*



- The lower the hypothesis space
  - the easier to find the correct hypothesis
  - the less examples you need

Challenges due to high dimensionality 5/5

- Patterns and models on high-dimensional data are often *hard to interpret*.
  - e.g., long decision rules
- *Efficiency* in high-dimensional spaces is often limited
  - index structures degenerate
  - distance computations are much more expensive
- Pattern might only be observable only in *subspaces* or *projected spaces*



- Cliques of correlated features dominate the object description

Outline

1. Introduction and challenges of high dimensionality
2. Feature Selection
3. Feature Reduction and Metric Learning
4. Clustering in High-Dimensional Data

Feature selection

- A task to remove irrelevant and/or redundant features
  - *Irrelevant* features: not useful for a given task
    - Relevant vs irrelevant
  - *Redundant* features: a relevant feature may be redundant in the presence of another relevant feature with which it is strongly correlated.

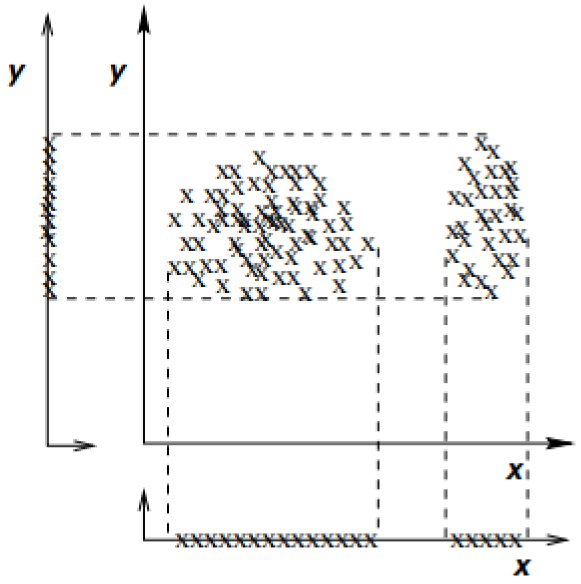
- Deleting irrelevant and redundant features can improve the *efficiency* as well as the *quality* of the found methods and patterns.

- New feature space: Delete all useless features from the original feature space.

- Feature selection ≠ Dimensionality reduction
- Feature selection ≠ Feature extraction

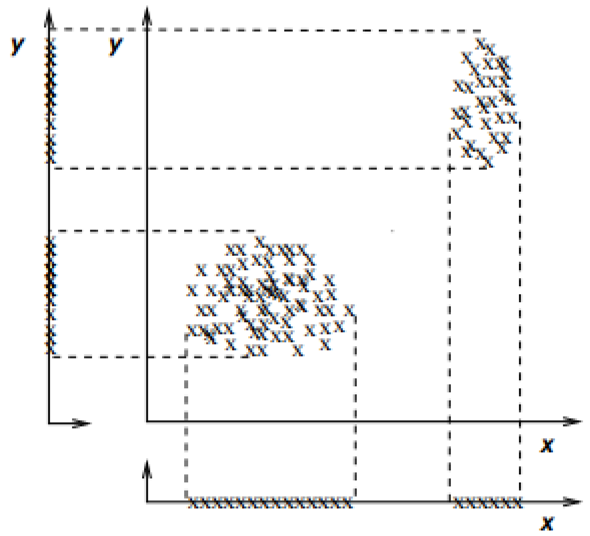
Irrelevant and redundant features (unsupervised learning case)

- Irrelevance



Feature *y* is irrelevant, because if we omit *x*, we have only one cluster, which is uninteresting.

- Redundancy

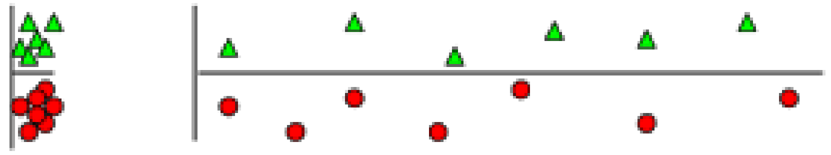


Features *x* and *y* are redundant, because *x* provides the same information as feature *y* with regard to discriminating the two clusters

Source: Feature Selection for Unsupervised Learning, Dy and Brodley, Journal of Machine Learning Research 5 (2004)

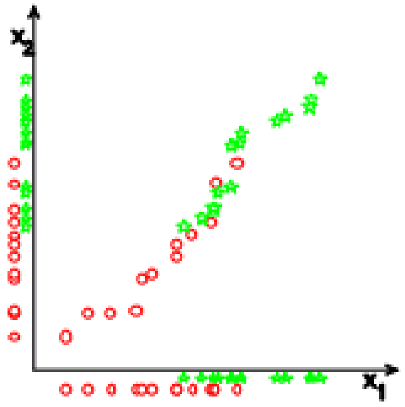
Irrelevant and redundant features (supervised learning case)

- Irrelevance



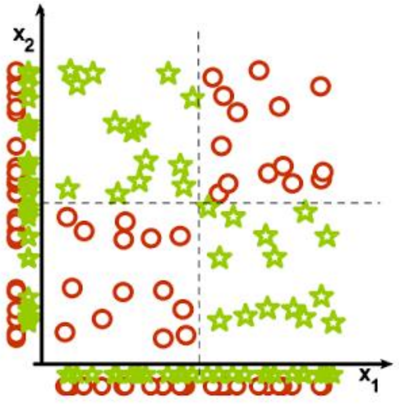
Feature *y* separates well the two classes.
Feature x is irrelevant.
Its addition “destroys” the class separation.

- Redundancy



Features *x* and *y* are redundant.

- Individually irrelevant, together relevant



Source: http://www.kdnuggets.com/2014/03/machine-learning-7-pictures.html

DATABASE SYSTEMS GROUP

Problem definition

- **Input:** Vector space $F = d_1 \times .. \times d_n$ *with* dimensions $D = \{d_1, .., d_n\}$.
- **Output:** a *minimal* subspace M over dimensions D`$\subseteq$ D which is *optimal* for a giving data mining task.
  - Minimality increases the efficiency, reduces the effects of the curse of dimensionality and increases interpretability.

Challenges:

- Optimality depends on the given task
- There are 2^d possible solution spaces (exponential search space)
- There is often no monotonicity in the quality of subspace

  (Features might only be useful in combination with certain other features)

$\Rightarrow$ For many popular criteria, feature selection is an exponential problem

$\Rightarrow$ Most algorithms employ search heuristics

2 main components

1. Feature subset generation
   - Single dimensions
   - Combinations of dimensions (subpaces)

2. Feature subset evaluation
   - Importance scores like information gain, χ^2
   - Performance of a learning algorithm

Feature selection methods 1/4

- Filter methods
  - Explores the general characteristics of the data, independent of the learning algorithm.
- Wrapper methods
  - The learning algorithm is used for the evaluation of the subspace
- Embedded methods
  - The feature selection is part of the learning algorithm

Feature selection methods 2/4

- Filter methods
  - Basic idea: assign an ``importance” score to each feature to filter out the useless ones
  - Examples: information gain, χ^2-statistic, TF-IDF for text
  - Disconnected from the learning algorithm.
  - Pros:
    - Fast
    - Simple to apply
  - Cons:
    - Doesn’t take into account interactions between features
    - Individually irrelevant features, might be relevant together (recall slide 14)

Feature selection methods 3/4

- Wrapper methods
  - A learning algorithm is employed and its performance is used to determine the quality of selected features.
  - Pros:
    - the ability to take into account feature dependencies.
    - interaction between feature subset search and model selection
  - Cons:
    - higher risk of overfitting than filter techniques
    - very computationally intensive, especially if building the classifier has a high computational cost.

Feature selection methods 4/4

- Embedded methods
  - Such methods integrate the feature selection in model building
  - Example: decision tree induction algorithm: at each decision node, a feature has to be selected.
  - Pros:
    - less computationally intensive than wrapper methods.
  - Cons:
    - specific to a learning method

Search strategies in the feature space

- Forward selection
  - Start with an empty feature space and add relevant features
- Backward selection
  - Start with all features and remove irrelevant features procedurally
- Branch-and-bound
  - Find the optimal subspace under the monotonicity assumption
- Randomized
  - Randomized search for a *k* dimensional subspace
- …

Selected methods in this course

1. Forward Selection and Feature Ranking
   - Information Gain , χ^2-Statistik, Mutual Information

2. Backward Elimination and Random Subspace Selection
   - Nearest-Neighbor criterion, Model-based search
   - Branch and Bound Search

3. *k*-dimensional subspace projections
   - Genetic Algorithms for Subspace Search
   - Feature Clustering for Unsupervized Problems

1. Forward Selection and Feature Ranking

Input: A *supervised* learning task

- Target variable *C*
- Training set of labeled feature vectors $<d_1, d_2, \ldots, d_n>$

Approach

- Compute the *quality* $q(d_i, C)$ for each dimension $d_i \in \{d_1,..,d_n\}$ to predict the correlation to *C*
- Sort the dimensions $d_1,..,d_n$ w.r.t. $q(d_i, C)$
- Select the k-best dimensions

Assumption:

Features are only correlated via their connection to *C*

=> it is sufficient to evaluate the connection between each single feature *d* and the target variable *C*

Statistical quality measures for features

How suitable is feature *d* for predicting the value of class attribute *C*?

Statistical measures :

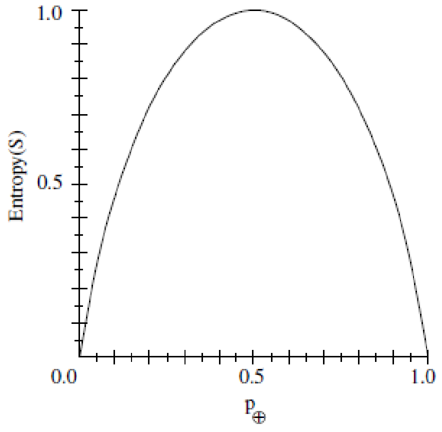
- Rely on distributions over feature values and target values.
  - For discrete values: determine probabilities for all value pairs.
  - For real valued features:
    - Discretize the value space (reduction to the case above)
    - Use probability density functions (e.g. uniform, Gaussian,..)
- How strong is the correlation between both value distributions?
- How good does splitting the values in the feature space separate values in the target dimension?
- Example quality measures:
  - Information Gain
  - Chi-square χ^2-statistics
  - Mutual Information

Information Gain 1/2

- Idea: Evaluate class discrimination in each dimension (Used in ID3 algorithm)
- It uses entropy, a measure of pureness of the data

$$Entropy(S) = \sum_{i=1}^{k} - p_i \log_2(p_i)$$

(p_i : relative frequency of *class* c_i in *S*)



- The information gain Gain(S,A) of an attribute A relative t measures the gain reduction in S due to splitting on A:

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

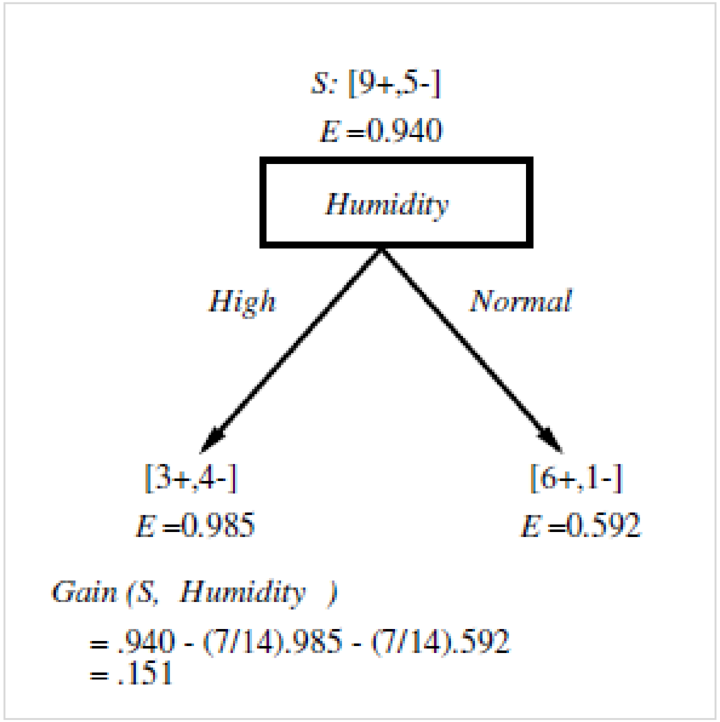
Before splitting

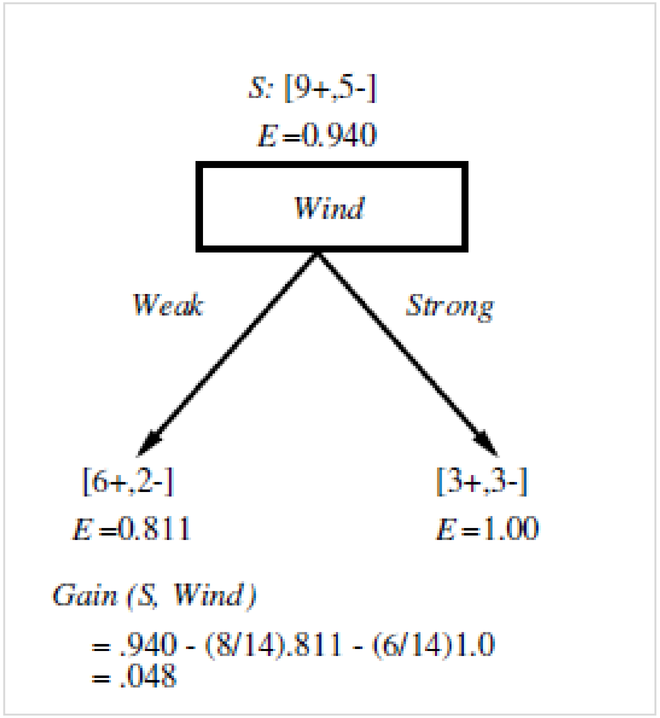
After splitting on A

- For nominal attributes: use attribute values
- For real valued attributes: Determine a splitting position in the value set.

Information Gain 2/2

- Which attribute, “Humidity” or “Wind” is better?





- Larger values better!

Chi-square χ2 statistics 1/2

- Idea: Measures the independency of a variable from the class variable.
- Contingency table
  - Divide data based on a split value s or based on discrete values
- Example: Liking science fiction movies implies playing chess?

Class attribute

Predictor attribute

	Play chess	**Not play chess**	Sum (row)
Like science fiction	250	200	450
Not like science fiction	50	1000	1050
Sum(col.)	300	1200	1500

- Chi-square χ^2 test

$$\chi^2 = \sum_{i=1}^{c}\sum_{j=1}^{r}\frac{(o_{ij}-e_{ij})^2}{e_{ij}}$$

o_{ij}:observed frequency
e_{ij}: expected frequency

$$e_{ij} = \frac{h_i h_j}{n}$$

Chi-square χ2 statistics 2/2

- Example

Class attribute

Predictor attribute

	Play chess	**Not play chess**	Sum (row)
Like science fiction	250 (90)	200 (360)	450
Not like science fiction	50 (210)	1000 (840)	1050
Sum(col.)	300	1200	1500

- χ^2 (chi-square) calculation (numbers in parenthesis are expected counts calculated based on the data distribution in the two categories)

$$\chi^2 = \frac{(250-90)^2}{90} + \frac{(50-210)^2}{210} + \frac{(200-360)^2}{360} + \frac{(1000-840)^2}{840} = 507.93$$

- Larger values better!

Mutual Information (MI)

- In general, MI between two variables *x*, y measures how much knowing one of these variables reduces uncertainty about the other
- In our case, it measures how much information a feature contributes to making the correct classification decision.
- Discrete case:

$$I(X,Y) = \sum_{y \in Y} \sum_{x \in X} p(x,y) \log \frac{p(x,y)}{p(x)\,p(y)}$$

p(*x*,*y*): the joint probability distribution function
p(x), p(y): the marginal probability distributions

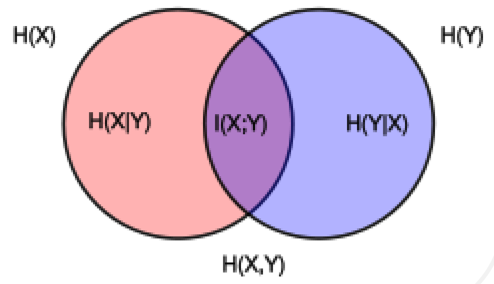
- Continuous case:

$$I(X,Y) = \int_Y \int_X p(x,y) \log \frac{p(x,y)}{p(x)\,p(y)} dx\,dy$$

- In case of statistical independence:
  - p(x,y)= p(x)p(y) → log(1)=0
  - knowing *x* does not reveal anything about *y*

Relation to entropy

$$\begin{aligned} I(X;Y) &= H(X) - H(X|Y) \\ &= H(Y) - H(Y|X) \\ &= H(X) + H(Y) - H(X,Y) \\ &= H(X,Y) - H(X|Y) - H(Y|X) \end{aligned}$$



Forward Selection and Feature Ranking - overview

Advantages:

- Efficiency: it compares $\{d_1, d_2, \ldots, d_n\}$ features to the class attribute C instead of $\binom{n}{k}$ subspaces
- Training suffices with rather small sample sets

Disadvantages:

- Independency assumption: Classes and features must display a direct correlation.
- In case of correlated features: Always selects the features having the strongest direct correlation to the class variable, even if the features are strongly correlated with each other.
  (features might even have an identical meaning)

Selected methods in this course

1. Forward Selection and Feature Ranking
   – Information Gain , χ^2-Statistik, Mutual Information

2. Backward Elimination and Random Subspace Selection
   – Nearest-Neighbor criterion, Model-based search
   – Branch and Bound Search

3. *k*-dimensional projections
   – Genetic Algorithms for Subspace Search
   – Feature Clustering for Unsupervized Problems

2. Backward Elimination

Idea: Start with the complete feature space and delete redundant features

Approach: Greedy Backward Elimination

1. Generate the subspaces *R* of the feature space *F*
2. Evaluate subspaces *R* with the quality measure *q(R)*
3. Select the best subspace *R** w.r.t. *q*(*R*)
4. If *R** has the wanted dimensionality, terminate
   else start backward elimination on *R**.

Applications:

- Useful in supervised and unsupervised setting
  - in unsupervised cases, *q*(*R*) measures structural characteristics
- Greedy search if there is no monotonicity on *q*(*R*)
  => for monotonous *q*(*R*) employ branch and bound search

Distance-based subspace quality

- **Idea:** Subspace quality can be evaluated by the distance between the within-class nearest neighbor and the between-classes nearest neighbor
- **Quality criterion:**

  For each $o \in D$, compute the closest object having the same class $NN_C(o)$ (*within-class nearest neighbor*) and the closest object belonging to another class $NN_{K \neq C}(o)$ (*between-classes nearest neighbor*) where $C = class(o)$.

  Quality of subspace U: $$q(U) = \frac{1}{|D|} \cdot \sum_{o \in D} \frac{NN^U_{K \neq C}(o)}{NN^U_C(o)}$$

- **Remark**: $q(U)$ is not monotonous.

  →By deleting a dimension, the quality can increase or decrease.

Model-based approach

- **Idea**: Directly employ the data mining algorithm to evaluate the subspace.
- **Example:** Evaluate each subspace by training a Naive Bayes classifier

Practical aspects:

- Success of the data mining algorithm must be measurable (e.g. class accuracy)
- Runtime for training and applying the classifier should be low
- The classifier parameterization should not be of great importance
- Test set should have a moderate number of instances

Backward Elimination - overview

Advantages:

- Considers complete subspaces (multiple dependencies are used)
- Can recognize and eliminate redundant features

Disadvantages:

- Tests w.r.t. subspace quality usually requires much more effort
- All solutions employ heuristic greedy search which do not necessarily find the optimal feature space.

Backward elimination: Branch and Bound Search

- **Given:** A classification task over the feature space *F*.
- **Aim:** Select the *k* best dimensions to learn the classifier.
- Backward elimination approach "Branch and Bound", by Narendra and Fukunaga, 1977 is guaranteed to find the optimal feature subset under the monotonicity assumption
- The monotonicity assumption states that the addition of features can only increase the value of the objective function *J*, this is

$$J(x_{i_1}) < J(x_{i_1}, x_{i_2}) < J(x_{i_1}, x_{i_2}, x_{i_3}) < \cdots < J(x_{i_1}, x_{i_2}, \cdots, x_{i_N})$$

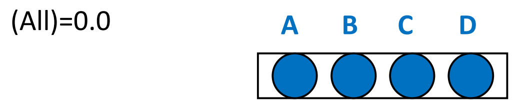
- Branch and Bound starts from the full set and removes features using *a depth-first strategy*
  - Nodes whose objective function are lower than the current best are not explored since the monotonicity assumption ensures that their children will not contain a better solution.

Slide adapted from: http://courses.cs.tamu.edu/rgutier/cs790_w02/l17.pdf

Example: Branch and Bound Search 1/8

Example: Original dimensionality 4, <A,B,C,D>. Target dimensionality *d* = 1.

○ selected feature ● removed feature

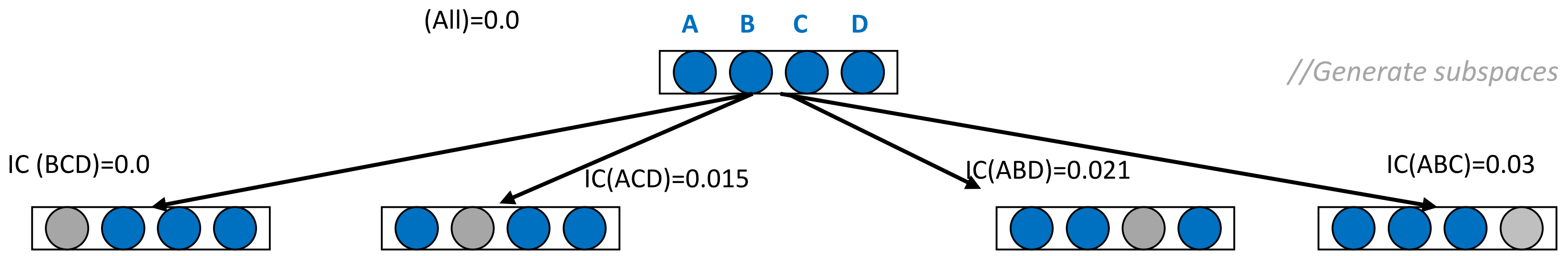


//Start from the full set

Example: Branch and Bound Search 2/8

Example: Original dimensionality 4, <A,B,C,D>. Target dimensionality *d* = 1.

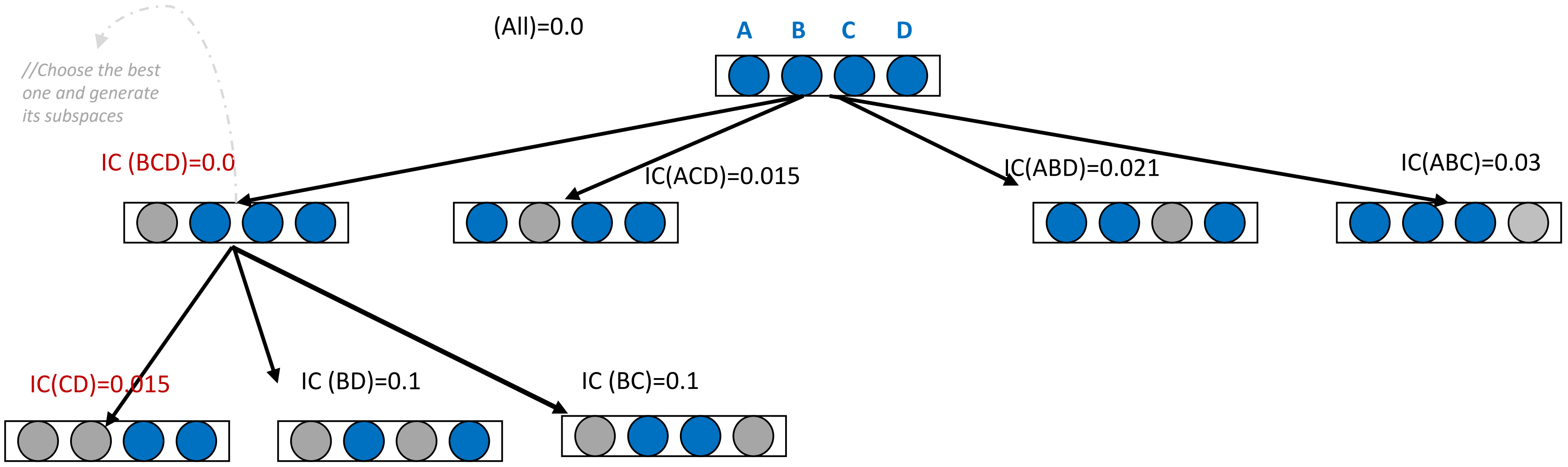
○ selected feature ● removed feature



Example: Branch and Bound Search 3/8

Example: Original dimensionality 4, <A,B,C,D>. Target dimensionality *d* = 1.

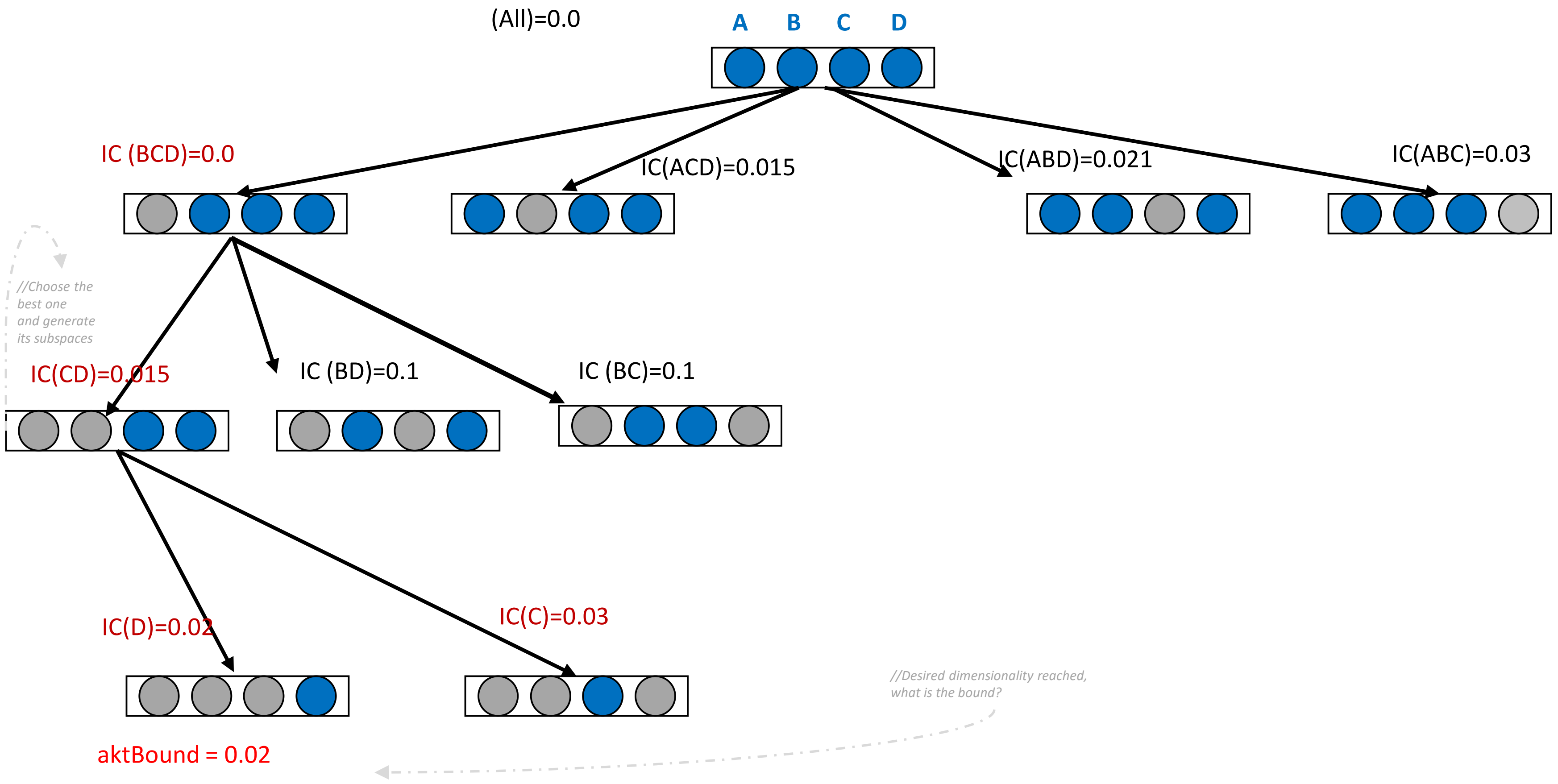
● selected feature ● removed feature



Example: Branch and Bound Search 4/8

Example: Original dimensionality 4, <A,B,C,D>. Target dimensionality *d* = 1.

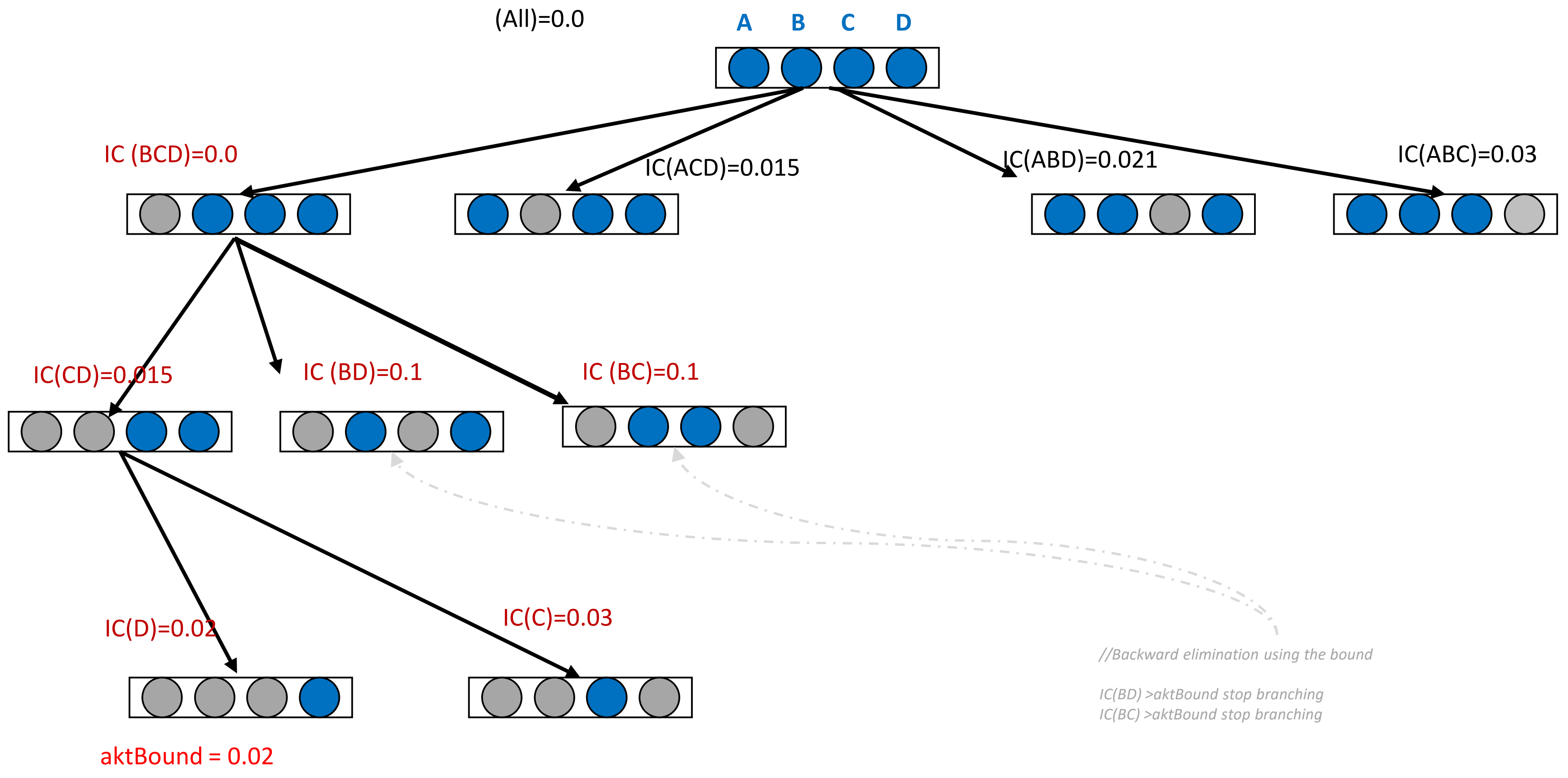
⚪ selected feature   🔵 removed feature



Example: Branch and Bound Search 5/8

Example: Original dimensionality 4, <A,B,C,D>. Target dimensionality *d* = 1.

● selected feature   ● removed feature



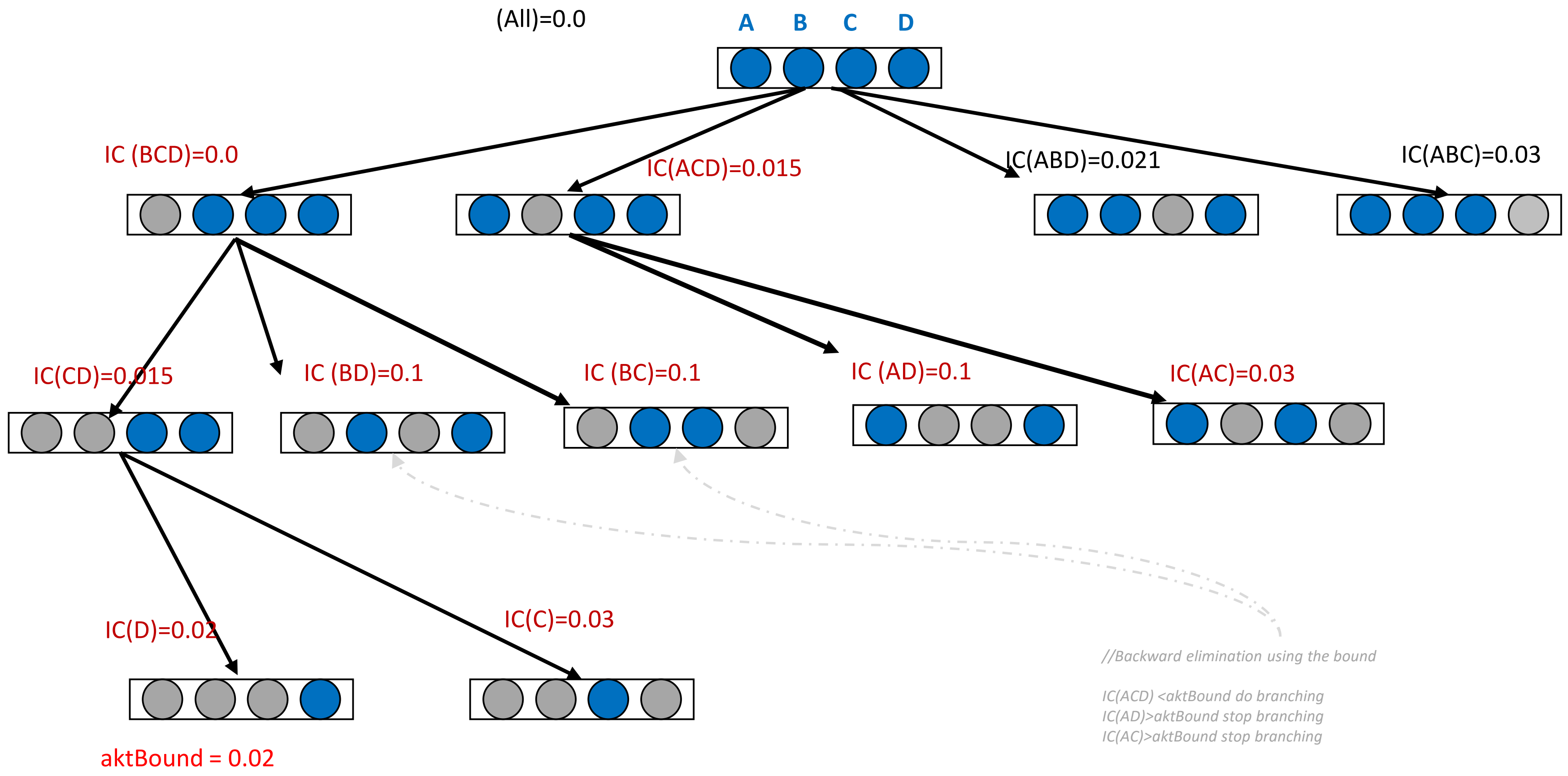
//Backward elimination using the bound

IC(BD) >aktBound stop branching
IC(BC) >aktBound stop branching

Example: Branch and Bound Search 6/8

Example: Original dimensionality 4, <A,B,C,D>. Target dimensionality *d* = 1.

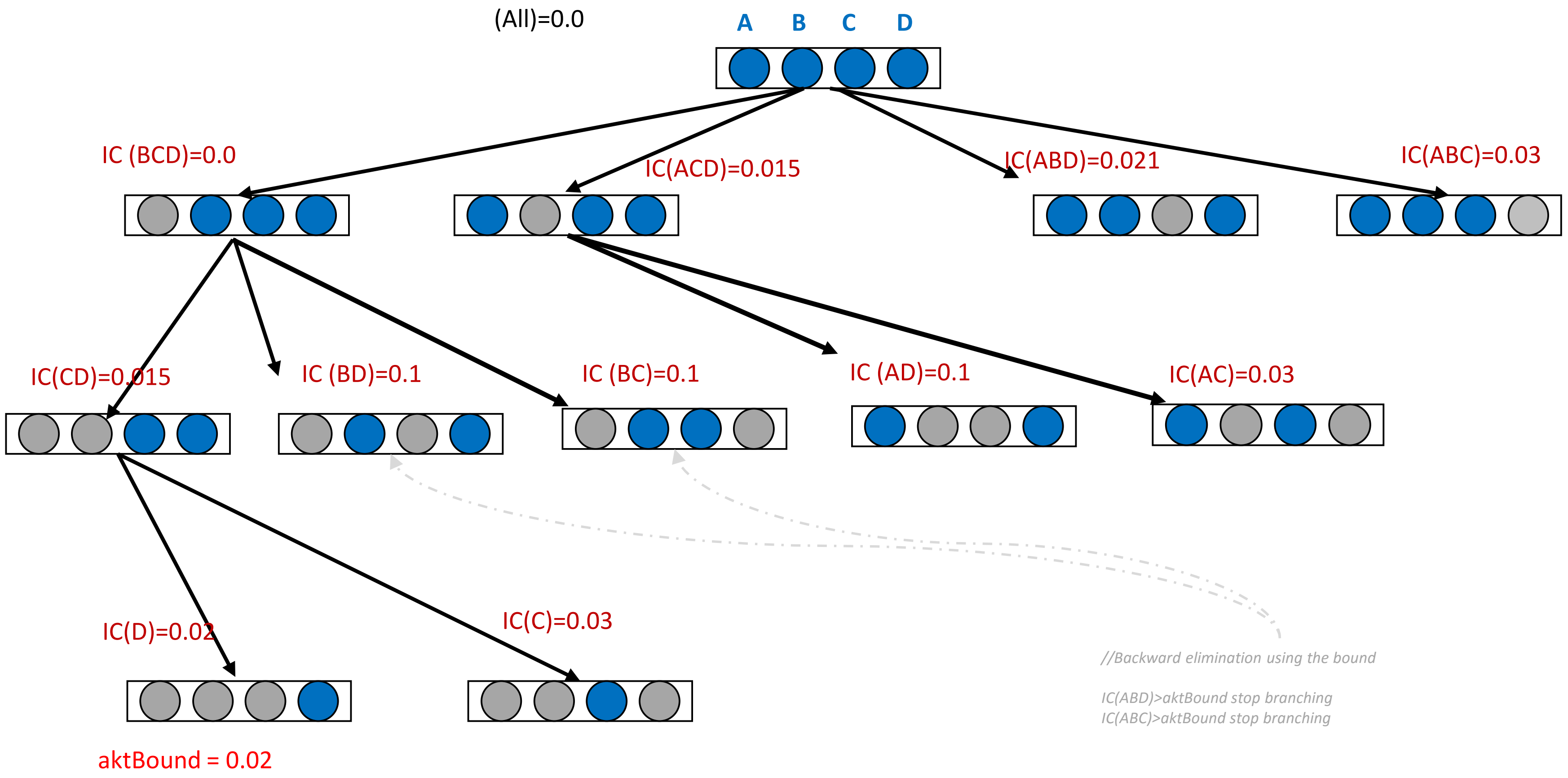
● selected feature ● removed feature



Example: Branch and Bound Search 7/8

Example: Original dimensionality 4, <A,B,C,D>. Target dimensionality *d* = 1.

⚪ selected feature 🔵 removed feature



Backward elimination: Branch and Bound Search

Given: A classification task over the feature space *F*.

Aim: Select the *k* best dimensions to learn the classifier.

Backward-Elimination based in Branch and Bound:

```
FUNCTION BranchAndBound(Featurespace F, int k)
        queue.init(ASCENDING);
        queue.add(F, quality(F))
        curBound:= INFINITY;
        WHILE queue.NotEmpty() and aktBound > queue.top() DO
                curSubSpace:= queue.top();
                FOR ALL Subspaces U of curSubSpace DO
                        IF U.dimensionality() = k THEN
                            IF quality(U)< curBound THEN
                                curBound  := quality(U);
                                BestSubSpace := U;
                        ELSE
                            queue.add(U, quality(U));
        RETURN BestSubSpace
```

Subspace Inconsistency (IC)

- **Idea:** Having identical vectors u, v ($v_i = u_i \;\; 1 \leq i \leq d$) in subspace U but the class labels are different ($C(u) \neq C(v)$)
  - ➔ the subspace displays an inconsistent labeling
- Measuring the inconsistency of a subspace *U*
  - – $X_U(A)$: Amount of all identical vectors *A* in *U*
  - – $X^c_U(A)$: Amount of all identical vectors in *U* having class label *C*
- $IC_U(A)$: inconsistency w.r.t. *A* in *U*

$$IC_U(A) = X_U(A) - \max_{c \in C} X_U^c(A)$$

Inconsistency of U: $$IC(U) = \frac{\sum_{A \in DB} IC_U(A)}{|DB|}$$

Monotonicity: $$U_1 \subset U_2 \Rightarrow IC(U_1) \geq IC(U_2)$$

Branch and Bound search - overview

Advantage:

- Monotonicity allows efficient search for optimal solutions
- Well-suited for binary or discrete data
  (identical vectors are very likely with decreasing dimensionality)

Disadvantages:

- Useless without groups of identical features (real-valued vectors)
- Worse-case runtime complexity remains exponential in *d*

Selected methods in this course

1. Forward Selection and Feature Ranking
   – Information Gain , χ^2-Statistik, Mutual Information

2. Backward Elimination and Random Subspace Selection
   – Nearest-Neighbor criterion, Model-based search
   – Branch and Bound Search

3. *k*-dimensional projections
   – Genetic Algorithms for Subspace Search
   – Feature Clustering for Unsupervised Problems

k-dimensional projections

- Idea: Select *n* random subspaces having the target dimensionality *k* out of the $\binom{d}{k}$ possible subspaces and evaluate each of them.

- Application:
  - Needs quality measures for complete subspaces
  - Trade-off between quality and effort depends on *k.*
- Disadvantages:
  - No directed search for combining well-suited and non-redundant features.
  - Computational effort and result strongly depend on the used quality measure and the sample site.
- Randomization approaches
  - Genetic algorithms
  - *k*-medoids feature clustering

Genetic Algorithms

- Idea: Randomized search through genetic algorithms

Genetic Algorithms:

- Encoding of the individual states in the search space: bit-strings
- Population of solutions := set of *k*-dimensional subspaces
- Fitness function: quality measure for a subspace
- Operators on the population:
  - Mutation: dimension d_i in subspace U is replaced by dimension d_j with a likelihood of *x*%
  - Crossover: combine two subspaces U_1, U_2
    - Unite the features sets of U_1 and U_2.
    - Delete random dimensions until dimensionality is *k*
- Selection for next population: All subspaces having at least a quality of *y*% of the best fitness in the current generation are copied to the next generation.
- Free tickets: Additionally each subspace is copied to the next generation with a probability of *u%* into the next generation.

Genetic Algorithm: Schema

```
Generate initial population
WHILE Max_Fitness > Old_Fitness  DO
        Mutate current population
        WHILE nextGeneration < PopulationSize DO
            Generate new  candidate from pairs of old subspaces
            IF K has a free ticket or K is fit enough THEN
                    copy K to the next generation
 RETURN fittest subspace
```

Genetic Algorithms

Remarks:

- Here: only basic algorithmic scheme (multiple variants)
- Efficient convergence by “Simulated Annealing”

  (Likelihood of free tickets decreases with the iterations)

Advantages:

- Can escape from local extreme values during the search
- Often good approximations for optimal solutions

Disadvantages:

- Runtime is not bounded can become rather inefficient
- Configuration depends on many parameters which have to be tuned to achieve good quality results in efficient time

Feature-clustering

Given: A feature space *F* and an unsupervised data mining task.

Target: Reduce *F* to a subspace of *k* (original) dimensions while reducing redundancy.

Idea: Cluster the features in the space of objects and select one representative feature for each of the clusters.
(This is equivalent to clustering in a transposed data matrix)

Typical example: item-based collaborative filtering

	1 (Titanic)	2 (Braveheart)	3 (Matrix)	4 (Inception)	5 (Hobbit)	6 (300)
Susan	5		5	5	4	
Bill	3	3		1		1
Jenny	5	4	1	1		4
Tim			4	4	3	3
Thomas		1	1	4	?	4

Feature-clustering

- Feature similarity, e.g.,
  - Cosine similarity
$$sim(i,j) = \cos(\vec{i},\vec{j}) = \frac{\vec{i}\cdot\vec{j}}{\|\vec{i}\|_2 * \|\vec{j}\|_2}$$
  - Pearson correlation:
$$sim(i,j) = \frac{\sum_{u\in U}(R_{u,i}-\bar{R}_i)(R_{u,j}-\bar{R}_j)}{\sqrt{\sum_{u\in U}(R_{u,i}-\bar{R}_i)^2}\sqrt{\sum_{u\in U}(R_{u,j}-\bar{R}_j)^2}}$$
- Algorithmic scheme:
  - Cluster features with a *k*-medoid clustering method based on correlation
  - Select the medoids to span the target data space
- Remark:
  - For group/cluster of dependent features there is one representative feature
  - Other clustering algorithms could be used as well.
  - For large dimensionalities, approximate clustering methods are used due to there linear runtime (c.f., BIRCH upcoming lectures)

Feature-Clustering based on correlation

Advantages:

- Depending on the clustering algorithm quite efficient
- Unsupervised method

Disadvantages:

- Results are usually not deterministic (partitioning clustering)
- Representatives are usually unstable for different clustering methods and parameters.
- Based on pairwise correlation and dependencies

  => multiple dependencies are not considered

Feature selection: overview

- *Forward-Selection*: Examines each dimension D‘ $\in\{D_{1,},..,D_d\}$. and selects the k-best to span the target space.
  - Greedy Selection based on Information Gain, $\chi 2$ Statistics or Mutual Information
- *Backward-Elimination*: Start with the complete feature space and successively remove the worst dimensions.
  - Greedy Elimination with model-based and nearest-neighbor based approaches
  - Branch and Bound Search based on inconsistency
- *k-dimensional Projections*: Directly search in the set of k-dimensional subspaces for the best suited
  - Genetic algorithms (quality measures as with backward elimination)
  - Feature clustering based on correlation

Feature selection: discussion

- Many algorithms based on different heuristics
- There are two reason to delete features:
  - Redundancy: Features can be expressed by other features.
  - Missing correlation to the target variable
- Often even approximate results are capable of increasing efficiency and quality in a data mining tasks
- **Caution**: Selected features need not have a causal connection to the target variable, but both observation might depend on the same mechanisms in the data space (hidden variables).
- Different indicators to consider in the comparison of before and after selection performance
  - Model performance, time, dimensionality, …

What you should know by now

- Reasons for using feature selection
- Curse of dimensionality
- Forward selection and feature ranking
  - Information Gain
  - $\chi 2$ Statistics
  - Mutual Information
- Backward Elimination
  - Model-based subspace quality
  - Nearest neighbor-based subspace quality
  - Inconsistency and Branch & Bound search
- k-dimensional projections
  - Genetic algorithms
  - Feature clustering

Literature

- I. Guyon, A. Elisseeff: An Introduction to Variable and Feature Selection, Journal of Machine Learning Research 3, 2003.
- H. Liu and H. Motoda, *Computations methods of feature selection*, Chapman & Hall/ CRC, 2008.
- A.Blum and P. Langley: *Selection of Relevant Features and Examples in Machine Learning*, Artificial Intelligence (97),1997.
- H. Liu and L. Yu: *Feature Selection for Data Mining* (WWW), 2002.
- L.C. Molina, L. Belanche, Â. Nebot: Feature Selection Algorithms: *A Survey and Experimental Evaluations*, ICDM 2002, Maebashi City, Japan.
- P. Mitra, C.A. Murthy and S.K. Pal: *Unsupervised Feature Selection using Feature Similarity*, IEEE Transacitons on pattern analysis and Machicne intelligence, Vol. 24. No. 3, 2004.
- J. Dy, C. Brodley: *Feature Selection for Unsupervised Learning*, Journal of Machine Learning Research 5, 2004.
- M. Dash, H. Liu, H. Motoda*: Consistency Based Feature Selection*, 4th Pacific-Asia Conference, PADKK 2000, Kyoto, Japan, 2000.