Data Mining

Databases, Aarhus University

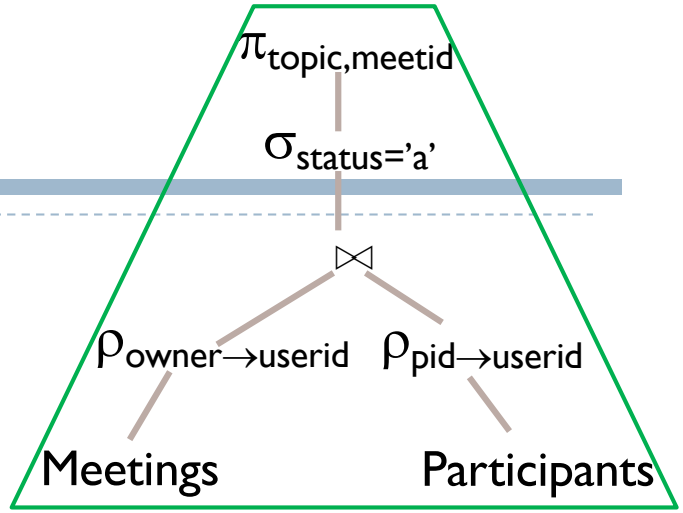
Ira Assent

Intended learning outcomes

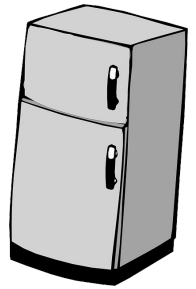
- Be able to
  - Describe the goals and applications of common data mining approaches
  - Discuss the basic steps in k-means clustering, decision tree classification, and association rule mining

ira@cs.au.dk

Review: query optimization



- Heuristic query optimization
  - Sequence of single query plans
  - Each (presumably) more efficient than previous
  - apply first the operations that reduce the size of intermediate results
    - select and project operations as far down the tree as possible
    - More restrictive select and join operations earlier than less restrictive ones
- Cost-based query optimization
  - Many query plans generated
  - The cost of each is estimated, with the most efficient chosen
    - E.g. I/O cost in blocks
  - Checking available statistics, access paths, use cost formula



- Heuristic optimization is more efficient to generate, but may not yield the optimal query evaluation plan
- Cost-based optimization relies on statistics gathered on the relations

ira@cs.au.dk

Textbook example cost-based estimate of join

- SELECT * FROM DEPARTMENT JOIN EMPLOYEE ON Dnumber=Dno;
- DEPARTMENT $\bowtie_{Dnumber=Dno}$ EMPLOYEE
- Statistics and access paths for Department file:
  - Department r_D=125 records in b_D=13 blocks
  - Primary index on Dnumber $x_{Dnumber}$=1
  - Secondary index on Mgr_ssn s_{Mgr_ssn}=1, x_{Mgr_ssn}=2
  - Join selectivity 1/125 because Dnumber is key
  - Assume bfr_{ED}=4 for join result relation

The cheat sheet (brightspace) is your friend

- Employee file
  - r_E = 10000 records with blocking factor (number of records per block) bfr_E=5, so b_E=2000 blocks
  - selection cardinality for Salary of 20 on average (s_{Salary}=20)
  - Selection cardinality key Ssn s_{Ssn}=1
  - Selection cardinality for Dno s_{Dno}=80
  - Selection cardinality for Sex s_{Sex}=5000

ira@cs.au.dk

Textbook example cost-based estimate of join

Department r_D=125 records, b_D=13 blocks
Primary index on Dnumber $x_{Dnumber}$=1
Secondary index on Mgr_ssn s_{Mgr_ssn}=1, x_{Mgr_ssn}=2
js_Q 1/125, bfr_{ED}=4

$DEPARTMENT \bowtie_{Dnumber=Dno} EMPLOYEE$

Employee r_E = 10000 records, bfr_E=5, b_E=2000 blocks
Selection cardinalities s_{Salary}=20, s_{Ssn}=1, s_{Dno}=80, s_{Sex}=5000

- Possible plans if 3 main memory buffers and their cost calculation:
  - Nested loop join, EMPLOYEE outer relation:
    - $b_E+(b_E*b_D)+((js_Q*r_E*r_D)/bfr_{ED})=2000+(2000*13)+(((1/125)*10000*125)/4)=30500$
  - Nested loop join, DEPARTMENT outer relation
    - $b_D+(b_E*b_D)+((js_Q*r_E*r_D)/bfr_{ED})=13+(2000*13)+(((1/125)*10000*125)/4)=28513$
  - Index-based join with EMPLOYEE outer relation:
    - $b_E+(r_E*(x_{Dno}+1))+((js_Q*r_E*r_D)/bfr_{ED})=2000+(10000*2)+(((1/125)*10000*125)/4)=24500$
  - Index-based join with DEPARTMENT outer relation:
    - $b_D+(r_D*(x_{Dno}+s_{Dno}))+((js_Q*r_E*r_D)/bfr_{ED})=13+(125*2+80)+(((1/125)*10000*125)/4)=12763$
  - Hash join: $3*(b_D+b_E)+((js_Q*r_E*r_D)/bfr_{ED})=3*(13+2000)+2500=8539$

ira@cs.au.dk

If we had more main memory…

Department r_D=125 records, b_D=13 blocks
Primary index on Dnumber $x_{Dnumber}$=1
Secondary index on Mgr_ssn s_{Mgr_ssn}=1, x_{Mgr_ssn}=2
js_Q 1/125, bfr_{ED}=4

$DEPARTMENT \bowtie_{Dnumber=Dno} EMPLOYEE$

Employee r_E = 10000 records, bfr_E=5, b_E=2000 blocks
Selection cardinalities s_{Salary}=20, s_{Ssn}=1, s_{Dno}=80, s_{Sex}=5000

If we had n_B=15 main memory buffers (or more), we could have the smaller relation Department with 13 blocks in main memory, then e.g.

- Nested loop join, DEPARTMENT outer relation
  - $b_D+(b_E*(b_D/n_B-2))+((js_Q*r_E*r_D)/bfr_{ED})=13+(2000*1)+(((1/125)*10000*125)/4)=4513$

- Nested loop join, EMPLOYEE outer relation:
  - $b_E+(b_E*b_D)+((js_Q*r_E*r_D)/bfr_{ED})=2000+(2000*13)+(((1/125)*10000*125)/4)=30500$
- Nested loop join, DEPARTMENT outer relation
  - $b_D+(b_E*b_D)+((js_Q*r_E*r_D)/bfr_{ED})=13+(2000*13)+(((1/125)*10000*125)/4)=28513$
- Index-based join with EMPLOYEE outer relation:
  - $b_E+(r_E*(x_{Dno}+1))+((js_Q*r_E*r_D)/bfr_{ED})=2000+(10000*2)+(((1/125)*10000*125)/4)=24500$
- Index-based join with DEPARTMENT outer relation:
  - $b_D+(r_D*(x_{Dno}+s_{Dno}))+((js_Q*r_E*r_D)/bfr_{ED})=13+(125*2+80)+(((1/125)*10000*125)/4)=12763$
- Hash join: $3*(b_D+b_E)+((js_Q*r_E*r_D)/bfr_{ED})=3*(13+2000)+2500=8539$

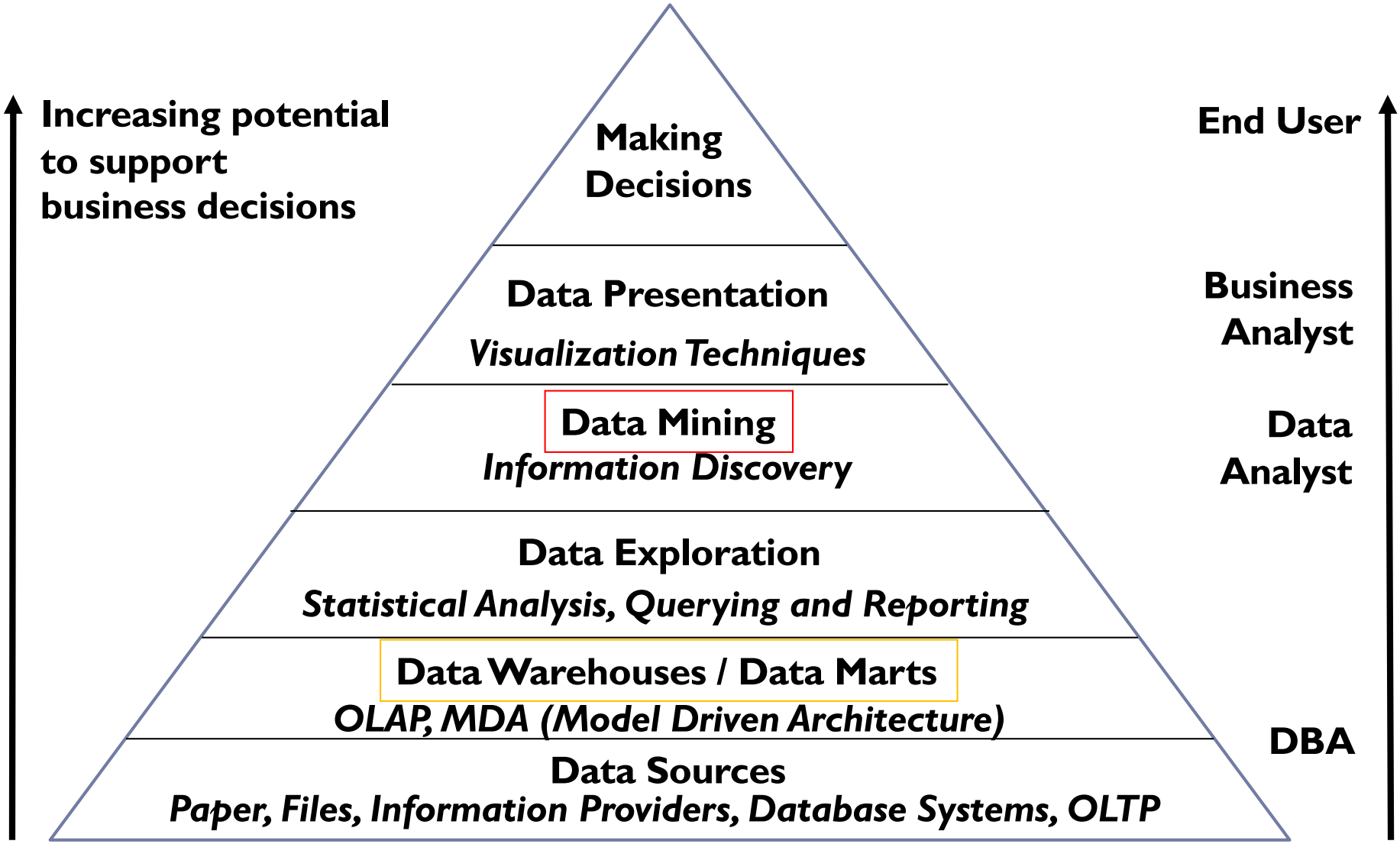
Comparison

- Heuristic query optimization
  - Sequence of single query plans
  - Each plan is (presumably) more efficient than the previous
  - Search is linear
- Cost-based query optimization
  - Many query plans generated
  - The cost of each is estimated, with the most efficient chosen

- Heuristic optimization is more efficient to generate, but may not yield the optimal query evaluation plan
- Cost-based optimization relies on statistics gathered on the relations

ira@cs.au.dk

Data Mining and Business Intelligence



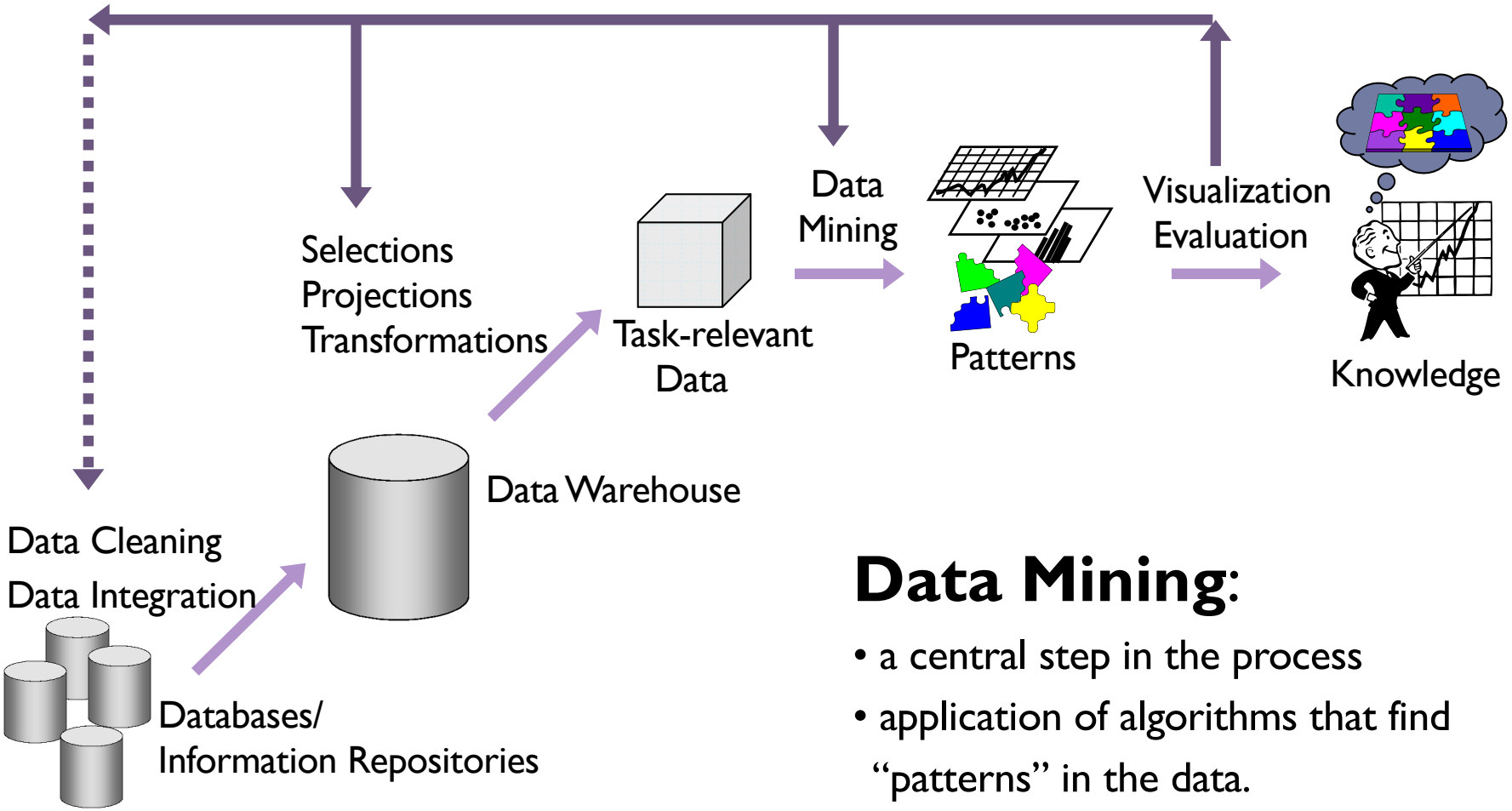
ira@cs.au.dk

Definitions of Data Mining

- Discovery of new information in terms of patterns or rules from vast amounts of data
- Process of finding interesting structure in data
- Process of employing one or more computer learning techniques to automatically analyze and extract knowledge from data
- Data mining may generate thousands of patterns: not all interesting
  - Pattern is interesting if easily understood by humans, valid on new or test data with some degree of certainty, potentially useful, novel, or validates some hypothesis that a user seeks to confirm
- Objective vs. subjective interestingness measures
  - Objective: based on statistics and structures of patterns, e.g., support, confidence, etc.
  - Subjective: based on user's belief in the data, e.g., unexpectedness, novelty, actionability, etc.

ira@cs.au.dk

KDD (knowledge discovery in databases)



Data Mining:

- a central step in the process
- application of algorithms that find "patterns" in the data.

ira@cs.au.dk

Types of Discovered Knowledge

- Association Rules
- Classification Models and Predictions
- Sequential Patterns such as trends, motifs
- Clustering: groups of related objects
- …
- Applications
  - Marketing
    - Marketing strategies and consumer behavior
  - Finance
    - Fraud detection, creditworthiness and investment analysis
  - Manufacturing
    - Resource optimization
  - Health
    - Image analysis, side effects of drug, and treatment effectiveness

ira@cs.au.dk

Association Rule Mining

- Example: Basket Data Analysis
- Transaction database
  - {butter, bread, milk, sugar}
  - {butter, flour, milk, sugar}
  - {butter, eggs, milk, salt}
  - {eggs}
  - {butter, flour, milk, salt, sugar}
- Which items are bought together frequently?
  - Association rule
    - Butter $\Rightarrow$ Milk
- Applications
  - Improved store layout
  - Cross marketing
  - Focused attached mailings / add-on sales

ira@cs.au.dk

Which rule is best given the database?

1. Eggs ⇒ Salt
2. Butter ⇒ Salt
3. Sugar ⇒ Butter
4. Bread ⇒ Sugar

{butter, bread, milk, sugar}
{butter, flour, milk, sugar}
{butter, eggs, milk, salt}
{eggs}
{butter, flour, milk, salt, sugar}

ira@cs.au.dk

Rule Measures: Support and Confidence

- We would like to express how interesting rules are
  - Support: percentage of transactions that contain all items in the rule (prevalence)
    - Can be understood as the probability of that a transactions contains all rule items
  - Confidence: fraction of transactions containing right hand side (RHS) items of the rule to transactions containing left hand side (LHS) items of the rule (strength)
    - Can be understood as the conditional probability given the left hand side items that the right hand side items are contained as well
  - E.g. butter $\Rightarrow$ milk: LHS = {butter}, RHS = {milk}
  - Eggs $\Rightarrow$ Salt: support 1/5 =20%, confidence 1/2=50%
    - Eggs and salt occur together only in one transaction (out of 5 in total), out of those that contain eggs (LHS) – the 3rd and 4th, only half contain salt (RHS)
  - Butter $\Rightarrow$ Salt: support 2/5=40%, confidence 1/2=50%
  - Sugar $\Rightarrow$ Butter: support 3/5=60%, confidence 1=100%
  - Bread $\Rightarrow$ Sugar: support 1/5=20%, confidence 1=100%

1. {butter, bread, milk, sugar}
2. {butter, flour, milk, sugar}
3. {butter, eggs, milk, salt}
4. {eggs}
5. {butter, flour, milk, salt, sugar}

ira@cs.au.dk

Mining Frequent Itemsets

- We want to find all rules that exceed two user specified thresholds
  - Minimum support and minimum confidence
- We do so by finding **itemsets** (= sets of items) that are **frequent** (=exceed minimum support threshold **mins**) first
  - From these, we will later generate rules

- Naïve frequent itemset mining algorithm
  - count the frequency of for all possible subsets of I in the database
  - too expensive since there are 2^m such itemsets for |I| = m items

- The Apriori principle (monotonicity):

  Any subset of a frequent itemset must be frequent

  e.g. if {butter, milk} frequent, then {butter} must also be frequent

  - Use opposite to exclude possible sets: any set that is not frequent cannot be the subset of a frequent one (downward closure)

ira@cs.au.dk

The Apriori Algorithm

- Method based on the apriori principle
  - First count the 1-itemsets, then the 2-itemsets, then the 3-itemsets, and so on…
    - `variable` L_k`: frequent itemsets of size k`
    - So work in increasing length, i.e., increasing number of items in the set
    - 1-itemsets L_1 could be {butter}, {salt}, {sugar},
    - L_2 2-itemsets {butter, salt}, {butter, sugar},
    - L_3 …
  - When counting (k+1)-itemsets, only consider those (k+1)-itemsets where all subsets of length k found to be frequent in previous step
    - Else cannot be frequent
    - This is the use of the apriori principle!

ira@cs.au.dk

The Apriori Algorithm

- Order the items (typically use lexicographic order, i.e., alphabetic order)

  variable L_k: itemsets of size k

  variable C_k: candidate itemsets of size k
- Scan the database to produce L_1: 1-itemset
- repeat
  - Step 1: Join frequent itemsets to produce candidates of length increased by one
    - frequent (k – 1)-itemsets p and q are joined if they share the same first k – 2 items

      e.g. if k=3: {butter, salt} and {butter, sugar} joined to form {butter, salt, sugar}
  - Step 2: Prune candidate k-itemsets which contain a non-frequent (k – 1)-subset s, i.e., $s \notin L_{k-1}$
    - E.g. {salt, sugar} not frequent, then discard {butter, salt, sugar}
    - Scan the database to check if new candidates Ck are indeed frequent
- until no more self-joins possible

ira@cs.au.dk

Apriori Algorithm Example

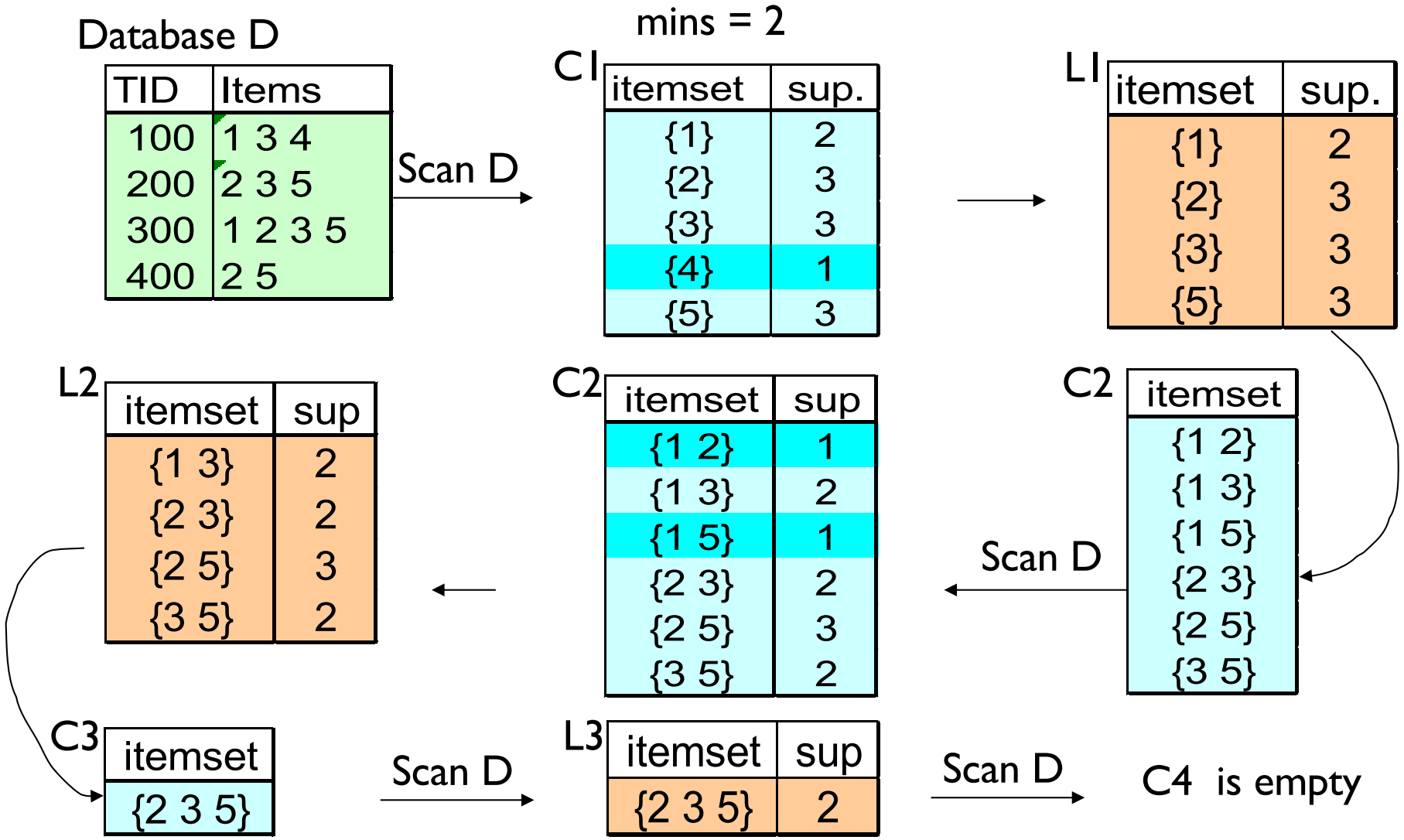
- Scan to produce L_1: 1-itemset
- repeat
  - Step 1: join frequent (k – 1)-itemsets if they share the same first k – 2 items
  - Step 2: Prune candidate k-itemsets which contain a non-frequent (k – 1)-subset
  - Scan the database to check if new candidates are indeed frequent
- until no more self-joins possible

Database D

TID	Items
100	1 3 4
200	2 3 5
300	1 2 3 5
400	2 5

ira@cs.au.dk

Apriori Algorithm Example



ira@cs.au.dk

Generating Rules from Frequent Itemsets

- For each frequent itemset X
  - For each subset A of X, form a rule A $\Rightarrow$ (X – A)
  - Delete those rules that do not have minimum confidence
- Computation of the confidence of a rule A $\Rightarrow$ (X – A)

$$confidence(A \Rightarrow (X - A)) = \frac{support(X)}{support(A)}$$

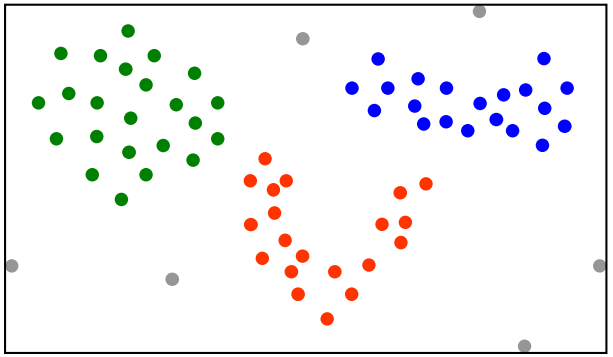
- Example: X = {A, B, C}, minConf = 60%
  - conf (A $\Rightarrow$ B, C) = 1; conf (B, C $\Rightarrow$ A) = 1/2
  - conf (B $\Rightarrow$ A, C) = 1/2; conf (A, C $\Rightarrow$ B) = 1
  - conf (C $\Rightarrow$ A, B) = 2/5; conf (A, B $\Rightarrow$ C) = 2/3

itemset	support
{A}	2
{B}	4
{C}	5
{A, B}	3
{A, C}	2
{B, C}	4
{A, B, C}	2

ira@cs.au.dk

Clustering

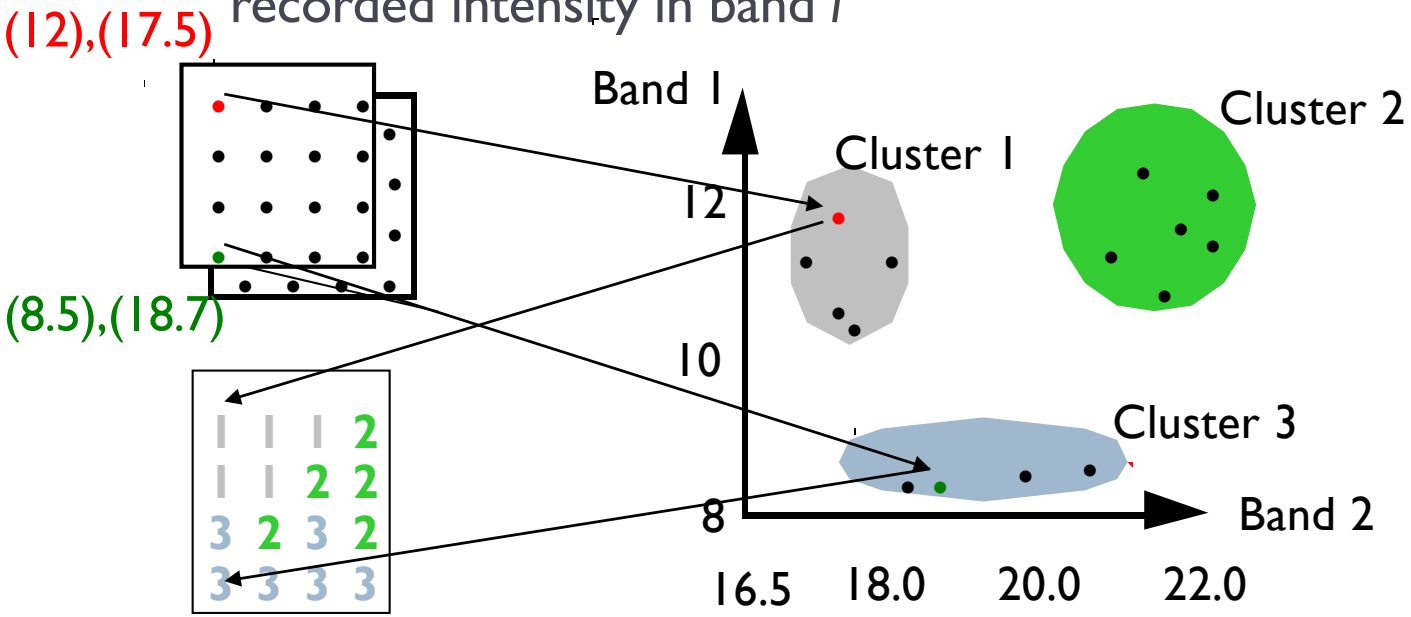
- Class labels are unknown:
  Group objects into sub-groups (clusters)
  - Similarity function (or dissimilarity fct. = distance) to measure similarity between objects
  - Objective: "maximize" intra-class similarity and "minimize" interclass similarity
- Clustering = **unsupervised classification** (no predefined classes)
- Typical usage
  - As a stand-alone tool to get insight into data distribution
  - As a preprocessing step for other algorithms
- Applications
  - Customer profiling/segmentation
  - Document or image collections
  - Web access patterns
  - …

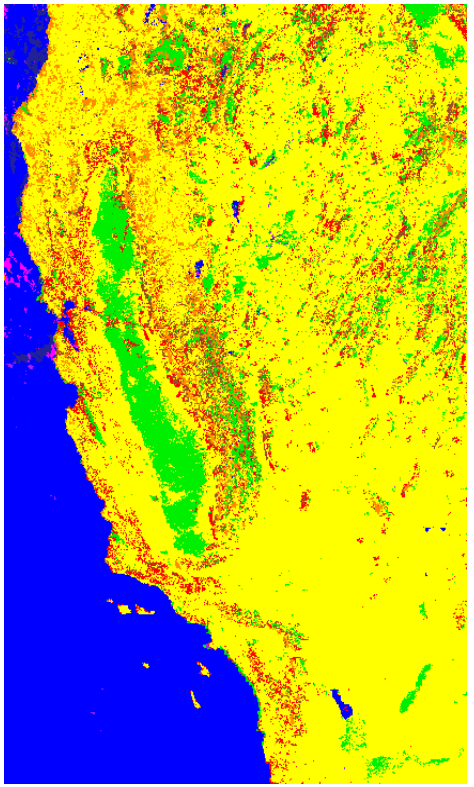


ira@cs.au.dk

A Typical Application: Thematic Maps

- Satellite images of a region in different wavelengths
  - Different land-uses reflect and emit light of different wavelengths in characteristic way
  - Each point on surface $p = (x_1, \ldots, x_d)$ has d values x_i of recorded intensity in band i

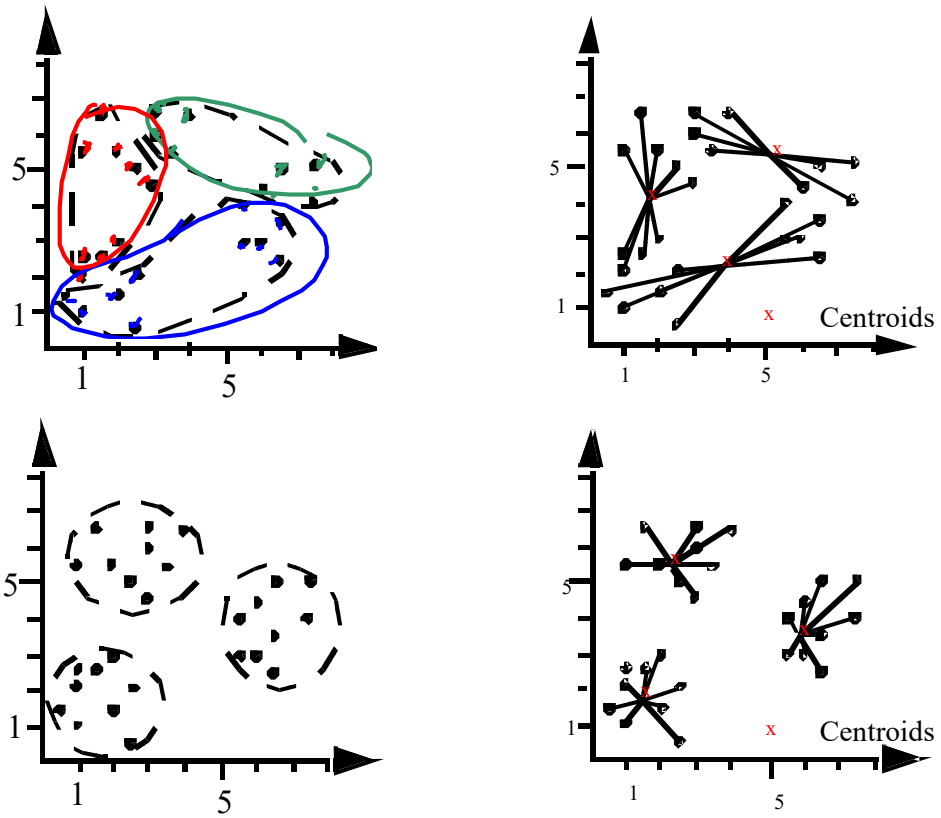




ira@cs.au.dk

K-Means Clustering: Basic Idea

- Objective: For a given *k*, form *k* groups so that the sum of the (squared) distances between the mean of the groups and their elements is minimal.
  - Poor Clustering
  - Optimal Clustering



ira@cs.au.dk

K-Means Clustering: Algorithm

Given *k*, the *k*-means algorithm is implemented in 4 steps:

1. Initialize: partition the objects into *k* nonempty subsets
2. Compute the centroids of the clusters of the current partition. The centroid is the center (mean point) of the cluster.
3. Assign each object to the cluster with the nearest representative.
4. Go back to Step 2, stop when representatives do not change.

- Note: we can also initialize by picking k random means first, then assign points to initial partitions based on these random means

ira@cs.au.dk

K-Means Clustering: Basic Notions

- Objects x are points in a d-dimensional vector space
- Centroid μ_C: mean of all points in a cluster C $\quad \mu_C = \frac{1}{|C|} \sum_{x_i \in C} x_i$
- Measure for the compactness („Total Distance“) of a **cluster** C_j:

$$TD(C_j) = \sqrt{\sum_{x \in C_j} dist(x, \mu_{C_j})^2}$$

- Measure for the compactness of a **clustering**

$$TD = \sqrt{\sum_{j=1}^{k} TD^2(C_j)}$$

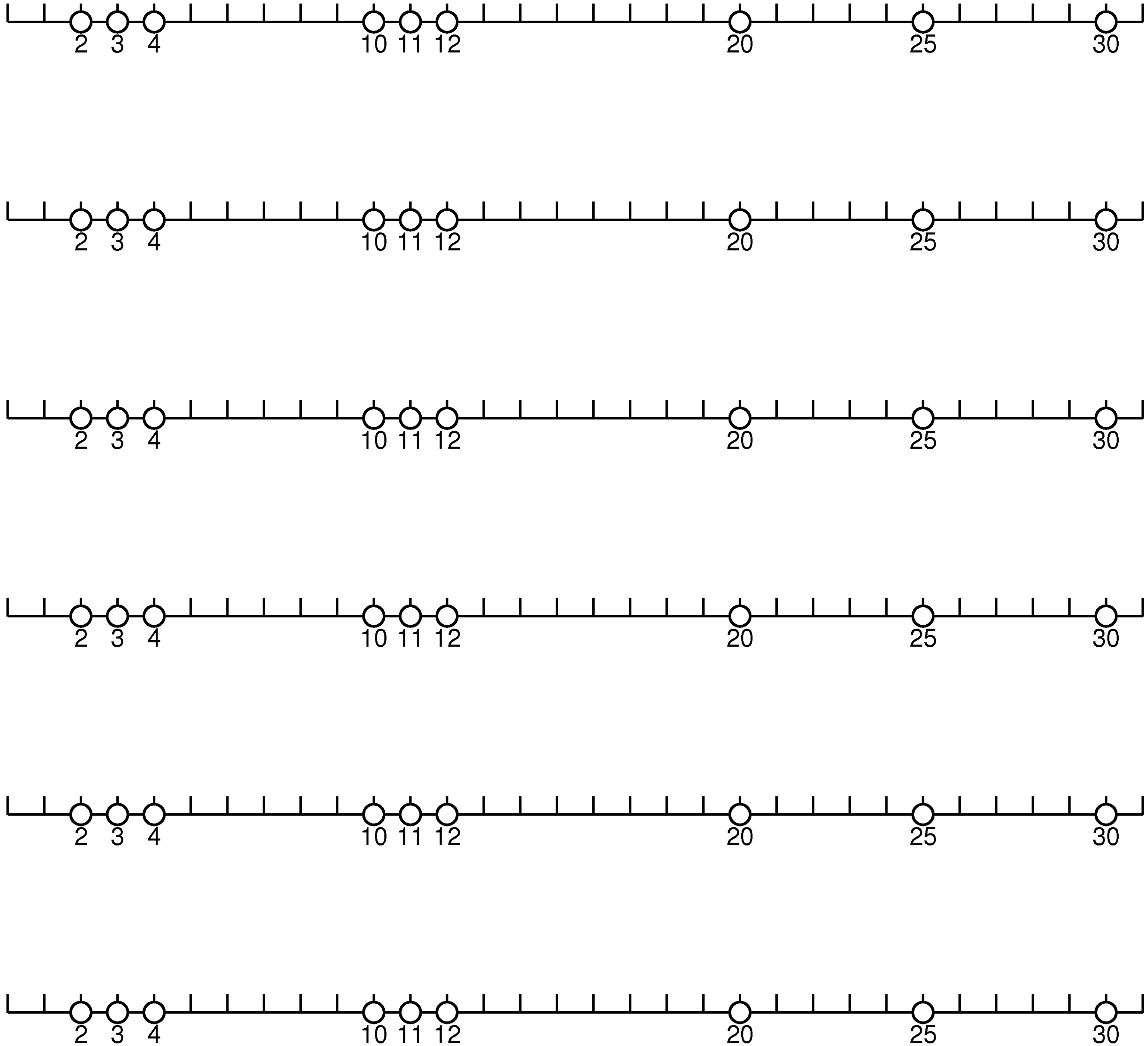
ira@cs.au.dk

What is a good k-means clustering?

A. TD is low.
B. TD is high.
C. The centroid is stable.
D. The centroid is equally far from all points in the cluster.

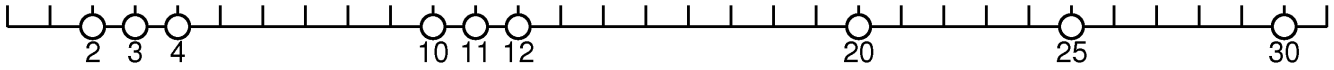
ira@cs.au.dk

K-Means example in one dimension

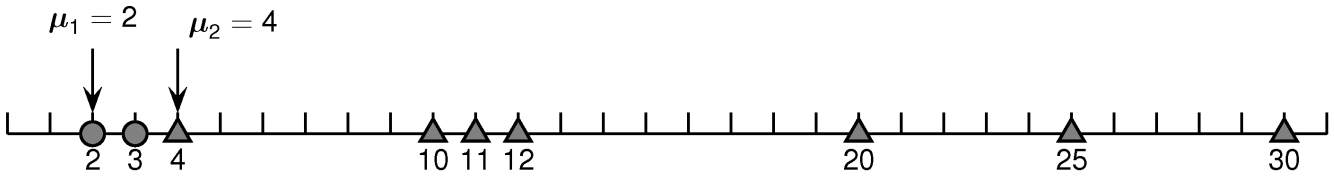


ira@cs.au.dk

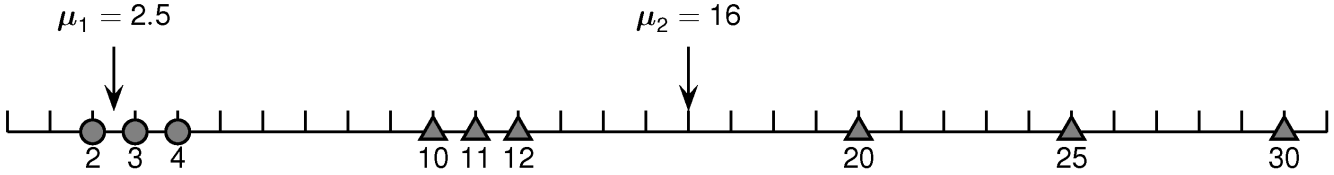
K-Means example in one dimension



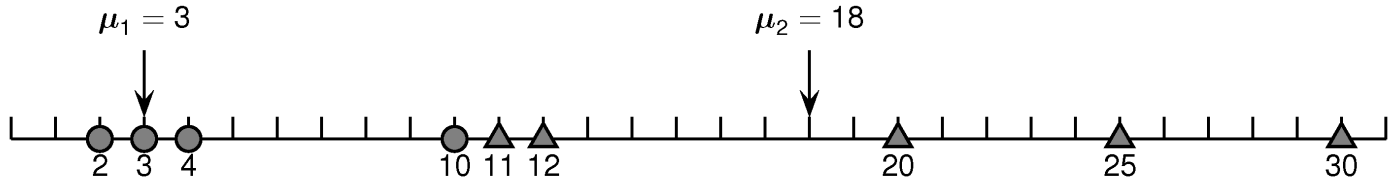
(a) Initial dataset



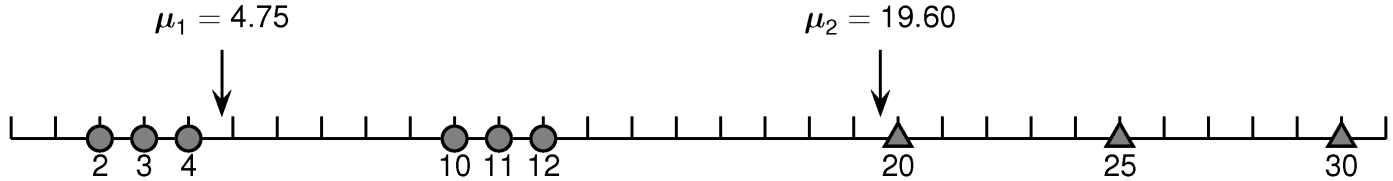
(b) Iteration: $t = 1$



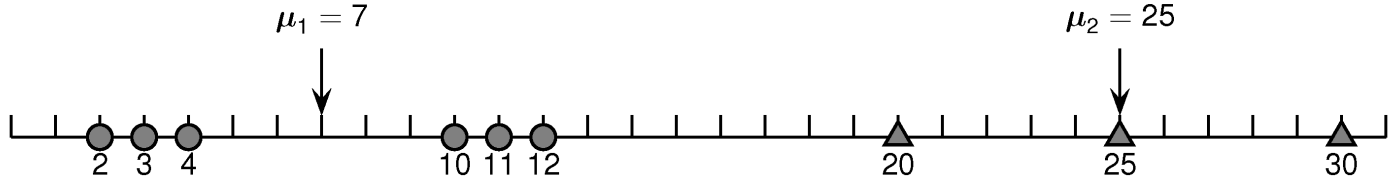
(c) Iteration: $t = 2$



(d) Iteration: $t = 3$



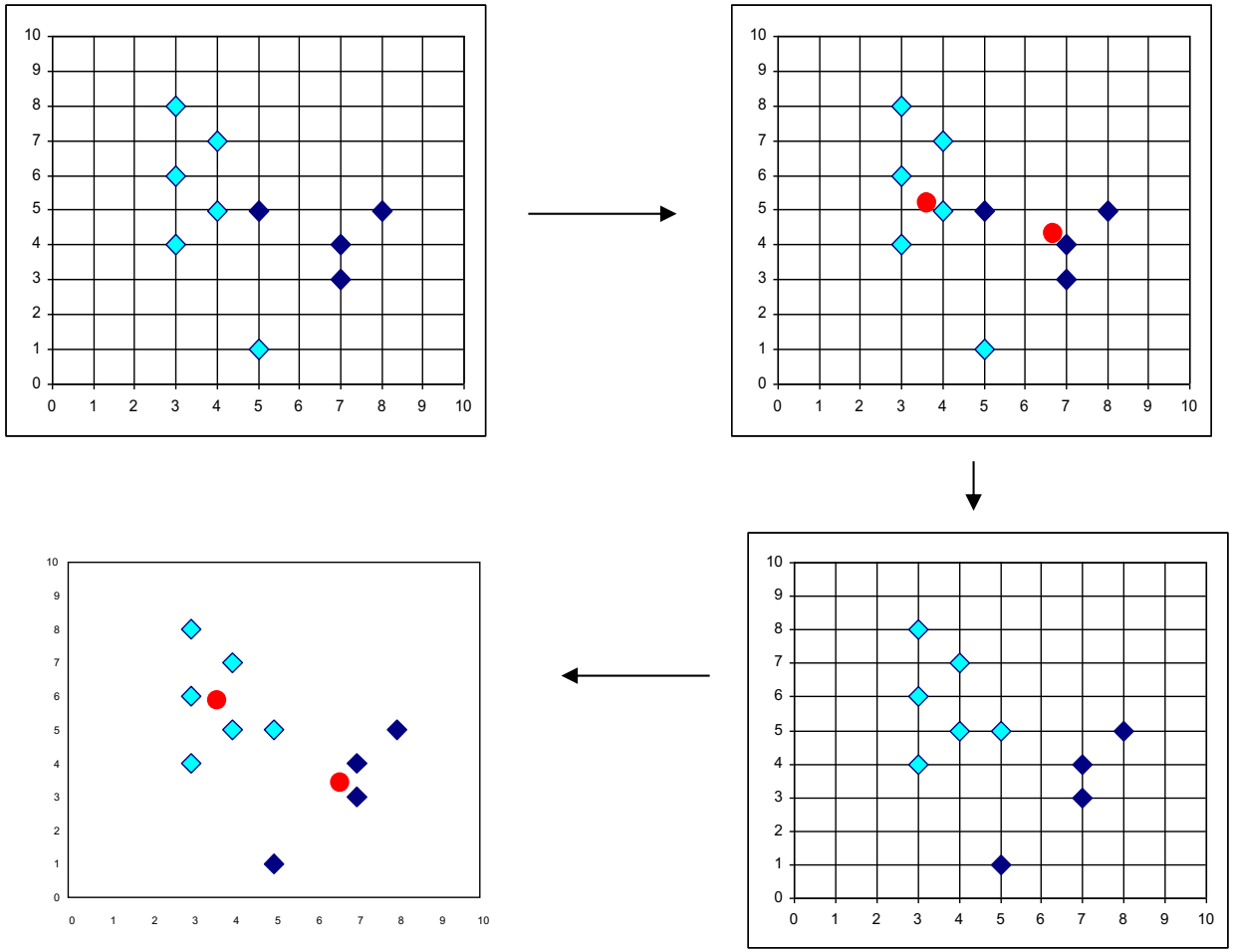
(e) Iteration: $t = 4$



(f) Iteration: $t = 5$ (converged)

ira@cs.au.dk

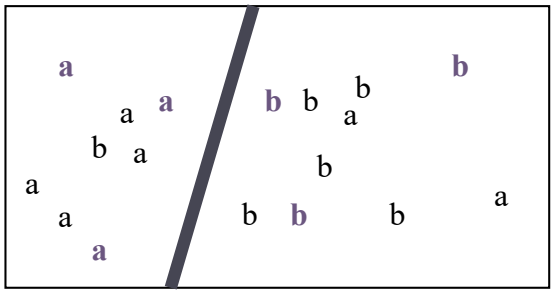
K-Means Clustering Example in 2 Dimensions



ira@cs.au.dk

Classification

- Learning a model able to describe different classes of data
- Supervised as the classes to be learned are predetermined
- Class labels are known for a small set of "training data": Find models/functions/rules (based on attribute values of the training examples) that
  - describe and distinguish classes
  - predict class membership for "new" objects

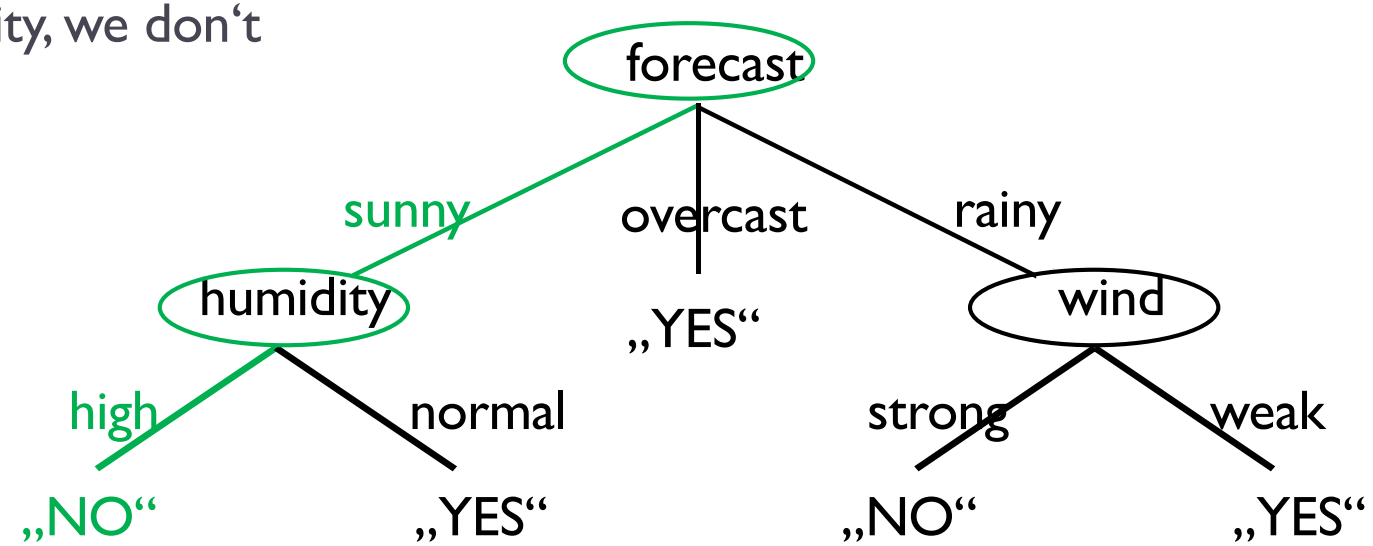


- Applications
  - Classify gene expression values for tissue samples to predict disease type and suggest best possible treatment
  - Automatic assignment of categories to large sets of newly observed celestial objects
  - Predict unknown or missing values (→ KDD pre-processing step)
  - …

ira@cs.au.dk

Decision Tree Classifiers

- Decision Tree
  - Flow-chart like graphical representation
  - Intuitive and interpretable classification model
- Structure
  - Nodes are attributes, branches are outcomes, leaves are class predictions
- Example: are we going to play tennis?
  - Let‘s check the weather: if today‘s forecast is sunny, and we have high humidity, we don‘t

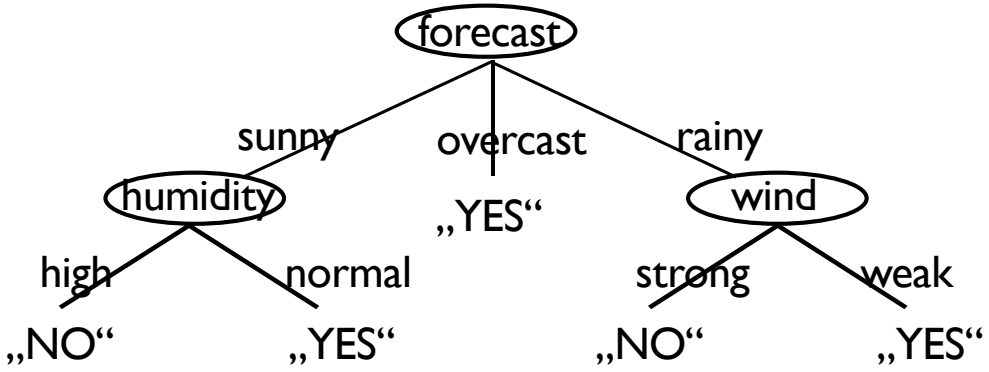


ira@cs.au.dk

Learning Decision Tree Classifiers

- Idea: make use of **training data**
  - „historical“ data for which we know the outcome (play tennis NO/YES)
  - Many records, each of which has attribute values for a number of attributes and a value for the class (target attribute: tennis)
  - To build the tree:
    - Start top-down from the root
      - Group data according to values for different attributes
        - E.g. forecast sunny/overcast/rainy gives two groups, wind strong/weak gives two groups
- Training data set:

day	forecast	temperature	humidity	wind	tennis
1	sunny	hot	high	weak	no
2	sunny	hot	high	strong	no
3	overcast	hot	high	weak	yes
4	rainy	mild	high	weak	yes
5	rainy	cool	normal	weak	yes
6	rainy	cool	normal	strong	no
7	. . .	. . .	. . .	. . .	. . .



ira@cs.au.dk

Which attribute should be root?

A. The one with the most frequent attribute value
B. The one that has the attribute value with the purest class label
C. The one that has a value distribution to balance the class label distribution
D. The one that has a value distribution to separate the class labels

ira@cs.au.dk

Building Decision Trees

- Tree is created top-down
  - We greedily try to reduce our uncertainty about the class outcome (YES/NO)
    - Root is decision on the attribute that helps most in this regard
  - Training examples T recursively partitioned into $T_1, T_2, \ldots, T_m$
    - Entropy for k classes with frequencies p_i (information theory: measure of uncertainty)

$$entropy(T) = -\sum_{i=1}^{k} p_i \cdot log_2 p_i$$

    - Compute the information gain of a split using an attribute, such as humidity, by comparing the entropy before the split with the entropy of the split
    - Always pick the split that reduces uncertainty most
      - highest information gain

$$information\ gain(T, A) = entropy(T) - \sum_{i=1}^{m} \frac{|T_i|}{|T|} \cdot entropy(T_i)$$

ira@cs.au.dk

Example: building Decision Trees

$$entropy(T) = -\sum_{i=1}^{k} p_i \cdot log_2 p_i$$

$$information\ gain(T, A) = entropy(T) - \sum_{i=1}^{m} \frac{|T_i|}{|T|} \cdot entropy(T_i)$$

Day	Outlook	Temp	Humid	Wind	Play?
1	Sunny	Hot	High	Weak	No
2	Sunny	Hot	High	Strong	No
3	Overcast	Hot	High	Weak	Yes
4	Rain	Mild	High	Weak	Yes
5	Rain	Cool	Normal	Weak	Yes
6	Rain	Cool	Normal	Strong	No
7	Overcast	Cool	Normal	Strong	Yes
8	Sunny	Mild	High	Weak	No
9	Sunny	Cool	Normal	Weak	Yes
10	Rain	Mild	Normal	Weak	Yes
11	Sunny	Mild	Normal	Strong	Yes
12	Overcast	Mild	High	Strong	Yes
13	Overcast	Hot	Normal	Weak	Yes
14	Rain	Mild	High	Strong	No

ira@cs.au.dk

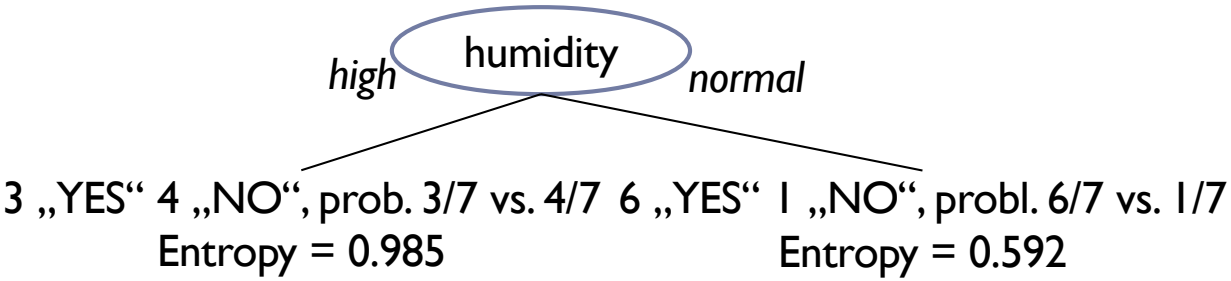
Example: building Decision Trees

$$entropy(T) = -\sum_{i=1}^{k} p_i \cdot log_2 p_i$$

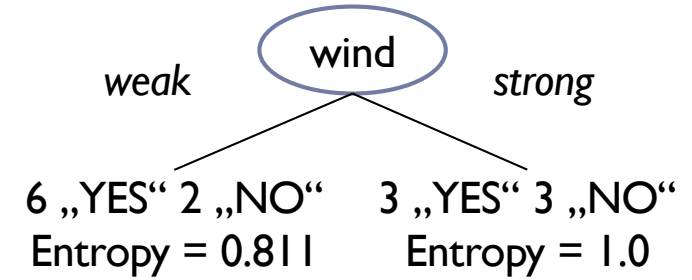
- Start from the root: all training data
  - What is the uncertainty?
  - Entropy (T) uses number of samples in each class, 9 yes, 5 no
    - Entropy(T)=-((5/14)*log(5/14)+(9/14)*log(9/14))=0.940
  - Which possible splits can we do?
    - For each attribute compute entropy based on splits it generates
    - Compare resulting information gain for all four attributes
      - Pick the one with highest information gain
    - For example: humidity, wind
  - Assume for sake of example only, we pick humidity, then below humidity, we check all three remaining attributes to determine the best split – both for the "high" branch and for the "normal" branch outcome likely to differ

$$information\ gain(T, A) = entropy(T) - \sum_{i=1}^{m} \frac{|T_i|}{|T|} \cdot entropy(T_i)$$

Day	Outlook	Temp	Humid	Wind	Play?
1	Sunny	Hot	High	Weak	No
2	Sunny	Hot	High	Strong	No
3	Overcast	Hot	High	Weak	Yes
4	Rain	Mild	High	Weak	Yes
5	Rain	Cool	Normal	Weak	Yes
6	Rain	Cool	Normal	Strong	No
7	Overcast	Cool	Normal	Strong	Yes
8	Sunny	Mild	High	Weak	No
9	Sunny	Cool	Normal	Weak	Yes
10	Rain	Mild	Normal	Weak	Yes
11	Sunny	Mild	Normal	Strong	Yes
12	Overcast	Mild	High	Strong	Yes
13	Overcast	Hot	Normal	Weak	Yes
14	Rain	Mild	High	Strong	No



$$IG(T, hum) = 0.94 - \frac{7}{14} * 0.985 - \frac{7}{14} * 0.592 = 0.151$$



$$IG(T, wind) = 0.94 - \frac{8}{14} * 0.811 - \frac{6}{14} * 1.0 = 0.048$$

ira@cs.au.dk

Evaluation of Classifiers

- Classification Accuracy
  - Predict class label for each object o
  - Determine the fraction of correctly predicted class labels:

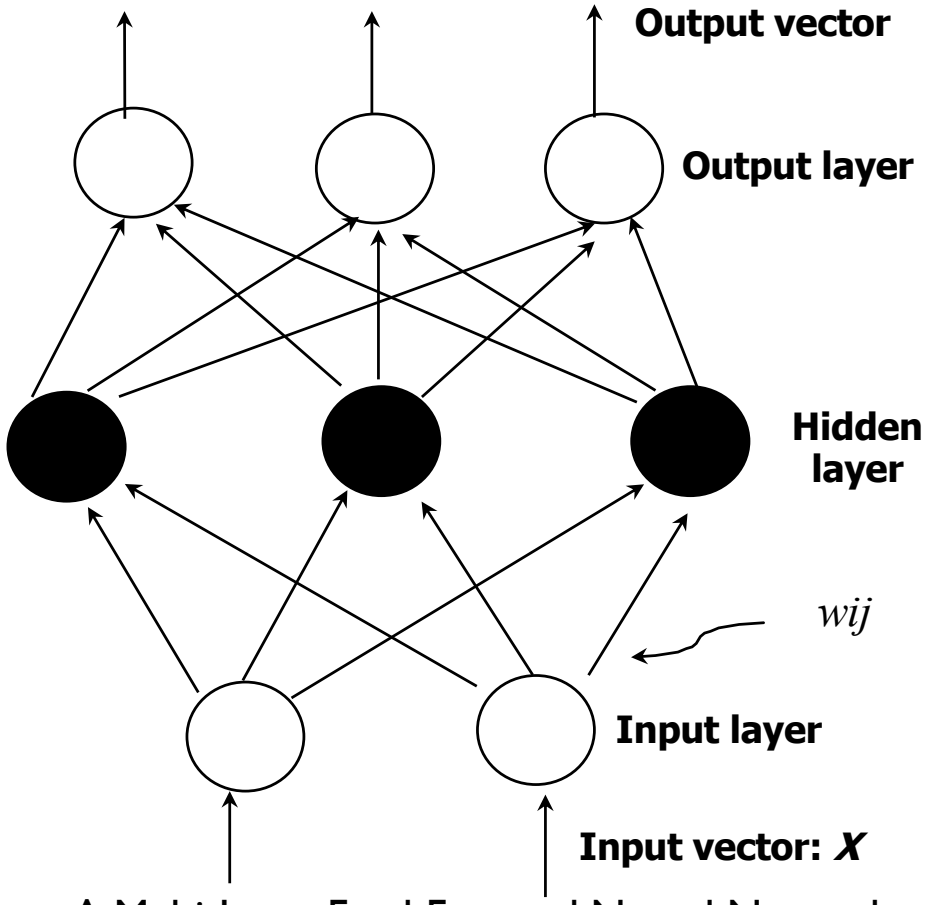
$$classification\ accuracy = \frac{count(\text{correctly predicted class label})}{count(o)}$$

  - Classification error = 1 – classification accuracy
- Overfitting: accuracy worse on entire data than on training data
  - bad quality of training data (noise, missing values, wrong values)
  - different statistical characteristics of training data and test data
- Train-and-Test: decomposition of data set into two partitions
  - Training data to train the classifier
    - Model construction using information also class labels
  - Test data to evaluate the classifier
    - temporarily hide class labels, predict them and compare

ira@cs.au.dk

Neural Networks

- Neural network: set of interconnected nodes inspired by human brain (not model of brain!)
  - Nodes in one layer connected to nodes in next layer
  - Node connections have weights
- Used for supervised learning and unsupervised clustering
- Recently dramatic improvements in performance
  - Big Data: lots of training data
  - New training methods and network architectures
- The output of a neural network is quantitative and not easily understood



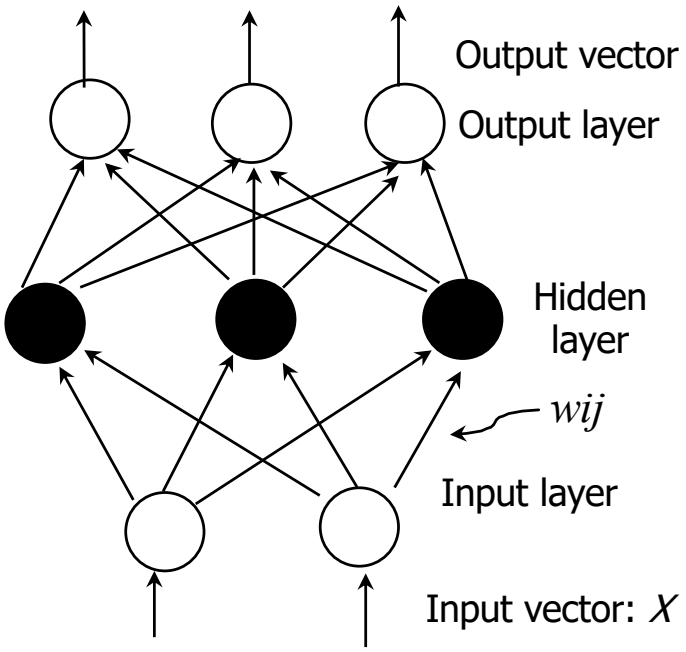
A Multi-Layer Feed-Forward Neural Network

ira@cs.au.dk

How A Multi-Layer Neural Network Works

- The **inputs** to the network correspond to the attributes measured for each training tuple
  - Inputs are fed simultaneously into the units making up the **input layer**
  - They are then weighted and fed simultaneously to frist **hidden layer, then to second etc**
- The number of hidden layers is arbitrary, many layers: deep learning
- The weighted outputs of the last hidden layer are input to units making up the **output layer**, which emits the network's prediction
- The network is **feed-forward**: None of the weights cycles back to an input unit or to an output unit of a previous layer
- From a statistical point of view, networks perform **nonlinear regression**: Given enough hidden units and enough training samples, they can closely approximate any function

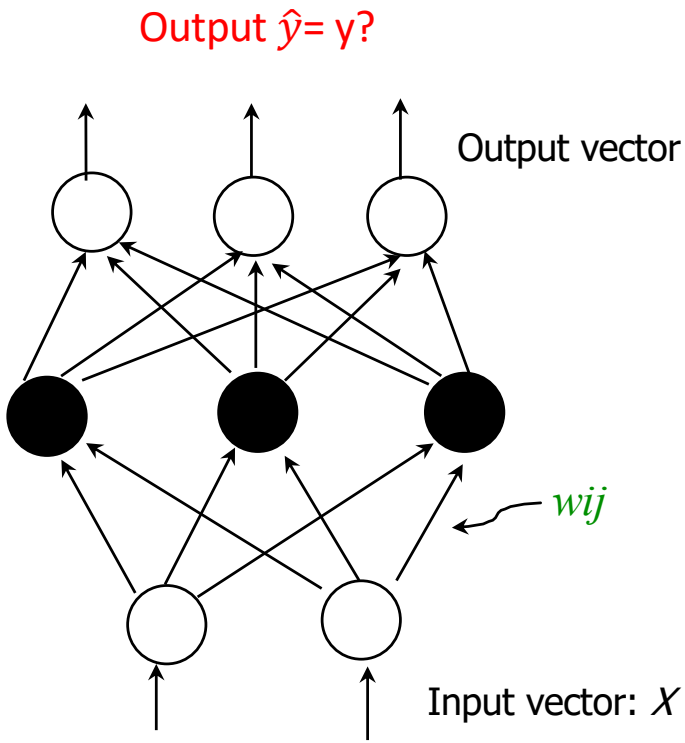
$$w_j^{(k+1)} = w_j^{(k)} + \lambda(y_i - \hat{y}_i^{(k)})x_{ij}$$



ira@cs.au.dk

Training neural networks

- Iteratively process a set of training tuples and compare the network's prediction with the actual known target value
- For each training tuple, the weights are modified in the opposite direction of the error (gradient descent)
- minimize a **loss function** (error) between the network's prediction and the actual target
  - Use backpropagation to compute gradient on loss function



ira@cs.au.dk

Why are NN popular now?

- Deep learning
  - Artificial neural networks with many layers
  - Can learn very complex functions
  - Impressive performance on some tasks
    - E.g. object recognition: better than "average human"
- Big Data
  - We have enough data to train large / complex networks
- Modern hardware
  - We have the computational power to train large networks with lots of data
- New architectures and training algorithms
  - We have new modules (ways of connecting nodes) for specific learning goals
  - We have efficient and effective training algorithms

Further Data Mining / Machine Learning Methods

- Sequential pattern analysis
  - find **subsequences** from given set of sequences that exceed minimum support
    - Sequence $S_1, S_2, S_3, ..$ customer purchasing itemset S_1 likely to buy S_2, then $S_3, \ldots$
    - Temporal order relevant: buy baby milk, then buy children's food; not so much the other way around
- Time Series Analysis
  - Identify the price trends of a stock or mutual fund
  - Generally temporal trends of values
- Regression
  - Regression equation estimates dependent numeric variable Y using set of independent numeric variables x_i and set of constants: $Y=f(x_1,x_2,\ldots,x_n)$
    - Linear regression: f linear in domain variables x_i,

- Machine Learning course (Bachelor, 3rd year)
- Cluster Analysis (Master)
- Data Mining course (Master)
- Advanced Data Management and Analysis course (Master)
- Computational Learning Theory (Master)

ira@cs.au.dk

What is supervised learning?

A. Clustering based on total distance.
B. Association rule mining with fixed minimum support.
C. Model creation based on class label information.
D. Learning with user input.

ira@cs.au.dk

Intended learning outcomes

- Be able to
  - Describe the goals and applications of common data mining approaches
  - Discuss the basic steps in k-means clustering, decision tree classification, and association rule mining

ira@cs.au.dk

What was this all about?

Guidelines for your own review of today's session

- In data mining, the goal is to…
  - The KDD process involves…
- Clustering is also called…
  - The learning goal is to…
- Classification is also called…
  - The learning goal is to…
  - Overfitting is the problem of… and can be addressed using…
- Association rule mining tries to…
  - The general idea in apriori makes use of…
- Other data mining tasks are…

ira@cs.au.dk