Data Mining with Multilayer Perceptrons: Intensive Care Medicine and Internet Traffic Forecasting

Paulo Cortez
pcortez@dsi.uminho.pt
http://www.dsi.uminho.pt/~pcortez

Departament of Information Systems
Universidade of Minho - Campus de Azurém
4800-058 Guimarães - Portugal

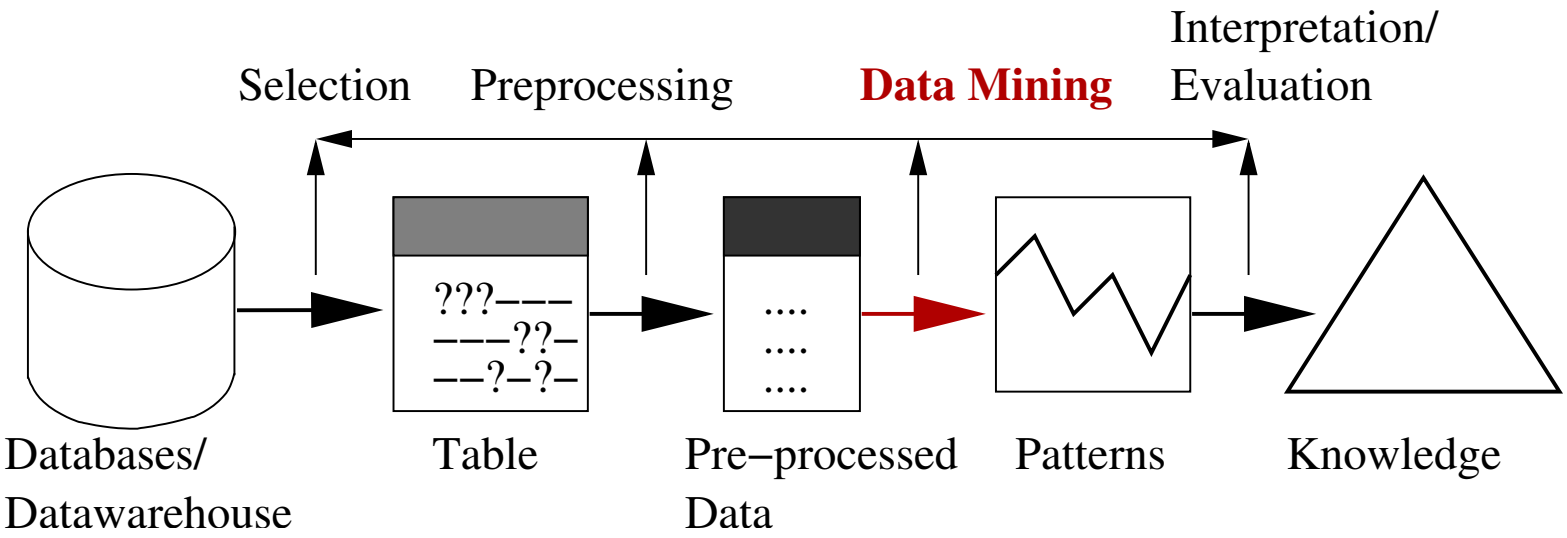
Motivation

- With the advances in information technology, there has been an **ever-increasing load of data** in organizations (massive datasets are commonplace);
- All this data, often with high complexity, holds **valuable information**;
- Human experts are limited and may overlook relevant details;
- Moreover, classical statistical analysis breaks down when such vast and/or complex data are present;

- A better alternative is to use **automated discovery tools** to analyze the raw data and extract high-level information for the decision maker [Hand et al., 2001];

Knowledge Discovery from Databases and Data Mining

For more details, consult [Fayyad et al., 1996].



Knowledge Discovery in Databases (KDD)

"the overall process of discovering useful knowledge from data".

Data Mining (DM)

"application of algorithms for extracting patterns (models) from data".
This is a particular step of the KDD process.

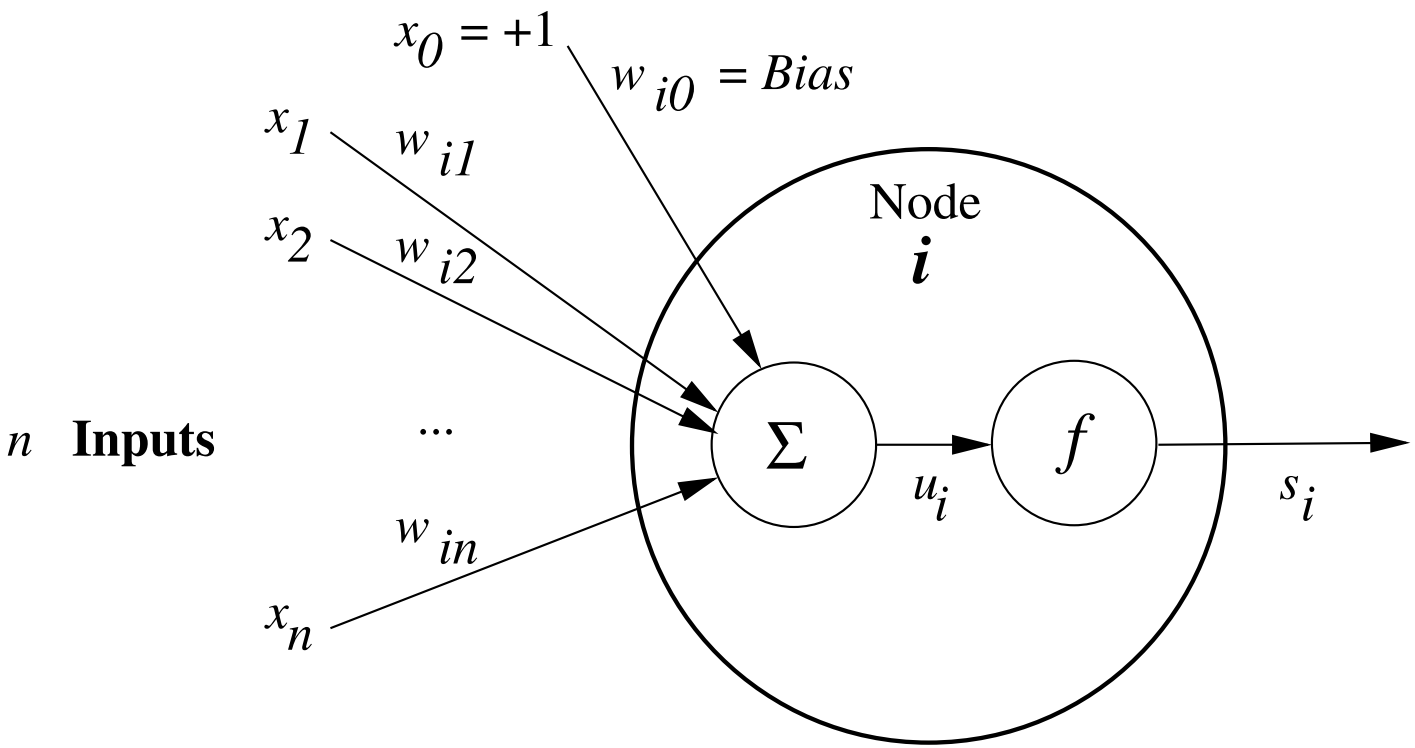
KDD consists of several iterative steps:

- **Understanding** the application domain;
- Acquiring or **selecting** a target data set;
- Data cleaning, **preprocessing** and transformation;
- Choosing the **DM goals**, **DM algorithm** and searching for **patterns of interest**;
- Result **interpretation** and verification; and
- **Using** and maintaining the discovered knowledge.

DM goals [Fayyad et al., 1996]

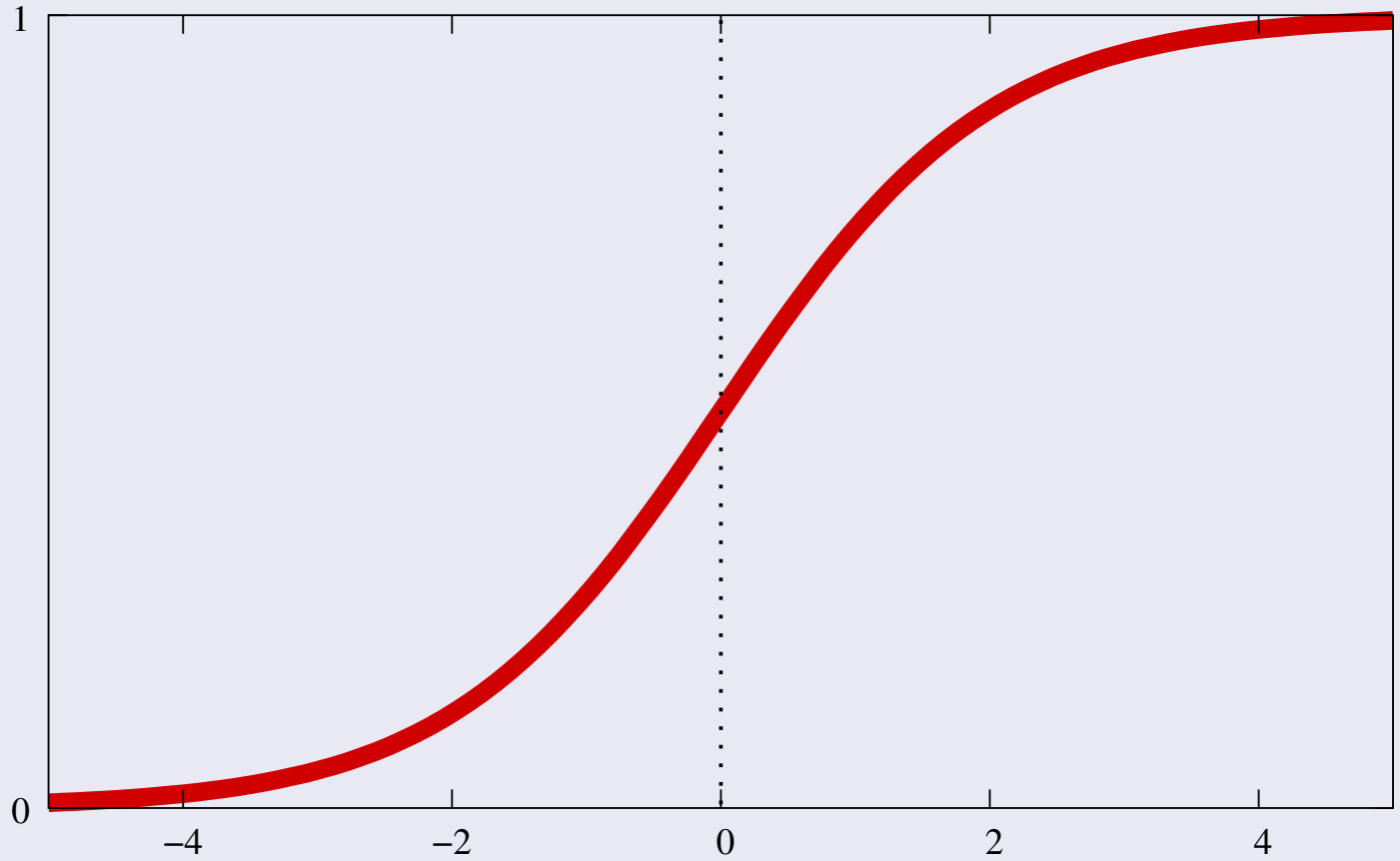
- **Classification** – labeling a data item into one of several predefined classes (e.g. diagnosing a disease according to patient's symptoms);
- **Regression** – mapping a set of attributes into a real-value variable (e.g. stock market prediction);
- **Clustering** – searching for natural groupings of objects based on similarity measures (e.g. segmenting clients of a database marketing study); and
- **Link analysis** – identifying useful associations in transactional data (e.g. "64% of the shoppers who bought milk also purchased bread").

- Feedforward neural network where each node **outputs** an activation function applied over the weighted sum of its **inputs**:
  $s_i = f(w_{i,0} + \sum_{j \in I} w_{i,j} \times s_j)$



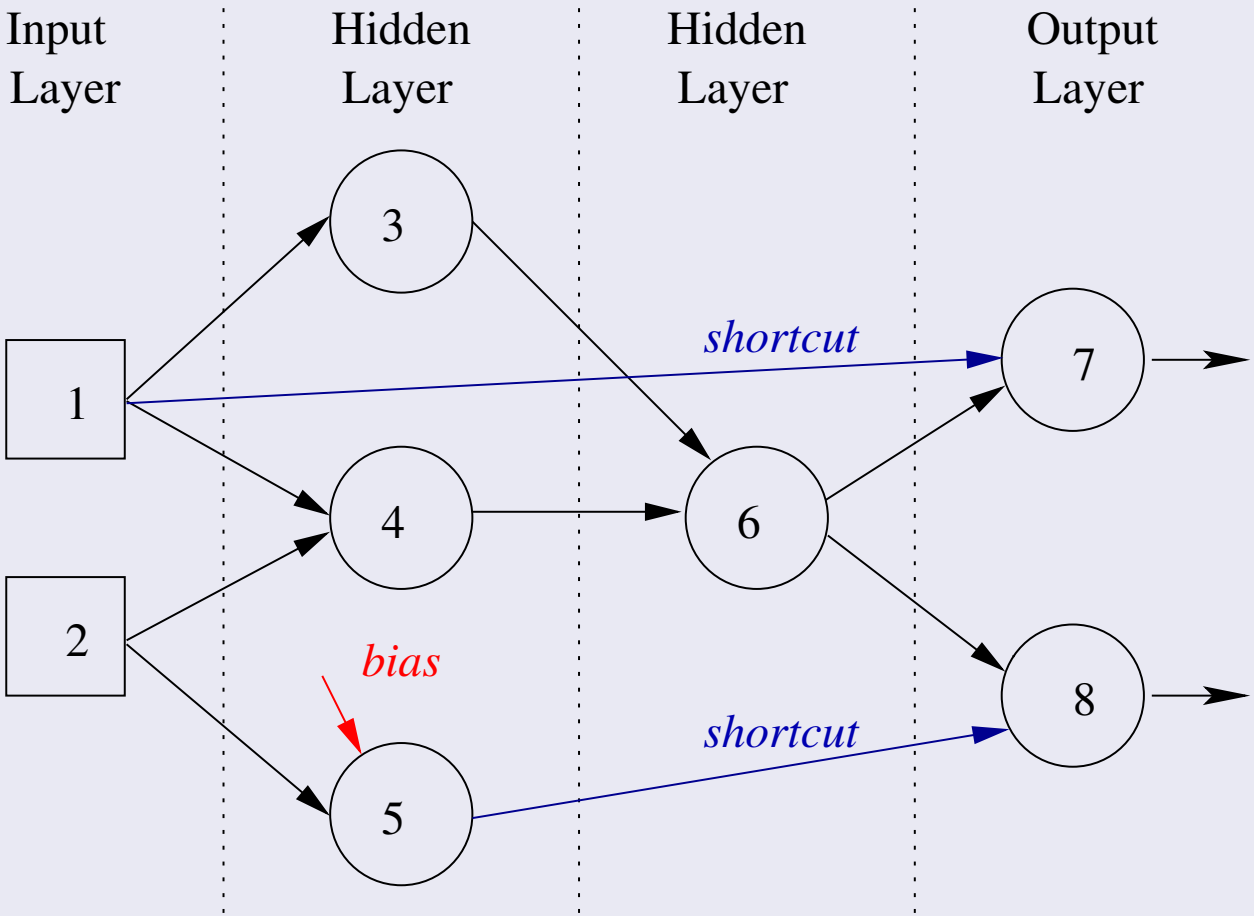
Activation functions

- Linear: $y = x$;
- Tanh: $y = tanh(x)$;
- **Logistic** or Sigmoid (most used): $y = \frac{1}{1+e^{-x}}$;



Architecture/Topology

- Only **feedforward** connections exist;
- Nodes are organized in **layers**;



Why Data Mining with MLPs? [Sarle, 2005]

- **Popularity** - the most used Neural Network, with several off-the-shelf packages available;
- **Universal Approximators** - general-purpose models, with a huge number of applications (e.g. classification, regression, forecasting, control or reinforcement learning);
- **Nonlinearity** - when compared to other data mining techniques (e.g. decision trees) MLPs often present a higher predictive accuracy;
- **Robustness** - good at ignoring irrelevant inputs and noise;
- **Explanatory Knowledge** - Difficult to explain when compared with other algorithms (e.g. decision trees), but it is possible to extract rules from trained MLPs.

Software [Sarle, 2005]

Commercial:

- SAS Enterprise Miner Software (`www.sas.com`);
- Clementine (`http://www.spss.com/spssbi/clementine/`);
- MATLAB Neural Network Toolbox (`www.mathworks.com/products/neuralnet`);
- STATISTICA: Neural Networks (`www.statsoft.com`);

Free:

- WEKA (data mining, Java source code available) (`www.cs.waikato.ac.nz/~ml/weka`);
- R (statistical tool, package **nnet**) (`www.r-project.org`);

Build Your Own Code:

Better control, but take caution!

Data Mining with MLPs

Supervised Learning – input/output mapping (e.g. **classification** or **regression**):

- **Data Collection** - learning samples must be representative, hundred/thousand of examples are required;
- **Feature Selection** - what are the relevant inputs?
- **Preprocessing** - data transformation, dealing with missing data, outliers, ...;
- **Modeling** – network design, training and performance assessment;
- **Prediction** – feed the trained MLP with new data and interpret the output;
- **Explanatory Knowledge** – input importance by sensitivity analysis and extraction of rules from trained MLPs;

Feature Selection

- Selection of the subset of relevant features. Why?
  - To reduce storage and measurement requirements;
  - To facilitate data visualization/comprehension;
  - Non relevant features/attributes will **increase the MLP complexity** and **worst performances** will be achieved.

Feature Selection methods [Witten and Frank, 2005]:

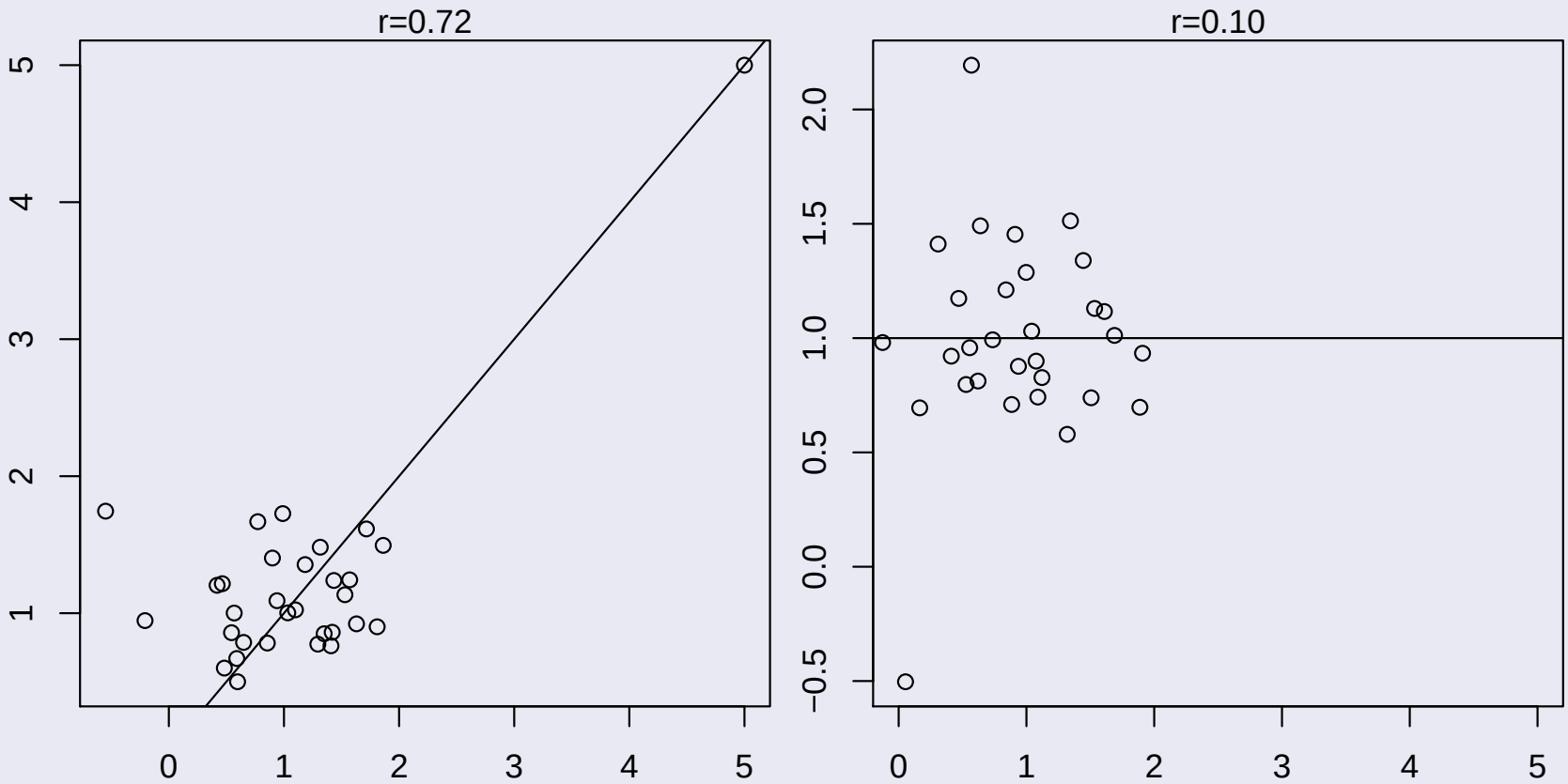
- A priori knowledge (e.g. the use of experts);
- **Filter** and **Wrapper** algorithms;
- Correlation analysis (only measures linear effects);
- Trial-and-error blind search (e.g. test some subsets and select the subset with the best performance);
- Hill-climbing search (e.g. sensitivity analysis, forward and backward selection);
- Beam search (e.g. genetic algorithms);

Handling Missing Data ('?', 'NA', ...) [Brown and Kros, 2003]:

- Use complete data only (delete cases or variables);
- **Data Imputation**, substitute by:
  - Value given by an expert (**case substitution**);
  - Mean, median or mode;
  - Value from another database (**cold deck**);
  - Value of most similar example (**hot deck**);
  - Value estimated by a regression model (e.g. linear regression);
  - Combination of previous methods (**multiple imputation**);

Outliers

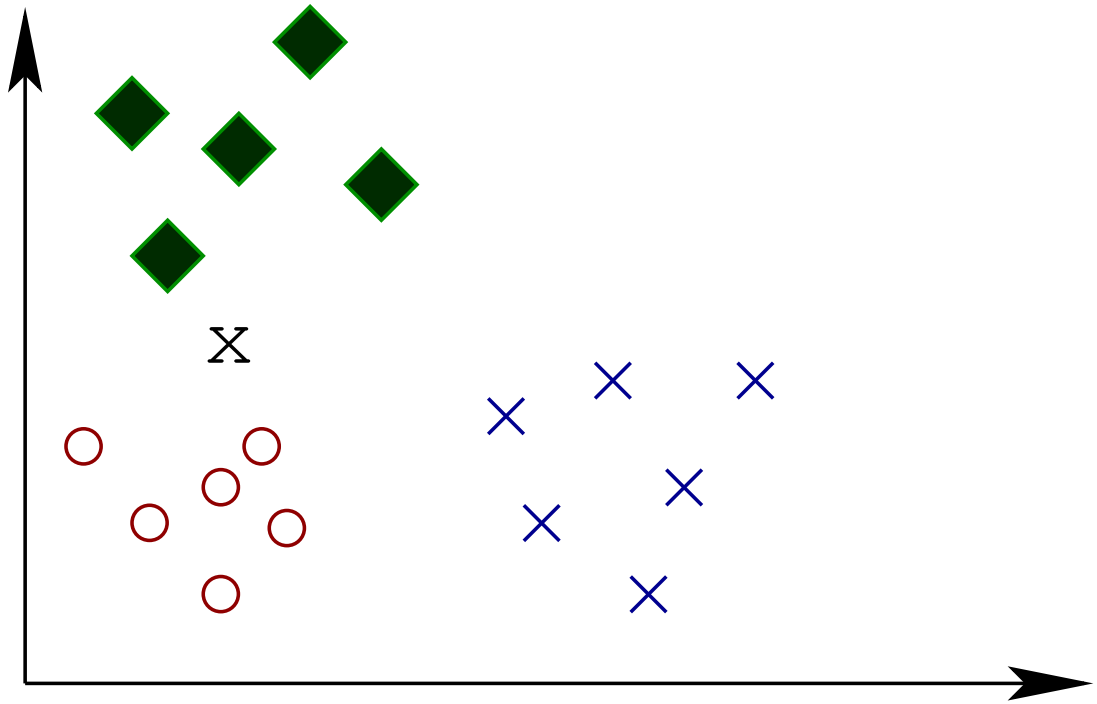
- Due to errors in data collection or rare events;
- Not related with the target variable, they prejudice the learning;
- Solution: use of experts, data visualization, statistical analysis, ...



Nonnumerical variable remapping [Pyle, 1999]

- Only numeric data can be fed into MLPs;
- **Binary** attributes can be coded into 2 values (e.g. $\{-1, 1\}$ or $\{0, 1\}$);
- **Ordered** attributes should be encoded by preserving the order (e.g. $\{\text{low} \rightarrow -1, \text{medium} \rightarrow 0, \text{high} \rightarrow 1\}$);
- **Nominal** (non-ordered with 3 or more classes) attributes:
  - **1-of-C** remapping – use one binary variable per class (generic);
  - **M-of-C** remapping – requires domain knowledge (e.g. a **state** can be coded into 2 variables, the horizontal and vertical position in a 2D map);

Example of 1-of-C remapping



- Attribute **color** = {Red, Blue, Green};
- With the linear mapping {Red $\rightarrow$ -1 , Blue $\rightarrow$ 0, Green $\rightarrow$ 1} it is impossible to describe **X**, which is half green and half red;
- With the **1-of-C** mapping { Red $\rightarrow$ 1 0 0, Blue $\rightarrow$ 0 1 0, Green $\rightarrow$ 0 0 1 }, **X** could be represented by: 0.5 0 0.5;

Rescaling/Normalization [Sarle, 2005]

- MLP learning improves if all **Inputs** are rescaled into the same range with a 0 mean:
  - $y = \frac{x-(max+min)/2}{(max-min)/2}$ (**linear scaling** with range [-1,1])
  - $y = \frac{x-\bar{x}}{s}$ (**standardization** with mean 0 and standard deviation 1)
- **Outputs** limited to the [0,1] range if logistic function is used ([-1,1] if tanh).
  - $y = \frac{(x-min)}{max-min}$ (**linear scaling** with range $[0, 1]$)

Confusion matrix [Kohavi and Provost, 1998]

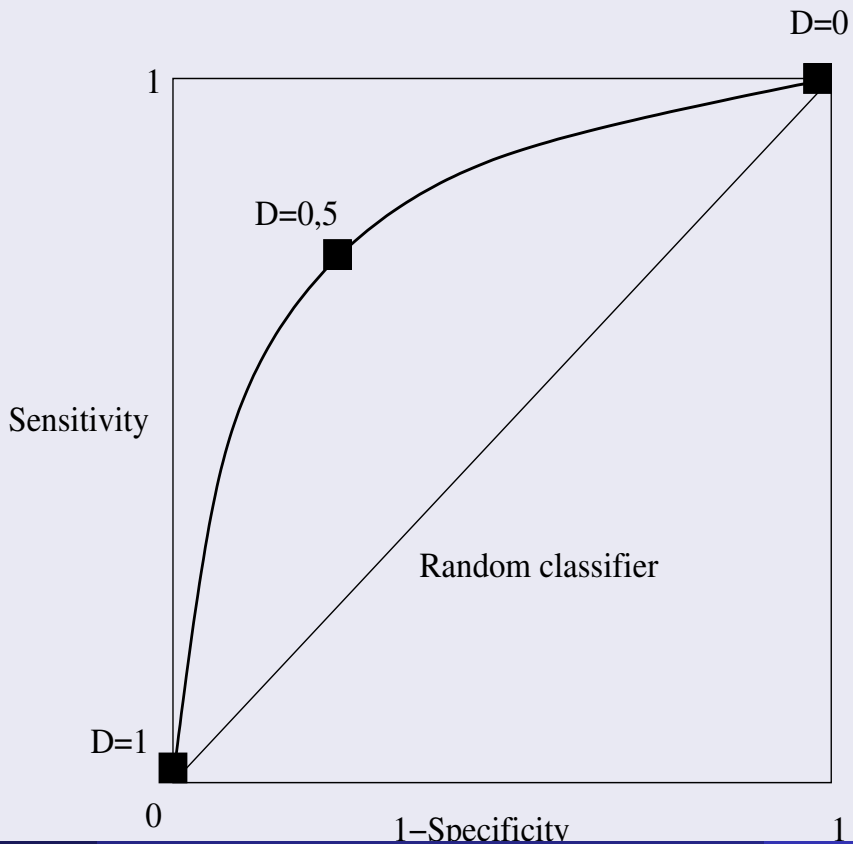
- Matches the **predicted** and **actual** values;
- The 2×2 *confusion matrix*:

$\downarrow$ **actual** $\setminus$ **predicted** $\rightarrow$	**negative**	**positive**
negative	*TN*	*FP*
positive	*FN*	*TP*

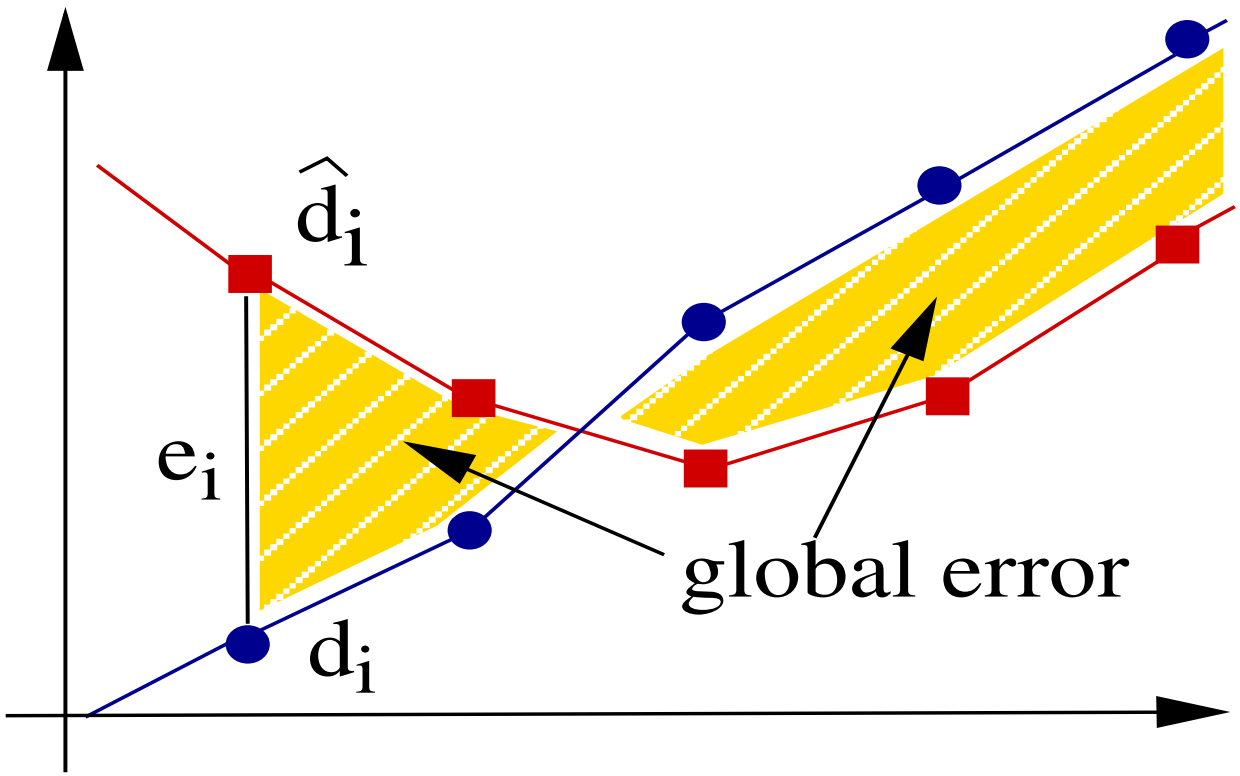
- Three accuracy measures can be defined:
  - the **Sensitivity** (*Type II Error*) $= \frac{TP}{FN+TP} \times 100\,(\%)$;
  - the **Specificity** (*Type I Error*) ; $= \frac{TN}{TN+FP} \times 100\,(\%)$
  - the **Accuracy** $= \frac{TN+TP}{TN+FP+FN+TP} \times 100\,(\%)$;

Receiver Operating Characteristic (ROC) [Fawcett, 2003]

- Shows the behavior of a 2 class classifier when varying a decision parameter D;
- The curve plots 1−Specificity ($x-$axis) vs the Sensitivity;
- Global performance measured by the **Area Under the Curve (AUC)**: $AUC = \int_0^1 ROC\,dD$ (the perfect AUC value is 1.0);



Regression Metrics



- The **error** e is given by: $e = d - \widehat{d}$ where d denotes the desired value and the $\widehat{d}$ estimated value (given by the model);

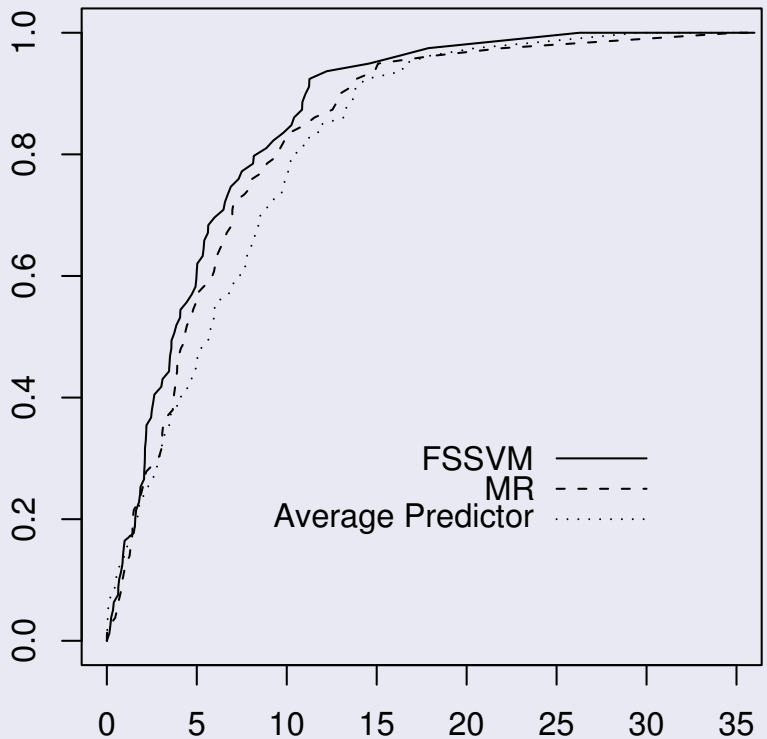
Given a dataset with the function pairs $x_1 \rightarrow d_1, \cdots, x_N \rightarrow d_N$, we can compute:

Error metrics

- **Mean Absolute Deviation (MAD)**: $MAD = \frac{\sum_{i=1}^{N} |e_i|}{N}$
- **Sum Squared Error (SSE)**: $SSE = \sum_{i=1}^{N} e_i^2$
- **Mean Squared Error (MSE)**: $MSE = \frac{SSE}{N}$
- **Root Mean Squared Error (RMSE)**: $RMSE = \sqrt{MSE}$
- **Relative Root Mean Squared** (RRMSE, scale independent): $RRMSE = RMSE / RMSE_{\text{baseline}} \times 100\,(\%)$, where baseline often denotes the average predictor.
- **Normalized Mean Square Error** (NMSE, scale independent); $NMSE = SSE / SSE_{\text{baseline}} \times 100\,(\%)$

Regression Error Characteristic (REC) curves [Bi and Bennett, 2003]

- Used to compare regression models;
- The curve plots the error tolerance (x-axis), given in terms of the absolute or squared deviation, versus the percentage of points predicted within the tolerance (y-axis);
- Example [Cortez et al., 2006a]:



Validation method: how to estimate the performance? [Flexer, 1996]

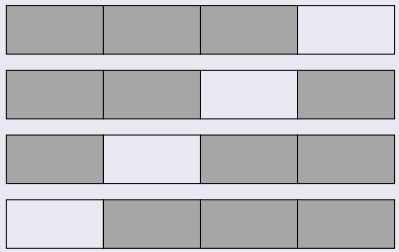
Holdout

Split the data into two exclusive sets, using random sampling:

- **training**: used to set the **MLP** weights (2/3);
- **test**: used to infer the **MLP** performance (1/3).

K-fold, works as above but uses rotation:

- data is split into K exclusive folds of equal size;
- each part is used as a test set, being the rest used for training;
- the overall performance is measured by averaging the K runs;
- **10-fold** most used if hundreds of examples;



Training Algorithm

Gradient-descent [Riedmiller, 1994]:

- **Backpropagation (BP)** - most used, yet slow;
- **Backpropagation with Momentum** - faster than **BP**, requires additional parameter tuning;
- **QuickProp** - faster than **BP with Momentum**;
- **RPROP** - faster than **QuickProp** and stable in terms of its internal parameters;

Evolutionary Computation [Rocha et al., 2003]

- May overcome local minima problems;
- Can be applied when no gradient information is available (reinforcement learning);

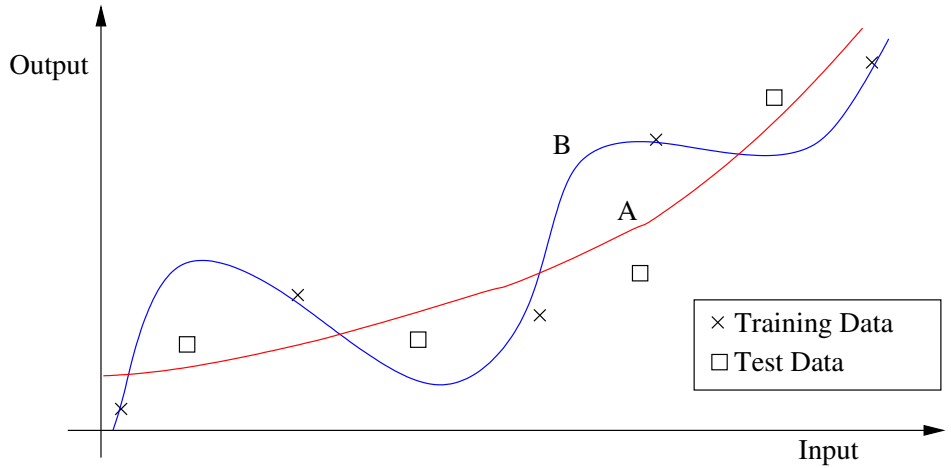
Local Minima [Hastie et al., 2001]

- The MLP weights are randomly initialized within small ranges ([-1,1] or [-0.7;0.7]);
- Thus, each training may converge to a different (local) minima;

Solutions

- Use of **multiple** trainings, selecting the *MLP* with lowest error;
- Use of **multiple** trainings, computing the average error of the MLPs;
- Use of **ensembles**, where the final output is given as the average of the MLPs;

Overfitting [Sarle, 2005][Hastie et al., 2001]



If possible use **large datasets**: $N \gg p$ (weights);

Model Selection

Apply several models and then choose the best generalization MLP;

Regularization

Use learning penalties or restrictions:

- **Early stopping** - stop when the validation error arises;
- **Weight decay** - in each epoch slightly decrease the weights;

MLP Capabilities

Linear learning when:

- there are no hidden layers; or
- only linear activation functions are used.

Nonlinear learning:

- Any continuous function mapping can be learned with one hidden layer;
- Complex discontinuous functions can be learned with more hidden layers;

Some MLP design rules

Output nodes:

It is better to perform only **one** classification/regression task per network; i.e., use **C/1** output node(s).

Activation Functions:

- Hidden Nodes: use the **logistic**;
- Output Nodes: if outputs bounded, apply the same function; else use the **linear** function;

Design Approaches [Rocha et al., 2006]

Blind Search

- Only tests a small number of alternatives.
- Examples: **Grid-Search**/**Trial-and-error** procedures.

Hill-climbing

- Only one solution is tested at a given time.
- Sensitivity to local minima.
- Examples: **Constructive** and **Pruning** methods.

Beam search

- Uses a population of solutions.
- Performs global multi-point search.
- Examples: **Evolutionary Computation (EC)**.

Explanatory Knowledge

In DM, besides obtaining a high predictive performance, it is also important to provide **explanatory knowledge**: what has the model learned?

Measuring Input Importance [Kewley et al., 2000]

- Use of sensitivity analysis, measured as the variance (V_a) produced in the output (y) when the input attribute (a) is moved through its entire range:

$$\begin{aligned} V_a &= \sum_{i=1}^{L} (y_i - \overline{y})/(L-1) \\ R_a &= V_a / \sum_{j=1}^{A} V_j \end{aligned} \quad (1)$$

- A denotes the number of input attributes and R_a the relative importance of the a attribute;
- The y_i output is obtained by holding all input variables at their average values; the exception is x_a, which varies through its range with L levels;

Extraction of rules from trained MLPs [Tickle et al., 1998]

Two main approaches:

- **Decompositional** algorithms start at the minimum level of granularity: first, rules are extracted from each individual neuron (hidden and output); then, the subsets of rules are aggregated to form a global relationship.
- **Pedagogical** techniques extract the direct relationships between the inputs and outputs of the MLP; By using a black-box point of view, less computation is required and a simpler set of rules may be achieved.

MLPs vs Support Vector Machines (SVMs)

SVMs present **theoretical advantages** (e.g. absence of local minima) over MLPs and several comparative studies have reported **better predictive performances**!

Yet:

- Few data mining packages with SVM algorithms are available;
- SVM algorithms require more computational effort for large datasets;
- Under reasonable assumptions, MLPs require the search of one parameter (hidden nodes or the decay) while SVMs require two or more (C, γ, ϵ, ...);
- MLPs can be applied in **real-time**, control & reinforcement or dynamic/changing environments;

The most used DM algorithms

KDNUGGETS poll, May 2006 (`www.kdnuggets.com`):

- Decision Trees/Rules: 51.1%
- Clustering: 39.8%
- Regression: 38.1%
- Statistics: 36.4%
- Association rules: 30.7%
- Visualization: 21.6%
- SVM: 17.6%
- Neural Networks: 17.6%
- Time Series: 13.6
- Bayesian: 13.6
- ...

Case Study I: Intensive Care Medicine (Classification) [Silva et al., 2006]

Intensive Care Units

- In the last decades, a worldwide expansion occurred in the number of **Intensive Care Units (ICUs)**;
- Scoring the severity of illness has become a daily practice, with several metrics available (e.g. APACHE II, SAPS II, MPM);
- The intensive care improvement comes with a price, being ICUs responsible for an increasing percentage of the health care budget;
- Resource limitations force Physicians to apply intensive care only to those who are likely to benefit from it;
- Critical decisions include interrupting life-support treatments and writing do-not-resuscitate orders;
- Under this context, **Mortality Assessment** is a crucial task;

SAPS II Prognostic Model

- The SAPS II is the most widely European score for mortality assessment, ranging from 0 to 163 (highest death probability);
- The model encompasses a total of 17 variables (e.g. age, previous health status or diagnosis), which are collected within the first 24 hours of the patient's internment;
- Some of these variables imply the use of clinical tests (e.g. blood samples) which require costs and time;
- The prognostic model is given by a **Logistic Regression**:

$$\begin{aligned} logit_k &= B_0 + B_1 \times SAPSII_k + B_2 \times ln(SAPSII_k + 1) \\ pdr_k &= exp(logit_k)/(1 + exp(logit_k)) \end{aligned}$$

- Implemented in the **R** statistical environment;

Motivation

- NNs are more flexible than the Logistic Regression;
- Current prognostic models only use static data (first 24 h);
- Most ICU beds already perform automatic measurement of four biometrics: **Blood Pressure (BP)**, **Heart Rate (HR)**, **Oxygen saturation (O2)** and **Urine output (UR)**;
- Physicians can consult the history of these variables, although very often this valuable data is discarded;

Aim

- Use dynamic information defined by **events** (out of range values) obtained from the four variables;
- Adopt a KDD/Data Mining (and in particular NN) based approach for ICU mortality assessment;

Data Collection

- A **EURICUS II** derived database was adopted, with 13165 records of patients from 9 EU countries, during 10 months, from 1998 until 1999;
- Data manually collected by the nursing staff (every hour);
- The whole data was gathered at the Health Services Research Unit of the Groningen University Hospital, the Netherlands;
- Each example reports over a patient's full length of stay;

Preprocessing

- After a consult with ICU specialists, the patients with age lower than 18, burned or bypass surgery were discarded, remaining a total of 13165 records;
- Four entries discarded due to the presence of missing values;
- The event based variables were transformed into daily averages;

Clinical Data Attributes

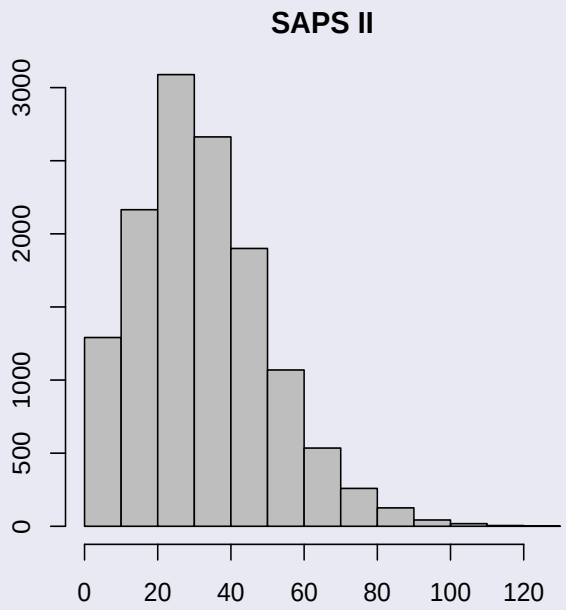
Attribute	Description	Domain
SAPS II	SAPS II score	$\{0, 1, \ldots, 163\}$
age	Patients' age	$\{18, 19, \ldots, 100\}$
admtype	Admission type	$\{1, 2, 3\}^a$
admfrom	Admission origin	$\{1, 2, \ldots, 7\}^b$
NBP	Daily number of blood pressure events	$[0.0, \ldots, 33.0]$
NCRBP	Daily number of critical blood pressure events	$[0.0, \ldots, 6.0]$
NHR	Daily number of heart rate events	$[0.0, \ldots, 42.0]$
NCRHR	Daily number of critical heart rate events	$[0.0, \ldots, 6.0]$
NO2	Daily number of oxygen events	$[0.0, \ldots, 28.0]$
NCRO2	Daily number of critical oxygen events	$[0.0, \ldots, 6.0]$
NUR	Daily number of urine events	$[0.0, \ldots, 38.0]$
NCRUR	Daily number of critical urine events	$[0.0, \ldots, 8.0]$
TBP	Daily time of blood pressure events	$[0.0, \ldots, 36.0]$
THR	Daily time of heart rate events	$[0.0, \ldots, 33.0]$
TO2	Daily time of oxygen events	$[0.0, \ldots, 33.0]$
TUR	Daily time of urine events	$[0.0, \ldots, 40.0]$
death	The occurrence of death	$\{0, 1\}^c$

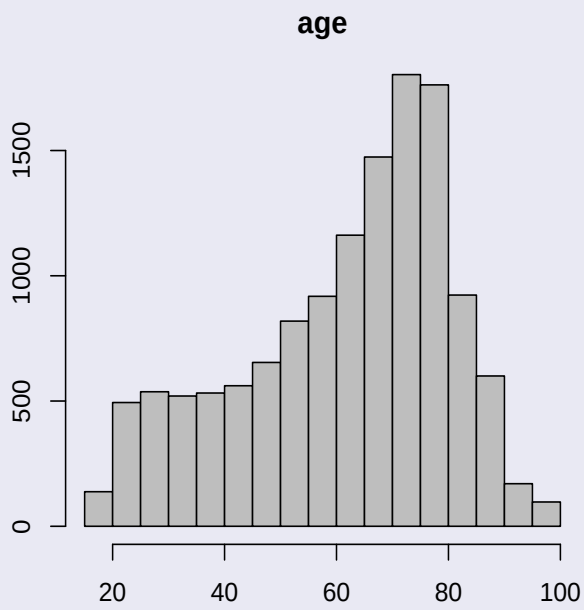
^{a} : 1 - Non scheduled surgery, 2 - Scheduled surgery, 3 - Physician.

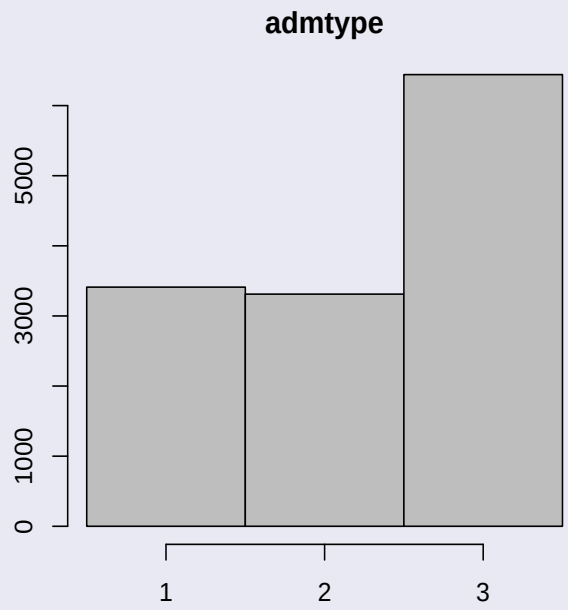
^{b} : 1 - Surgery block, 2 - Recovery room, 3 - Emergency room, 4 - Nursing room, 5 - Other ICU, 6 - Other hospital, 7 - Other

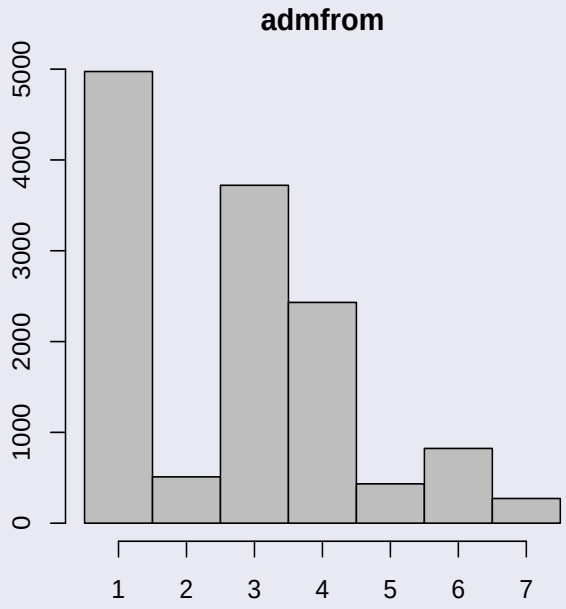
^{c} : 0 - No death, 1 - Death.

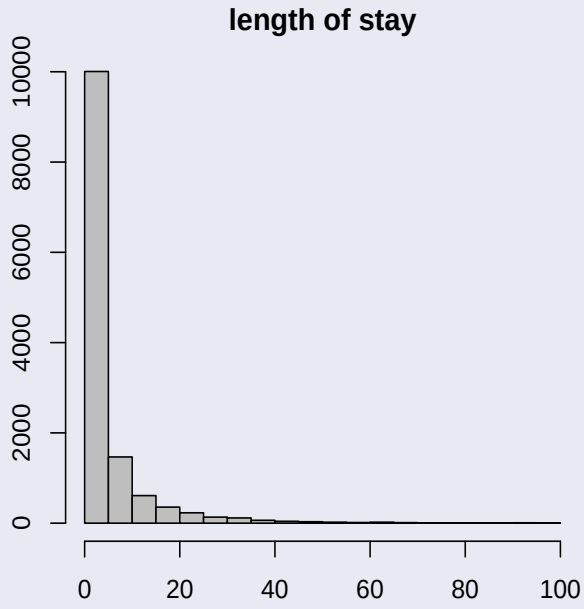
Clinical Data Histograms

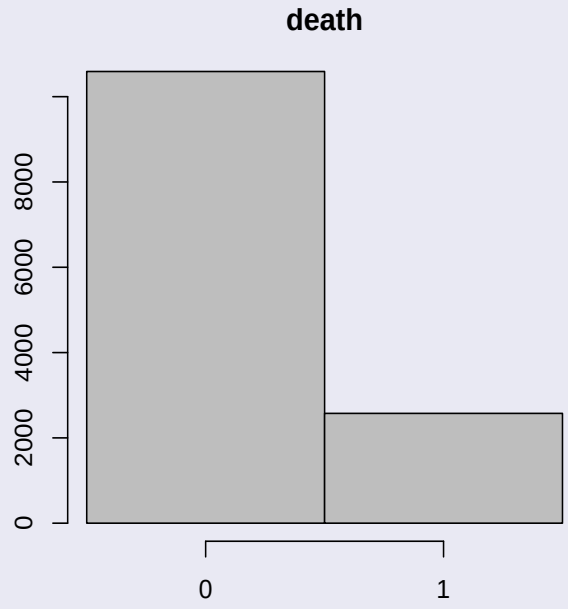












Neural Network Design

- The input data was preprocessed with a $max - min$ scaling within $[-1.0, 1.0]$ and a **1-of-C** coding for the nominal attributes (e.g. **admtype**);
- The output was preprocessed to the values: 0 - no death, 1 - death;
- Predicted class given by the nearest value to the decision threshold D;
- Fully connected MLPs with **bias** connections, **one hidden layer** (with a fixed number of hidden nodes: $input\ nodes/2$) and **logistic** activation functions;
- RPROP training, random initialization within $[-1.0, 1.0]$, stopped after 100 epochs;
- Implemented in a **JAVA** package developed by the authors.

Performance Assessment

- **Metrics: Sen.** (sensitivity), **Spe.** (specificity), **Acc.** (accuracy) and *AUC* (ROC area);
- **Validation:** Stratified **hold-out** with 2/3 training and 1/3 test sizes;
- 30 runs for each model;
- $D = 0.5$ (the middle of the interval);

Training Results

Setup	*Acc*	*Sen*	*Spe*
Normal	**85.75**±0.10	41.78±0.34	**96.43**±0.08
Balanced	78.22±0.20	76.45±0.37	79.99±0.36
Log Balanced	79.87±0.19	**78.99**±0.22	80.76±0.26

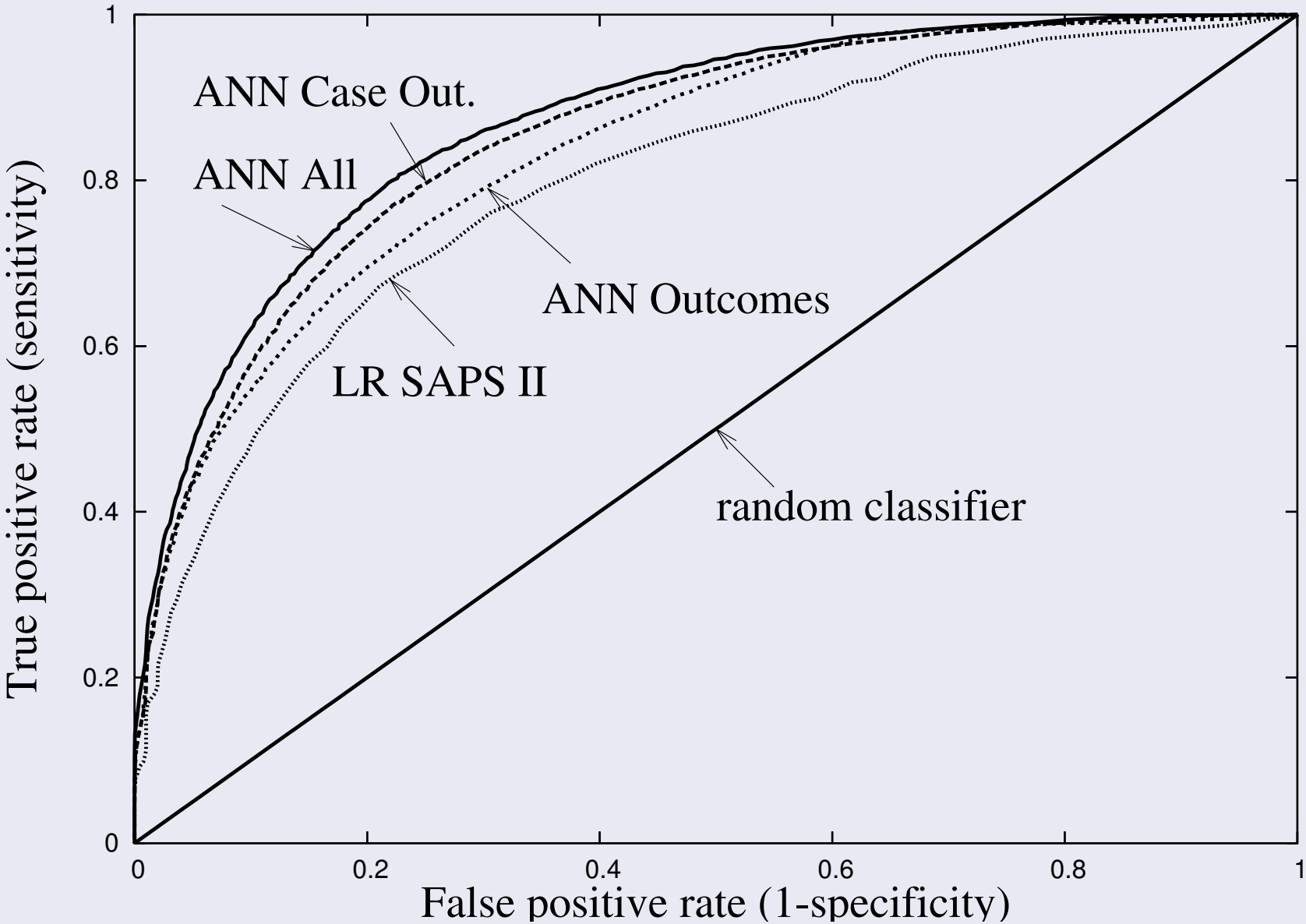
Balanced training with under sampling (equal number of true/false cases);
Log – application of the transform $y = \log(x + 1)$ in the event variables;

Test Set Performances

Setup	Acc	Sen	Spe	AUC
ANN All	79.21±0.24	78.11±0.51	79.48±0.35	87.12±0.21
ANN Case Out.	78.22±0.26	75.78±0.66	78.82±0.36	85.52±0.20
ANN Outcomes	77.60±0.31	70.00±0.59	79.45±0.48	83.88±0.23
LR All	75.97±0.29	77.06±0.68	75.71±0.40	85.17±0.27
LR Case Out.	77.15±0.26	70.63±0.67	78.73±0.41	83.78±0.24
LR Outcomes	77.57±0.15	65.91±0.62	80.41±0.21	82.59±0.23
LR SAPS II	82.60±0.14	42.57±0.50	92.33±0.12	79.84±0.26
LR SAPS II B	69.37±0.32	77.57±0.64	67.37±0.48	80.04±0.25

ALL - All inputs except the age;
Case Outcomes - All inputs except the SAPS II;
Outcomes - Only uses the intermediate outcomes;
LR SAPS II - the most used European prognostic model;
LR SAPS II B - equal to previous model, except that the parameters are calibrated to a balanced training.

ROC (test set)


ANN Case Out.
ANN All
ANN Outcomes
LR SAPS II
random classifier
True positive rate (sensitivity)
False positive rate (1-specificity)
0
0.2
0.4
0.6
0.8
1

Sensitivity Analysis [Kewley et al., 2000]

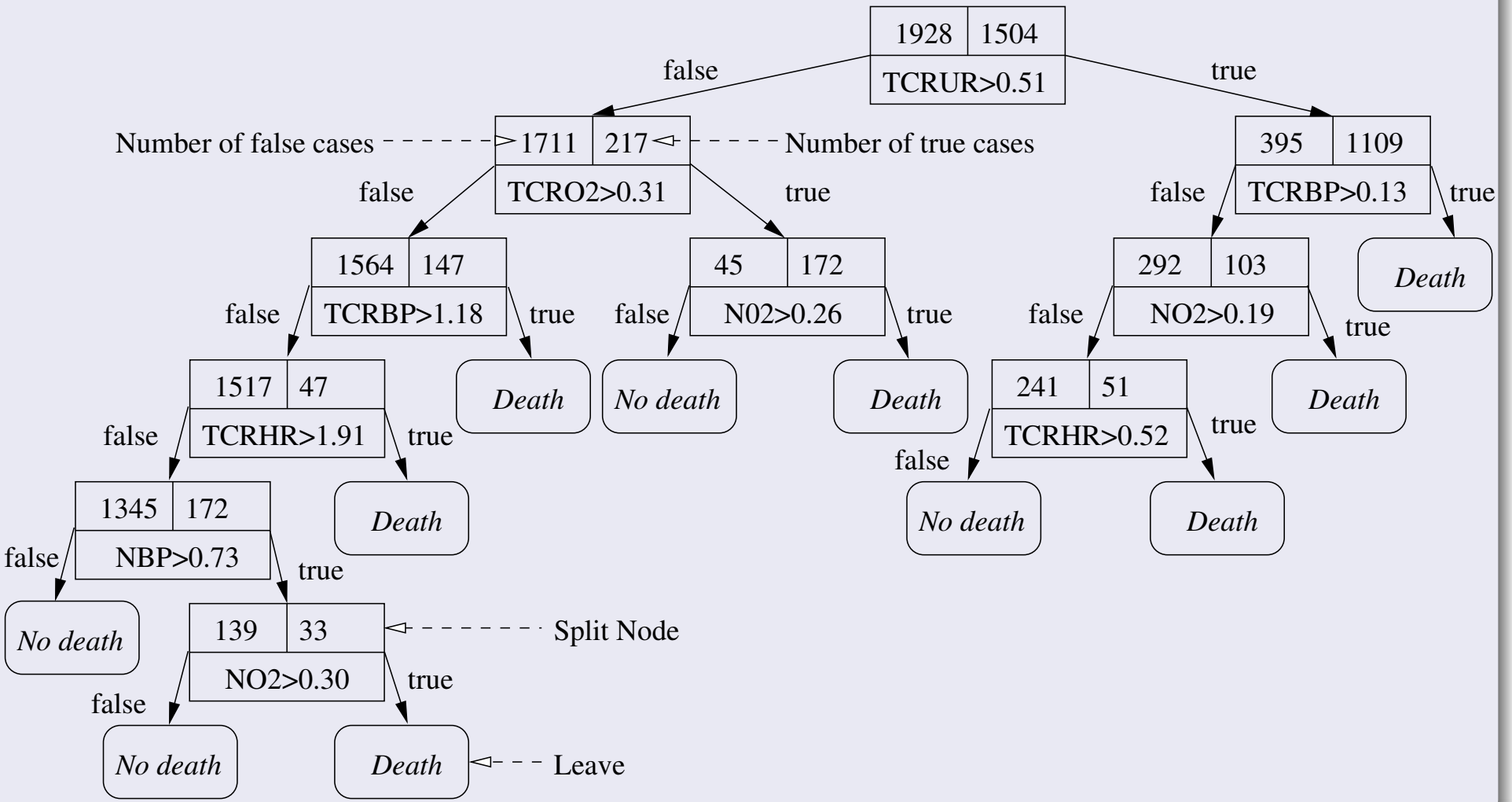
- It can be measured as the variance produced in the output of a trained NN when a given input attribute is moved through its entire range, while the remaining inputs are kept at their average values;

Input Importance by Sensitivity Analysis

Setup	SAPSII	age	admtype	admfrom	BP⋆	HR⋆	O2⋆	UR⋆
All	16.8	–	1.0	2.0	15.9	14.4	30.7	19.2
Case Outcomes	–	14.3	5.9	11.7	13.1	16.7	22.5	15.8
Outcomes	–	–	–	–	16.9	15.8	21.8	**45.5**

⋆ All attributes related to the variable where included: number of events/critical events and the time.

Extraction of Rules using a Decision Tree



Conclusions

- The event based models are more accurate than the static models (SAPS II);
- With the same inputs, the NNs outperform the Logistic Regression;
- Off-line learning study (data manually collected). However, there is a potential for real-time and low cost modeling;
- **Future work**: test this approach in a real environment with an on-line learning (pilot project **INTCare**, Hospital S. António);
- Some important issues:
  - Data cleansing and validation by ICU staff;
  - Study the concept of **lead time**: how soon is it possible to predict death and with which accuracy?

Artificial Intelligence Medicine journal article available at:

http://www.dsi.uminho.pt/~pcortez/death4.pdf

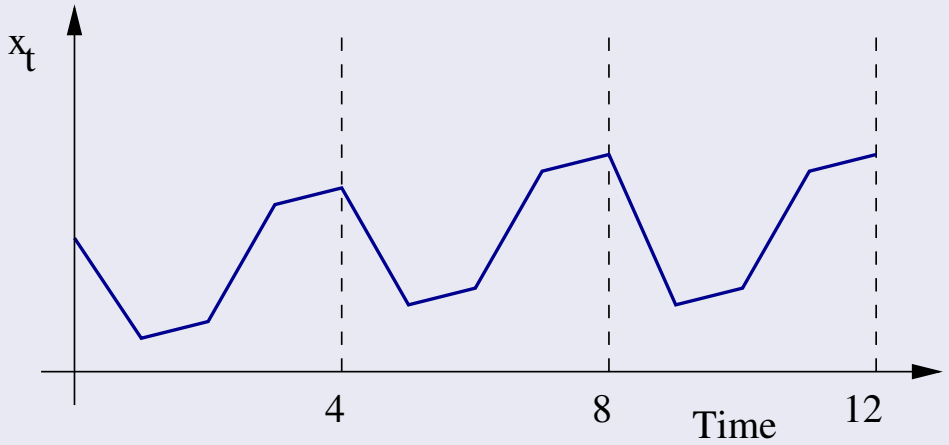
Case Study II: Internet Traffic Forecasting (Regression) [Cortez et al., 2006b]

Motivation

- TCP/IP **traffic forecasting** is a crucial task for any medium/large **Internet Source Provider (ISP)** and it has received little attention from the computer networks community;
- With better forecasts, the resources of the network can be **optimized**;
- Traffic forecasting can also help to **detect anomalies** in the networks. Security attacks like Denial-of-Service, viruses, or even an irregular amount of SPAM can in theory be detected by comparing the real traffic with the predicted values;
- Nowadays, this task is often done **intuitively** by experienced network administrators;
- Yet, the **Operational Research** and **Computer Science** disciplines led to solid forecasting methods (e.g. ARIMA) that replaced intuition approaches in several fields (e.g. agriculture or economy);

Time Series Forecasting (TSF)

- A **Time Series** contains **time ordered** observations of an event:



- **Time Series Forecasting (TSF)** uses past patterns to predict the future;
- Several **TSF** methods available:
  - **Exponential Smoothing (ES)** (e.g. Holt-Winters) [Winters, 1960];
  - **Box-Jenkins** methodology (ARIMA) [Box and Jenkins, 1976];
  - **Neural Networks (NNs)** [Cortez et al., 2005].
- The forecast **horizon** (or **lead time**) is defined by the time in advance that a forecast is issued;

Aim

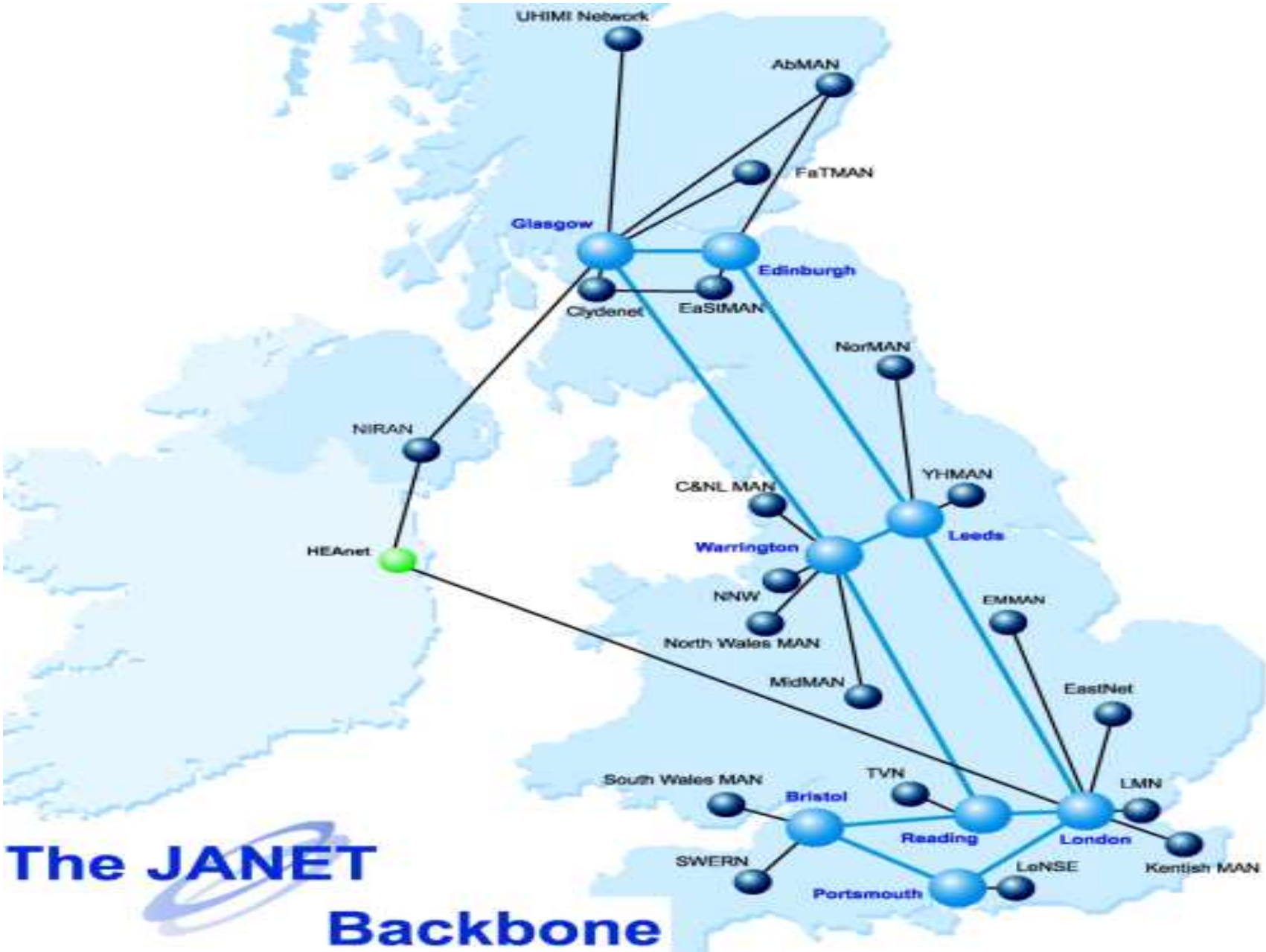
- Use the **Simple Network Management Protocol (SNMP)** that quantifies the traffic passing through every network interface with reasonable accuracy. Furthermore, SNMP does not induce extra traffic on the network;
- Forecast Internet traffic with a pure TSF approach (i.e., only past values are used as inputs);
- The predictions are analyzed at different time scales (e.g. five minutes, hourly) and considering distinct lookahead horizons (from 1 to 24);
- Test several TSF methods: Holt-Winters (both traditional and recent double seasonal versions), the ARIMA methodology and a NN based approach;

Internet Traffic Data

- This work analyzed traffic data (in bits) from two different ISPs: **A** and **B**;
- The **A** dataset belongs to a private ISP with centers in 11 European cities. The data corresponds to a transatlantic link and was collected from 6:57 AM 7th June 2005;
- Dataset **B** comes from UKERNA[a] and represents aggregated traffic in the Janet[b] (the UK academic network) backbone. It was collected between 9:30 AM, 19th November 2004 and 11:11 AM, 27th January 2005;
- The **A** time series was registered every 30 seconds, while the **B** data was recorded at a five minute period;
- The first series (**A**) included 8 missing values, which were replaced by using a regression imputation (e.g. linear interpolation);

[a] United Kingdom Education and Research Networking Association.
[b] http://www.ja.net

JANET Backbone (UKERNA)

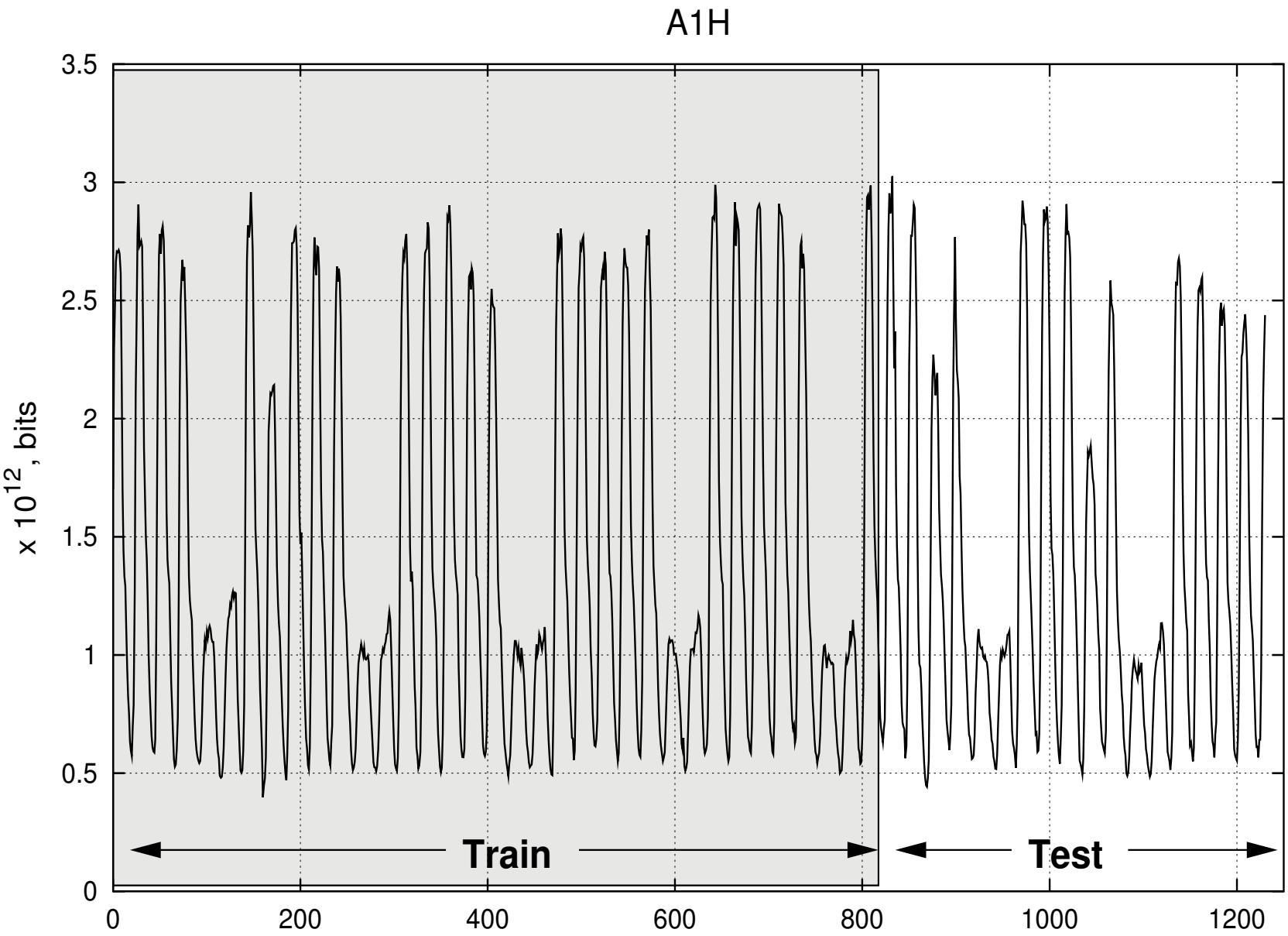


Internet Traffic Data (cont.)

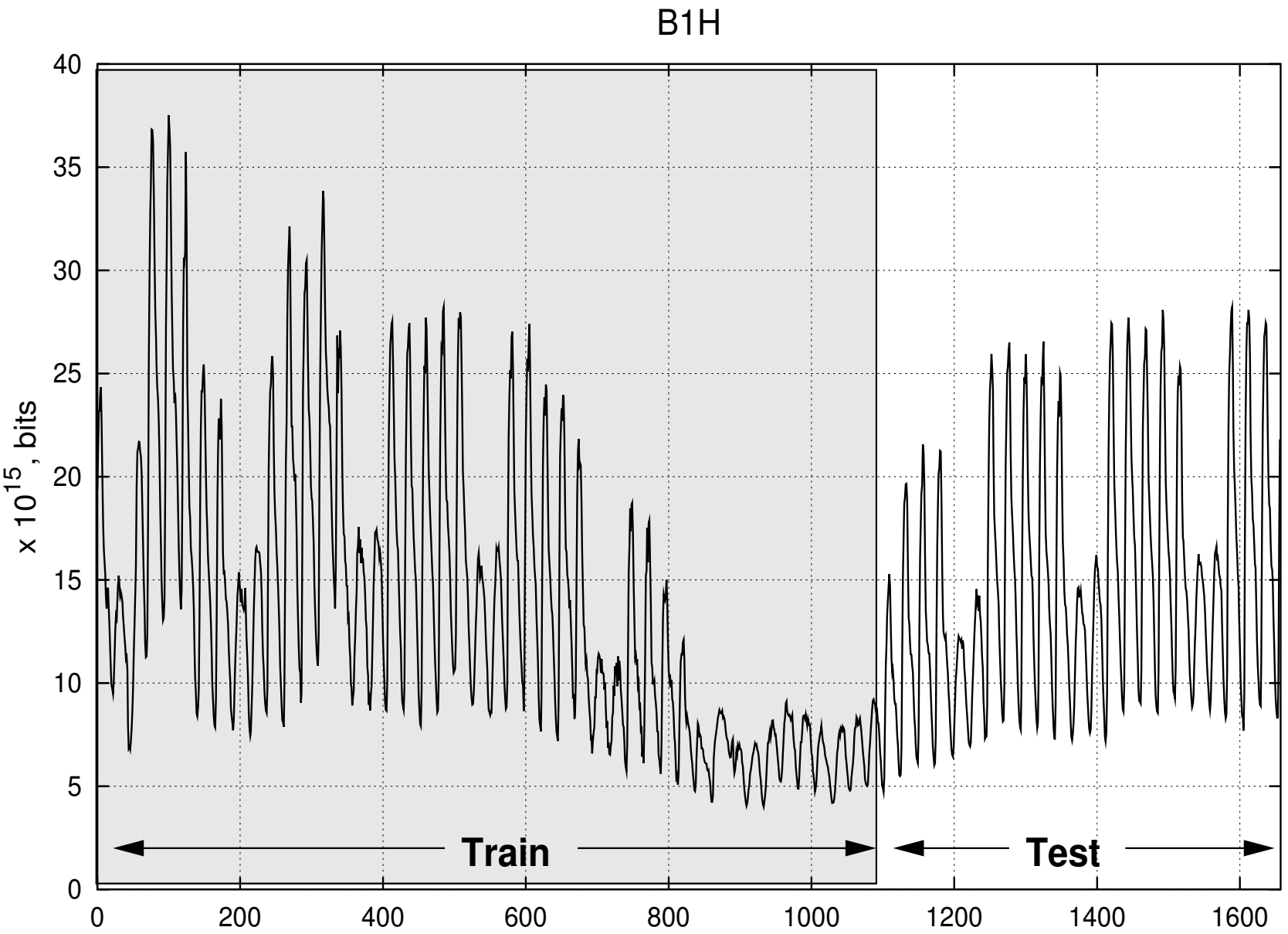
- Two new time series were created for each ISP by aggregating the original values;
- The selected time scales were every five minutes (**A5M** and **B5M**) and every hour (**A1H** and **B1H**);
- For each series, the first 2/3 of the data will be used to create the forecasting models (**train**) and the remaining last 1/3 to evaluate (test) the forecasting accuracies (fixed **hold-out**);

Series	Time Scale	Train Length	Test Length	Total Length
A5M	5 min.	9848	4924	14772
B5M	5 min.	13259	6629	19888
A1H	1 hour	821	410	1231
B1H	1 hour	1105	552	1657

Hourly A Data

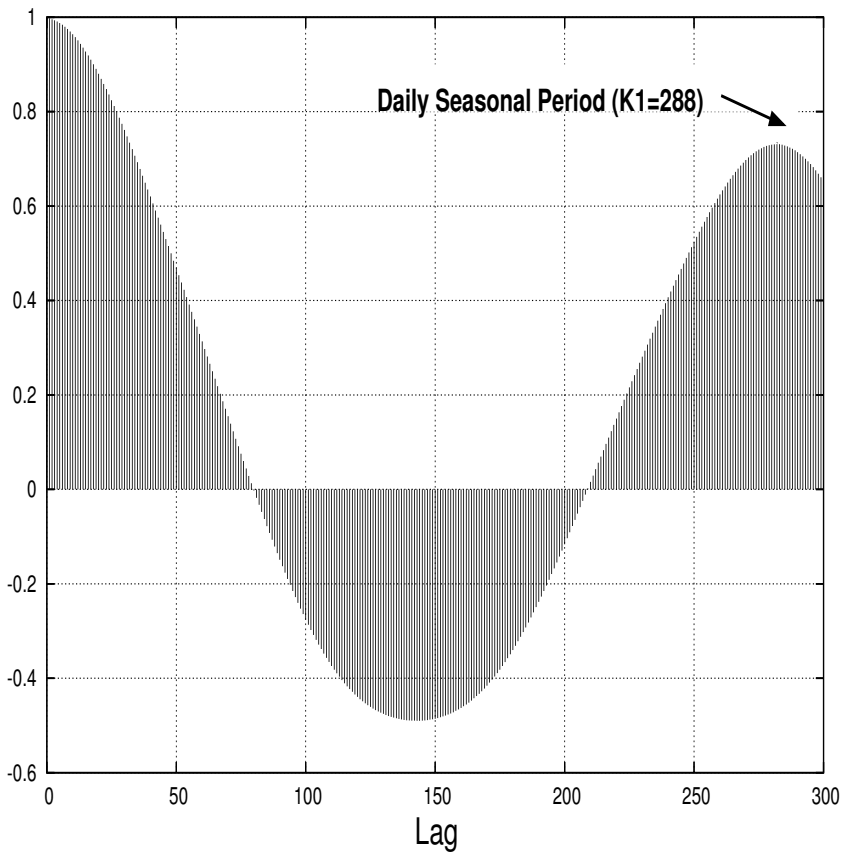


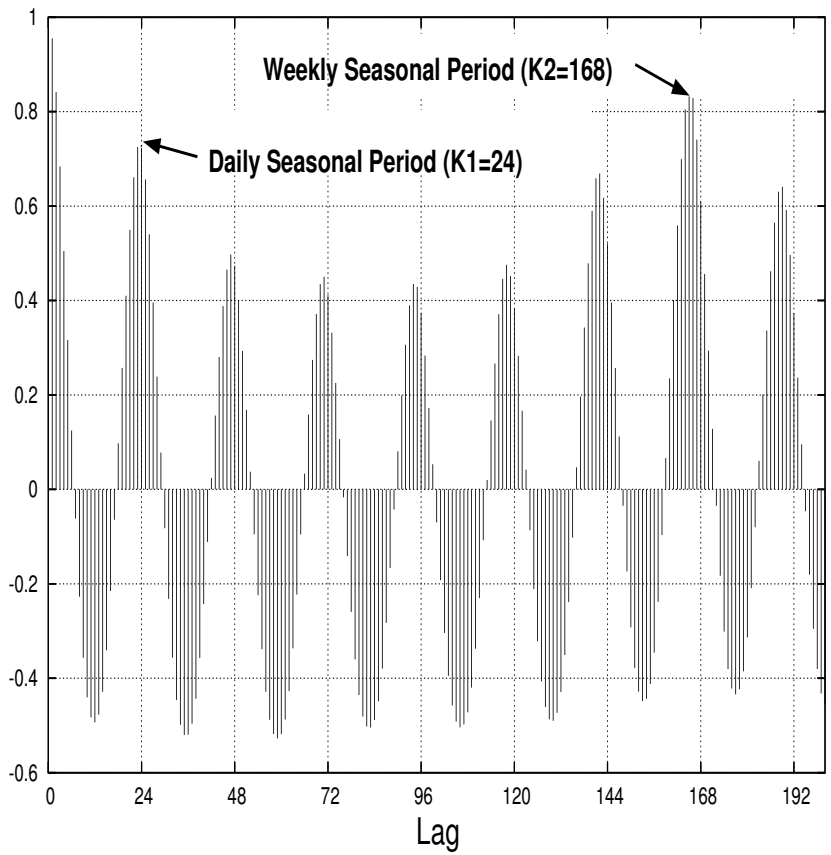
Hourly B Data



Time Series Decomposition (A5M and A1H autocorrelations)

Two seasonal cycles: intraday ($K_1 = 288/24$) and intraweek ($K_2 = 168$);





Naive Benchmark

- The **seasonal version** of the random walk is used. Forecast given by the observed value for the same period related to the previous longest seasonal cycle (last week).

Holt-Winters Method (Exponential Smoothing)

- Very popular and simple predictive model, based on some underlying patterns such as **trend** and **seasonal** components (K_1) [Winters, 1960];
- Three parameters: α, β and γ;
- This model has been extended to encompass **two seasonal cycles** (K_1, K_2 and four parameters: α, β, γ and ω) [Taylor, 2003];
- The parameters will be optimized by a 0.01/0.05 grid-search for the best training error;

ARIMA Methodology [Box and Jenkins, 1976]

- An important forecasting approach, going over model identification, parameter estimation, and model validation.
- The global model is based on a linear combination of past values (AR) and errors (MA):

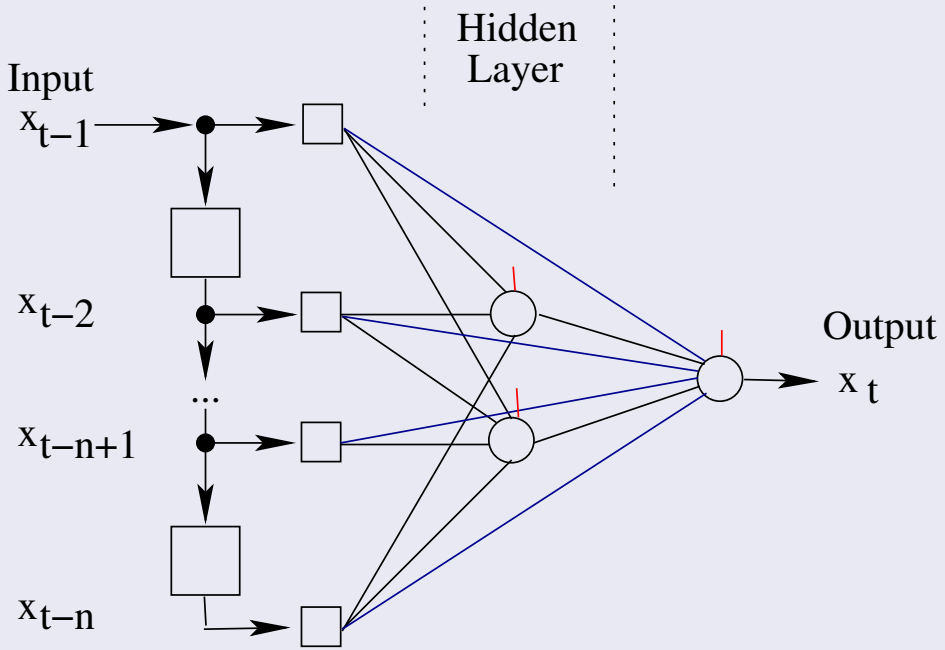
$ARIMA(p, d, q)$	non seasonal model
$ARIMA(p, d, q)(P_1, D_1, Q_1)$	seasonal model
$ARIMA(p, d, q)(P_1, D_1, Q_1)(P_2, D_2, Q_2)$	double seasonal model

- Since the model selection is a non-trivial process, the forecasting Fortran package **X-12-ARIMA** [Findley et al., 1996], from the U.S. Bureau of the Census, was used for the model and parameter estimation phases;
- The **BIC** statistic, which penalizes model complexity and is evaluated over the training data, will be the criterion for the model selection;

Neural Network (NN) Design

- Multilayer perceptrons with **one hidden layer**, **logistic** functions on the **hidden nodes**, **linear function** in the output node, **bias** and **shortcut** connections;
- The NN initial weights are randomly set within the range $[-1.0; 1.0]$ and the RPROP algorithm is used for training (stopped at 1000 epochs);
- A **NN Ensemble (NNE)** is used, where $R = 5$ different networks are trained and the final prediction is given by the average of the individual predictions;
- The NNE based forecasting method will depend solely on two parameters: the **input time lags** and **number of hidden nodes** (H);

Neural Network (NN) Design (cont.)



- The inputs will be defined by a sliding time window (with several time lags);
- Example: with the series $1, 2, 3, 4, 5, 6$ and time lags $\{1, 2, 4\}$, the following examples can be built: $1, 3, 4 \rightarrow 5$ and $2, 4, 5 \rightarrow 6$;

Neural Network (NN) Design (cont.)

- The **training data** will be divided into **training** (2/3) and **validation** sets (1/3);
- Several NN combinations were tested, using $H \in \{0, 2, 4, 6, 8\}$ and:

Scale	Time Lags
5min.	{1,2,3,5,6,7,287,288,289}
	{1,2,3,5,6,7,11,12,13}
	{1,2,3,4,5,6,7}
1hour	{1,2,3,24,25,26,168,167,169}
	{1,2,3,11,12,13,24,25,26}
	{1,2,3,24,25,26}

- The NN with the lowest validation error (average of all $MAPE_h$ values) will be selected. After this model selection phase, the final NNs are retrained with all training data.

Implementation

All methods (except ARIMA model estimation) implemented in a **JAVA** package developed by the authors;

Best Neural Models

Series	Hidden Nodes (H)	Input Time Lags
A5M	6	{1,2,3,5,6,7,11,12,13}
A1H	8	{1,2,3,24,25,26}
B5M	0	{1,2,3,5,6,7,287,288,289}
B1H	0	{1,2,3,24,25,26,168,167,169}

- The number of hidden nodes suggest that the **A** datasets are nonlinear while the data from the ISP **B** are linear.

Performance Assessment

- The **Mean Absolute Percentage Error (MAPE)** is a popular metric, with the advantage of being scale independent:

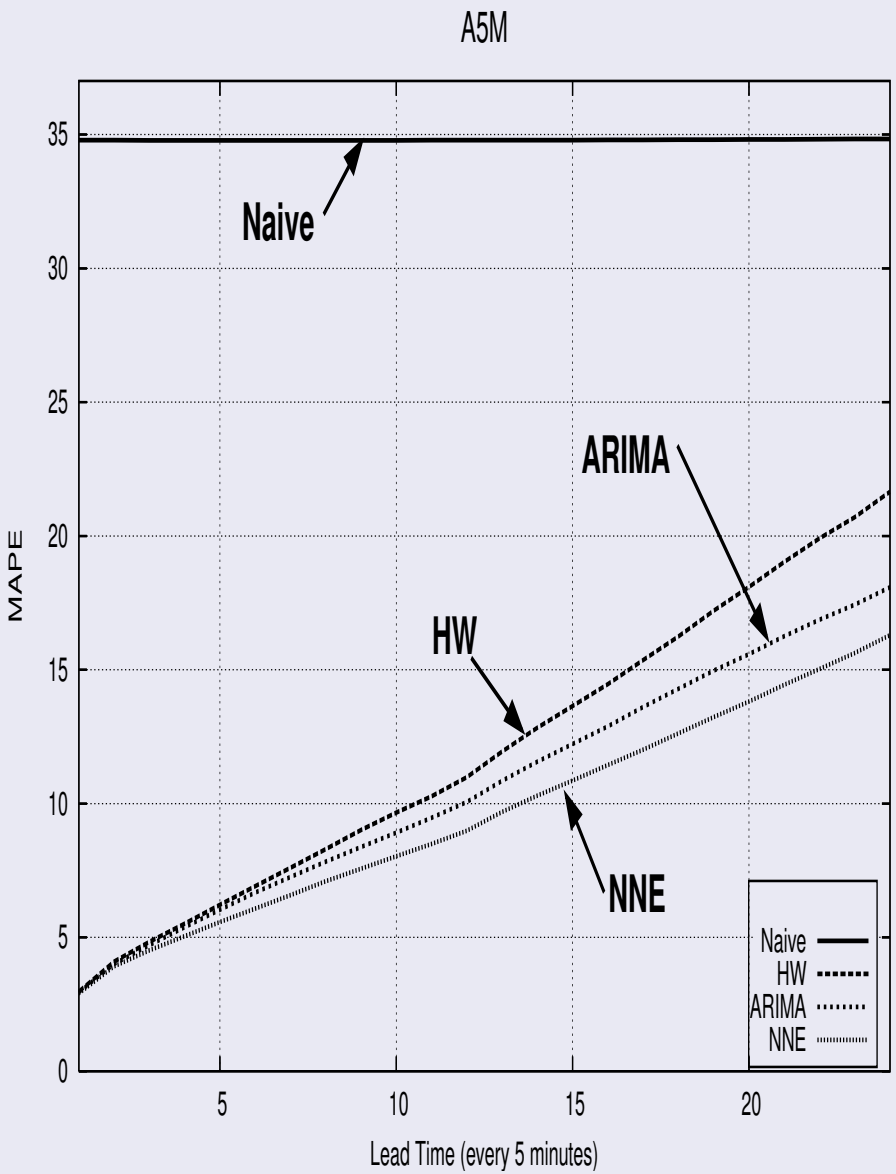
$$
\begin{aligned}
e_t &= y_t - \widehat{y}_{t,t-h} \\
MAPE_h &= \sum_{i=P+1}^{P+N} \frac{|y_i - \widehat{y}_{i,i-h}|}{y_i \times N} \times 100\%
\end{aligned}
\tag{2}
$$

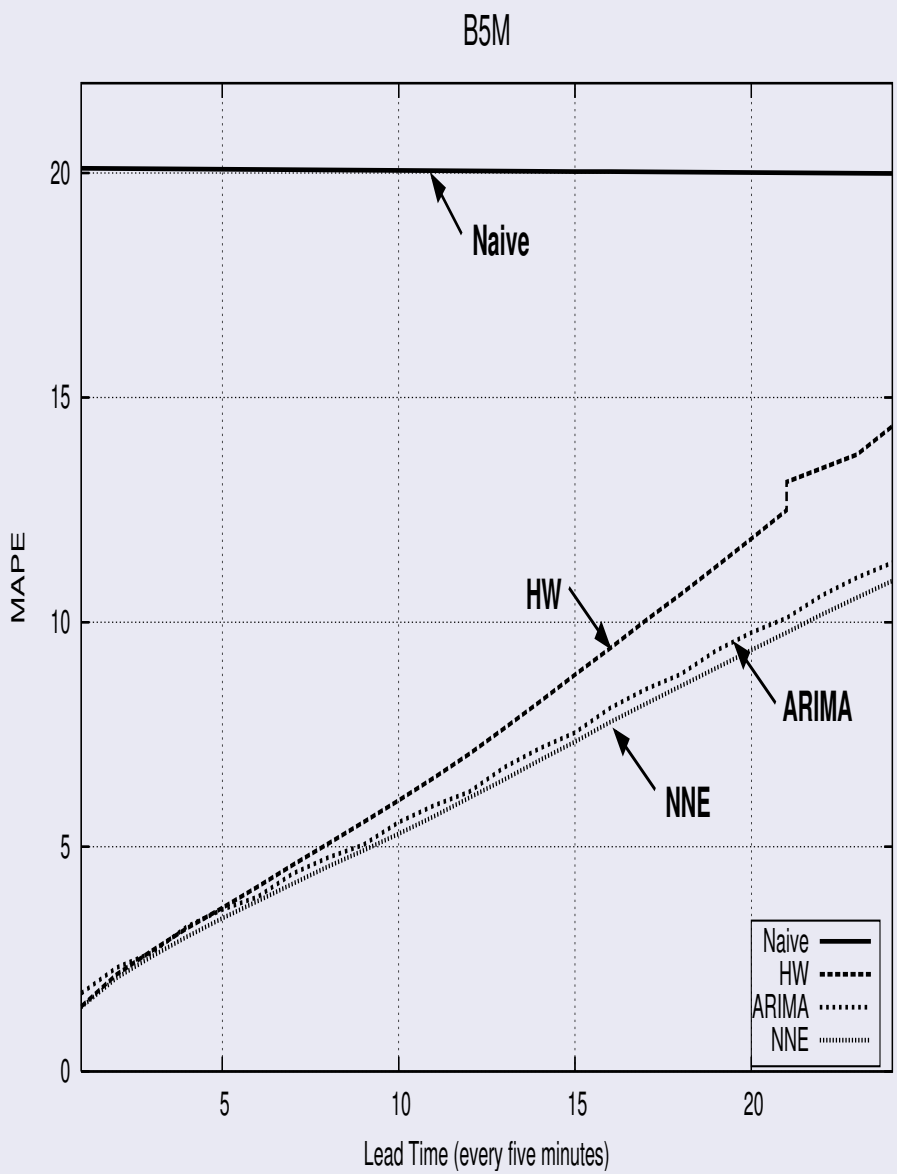
where e_t denotes the forecasting error at time t; y_t the desired value; $\widehat{y}_{t,p}$ the predicted value for period t and computed at period p; P is the present time and N the number of forecasts.

Test Results (Average $MAPE_h$ values, $h \in \{1, \ldots, 24\}$)

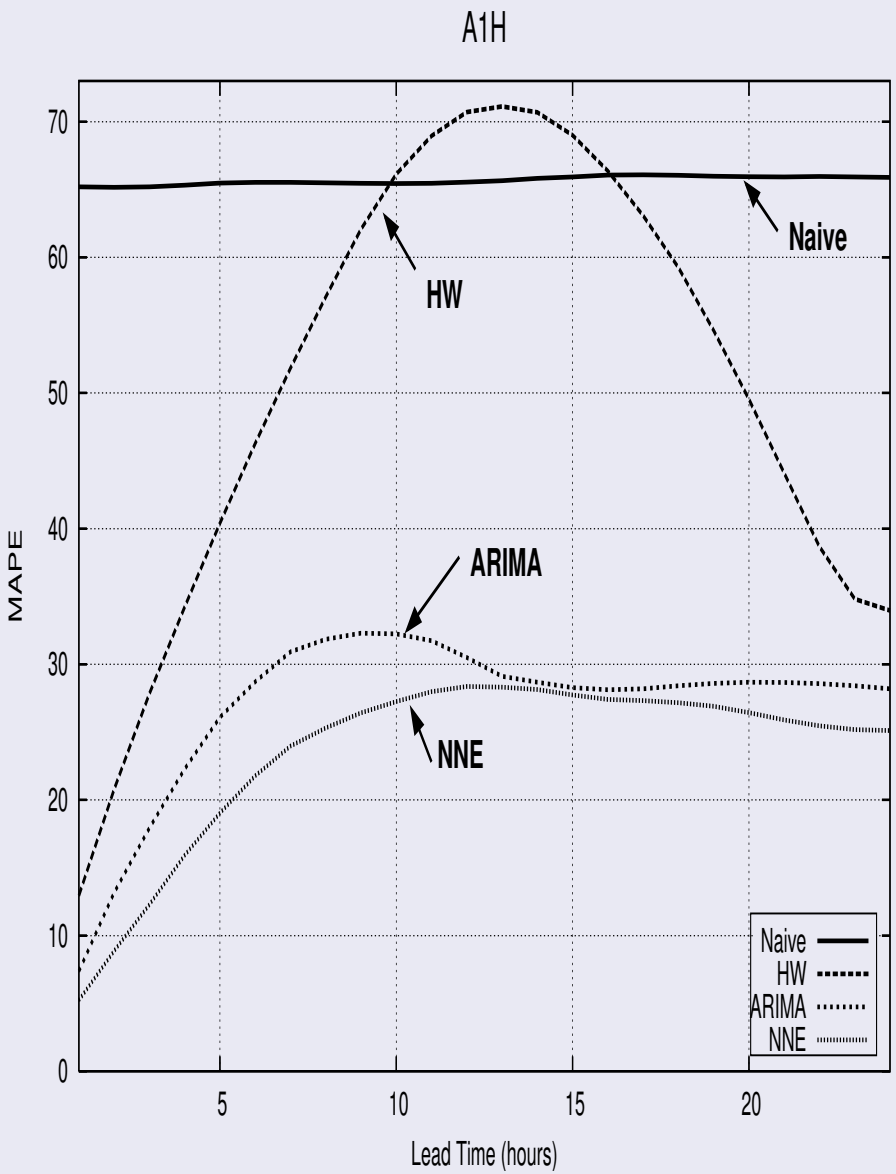
Series	Naive	Holt-Winters	ARIMA	NNE
A5M	34.80	11.98	10.68	**9.59**$\pm$0.08
B5M	20.05	7.65	6.60	**6.34**$\pm$0.11
A1H	65.67	50.60	26.96	**23.48**$\pm$0.49
B1H	35.18	13.69	12.69	**12.26**$\pm$0.03

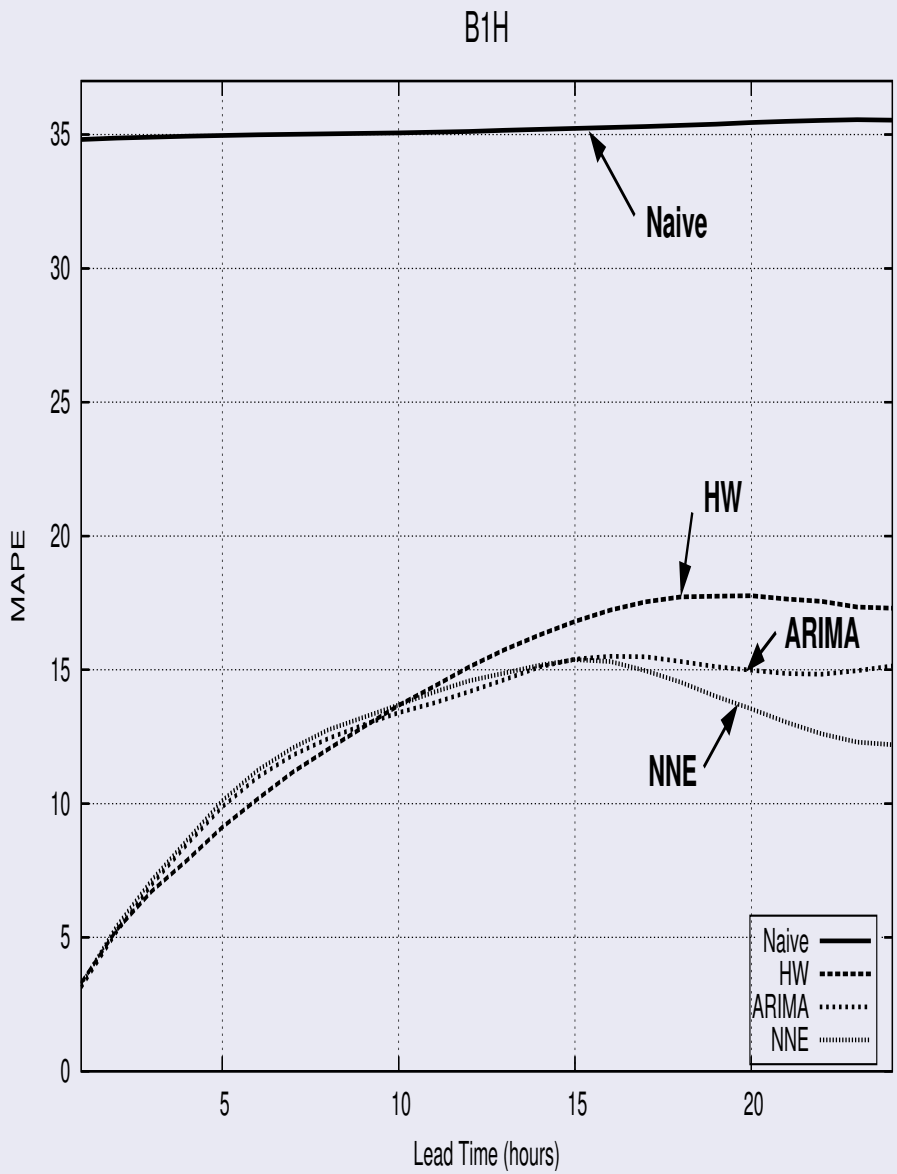
Five Minute Test Results (horizon vs MAPE values)





Hourly Test Results (horizon vs MAPE values)





Conclusions

- The experimental results reveal promising performances:
  - **Real-time**: 1–3% error for five minute lookahead forecasts and 11–17% error for 2 hours in advance;
  - **Short-term**: 3–5% (one hour ahead) to 12–23% (1 day lookahead);
- Both ARIMA and NNE produce the lowest errors for the five minute and hourly data (NN is the **best** for A5M, B5M and A1H);
- The computational effort criterion discards the ARIMA methodology, which is impractical for on-line forecasting systems [Taylor et al., 2006]. Thus, we **advise the use of the NNE approach**, which can be used in **real-time**;
- **Future Work**:
  - Apply similar methods to active measurement scenarios in which real-time packet level information is fed into the forecasting engine;
  - Forecast of traffic demands associated with specific Internet applications (e.g. peer to peer);

At last, a nice demo...

Artificial Life Environment (reinforcement learning)

- There is energy (e.g. grass) all over the field;
- Two populations of beings: preys and predators;
- Each artificial being is modeled by a MLP:
  - Inputs: **vision** within a certain range and angle (nothing, prey or predator);
  - Outputs: **actions** (nothing, move forward, rotate left or right);
  - Weights are directly coded into chromosomes, using real-valued representations;
- Evolutionary learning;


Bi, J. and Bennett, K. (2003).
Regression Error Characteristic curves.
In *Proceedings of 20th Int. Conf. on Machine Learning (ICML)*, Washington DC, USA.


Bishop, C. (1995).
Neural Networks for Pattern Recognition.
Oxford University Press.


Box, G. and Jenkins, G. (1976).
Time Series Analysis: Forecasting and Control.
Holden Day, San Francisco, USA.


Brown, M. and Kros, J. (2003).
Data mining and the impact of missing data.
Industrial Management & Data Systems, 103(8):611–621.


Cortez, P., Portelinha, M., Rodrigues, S., Cadavez, V., and Teixeira, A. (2006a).
Lamb Meat Quality Assessment by Support Vector Machines.
Neural Processing Letters.


Cortez, P., Rio, M., Rocha, M., and Sousa, P. (2006b).
Internet Traffic Forecasting using Neural Networks.
In *Proceedings of the 2006 International Joint Conference on Neural Networks (IJCNN 2006)*, Vancouver, Canada. IEEE.


Cortez, P., Rocha, M., and Neves, J. (2005).
Time Series Forecasting by Evolutionary Neural Networks.
chapter III, *Artificial Neural Networks in Real-Life Applications*, Idea Group Publishing, USA, pages 47–70.


Fawcett, T. (2003).
Roc graphs: Notes and practical considerations for data mining researchers.
Technical Report HPL-2003-4, HP Laboratories Palo Alto.


Fayyad, U., Piatetsky-Shapiro, G., and Smyth, P. (1996).
Advances in Knowledge Discovery and Data Mining.
MIT Press.


Findley, D., Monsell, B., Bell, W., Otto, M., and Chen, B. (1996).

New Capabilities and Methods of the X-12 Seasonal Adjustment Program.
Journal of Business and Economic Statistics.


Flexer, A. (1996).
Statistical evaluation of neural networks experiments: Minimum requirements and current practice.
In *Proceedings of the 13th European Meeting on Cybernetics and Systems Research*, volume 2, pages 1005–1008, Vienna, Austria.


Hand, D., Mannila, H., and Smyth, P. (2001).
Principles of Data Mining.
MIT Press, Cambridge, MA.


Hastie, T., Tibshirani, R., and Friedman, J. (2001).
The Elements of Statistical Learning: Data Mining, Inference, and Prediction.
Springer-Verlag, NY, USA.


Kewley, R., Embrechts, M., and Breneman, C. (2000).

Data Strip Mining for the Virtual Design of Pharmaceuticals with Neural Networks.
IEEE Trans. on Neural Networks, 11(3):668–679.


Kohavi, R. and Provost, F. (1998).
Glossary of Terms.
Machine Learning, 30(2/3):271–274.


Pyle, D. (1999).
Data Preparation for Data Mining.
Morgan Kaufmann, S. Francisco CA, USA.


Riedmiller, M. (1994).
Supervised Learning in Multilayer Perceptrons - from Backpropagation to Adaptive Learning Techniques.
Computer Standards and Interfaces, 16.


Rocha, M., Cortez, P., and Neves, J. (2003).
Evolutionary Neural Network Learning.

In Pires, F. and Abreu, S., editors, *Progress in Artificial Intelligence, EPIA 2003 Proceedings, LNAI 2902*, pages 24–28, Beja, Portugal. Springer.


Rocha, M., Cortez, P., and Neves, J. (2006).
Evolution of Neural Networks for Classification and Regression.
Neurocomputing.


Sarle, W. (2005).
Neural Network Frequently Asked Questions.
Available from ftp://ftp.sas.com/pub/neural/FAQ.html.


Silva, A., Cortez, P., Santos, M. F., Gomes, L., and Neves, J. (2006).
Mortality assessment in intensive care units via adverse events using artificial neural networks.
Artificial Intelligence in Medicine.


Taylor, J. (2003).
Short-Term Electricity Demand Forecasting Using Double Seasonal Exponential Smoothing.
Journal of Operational Research Society, 54:799–805.


Taylor, J., Menezes, L., and McSharry, P. (2006).
A Comparison of Univariate Methods for Forecasting Electricity Demand Up to a Day Ahead.
Int. Journal of Forecasting, In press.


Tickle, A., Andrews, R., Golea, M., and Diederich, J. (1998).
The Truth Will Come to Light: Directions and Challenges in Extracting the Knowledge Embedded Within Trained Artificial Neural Networks.
IEEE Transactions on Neural Networks, 9(6):1057–1068.


Winters, P. R. (1960).
Forecasting sales by exponentially weighted moving averages.
Management Science, 6:324–342.


Witten, I. and Frank, E. (2005).
Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations.
Morgan Kaufmann, San Francisco, CA.