Intelligent Data Mining Techniques
(tutorial presented at ANNIE’2003)

Iveta Mrázová
Department of Software Engineering
Charles University Prague
Content outline

- Intelligent Data Mining: introduction and overview of Intelligent Data Mining Techniques (20 min)
- Selected Data Mining Techniques: principles and examples
  - undirected DM-techniques:
    - Market Basket Analysis (MBA) - (20 min)
    - Link Analysis and Scale-Free Networks (10 min)
    - Automatic Cluster Detection and Fuzzy Systems: Clustering the World Bank Data (20 min)
  - directed DM-techniques:
    - Internal Knowledge Representation in BP-Networks (20 min)
    - Modular Networks, Sensitivity Analysis and Feature Selection (20 min)
    - Neural Networks and Decision Trees: Students’ Questionnaire (20 min)
    - Genetic Algorithms and BP-networks: Generating Melodies (10 min)
- Conclusions, Questions + Answers (10 min)
Intelligent Data Mining: References

- M. J. A. Berry, G. Linoff: *Mastering Data Mining*, John Wiley & Sons, 2000
- J. Han, M. Kamber: *Data Mining: Concepts and Techniques*, Morgan Kaufmann Publishers, 2001
- D. Pyle: *Data Preparation for Data Mining*, Morgan Kaufmann Press, 1999
  
  http://www.mkp.com/datamining
  
  http://www.cs.waikato.ac.nz/ml/weka
What is Data Mining?

- discovering patterns in data
  - discovered patterns should be meaningful
  - should lead to some advantage, e.g. economic, ...
  - allows to make non-trivial predictions on new data
- the data is present in substantial quantities
- automatic or semi-automatic process
- two extremes for the form of discovered patterns:
  - black box - e.g. neural networks
  - transparent box - more structured, capture the decision structure in an explicit way
Building models for the data

- Classification model:
  - assigns an existing classification to new records
- Predictive model
  - Time-series model
- Clustering model
Data Analysis:
Influence of other disciplines

- statistics
- sampling
- regression analysis
  - linear regression
- correlation analysis
- memory-based reasoning
- link analysis
- genetic algorithms and neural networks

→ interpret observations
→ reduce the size of data
→ inter- and extrapolate observations
→ fit a line to observed data
→ mutual occurrence of observations
→ directly from AI
→ graph theory
→ model biological processes
Intelligent DM-Techniques: an overview

- Market Basket Analysis (MBA)
- Memory-Based Reasoning (MBR)
- Automatic Cluster Detection
- Fuzzy Systems (FS)
- Link Analysis
- Decision Trees
- Artificial Neural Networks (ANN)
- Genetic Algorithms (GA)
Market Basket Analysis (MBA)

Analyses in the retail industry:

What items occur together in a “basket”?

Results:
- expressed as rules
- highly actionable

Applications:
- planning store layouts
- offering coupons, limiting specials
- bundling products
Memory-Based Reasoning (MBR)

*Look for the nearest “known” neighbor to classify or predict value!*

- applicable to virtually any data
- new instances learned by adding them to the data set
- distance to neighbors estimates the correctness of the results

**Key elements in MBR:**
- *distance function* - to find nearest neighbors
- *combination function* - combine values at nearest neighbors to classify or predict
Link Analysis

▲ Goals:
- **find patterns in relationships between records**
- **visualize the links**

▲ Application areas:
- telecommunications
- law enforcement - clues about crimes are linked together to solve them
- marketing - relationships between customers
Goal:

*Find previously unknown similarities in the data!*

- Build models that find data records similar to each other
- Good as an initial analysis of the data
- Undirected data mining
Decision Trees and Rule Induction

Divide the data into disjoint subsets characterized by simple rules!

- Directed data mining (classification)
- Explainable rules applicable directly to new records

Techniques:
- Classification And Regression Trees (CART)
- Chi-squared Automatic Induction (CHAID)
- C4.5
Artificial Neural Networks (ANN)

Detect patterns in the data in a way “similar” to human thinking!

- Directed data mining (classification and prediction)
- Applicable also to undirected data mining (SOMs)
- Two major drawbacks:
  - difficulty in understanding the models they produce
  - sensitivity to the format of incoming data
Genetic Algorithms (GA)

*Apply genetics and natural selection to find optimal parameters of a predictive function!*

- GA use “genetic” operators to evolve successive generations of solutions:
  - selection
  - crossover
  - mutation
- Best candidates “survive” to further generations until convergence is achieved
- Directed data mining
On-Line Analytic Processing (OLAP)

- an important tool for extracting and presenting information
- facilitates understanding of the data and important patterns inside it
- a way of presenting relational data to users
- multi-dimensional databases (MDDs):
  - a representation of data
  - allows users to drill down into the data and understand various important summarizations
Market Basket Analysis (MBA)

- Analyses in the retail industry:
  - What items occur together in a “basket”?

- Results:
  - expressed as rules
  - highly actionable

- Applications:
  - planning store layouts
  - offering coupons, limiting specials
  - bundling products
Association rules

**How do the products relate one to each other?**

- Association rules should be:
  - *easy to understand*: once the pattern is found, it is easy to justify it
  - *useful*: contain actionable information leading to other interventions

- Association rules should not be:
  - *trivial*: results are already known by anyone familiar with the business
  - *inexplicable*: seem to have no explanation and do not suggest any action
MBA to compare stores

- Virtual items:
  - specify which group the transaction comes from
  - do not correspond to a product or service

- Comparison between new and existing stores:
  1. Gather data for a specific period from store openings
  2. Gather about the same amount of data from existing stores
  3. Apply MBA to find association rules in each set
  4. Consider especially association rules containing the virtual items
MBA - how does it work?

- **Items** - products or service offerings
- **Transactions** contain one or more **items**
- **Co-occurrence table**
  - indicates the number of times that any two *items co-occur* in a *transaction* (i.e. these products were purchased together)
  - values along the diagonal represent the *number of transactions* containing just that one item
MBA - example

- Grocery transactions:

<table>
<thead>
<tr>
<th>Customer</th>
<th>Items</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>bread, butter</td>
</tr>
<tr>
<td>2</td>
<td>milk, bread, butter</td>
</tr>
<tr>
<td>3</td>
<td>bread, coffee</td>
</tr>
<tr>
<td>4</td>
<td>bread, butter, coffee</td>
</tr>
<tr>
<td>5</td>
<td>coffee, butter</td>
</tr>
</tbody>
</table>

- Co-occurrence of products:

<table>
<thead>
<tr>
<th></th>
<th>bread</th>
<th>butter</th>
<th>milk</th>
<th>coffee</th>
</tr>
</thead>
<tbody>
<tr>
<td>bread</td>
<td>4</td>
<td>3</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>butter</td>
<td>3</td>
<td>4</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>milk</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>coffee</td>
<td>2</td>
<td>2</td>
<td>0</td>
<td>3</td>
</tr>
</tbody>
</table>

Sales patterns apparent from the co-occurrence table:
Bread and butter are likely to be purchased together.
Milk is never purchased with coffee.
MBA - Association rules

Rule: \[ \text{IF } \text{Condition} \text{ THEN Result.} \]
\[ (\text{Rule}_r : \text{IF Item}_i \text{ THEN Item}_j .) \]

Questions:
- How good are the found association rules?
  - support
  - confidence
  - improvement
- How to find association rules automatically?
Support and confidence

Support: How frequently can the rule be applied?

\[
Support(Rule_r) = \frac{Nr\_of\_Transactions\_containing\_i\_and\_j}{Number\_of\_all\_Transactions} \cdot 100 \%
\]

Confidence: How much can we rely on the result of the rule?

\[
Confidence(Rule_r) = \frac{Nr\_of\_Transactions\_containing\_i\_and\_j}{Nr\_of\_Transactions\_containing\_i} \cdot 100 \%
\]
Support and confidence - example

Rule 1: If a customer purchases bread then the customer also purchases butter.

Rule 2: If a customer purchases coffee then the customer also purchases butter.

Support (Rule_1) = \( \frac{3}{5} \times 100\% = 60\% \)

Support (Rule_2) = \( \frac{2}{5} \times 100\% = 40\% \)

Confidence (Rule_1) = \( \frac{3}{4} \times 100\% = 75\% \)

Confidence (Rule_2) = \( \frac{2}{3} \times 100\% = 66\% \)
Improvement of a rule

**Improvement:** How much is a rule better at predicting the result than just assuming it?

$$\text{Improvement}(\text{Rule}_r) = \frac{p(i \_and\_j)}{p(i) \cdot p(j)}$$

**If Improvement < 1:**
- rule is worse at predicting the result than random chance
- NEGATING the result might produce a better rule

**IF Condition THEN NOT Result.**
Improvement of a rule - example

**Rule:** *If a customer purchases milk then the customer also purchases butter.*

\[
\text{Support (Rule}_1) = \frac{1}{5} \cdot 100\% = 20\%
\]

\[
\text{Confidence (Rule}_1) = \frac{1}{1} \cdot 100\% = 100\%
\]

\[
\text{Improvement (Rule}_1) = \left( \frac{1}{5} \right) / \left( \frac{1}{5} \cdot \frac{4}{5} \right) = \frac{5}{4} = 1.25
\]
Basic steps of MBA

- **Choose** the right set of **items** and the right level
- **Generate rules** by deciphering the co-occurrence matrix
  - calculate the probabilities and joint probabilities of items and their combinations in transactions
  - limit the search with thresholds set on support
- **Analyze probabilities to determine best rules**
  - overcome limits imposed by the number of items and their combinations in “interesting” transactions
MBA - the choice of right items

Gathering transaction data:
- often bad quality requiring extensive pre-processing
- items of interest may change over time
- the right level of detail:
  - a growing number of item combinations
  - actionable results (specific items)
  - rules with sufficient support (frequent occurrence in the data set)
**Taxonomies:** hierarchical categories

**MBA - Complexity of generated rules:**
- Use more general items initially
- Then, generate rules for more specific items using only transactions containing these items

**MBA - Actionable results:**
Items should occur in roughly the same number of transactions:
- roll up rare items to higher levels in the taxonomy (to become more frequent)
- keep more common items at lower levels (to prevents rules from being dominated by the most common items)
Virtual items: go beyond the taxonomy

- cross product boundaries of original items
  - e.g. designer labels - Calvin Klein
- may include information about the transaction itself
  - anonymous (day of week, time, etc.)
  - signed (info about customers and their behavior over time)
- might be a cause of redundant rules
  - items from the taxonomy are associated with just one virtual item ("If Coke product then Coke.")
  - virtual and generalized items appear together in a rule ("If Coke product and diet soda then pretzels" instead of "If diet coke then pretzels")
MBA - generating rules

- **Compute the co-occurrence table:**
  - provides the information about which combinations of items occur most commonly in the transactions
  - applicable for evaluating basic probabilities necessary to evaluate the importance of generated rules

- **Provide useful rules:**
  - improvement should be greater than 1
    - low improvement can be increased by negating the rules
    - negated rules might be less actionable than original rules
  - reduce the number of generated rules - **PRUNING**
Minimum support pruning

Eliminate less frequent items

- actions should *affect enough transactions*

- two possibilities:
  - eliminate rare items from consideration (then, eliminate their respective associative rules)
  - use taxonomy to generalize items (then, resulting generalized items should meet the threshold criteria)

- *variable minimum support* - a cascading effect
MBA - Dissociation rules

- **Rule:** \( \text{IF } A \text{ AND NOT } B \text{ THEN } C. \)
  - Introduce new items inverse to original ones
  - Each transaction will contain an inverse item if it does not contain the original one

- **Drawbacks:**
  - doubled number of items
  - growing size of transactions
  - inverse items tend to occur more frequently than original (leading to less actionable rules with all items inverted: “IF \text{ NOT } A \text{ AND NOT } B \text{ THEN NOT } C.”)
Time-series analysis with MBA

- **Analyze cause and effects:**
  - time- or sequencing information to determine when transactions occurred relative to each other
  - usually requires some way of identifying the customer

- **Conversions to an MBA-problem:**
  - include in transactions items before the event of interest (for *causes*) or after the event of interest (for *effects*); then, remove duplicate items from the transaction
  - *time-window:* a “snapshot” of all items that occur within a certain period (e.g. all transactions within a month)
    - trends for rare items
Strengths of MBA

- Produces clear and understandable results
  - *actionable IF - THEN - rules*

- Supports *undirected data mining*
  - important when approaching large data sets with no prior knowledge

- Works on *variable-length data*

- Computations are *easy to understand*
  - Computational costs grow exponentially with the number of items!
Weaknesses of MBA

- Exponentially growing computational costs
  - necessity for item taxonomies and virtual items
- Limited support for attributes on the data
  - pruning of less actionable general items
- Difficult to determine the right number of items
  - items should have approximately the same frequency
- Discounts rare items
  - variable thresholds for minimum support pruning
  - higher levels in item taxonomies
Link Analysis

- **Goals:**
  - find patterns in relationships between records
  - visualize the links

- **Application areas:**
  - telecommunications
  - law enforcement - clues about crimes are linked together to solve them
  - marketing - relationships between customers
Scale-Free Networks

- Some nodes have an extremely large number of links (edges) to other nodes - hubs
- Most nodes have just a few links to other nodes
- Robust against accidental failures
- Vulnerable to coordinated attacks
- New application areas
  - preventing computer viruses spreading through the Internet
  - medicine (vaccinations)
  - business (marketing)
Scale-Free Networks

A random graph

A scale-free network

Distribution of edges

Distribution of edges

nr. of nodes

number of edges

nr. of nodes

number of edges

adapted from “A. L. Barabasi and E. Bonabeau: Scale-Free Networks, Scientific American, May 2003”
Examples of Scale-Free Networks

- **Social networks**
  - research collaboration (scientists, co-authorship of papers)
  - Hollywood (actors, appearance in the same movie)

- **Biological networks**
  - cellular metabolism (molecules involved in energy production, participation in the same biological reaction)
  - protein regulatory network (proteins controlling cell activity, interactions among proteins)

- **Socio-technical networks**
  - Internet (routers, optical or other connections)
  - World Wide Web (Web-pages and URLs)
Scale-Free Networks: basic characteristics

- Two basic mechanisms:
  - growth
  - preferential attachment

- “The rich get richer” (hubs):
  - new nodes tend to connect to the more connected sites
  - “popular locations” acquire more links over time than less connected neighbors

- Reliability
  - accidental failures (80% of randomly selected nodes can fail without fragmenting the cluster)
  - coordinated attacks (eliminating 5-15% of all hubs can crash the system)
Scale-Free Networks

adapted from “A. L. Barabasi and E. Bonabeau: Scale-Free Networks, Scientific American, May 2003”
Implications of Scale-Free Networks

- **Computing**
  - networks with scale-free architectures

- **Medicine**
  - vaccination campaigns and new drugs

- **Business**
  - cascading financial failures
  - marketing
Implications of Scale-Free Networks

Computing

- computer networks with scale-free architectures (e.g. WWW)
  - highly resistant to accidental failures
  - very vulnerable to deliberate attacks and sabotage
- eradicating viruses from the Internet will be effectively impossible
Implications of Scale-Free Networks

**Medicine**

- Vaccination campaigns against serious viruses focused on hubs
  - People with many connections to others
  - Difficult to identify such people

- New drugs targeting the hub molecules involved in certain diseases

- Control the side-effects of drugs with maps of networks within cells
Implications of Scale-Free Networks

Business

- financial failures
  - understand how companies, industries and economies are inter-linked
  - monitor and avoid cascading financial failures

- marketing
  - study the spread of a contagion on a scale-free network
  - more efficient ways of propagating consumer buzz about new products
Goal:

*Find previously unknown similarities in the data!*

- Build models that find data records similar to each other
- Good as an initial analysis of the data
- Undirected data mining
Economies grouped according to their results

GDP growth rates

Cluster 1
Cluster 2
Cluster 3
Cluster 4
Cluster 5
Cluster 6
Cluster 7
Cluster 8
Cluster 9
projection

Cluster

0 0.2 0.4 0.6 0.8 1
0 0.2 0.4 0.6 0.8 1
gross national product
purchasing power parity
Mining the World Bank Data: the Fuzzy c-means Clustering Approach

with Cihan H. Dagli,
Engineering Management Department, University of Missouri - Rolla
FCM-clustering: introduction

- **World Development Indicators (WDI)**
  - published annually by the World Bank
  - reflect development process in the countries
  - incomplete and imprecise data

- **Previously applied techniques**
  - regression analysis - linear relationships
  - US-based grouping of countries (G. Ip, Wall Street Journal)
  - GDP-based grouping of economies (World Bank)
  - self-organizing feature maps (T. Kohonen, S. Kaski, G. Deboeck)
Poverty maps - T. Kohonen

- more neurons than countries
- only local geometric relations are important
- countries mapped close to each other have a similar state of development

Poverty maps - T. Kohonen, S. Kaski

U-matrix:
- illustrate “boundaries” between clusters
- represent average distances between neighboring neurons in a gray scale
  - small average distance ⇒ light shade
  - large average distance ⇒ dark shade

Our goal

- Cluster efficiently imprecise data
- Estimate the number of clusters
- Visualize the results
- Interpret the results
Our goal

- Cluster efficiently imprecise data
- Estimate the number of clusters
- Visualize the results
- Interpret the results

- **fuzzy c - means clustering (FCM)**
- **cluster validity indicators**
- **spread-sheet-like form**
- **find “landmarks”**
The objective function

- corresponds to the **weighted distance** between input patterns and cluster centers:

\[
J_s (U, v) = \sum_{p=1}^{P} \sum_{i=1}^{c} (u_{ip})^s \left[ \sum_{j=1}^{n} (x_{pj} - v_{ij})^2 \right]
\]

- membership degrees between 0 and 1: \(0 \leq u_{ip} \leq 1\)
- total membership of a pattern equals to 1: \(\forall p \mid \sum_{i=1}^{c} u_{ip} = 1\)
- no empty or full clusters: \(\forall i \mid 0 < \sum_{p=1}^{P} u_{ip} < P\)
Fuzzy $c$-means Clustering (FCM)

- **Step 1:** Initialize $c$, $s$, $\varepsilon$ and $t$. Choose randomly $U^{(0)}$.

- **Step 2:** Determine new fuzzy cluster centers:

  $$
  \bar{v}_i^{(t)} = \frac{1}{\sum_p (u_{ip}^{(t)})^s} \sum_p (u_{ip}^{(t)})^s \bar{x}_p
  $$

- **Step 3:** Calculate new partition matrix $U^{(t+1)}$:

  $$
  u_{ip}^{(t+1)} = \frac{\left(1 / \left\| \bar{x}_p - \bar{v}_i^{(t)} \right\|^2 \right)^{1 / s - 1}}{\sum_{k=1}^c \left(1 / \left\| \bar{x}_p - \bar{v}_k^{(t)} \right\|^2 \right)^{1 / s - 1}}
  $$

- **Step 4:** Evaluate $\Delta = \left\| U^{(t+1)} - U^{(t)} \right\| = \max_{i,p} \left| u_{ip}^{(t+1)} - u_{ip}^{(t)} \right|$

  If $\Delta > \varepsilon$ then set $t = t + 1$ and go to Step 2. If $\Delta \leq \varepsilon$ then Stop.

- **END of FCM**
Cluster validity criteria

**Partition coefficient:**

\[ F (U ; c ) = \frac{1}{P} \sum_{p=1}^{P} \sum_{i=1}^{c} (u_{ip})^2 \]

**Partition entropy:**

\[ H (U ; c ) = -\frac{1}{P} \sum_{p=1}^{P} \sum_{i=1}^{c} u_{ip} \ln(u_{ip}) ; \quad u_{ip} \ln(u_{ip}) = 0 \text{ for } u_{ip} = 0. \]

**Windham's proportion exponent:**

\[ W (U ; c ) = -\sum_{p=1}^{P} \ln \left[ \sum_{j=1}^{c} (-1)^{j+1} \binom{c}{j} (1 - j \cdot \mu_p)^{c-1} \right] ; \quad \mu_p = \max_{1 \leq i \leq c} \{ u_{ip} \} \]

Iveta Mrázová, ANNIE’03
How many clusters?

- **Partition coefficient:**
  \[
  \max_{2 \leq c \leq P-1} \left\{ \max_U \left[ F(U; c) \right] \right\}
  \]

- **Partition entropy:**
  \[
  \min_{2 \leq c \leq P-1} \left\{ \min_U \left[ H(U; c) \right] \right\}
  \]

- **Windham’s proportion exponent:**
  \[
  \max_{2 \leq c \leq P-1} \left\{ \max_U \left[ W(U; c) \right] \right\}
  \]
Supporting experiments - artificial data

Cluster validity indicators for artificial data

21 input patterns, \( s = 1.4, \ varepsilon = 0.05 \)

Fuzzy 4-partition of the data

‘\( \times \)’ indicates cluster centers, patterns from the same clusters are labeled identically
Supporting experiments - artificial data

Fuzzy 6-partition of the data

Fuzzy 8-partition of the data

‘×’ indicates cluster centers, patterns from the same clusters are labeled identically.

‘×’ indicates cluster centers, patterns from the same clusters are labeled identically.
Interpret the results!

Characteristic features for detected clusters:

- cluster centers - “fictive” patterns out of the data set
- “calibrate” clusters with the “most representative” patterns from the data set - based on just one pattern
- Determine outstanding properties for clusters:
  - compared to other properties within the cluster
  - compared to properties of other clusters
  - exception: “border areas”

fuzzy $c$-landmarks
Automatic landmark selection

**Fuzzy c-landmark for cluster** $i^*$: \(( j^*, v_{i^*j}^* )\)

- "fuzzy distance" from the cluster center should be small
- "fuzzy distance" from all other cluster centers should be large

\[
\min \sum_{p=1}^{P} u_{i^*p} \left| x_{pj} - v_{i^*j}^* \right| \\
\sum_{p=1}^{P} u_{i^*p} \\
\sum_{p=1}^{P} u_{i^*p} \left| x_{pj} - v_{ij} \right| \\
\min \sum_{i \neq i^*}^{1 \leq i \leq c} \sum_{p=1}^{P} u_{i^*p}
\]

Iveta Mrázová, ANNIE'03
Supporting experiments: The World Bank Data

- 99 state economies with 16 (latest) indicators for each country
- economical and social potential of countries and their citizens
- all indicators are relative to population
- element-wise transformation to (0,1) with:
  \[ x' = \frac{x - x_{\min}}{x_{\max} - x_{\min}} \quad \text{and} \quad x'' = \frac{1}{1 + e^{-k(x' - 1/2)}} \]
  
  minimum over all patterns
  maximum over all patterns

- the choice of other parameters \((k=4; s=1.4; \epsilon=0.05)\)
Used Development Indicators

- GNP per capita
- Purchasing Power Parity
- Growth rate of GDP per capita
- GDP implicit deflator
- External debt (% of GNP)
- Total debt service (% of export of goods and services)
- High technology exports (% of manufactured exports)
- Military expenditures (% of GNP)
- Expenditures for R&D (% of GNP)
- Total expenditures on health (% of GDP)
- Public expenditures on education (% of GNP)
- Male life expectancy at birth
- Fertility rates
- GINI-index (distribution of income/consumption)
- Internet hosts per 10000 people
- Mobile phones per 1000 people
Supporting experiments: the World Bank data

Cluster validity indicators for the WB-data

99 countries with 16 indicators

$s = 1.1, \varepsilon = 0.05$

Cluster validity indicators for the WB-data

99 countries with 16 indicators

$s = 1.4, \varepsilon = 0.05$
Fuzzy 7-partition of the WB data:

A part of the fuzzy 7-partition of the World Bank data:
99 countries with 16 indicators; $s = 1.4$, $\varepsilon = 0.05$
**Landmarks for the WB data**

“Representative patterns” and fuzzy 7-landmarks for the World Bank data:
99 countries with 16 indicators; \( s = 1.4, \varepsilon = 0.05 \)

<table>
<thead>
<tr>
<th>No.</th>
<th>Representant</th>
<th>1. char. feature</th>
<th>2. char. feature</th>
<th>3. char. feature</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Uzbekistan</td>
<td>GDP impl. defl. 330 % ann. growth</td>
<td>Hi-Tech exports 4 % of annual exp.</td>
<td>Gini-index 33.90</td>
</tr>
<tr>
<td>2</td>
<td>Vietnam</td>
<td>Fertility rate 2.57</td>
<td>Gini-index 36.73</td>
<td>Total exp. on health 4.94 % of GDP</td>
</tr>
<tr>
<td>3</td>
<td>Guinea</td>
<td>Internet hosts 0 per 10000 people</td>
<td>PPP per capita 1276 USD</td>
<td>GNP per capita 441.43 USD</td>
</tr>
<tr>
<td>4</td>
<td>Ghana</td>
<td>Fertility rate 3.94</td>
<td>Life exp. (males) 57.62 years</td>
<td>Gini-index 42.61</td>
</tr>
<tr>
<td>5</td>
<td>Slovenia</td>
<td>PPP per capita 13485 USD</td>
<td>Mobile phones 270 per 1000 people</td>
<td>Expend. on R&amp;D 0.98 % of GNP</td>
</tr>
<tr>
<td>6</td>
<td>Netherlands</td>
<td>GDP impl. defl. 2.3 % of ann. growth</td>
<td>Ext. debt 1.1 % of GNP</td>
<td>Tot. debt serv. 0.47 % of export</td>
</tr>
<tr>
<td>7</td>
<td>Peru</td>
<td>Gini-index 48.98</td>
<td>GDP growth rate -1.92 % per capita</td>
<td>Life exp. (males) 66.95 years</td>
</tr>
</tbody>
</table>
FCM-Clustering: conclusions

- **FCM-clustering**
  - efficiency and cluster validity
  - choice of the fuzziness parameter
  - grouping of country economies (World Bank, Ip, Kohonen, Deboeck)

- **Visualization**
  - membership degree
  - topological relationships

- **Landmarks and interpretation of the results**
  - formulation of “class discriminating” criteria
From FCM towards Fuzzy Systems

- Rule extraction:
  - fuzzy inference systems
  - (feed-forward) neural networks
    - back-propagation
    - RBF-networks

- Neuro-fuzzy systems with adaptive inputs
  - detection of significant input patterns
  - influence of internal knowledge representation
  - speed-up the training and recall process

Characteristics $\iff$ Economical results
Artificial Neural Networks (ANN)

Detect patterns in the data in a way “similar” to human thinking!

- Directed data mining (classification and prediction)
- Applicable also to undirected data mining (SOMs)
- Two major drawbacks:
  - difficulty in understanding the models they produce
  - sensitivity to the format of incoming data
Back-Propagation and GREN-networks
Introduction

■ Multi-layer feed-forward networks (BP-networks)
  - one of the most often used models
  - relatively simple training algorithm
  - relatively good results

■ Limits of the considered model
  - the speed of the training process
  - convergence and local minimums
  - generalization and “over-training”

_conversion demands on the desired network behavior
The error function

- corresponds to the difference between the actual and the desired network output:

\[
E = \frac{1}{2} \sum_{p} \sum_{j} (y_{j,p} - d_{j,p})^2
\]

- the goal of the training process is to minimize this difference on the given training set

⇒ Back-Propagation training algorithm
The Back-Propagation training algorithm

- computes the actual output for a given training pattern
- compares the desired and the actual output
- adapts the weights and the thresholds
  - against the gradient of the error function
  - backwards from the output layer towards the input layer
Drawbacks of the standard BP-model

- The error function
  - correspondence to the desired behavior
  - the form of the training set
    - requires the knowledge of desired network outputs
    - better performance for “larger” and “well-balanced” training sets

- Generalization abilities
  - ability to interpret and evaluate the “gained” experience
  - retraining for modified and/or developing task domains
Desired properties of trained networks

- Robustness against small deviations of those input patterns lying “close to the separating hyper-plane”
- Transparent network structure with a suitable internal knowledge representation
- A possible reuse of already trained networks under changed conditions
Condensed internal representation

- interpret the activity of hidden neurons:
  - 1 ↔ active ↔ YES
  - 0 ↔ passive ↔ NO
  - \( \frac{1}{2} \) ↔ silent ↔ “no decision possible”

- “clear” the inner network structure

- detect superfluous neurons and prune
How to force the condensed internal representation?

- formulate “the desired properties” in the form of an objective function:
  \[ G = E + cs F \]
  - Standard error function
  - Representation error function
  - the influence of F on G

- local minima of the representation error function correspond to active, passive and silent states:
  \[ F = \sum_p \sum_h y_{h,p}^s \left(1 - y_{h,p}^s\right)^s \left(y_{h,p}^s - 0.5\right)^2 \]
  - patterns
  - hidden neurons
  - passive state
  - active state
  - the shape of F
  - silent state
Influence of parameters

- slower forcing of the internal representation and the desired network function
- stability of the forced internal representation and an optimal network architecture
- the shape of the representation error function, the speed of the representation forcing process and its form
- the time-overhead of the weight adjustment

\[ w_{ij}(t + 1) = w_{ij}(t) + \alpha \delta_j y_i + \alpha_r \rho_j y_i + \alpha_m \left( w_{ij}(t) - w_{ij}(t - 1) \right) \]
Shape of the representation function

\[ F = y^s (1 - y)^s (y - 0.5)^t \]

\[ s=1 \quad t=2 \]

\[ s=4 \quad t=2 \]

\[ s=8 \quad t=2 \]

\[ s=5 \quad t=4 \]
Further modifications of the representation function

- **Discrete internal representation:**
  
  (S allowed output values $r_1, \ldots, r_s$ for neurons from the last hidden layer)

  \[
  F = \sum_p \sum_j \left( y_{j,p} - r_1 \right)^{2t_1} \cdots \left( y_{j,p} - r_s \right)^{2t_s} = \sum_p \sum_j \prod_s \left( y_{j,p} - r_s \right)^{2t_s}
  \]

- **Condensed internal representation for all hidden layers:**

  \[
  F = \sum_{l'} \sum_p \sum_{j_1'} \ y_{j_1',p}^s \left( 1 - y_{j_1',p} \right)^s \left( y_{j_1',p} - 0.5 \right)^2
  \]
Unambiguous internal representation

- Patterns with highly different outputs should form highly different internal representations.

- Formulate the requirements as a modified objective function:
  \[ G = E + F + H \]

- Ambiguity criterion for the internal representation:
  \[
  H = -\frac{1}{2} \sum_{p} \sum_{q \neq p} \sum_{j} \sum_{o} \left( d_{o,p} - d_{o,q} \right)^2 \left( y_{j,p} - y_{j,q} \right)^2
  \]

\[\text{patterns} \quad \text{hidden neurons} \quad \text{output neurons} \quad \text{const. for a given p} \quad \text{const. for a given p} \quad \text{const. for a given p}\]
Modular structure of BP-networks

- Decompose the task into the particular subtasks
- Propose and form the modular architecture
  - strategy for extracting $\varepsilon$-equivalent BP-modules
    - elimination of superfluous hidden and/or input neurons
    - suitable for "already trained" networks
    - a compromise between the desired accuracy of the extracted module and its optimal architecture
- Communication between the particular modules
  - serial and parallel composition of BP-networks
Extracting BP-modules
- allowed potential deviations

- The potential change $\delta_r^-(\xi)$ is in this case smaller than the potential change $\delta_r^+(\xi)$
- The potential should change "towards the separating hyperplane"
- The changed potential should preserve the location of the input pattern in the same half-space
- The allowed potential changes should be independent of each particular input pattern (from S)
Notes on the construction of an $\varepsilon$-equivalent network

- Possible improvements of network properties:
  - “egalitarian” versus “differentiated” approach

- The relationship of the construction to “more robust” networks
  - Necessary knowledge of $\varepsilon_r$-boundary regions
  - Preserve the created internal representation
Desired properties of “experts” for training (modular) BP-networks

- evaluate the error connected with the actual response of a BP-network
- “explain” the BP-network its error during training
- not require the knowledge of the desired network output
- but should recognize a correct behavior
- “suggest” a “better” behavior
Desired properties of “experts” for training BP-networks

- evaluate the error connected with the actual response of a BP-network
- “explain” the BP-network its error during training
- not require the knowledge of the desired network output
- but should recognize a correct behavior
- “suggest” a “better” behavior
Desired properties of “experts” for training BP-networks

- evaluate the error connected with the actual response of a BP-network
- “explain” the BP-network its error during training
- not require the knowledge of the desired network output
- but should recognize a correct behavior
- “suggest” a “better” behavior
GREN-networks: Generalized relief error networks

- assign the error to the pairs \([input\ pattern,\ actual\ output]\)
- trained e.g. by the standard BP-training algorithm
- should have good approximation and generalization abilities
- “approximates” the error function by:

\[
E = \sum_p \sum_e e^{GR}_{e,p}
\]

patterns \(p\)

output values of the GREN-network \(e\)

output neurons of the GREN-network

Iveta Mrázová, ANNIE’03
A modular system for training BP-networks with GREN-networks
Training with a GREN-network

- Applied the basic idea of Back-Propagation
- How to determine the error terms at the output of the trained BP-network?

  Use error terms back-propagated from the GREN-network

- Weight adjustment rules similar to the standard Back-Propagation
Training with a GREN-network

- Applied the basic idea of Back-Propagation

\[ \Delta_{Ew}^{B} = - \frac{\partial E}{\partial w_{ij}} = - \frac{\partial E}{\partial y_{j}} \frac{\partial y_{j}^{B}}{\partial \xi_{j}^{B}} \frac{\partial \xi_{j}^{B}}{\partial w_{ij}} \]

- How to determine \( \frac{\partial E}{\partial y_{j}^{B}} \) at the output layer of the BP-network \( B \)?
Weight adjustment rules

- Use error terms back-propagated from the GREN-network
- Rules similar to the standard Back-Propagation

\[
 w_{ij}^B (\text{new } ) = w_{ij}^B (\text{old } ) + \alpha \delta_j^B y_i^B \]

- For output neurons, compute \( \delta_j^B \) by means of \( \delta_k^{GR_B} \) propagated from the GREN-network \( GR_B \)

\[
 \delta_k^{GR_B} = - \frac{\partial E}{\partial y_k^{GR_B}} \frac{\partial y_k^{GR_B}}{\partial \xi_k^{GR_B}} \]

Iveta Mrázová, ANNIE’03
Error terms for the trained BP-network

The back-propagated error terms $\delta_j^B$ correspond for $E = \sum e_e$ to:

\[
\delta_j^B = \begin{cases} 
- \left( \sum_e e_e^{GRB} \left(1 - e_e^{GRB}\right) w_{je}^{GRB} \right) f''(\xi_j^B) 
& \text{for an output neuron of } B \text{ and } GRB \text{ with no hidden layer} \\
- \left( \sum_k \delta_k^{GRB} w_{jk}^{GRB} \right) f''(\xi_j^B) 
& \text{for an output neuron of } B \text{ and } GRB \text{ with hidden layers} \\
\left( \sum_k \delta_k^B w_{jk}^B \right) f''(\xi_j^B) 
& \text{for a hidden neuron of } B
\end{cases}
\]
Is the GREN-network an “expert”? 

- Has not to “know the right answer”
- But should “recognize the correct answer”

for an input pattern, yield the minimum error only for one actual output - the right one

- Simple test for “problematic” GREN-experts:
  - zero-weights from the actual output $y^B$
  - zero “$y$-terms” of potentials in the 1. hidden layer
  - “too many large negative weights” $\sum_i |w_i^-| >> \sum_i w_i^+$
Find “better” input patterns!

- input patterns of a GREN-network
- “similar” to those presented to and recalled by the BP-network
- with a smaller error

- **minimize** the error at the output of the GREN-network, e.g. **by back-propagation**
- **adjust input patterns** against the gradient of the GREN-network error function
Avoid “problematic” GREN-networks!

- Insensitive to the outputs of trained BP-networks
  - inadequately small error terms back-propagated by the GREN-network
- Incapable of training further BP-networks
  - small error terms even for large errors

Our goal:
Increase the sensitivity of GREN-networks to their inputs!
How to handle the sensitivity of BP-networks?

Increase their robustness:

- over-fitting leads to functions with a lot of structure and a relatively high curvature
- favor “smoother” network functions
- alternative formulation of the objective function
  - penalizing large second-order derivatives of the network function
  - penalizing large second-order derivatives of the transfer function for hidden neurons
  - weight-decay regularizers
Controlled learning of GREN-networks

- Require GREN-networks sensitive to their inputs
  - non-zero error terms for incorrect BP-network outputs
- Favor larger values of the error terms

\[
E_{\text{REG}} = \sum_q E^\text{REG}_q = -\sum_q \sum_s \sum_{r>n} \left( \frac{\partial y_{s,q}}{\partial y_{r,q}} \right)^2
\]

Minimize during training

Patterns → Output neurons → Controlled input neurons → Controlled input values → Output values
Weight adjustment rules

- Regularization by means of

\[
\Delta_{E^{REG}} w_{ij} = - \frac{\partial E^{REG}}{\partial w_{ij}} = - \frac{\partial}{\partial w_{ij}} \left( - \sum_{s} \sum_{r>n} \left( \frac{\partial y_{s}}{\partial y_{r}} \right)^2 \right)
\]

- Rules similar to the standard Back-Propagation

\[
w_{ij}^{GRB}(T+1) = w_{ij}^{GRB}(T) + \alpha \Delta_{E} w_{ij} + \alpha_c \Delta_{E^{REG}} w_{ij} + \alpha_m \left( w_{ij}^{GRB}(T) - w_{ij}^{GRB}(T-1) \right)
\]
Characteristics of the method

- Applicable to any BP-network and/or input neuron
- Quicker training of “actual” BP-networks
  - larger “sensitivity terms” \( \frac{\partial y_s}{\partial y_r} \) transfer better the errors from the GREN-network
- Oscillations during training “actual” BP-networks
  - due to the “linear” nature of the GREN-specified error function

\[
E = \sum_p \sum_e e^{GR_B}_{e,p}
\]

patterns
output neurons of the GREN-network
output values of the GREN-network
Modification of the method

- Use “quadratic” GREN-specified error terms for training “actual” BP-networks

\[
\hat{E} = \sum_p \sum_e \left( e_{e, p}^{GR_B} \right)^2
\]

- Considers both the GREN-network outputs \( e_{e, p}^{GR_B} \) and the “sensitivity” terms \( \partial e_{e, p}^{GR_B} / \partial y_{j, p}^B \)

- Crucial for low sensitivity to erroneous training patterns
Supporting experiments

Output of the standard BP-network

Output of the GREN-trained BP-network

3000 cycles, SSE = 0.89

3000 cycles, SSE = 0.05
Supporting experiments

BP-network output for a constant $y=0.25$

errorbars correspond to the GREN-error

GREN-adjusted input/output patterns for a constant $y=0.25$

errorbars correspond to the GREN-error

initial I/O_pattern_1=[0,0.25,0.197]
initial I/O_pattern_2=[0.5,0.25,0.388]
initial I/O_pattern_3=[1,0.25,0.932]
Supporting experiments

Output of the standard BP-network (with 8 hidden neurons)

1500 cycles, SSE = 0.51

Output of the GREN-trained BP-network (with 8 hidden neurons)

1500 cycles, SSE = 0.06, GREN-error = 1.2
Supporting experiments

Sensitivity and error for a standard BP-trained GREN-network

Sensitivity and error for a controlled-trained GREN-network (control rates = 0.2)
Supporting experiments

Sensitivities and error for a controlled-trained GREN-network (control rates = 0.2)

SSE

<table>
<thead>
<tr>
<th>Cycles</th>
<th>Sensitivity to BP-output</th>
<th>Network Sensitivity</th>
<th>Network Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>-2</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2500</td>
<td>0</td>
<td>0.5</td>
<td>0.2</td>
</tr>
<tr>
<td>5000</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Sensitivities and error for an over-trained GREN-network (control rates = 0.2)

SSE

<table>
<thead>
<tr>
<th>Cycles</th>
<th>Sensitivity to BP-output</th>
<th>Network Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>-2</td>
<td>0</td>
</tr>
<tr>
<td>2500</td>
<td>0</td>
<td>0.2</td>
</tr>
<tr>
<td>5000</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
GREN-networks: conclusions

- GREN-networks can train BP-networks without the knowledge of their desired outputs
- A simple detection of “problematic” GREN-experts
- GREN-networks can find “similar” input patterns with a lower error
Conclusions: Sensitivity of GREN-networks

- Increase the sensitivity of trained GREN-networks to their inputs
- Detect “over-training” in GREN-networks
- Train BP-networks more efficiently by minimizing squared GREN-network outputs instead of the “linear” ones
- Further research: simplified sensitivity control
Acoustic Emission and Feature Selection Based on Sensitivity Analysis

with M. Chlada and Z. Převorovský,
Institute of Thermomechanics, Academy of Science
Acoustic Emission and Feature Selection Based on Sensitivity Analysis

- BP-networks and sensitivity analysis
  - larger “sensitivity terms” $|\frac{\partial y_j}{\partial x_i}|$ indicate higher importance of the feature $i$

- numerical experiments
  - acoustic emission:
    - classification of simulated AE data
  - feature selection:
    - reduction of original input parameters (from 14 to 6)
    - model dependence between parameters
Simulation of AE-data

MODEL PULSES

WAVE 1

WAVE 2

WAVE 3

0.3*WAVE1 + 0.2*WAVE2 + 0.5*WAVE3
Simulation of AE-data

CONVOLUTION WITH THE GREEN FUNCTION

GREEN FUNCTION - 140mm

INPUT SIGNAL (a=0.3, b=0.2, c=0.5)

Iveta Mrázová, ANNIE’03 112
Original Features for AE-signals

- **amplitude:** $z_{\text{max}} = \max_{t \in T} \{|z(t)|\}$
- **rise time**
- **effective value (RMS)**
  $$\text{RMS} = \sqrt{\frac{1}{T} \int_T z^2(t) \, dt}$$
- **energy moment:** $T_E = \int_T t \cdot z^2(t) \, dt$
- **mean value:** $t_s = \left( \int_T t \cdot z(t) \, dt \right) / T$
- **deviation:** $\sigma^2 = \left( \int_T (t_s - z(t))^2 \, dt \right) / T$
- **asymmetry:** $\eta^2 = \left( \int_T (t_s - z(t))^3 \, dt \right) / \sigma^3$
- **excess:** $\xi^2 = \left( \int_T (t_s - z(t))^4 \, dt \right) / \sigma^4$
- **6 spectral parameters:**
  $$P_X = \frac{\int_X f(k) \, dk}{\int_G f(k) \, dk}; \quad X \in \{A, B, C, D, E, F\}$$
  with
  - $A = ([0.12 / 2, f_N / 2])$
  - $B = ([0.12, 0.24 / 2, f_N / 2])$
  - $C = ([0.24, 0.36 / 2, f_N / 2])$
  - $D = ([0.36, 0.48 / 2, f_N / 2])$
  - $E = ([0.48, 0.6 / 2, f_N / 2])$
  - $F = ([0.6 / 2, 1, f_N / 2])$
  and $G = ([0.1 / 2, f_N])$

$f_N$ is the Nyquist frequency
Factor analysis for input parameters

- 9 factors selected
- “explain” 98.4% of all variables, e.g.
  - higher energy of signals lead to higher amplitudes and RMS (parameters 1, 3, 4)
- allow to reduce linearly dependent input parameters
  - in our case to: 2, 3, 5, 6, 7, 8, 11, 12 and 2 new spectral parameters

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.04</td>
<td>0.91</td>
<td>0.16</td>
<td>0.10</td>
<td>0.01</td>
<td>0.07</td>
<td>0.06</td>
<td>0.01</td>
<td>0.23</td>
</tr>
<tr>
<td>0.09</td>
<td>0.02</td>
<td>0.01</td>
<td>0.19</td>
<td>0.03</td>
<td>0.03</td>
<td>0.95</td>
<td>0.03</td>
<td>0.16</td>
</tr>
<tr>
<td>0.10</td>
<td>0.96</td>
<td>0.15</td>
<td>0.00</td>
<td>0.03</td>
<td>0.00</td>
<td>0.02</td>
<td>0.05</td>
<td>0.07</td>
</tr>
<tr>
<td>0.13</td>
<td>0.91</td>
<td>0.02</td>
<td>0.03</td>
<td>0.05</td>
<td>0.06</td>
<td>0.04</td>
<td>0.06</td>
<td>0.20</td>
</tr>
<tr>
<td>0.30</td>
<td>0.05</td>
<td>0.04</td>
<td>0.41</td>
<td>0.06</td>
<td>0.49</td>
<td>0.17</td>
<td>0.07</td>
<td>0.66</td>
</tr>
<tr>
<td>0.26</td>
<td>0.00</td>
<td>0.04</td>
<td>0.03</td>
<td>0.08</td>
<td>0.93</td>
<td>0.02</td>
<td>0.10</td>
<td>0.20</td>
</tr>
<tr>
<td>0.29</td>
<td>0.06</td>
<td>0.03</td>
<td>0.27</td>
<td>0.03</td>
<td>0.17</td>
<td>0.16</td>
<td>0.04</td>
<td>0.86</td>
</tr>
<tr>
<td>0.12</td>
<td>0.06</td>
<td>0.01</td>
<td>0.88</td>
<td>0.02</td>
<td>0.02</td>
<td>0.24</td>
<td>0.03</td>
<td>0.36</td>
</tr>
<tr>
<td>0.90</td>
<td>0.15</td>
<td>0.09</td>
<td>0.08</td>
<td>0.15</td>
<td>0.17</td>
<td>0.06</td>
<td>0.21</td>
<td>0.18</td>
</tr>
<tr>
<td>0.93</td>
<td>0.14</td>
<td>0.06</td>
<td>0.08</td>
<td>0.10</td>
<td>0.17</td>
<td>0.06</td>
<td>0.09</td>
<td>0.18</td>
</tr>
<tr>
<td>0.25</td>
<td>0.10</td>
<td>0.12</td>
<td>0.03</td>
<td>0.25</td>
<td>0.11</td>
<td>0.03</td>
<td>0.90</td>
<td>0.05</td>
</tr>
<tr>
<td>0.20</td>
<td>0.07</td>
<td>0.14</td>
<td>0.02</td>
<td>0.93</td>
<td>0.09</td>
<td>0.03</td>
<td>0.23</td>
<td>0.04</td>
</tr>
<tr>
<td>0.04</td>
<td>0.09</td>
<td>0.97</td>
<td>0.00</td>
<td>0.05</td>
<td>0.00</td>
<td>0.00</td>
<td>0.06</td>
<td>0.02</td>
</tr>
<tr>
<td>0.08</td>
<td>0.18</td>
<td>0.95</td>
<td>0.02</td>
<td>0.09</td>
<td>0.04</td>
<td>0.01</td>
<td>0.06</td>
<td>0.02</td>
</tr>
</tbody>
</table>
Sensitivity analysis of trained BP-networks

- 2000 samples
  - 500 training s.
- 14-27-19-3
- 180 iterations
- selected inputs:
  - sensitivity analysis
    1, 3, 4, 5, 6, 13, 14
  - + factor analysis
    1, 3, 5, 6, 13, 14
- new architecture:
  6-13-7-3 (even with slightly better MSE-results)
Model dependence

SENSITIVITY COEFFICIENTS ... $X_4 = (X_1)^4$

<table>
<thead>
<tr>
<th>PARAMETER</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.32</td>
<td>0.16</td>
<td>0.08</td>
<td>0.04</td>
</tr>
<tr>
<td>2</td>
<td>0.04</td>
<td>0.92</td>
<td>0.03</td>
<td>0.04</td>
</tr>
<tr>
<td>3</td>
<td>0.02</td>
<td>0.05</td>
<td>0.98</td>
<td>0.03</td>
</tr>
<tr>
<td>4</td>
<td>0.16</td>
<td>0.14</td>
<td>0.05</td>
<td>0.35</td>
</tr>
</tbody>
</table>

TARGET - PARAMETER

$X_4 = (X_1)^4$
Model dependence

Sensitivity coefficients ...

\[ X_4 = \sin(9 \cdot X_1) \]
Knowledge extraction in neural networks (students’ questionnaire)

with Eva Poučková,
Department of Software Engineering, Charles University Prague
Knowledge representation in NN

Which inputs are the most important ones?

What and how does the network do?
Knowledge extraction in NN

- Dimension reduction and sensitivity analysis for inputs
- Rule extraction from trained networks
  - Structural learning with forgetting
    - BP-networks
    - GREN-networks
  - Babsi-trees (B. Hammer et al.)
    - GRLVQ
Dimension reduction

- **PCA**: linear transformation of the input data

- **Sensitivity analysis:**
  Feature Subset Selection (FSS)

- **Correlation-based Feature Selection (CFS):**
  select a group of features with a high average correlation \( \text{input feature - output} \) but with a low mutual correlation
Dimension reduction: results

**PCA:** method not suitable for further processing – knowledge and rule extraction

25 original features

FSS: 7 features

CFS: 7 features

8 features selected as a union of the results for FSS and CFS
Features selected for the overall evaluation

<table>
<thead>
<tr>
<th>Feature subset selection (FSS):</th>
<th>Correlation-based feature selection (CFS):</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1) understandable subject</td>
<td>(1) understandable subject</td>
</tr>
<tr>
<td>(2) structured and prepared presentations</td>
<td>(2) structured and prepared presentations</td>
</tr>
<tr>
<td>(3) interesting classes</td>
<td>(3) interesting classes</td>
</tr>
<tr>
<td>(4) quality of education</td>
<td>(4) quality of education</td>
</tr>
<tr>
<td>(5) understandable classes</td>
<td>(5) understandable classes</td>
</tr>
<tr>
<td>(6) start/end of class on time</td>
<td>(6) start/end of class on time</td>
</tr>
<tr>
<td>(7) relationship to students</td>
<td>(8) students prepare for classes</td>
</tr>
</tbody>
</table>
Methods for knowledge extraction

- SLF – Structural learning with forgetting
  - Learning with forgetting
  - Learning with forcing internal representations on hidden neurons
  - Learning with selective forgetting

- Babsi-trees
  - Form a tree from a neural network trained by means of the GRLVQ-method
Generalized relevance learning vector quantization (GRLVQ)

- a robust combination of GLVQ and RLVQ
- provides weighing factors ($\lambda$) for input features
  - larger $\lambda$ corresponds to a “more important” feature
- applicable to pruning of input features
- **GLVQ:** considers class representatives
  - separating surfaces approach the optimum Bayessian ones
- **RLVQ:** input features can have different importance
  - relatively unstable, sensitive to noise
Generalized LVQ - GLVQ

- Select a fixed number of representatives $w_1, \ldots, w_L$ for all classes $C_i, i=1, \ldots, q$.

- Receptive field of the class representative $w_i$

  $$R_i = \{x \in T \mid \forall k \neq i : \| x - w_i \| < \| x - w_k \| \}$$

- Receptive fields of class representatives should be as small as possible!

  - minimize $E = \sum_{k=1}^{p} \sigma(\eta(x^{(i)}))$; $\sigma$ denotes the sigmoid
  
  - and $\eta(x^{(i)}) = \frac{\| x^{(i)} - w^+ \| - \| x^{(i)} - w^- \|}{\| x^{(i)} - w^+ \| + \| x^{(i)} - w^- \|}$

  [Diagram showing correct and wrong classification]

Iveta Mrázová, ANNIE’03
Generalized LVQ - GLVQ

- **Weight adjustment:**

\[
\Delta w^+ = \alpha \sigma' \frac{\| x^{(t)} - w^- \|}{(\| x^{(t)} - w^+ \| + \| x^{(t)} - w^- \|)^2} (x^{(t)} - w^+)
\]

\[
\Delta w^- = -\alpha \sigma' \frac{\| x^{(t)} - w^+ \|}{(\| x^{(t)} - w^+ \| + \| x^{(t)} - w^- \|)^2} (x^{(t)} - w^-)
\]

with \( \sigma' = \sigma(\eta(x^{(t)}))' = \sigma(\eta(x^{(t)}))(1 - \sigma(\eta(x^{(t)}))) \)

and learning rates \( \alpha \)
Relevance LVQ - RLVQ

- Input features can have different importance $\lambda$

$$\text{dist} \left( \mathbf{x}, \mathbf{w} \right)_\lambda = \sum_{i=1}^{n} \lambda_i (x_i - w_i)^2 ; \quad \sum_{i=1}^{n} \lambda_i^2 = 1$$

- Receptive field of the class representative $\mathbf{w}_i$

$$R_{i\lambda} = \left\{ \mathbf{x} \in T \mid \forall j \neq i : \| \mathbf{x} - \mathbf{w}_i \|_\lambda < \| \mathbf{x} - \mathbf{w}_j \|_\lambda \right\}$$

- Weight adjustment according to GLVQ with adaptive importance factors $\lambda_i$ for input features $(0 < \varepsilon < 1)$:

$$\Delta \lambda_i^{(t)} = \begin{cases} \max \left( \lambda_i^{(t-1)} - \varepsilon (x_i^{(t)} - w_{ij})^2, 0 \right) & d^{(t)} = c_j \\ \lambda_i^{(t-1)} + \varepsilon (x_i^{(t)} - w_{ij})^2 & \text{else} \end{cases}$$
Generalized Relevance Learning
Vector Quantization (GRLVQ)

- Weight adjustment according to GLVQ with adaptive importance factors $\lambda_i$ for input features:

$$\Delta \lambda_i^{(t)} = -\varepsilon \sigma' \left( \frac{\| x^{(t)} - w^- \|}{(\| x^{(t)} - w^+ \| + \| x^{(t)} - w^- \|)^2} \left( x_i^{(t)} - w_i^+ \right)^2 - \right. $$

$$\left. - \frac{\| x^{(t)} - w^+ \|}{(\| x^{(t)} - w^+ \| + \| x^{(t)} - w^- \|)^2} \left( x_i^{(t)} - w_i^- \right)^2 \right)$$
Babsi-trees

Root-trees $G=(V,E)$ satisfying the following conditions:

- all vertices $v_i \in V$ can have an arbitrary number $n_i$ of sons
- all leaves $v_J$ are labeled with the corresponding classification class $C_J$
- all vertices $v_i$ which are not leafs are labeled with $I^{v_i}$
  - $I^{v_i}$ stands for the currently processed input dimension $i$
  - dimensions are “ordered” according to their importance ($\lambda$)
- All edges going from a vertex $v_i$ to its sons are labeled with intervals $\left( s^{v_i}_k, s^{v_i}_l \right)$
  - interval boundaries are placed in the middle between two neighboring cluster representatives
SLF for layered networks

feed-forward neural networks

GREN-networks
Results for the SLF-method

Both BP-networks and GREN-trained networks lead to similar sets of rules:
The resulting Babsi-tree

Relevant dimensions (features): 4, 8 and 7

Values for intervals:
1 $(-\infty, 1.5)$
2 $[1.5, 2.5)$
3 $[2.5, 3.5)$
4 $[3.5, 4.5)$
5 $[4.5, \infty)$

Overall evaluation

Iveta Mrázová, ANNIE’03
# Comparison of the results

**SLF**
- Few simple hierarchically ordered rules
- Possibility to add rules after achieving the desired accuracy
- Rule correctly applicable - 71% and 73%, resp.

**Babsi-trees**
- Many simple rules
- Quick training of the network
- Few training parameters
- Rule correctly applicable - 67%
Knowledge extraction: Conclusions

**Main results achieved:**
- dimension reduction for the input space
- analysis of various models for knowledge extraction
- rule extraction from GREN-networks
- comparison with other neural network models

**Further research:**
- adjusting rules extracted from a neural network trained with the GRLVQ-algorithm
- (automatic) selection of training parameters for the SLF-algorithm
Genetic Algorithms (GA)

Apply genetics and natural selection to find optimal parameters of a predictive function!

- GA use “genetic” operators to evolve successive generations of solutions:
  - selection
  - crossover
  - mutation

- Best candidates “survive” to further generations until convergence is achieved

- Directed data mining
The basic Genetic Algorithm

- **Step 1**: Create an initial population of individuals
- **Step 2**: Evaluate the fitness of all individuals in the population
- **Step 3**: Select candidates for the next generation
- **Step 4**: Create new individuals (use genetic operators - crossover and mutation)
- **Step 5**: Form a new population by replacing (some) old individuals by new ones
- GOTO Step 2
ANTARES

STUDENT SOFTWARE PROJECT
supervised by I. Mrázová, F. Mráz

participating students:
D. Bělonožník, D. J. Květoň, M. Šubert,
J. Tomaštík, J. Tupý

http://www.ms.mff.cuni.cz/~mraz/antares
Project ANTARES

- **Generate melodies** with genetic algorithms
  - the fitness of candidate solutions is evaluated by the cooperating feed-forward neural network

- Parallel implementation of genetic algorithms
  - open system for the design and testing of genetic algorithms and neural networks
  - supports mutual cooperation between neural networks and genetic algorithms
Fitness evaluation with neural networks

- For some problems, it might be difficult to define explicitly the fitness function
  - e.g. „evaluate“ the beauty of generated melodies
- Fitness of candidate melodies (generated by GA) is evaluated by the pre-trained NN:
  - provide a set of positive and negative examples (supervised learning)
  - train a feed-forward network to approximate the „unknown“ fitness function (on the training set)
Generating melodies: positive training samples
Generating melodies

- Positive training samples
- Negative training samples
- Test samples (with a high fitness value)
- Generated melodies
Thank you for your attention!