APPLIED STATISTICS

DATA MINING

COMPUTER SCIENCE

- Statistics plays a major role in data mining. However, computer scientists are leading changes and adapting methods to new problems and new environments.
- Contributions and enhancements within computer science:

1. Flexible predictive modeling methods (ANN, Decision Rules) – strong algorithmic foundation but weaker formal statistical justification.

2. The use of hidden variable models (ANN, Markov models) – data can be “compressed” into a relatively simple hidden state representation.


4. Analyze and discovery in heterogeneous data types – (multimedia – images, audio, video; Web and text documents).

5. The engineering of scale – scaling traditional (or modified) algorithms to handle massive data sets.
Tabular representation of a data set

FEATURES

SAMPLES

Feature’s value for the given sample

Features = Variables

- independent variables (input variables) – X
- dependent variables (output variables) - Y

DATA MINING ROOTS

1. **Statistics** - driven by the notion of a *model*,

2. **Machine learning** - emphasize *algorithms*,

3. **Data bases** - concentration on large amount of *data*.

4. **Control theory**
   - To predict a system’s behavior.
   - To explain the interaction and relationships between variables of a system.
DATA QUALITY

Requirements:

1. The data should be accurate.
2. The data should be stored according to data types.
3. The data has integrity.
4. The data is consistent.
5. The data is not redundant.
6. The data is timely.
7. The data is well understood.
8. The data set is complete.

DATA WAREHOUSING

The data warehouse is a collection of integrated, subject-oriented databases designed to support the decision support functions (DSF), where each unit of data is relevant to some moment in time.

- Data warehouse includes different types of data:
  1. Old detail data
  2. Current (new) detail data
  3. Lightly summarized data
  4. Highly summarized data
  5. Metadata (the data directory or guide).

- Data transformation, in a preparation of a data warehouse, are standardized:
  1. Simple transformations
  2. Cleansing and scrubbing
  3. Integration
  4. Aggregation and summarization
DATA WAREHOUSING

- Data mining represents one of the major applications for data warehousing

SQL : Data Mining

SQL is a standard relational database language that is good for queries that impose some kind of constraints on data in the database in order to extract an answer.

In contrast, data mining methods are good for queries that are exploratory in nature, trying to extract hidden, not so obvious information.

OLAP : Data Mining

OLAP tools make it very easy to look at dimensional data from any angle, or to “slice-and-dice” it.

OLAP tools do not learn from data, nor do they create new knowledge. They are usually special-purpose visualization tools that can help an end user to make his/her own conclusions and decisions based on graphically condensed data.

OLAP tools are very useful for data mining process, they can be a part of it but they are not its substitute.

REPRESENTATION OF RAW DATA

Common types:

1. Numeric
   A feature with numeric values has two important properties:
   a) Order relation (for example, 2 < 5 and 5 < 7),
   b) Distance relation (for example, d(2.3, 4.2) = 1.9).

2. Categorical
   Categorical (often called symbolic) variables have neither of these two relations.
   The two values of a categorical variable can be either equal or not equal: they only support equality relation (Blue = Blue, or Red ≠ Black)
   Coded categorical variables are known as “dummy variables” in statistics:

<table>
<thead>
<tr>
<th>Feature value</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>Black</td>
<td>1000</td>
</tr>
<tr>
<td>Blue</td>
<td>0100</td>
</tr>
<tr>
<td>Green</td>
<td>0010</td>
</tr>
<tr>
<td>Brown</td>
<td>0001</td>
</tr>
</tbody>
</table>
# REPRESENTATION OF RAW DATA

Another way of variable classification:

## 1. Continuous variables
(also known as quantitative or metric variables)

These variables are measured using either:

a) *interval scale*
   The zero point in the interval scale is placed arbitrarily. (Temperatures: 40°C and 80°C)

b) *ratio scale*
   It has an absolute zero point and the ratio relation holds. (Lengths: 2 ft. and 4 ft.)

## 2. Discrete variables
(also called qualitative variables)

They use one of two kinds of non-metric scales:

a) *nominal scale*
   
   A nominal scale is an order-less scale, (A, B and C values for the variable, or ZIP-code)

b) *ordinal scale*
   
   An ordinal scale consists of ordered discrete gradations, e.g. rankings. An order relation is defined but no distance relation. (gold, silver, and bronze medal, or students ranked as 15th and 16th).

* c) *Periodic variable* is a feature for which the distance relation exists, but there is no order relation. (Days of the week, month, or year).

---

# REPRESENTATION OF RAW DATA

Another way of variable classification with respect to time:

- Some data are not changing with time and we are considered them as *a static data*.

- On the other hand, there are attribute values that change with time, and this type of data we call *dynamic or temporal data*.

- The majority of the data mining methods are more suitable for static data.
“CURSE OF DIMENSIONALITY”

- The “curse of dimensionality” is due to the geometry of high-dimensional spaces.

- The properties of high-dimensional spaces often appear counterintuitive because our experience with the physical world is in low-dimensional space such as space with two or three dimensions.

- Conceptually objects in high-dimensional spaces have a larger amount of surface area for a given volume than objects in low-dimensional spaces.

- For example, a high-dimensional hypercube, if it could be visualized, would look like a porcupine. As the dimensionality grows larger, the edges grow longer relative to the size of a central part of the hypercube.

1. A size of a data set yielding the same density of data points in n-dimensional space, increase exponentially with dimensions.

   \[
   \text{SAME DENSITY OF DATA:} \\
   \begin{array}{c|c|c}
   \text{one-dimension} & k \text{ dimensions} & \\
   n & n^k & \\
   100 & k=5: 100^5 = 10^{10} & \\
   \end{array}
   \]

2. A larger radius is needed to enclose a fraction of data points in a high-dimensional space. The edge length \( e \) of the hypercube:

   \[
   e(p) = \frac{p}{d} 
   \]

   where \( p \) is the pre-specified fraction of samples and \( d \) is the number of dimensions.

   \[
   \begin{array}{c|ccc}
   \text{10% of the samples (p=0.1):} & \text{one-dimension} & 2 \text{ dimensions} & 3 \text{ dimensions} & 10-\text{dimensions} \\
   e_1(0.1) = 0.1 & e_2(0.1) = 0.32 & e_3(0.1) = 0.46 & e_{10}(0.1) = 0.80 \\
   \end{array}
   \]
“CURSE OF DIMENSIONALITY”

2. Almost every point is closer to an edge than to another sample point in a high-dimensional space.

For a sample size \( n \), the expected distance \( D \) between data points in \( d \)-dimensional space is:

\[
D(d, n) = \frac{1}{2} \left( \frac{1}{n} \right)^{1/d}
\]

- For a two-dimensional space with 10000 points \( \rightarrow D(2,10000) = 0.0005 \)
- For a 10-dimensional space with 10000 points \( \rightarrow D(10,10000) = 0.4 \)

3. Almost every point is an outlier.

As the dimension of the input space increases, the distance between the prediction point and the center of classified points increases.

- When \( d=10 \), the expected value of the prediction point is 3.1 SD away from the center of the data belonging to one class.
- When \( d=20 \), the distance is 4.4 SD.

CHARACTERISTICS OF RAW DATA

- Missing data,
- Misrecorded data,
- Data may be from the other population, ...

Two central tasks for the preparation of data:

- To organize data into a standard form (typically a standard form is a relational table)
- To prepare data sets that lead to the best data mining performances.
TRANSFORMATION OF RAW DATA

1. Normalizations
   a) Decimal scaling: \[ v'(i) = \frac{v(i)}{10^k} \]
      for the smallest \( k \) such that \( \max |v'(i)| < 1 \).
   b) Min-max normalization:
      \[ v'(i) = \frac{v(i) - \min(v(i))}{\max(v(i)) - \min(v(i))} \]
      for normalized interval \([0,1]\),
   c) Standard deviation normalization:
      \[ v'(i) = \frac{v(i) - \text{mean}(v)}{\text{sd}(v)} \]

2. Data smoothing
   \[ F = \{0.93, 1.01, 1.001, 3.02, 2.99, 5.03, 5.01, 4.98\}, \]
   \[ F_{\text{smoothed}} = \{1.0, 1.0, 1.0, 3.0, 3.0, 5.0, 5.0, 5.0\}. \]

3. Differences and ratios
   \[ s(t+1)/s(t) \]
   \[ s(t+1) - s(t) \]

4. Composing new features (For example: BMI=k f(W,H))

MISSING DATA

- Manually examine samples with missing data values

- Automatic replacement:
  - Replace all missing values with a single global constant (a selection of a global constant is highly application dependent).
  - Replace a missing value with its feature mean.
  - Replace a missing value with its feature mean for the given class (only for classification problems).
  - One possible interpretation of missing values is that they are "don’t care" values.

\[ X = \{1, ?, 3\} \rightarrow \text{for the feature domain } [0, 1, 2, 3, 4]: \]
\[ X_1 = \{1, 0, 3\}, X_2 = \{1, 1, 3\}, X_3 = \{1, 2, 3\}, X_4 = \{1, 3, 3\}, X_5 = \{1, 4, 3\} \]

- Data miner can generate model of correlation between features. Different techniques may be used such as regression, Bayesian formalism, clustering, or decision tree induction.

In general, it is speculative and often misleading to replace missing values using a simple, artificial schema of data preparation.

It is best to generate multiple solutions of data mining with and without features that have missing values, and then make analysis and interpretation.
TIME-DEPENDENT DATA

- The **time series** of values can be expressed as:

\[ X = \{t(1), t(2), t(3), \ldots, t(n)\} \]

where \( t(n) \) is the most recent value.

- For many problems based on time series the goal is to forecast \( t(n+1) \) from previous values of the feature, where these values are directly related to the predicted value.

- The most important step in preprocessing of row time-dependent data is specification of a **window** or a time lag.

- For example, if the time series consists of eleven measurements:

\[ X = \{t(0), t(1), t(2), t(3), t(4), t(5), t(6), t(7), t(8), t(9), t(10)\} \]

1. Transform data if the window for analysis of time-series is five.

2. More formally, given the time-dependent values: \( t(n-i), \ldots, t(n) \), it is necessary to predict the value \( t(n+j) \). Transform data for the previous example taking \( j=3 \).

<table>
<thead>
<tr>
<th>Sample</th>
<th>W</th>
<th>I</th>
<th>N</th>
<th>D</th>
<th>O</th>
<th>W</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>M1</td>
<td>M2</td>
<td>M3</td>
<td>M4</td>
<td>M5</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>t(0)</td>
<td>t(1)</td>
<td>t(2)</td>
<td>t(3)</td>
<td>t(4)</td>
<td>t(5)</td>
</tr>
<tr>
<td>2</td>
<td>t(1)</td>
<td>t(2)</td>
<td>t(3)</td>
<td>t(4)</td>
<td>t(5)</td>
<td>t(6)</td>
</tr>
<tr>
<td>3</td>
<td>t(2)</td>
<td>t(3)</td>
<td>t(4)</td>
<td>t(5)</td>
<td>t(6)</td>
<td>t(7)</td>
</tr>
<tr>
<td>4</td>
<td>t(3)</td>
<td>t(4)</td>
<td>t(5)</td>
<td>t(6)</td>
<td>t(7)</td>
<td>t(8)</td>
</tr>
<tr>
<td>5</td>
<td>t(4)</td>
<td>t(5)</td>
<td>t(6)</td>
<td>t(7)</td>
<td>t(8)</td>
<td>t(9)</td>
</tr>
<tr>
<td>6</td>
<td>t(5)</td>
<td>t(6)</td>
<td>t(7)</td>
<td>t(8)</td>
<td>t(9)</td>
<td>t(10)</td>
</tr>
</tbody>
</table>

1.) \( t=5, j=1 \)

2.) \( t=5, j=3 \)
TIME-DEPENDENT DATA

- One way of summarizing features in the data set is to average them producing so called “moving averages” (MA):

\[ MA(i, m) = \frac{1}{m} \sum_{j=i-m+1}^{i} t(j) \]

The objective is to smooth neighboring time points by a moving average to reduce the random variation and noise components:

\[ MA(i, m) = t(i) = \text{mean}(i) + \text{error} \]

- Another type of average is an exponential moving average (EMA) that gives more weight to the most recent time periods. It is described recursively:

\[ EMA(i, m) = p \cdot t(i) + (1-p) \cdot EMA(i-1, m-1) \]
\[ EMA(i, 1) = t(i) \]

where \( p \) is a value between 0 and 1.

For example if \( p=0.5 \):

\[ EMA(i, 2) = 0.5 \cdot t(i) + 0.5 \cdot t(i-1) \]
\[ EMA(i, 3) = 0.5 \cdot t(i) + 0.5 \left[ 0.5 \cdot t(i-1) + 0.5 \cdot t(i-2) \right] \]

OUTLIER ANALYSIS

- Samples, which are significantly different or inconsistent with the remaining set of data, are called outliers.

1. Outlier detection for one-dimensional samples:

**STATISTICAL APPROACH**

Age = \{3, 56, 23, 39, 156, 52, 41, 22, 9, 28, 139, 31, 55, 20, -67, 37, 11, 55, 45, 37\}

Statistical parameters are:

- Mean = 39.9
- Standard deviation = 45.65

If we select that the threshold value for normal distribution of data is

\[ \text{Threshold} = \text{Mean} \pm 2 \times \text{Standard deviation} \]

then all data out of range [-54.1, 131.2] will be potential outliers.

2. Outlier detection for n-dimensional samples:

**DISTANCE-BASED OUTLIER DETECTION:**

- Evaluate the distance measures between all samples in n-dimensional data set.
- A sample \( s_i \) in a data set \( S \) is an outlier if at least a fraction \( p \) of the samples in \( S \) lies at a distance greater than \( d \).

- Deviation-based techniques
DIMENSIONS REDUCTION OF LARGE DATA SETS

Main dimensions:

- columns (features),
- rows (cases or samples), and
- values of the features.

Analysis of data reduction operations:

1. **Computing time** – Simpler data due to data reduction process can hopefully lead to the time reduction for data mining.

2. **Predictive/descriptive accuracy** – We expect that by only using relevant features, a data-mining algorithm cannot only learn faster, but also with higher accuracy:
   - irrelevant data may mislead a learning process;
   - redundant data may complicate the task and cause unexpected data mining results.

3. **Representation of the data-mining model** – The simplicity of representation, often implies model’s better comprehensibility (small decrease of accuracy may be tolerable!).

DATA REDUCTION ALGORITHMS

Recommended characteristics:

1. **Measurable quality**
2. **Recognizable quality**
3. **Monotonicity**
4. **Consistency**
5. **Diminishing returns**
6. **Interuptibility**
7. **Preemptability**
FEATURES REDUCTION

Two standard tasks:

1. **Feature selection**
   - feature ranking algorithms, and
   - minimum subset algorithms.

2. **Feature composition**
   - Feature selection in general can be viewed as a search problem ($2^N$)
   - For practical methods, an optimal search is not feasible, and simplifications are made to produce acceptable and timely reasonable results:
     - heuristic criteria
     - bottom-up approach
     - top-down approach

1. **Comparison of means and variances:**

   - If one feature describes different classes of entities, samples of two (or more) different classes can be examined:

     \[
     SE(A-B) = \sqrt{\frac{\text{var}(A)}{n_1} + \frac{\text{var}(B)}{n_2}}
     \]

   TEST:

     \[
     \frac{|\text{mean}(A) - \text{mean}(B)|}{SE(A-B)} > \text{threshold-value}
     \]

     where A and B are sets of feature values measured for two different classes, and $n_1$ and $n_2$ are the corresponding number of samples

   - For k classes, k pairwise comparisons can be made, comparing each class to its complement.

   - We a priory assumed that the given feature is independent to the others.
FEATURES REDUCTION

Comparison of means and variances (EXAMPLE):

<table>
<thead>
<tr>
<th>X</th>
<th>Y</th>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.3</td>
<td>0.7</td>
<td>A</td>
</tr>
<tr>
<td>0.2</td>
<td>0.9</td>
<td>B</td>
</tr>
<tr>
<td>0.6</td>
<td>0.6</td>
<td>A</td>
</tr>
<tr>
<td>0.5</td>
<td>0.5</td>
<td>A</td>
</tr>
<tr>
<td>0.7</td>
<td>0.7</td>
<td>B</td>
</tr>
<tr>
<td>0.4</td>
<td>0.9</td>
<td>B</td>
</tr>
</tbody>
</table>

Threshold value is 0.5

$X_A = \{0.3, 0.6, 0.5\}$, $X_B = \{0.2, 0.7, 0.4\}$,

$Y_A = \{0.7, 0.6, 0.5\}$, and $Y_B = \{0.9, 0.7, 0.9\}$

and the results of applied tests are:

$SE(X_A - X_B) = \sqrt{\frac{0.0233}{3} + \frac{0.6333}{3}} = 0.4678$

$SE(Y_A - Y_B) = \sqrt{\frac{0.01}{3} + \frac{0.0133}{3}} = 0.0875$

$|\text{mean}(X_A) - \text{mean}(X_B)| / SE(X_A - X_B) = 0.4667 - 0.4333 / 0.4678 = 0.0735 < 0.5$

$|\text{mean}(Y_A) - \text{mean}(Y_B)| / SE(Y_A - Y_B) = 0.6 - 0.8333 / 0.0875 = 2.6667 > 0.5$

- $X$ is a candidate feature for reduction because its mean values are close, and therefore the final test is below threshold value.

EXAMPLE:

$\text{SE}(X_A - X_B) = (\text{var}(X_A)/n_1 + \text{var}(X_B)/n_2)^{1/2} = (0.0233/3 + 0.6333/3)^{1/2} = 0.4678$

$\text{SE}(Y_A - Y_B) = (\text{var}(Y_A)/n_1 + \text{var}(Y_B)/n_2)^{1/2} = (0.01/3 + 0.0133/3)^{1/2} = 0.0875$

$|\text{mean}(X_A) - \text{mean}(X_B)| / \text{SE}(X_A - X_B) = 0.4667 - 0.4333 / 0.4678 = 0.0735 < 0.5$

$|\text{mean}(Y_A) - \text{mean}(Y_B)| / \text{SE}(Y_A - Y_B) = 0.6 - 0.8333 / 0.0875 = 2.6667 > 0.5$

- $X$ is a candidate feature for reduction because its mean values are close, and therefore the final test is below threshold value.

FEATURES REDUCTION

2. Covariance matrix of the means:

Under assumption of normal distributions for features values:

$M$ – a vector of the $m$ feature means, and

$C$ – an $m \times m$ covariance matrix of the means, where $C_{i,i}$ are simply the variance of feature $i$, and $C_{i,j}$ terms are correlations between each pair of features:

$C_{i,j} = 1/n \sum_{k=1}^{n} ((v(k,i) - m(i)) * (v(k,j) - m(j)))$

where: $v(k,i)$ and $v(k,j)$ are the values of features indexed with $i$ and $j$, $m(i)$ and $m(j)$ are feature means, and $n$ is the number of dimensions.

3. Heuristic measure $DM$ for filtering features that separate two classes is defined as:

$DM = (M_1 - M_2) (C_1 + C_2)^{-1} (M_1 - M_2)^T$

where $M_1$ and $C_1$ are descriptors of samples for the first class, and $M_2$ and $C_2$ for the second class.

- Given the target of $k$ best features, all subsets of $k$ from $m$ features must be evaluated to find the subset with the largest $DM$. 
FEATURES REDUCTION

3. Principal Components Analysis:

- The features are examined collectively, merged and transformed into a new set of features that hopefully retain the original information content in a reduced form.

- Given m features, they can be transformed into a single new feature F’, by the simple application of weights:

\[
F' = \sum_{j=1}^{m} w(j) \cdot f(j)
\]

The first principal component is an axis in the direction of maximum variance.

- Most likely a single set of weights w(j) will not be adequate transformation, and up to m transformations are generated, where each vector of m weights is called a principal component and it generate a new feature. Eliminating the bottom ranked transformation will cause dimensions reduction.

1. We use covariance matrix S computation, as a first step in features transformation.

\[
S_{n \times n} = \frac{1}{(n-1)} \sum_{j=1}^{n} (x_j - x')^T (x_j - x')
\]

where \( x' = \frac{1}{n} \sum_{j=1}^{n} x_j \).

2. The eigenvalues of the covariance matrix S for the given data should be calculated in the next step:

- The eigenvalues of \( S_{n \times n} \) are \( \{\lambda_1, \lambda_2, \ldots, \lambda_n\} \) where \( \lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_n \geq 0 \).

3. The eigenvectors \( e_1, e_2, \ldots, e_n \) correspond to eigenvalues \( \lambda_1, \lambda_2, \ldots, \lambda_n \), and they are called the principal axes.

4. The criterion for features selection is based on the ratio of the sum of the m largest eigenvalues of S to the trace of S (for example R>90%):

\[
R = \frac{\sum_{i=1}^{m} \lambda_i}{\sum_{i=1}^{n} \lambda_i}
\]
FEATURES REDUCTION

4. Entropy measure for features ranking:

1.1 The similarity measure $S$ of two numerical samples is:

$$S_{ij} = e^{-\alpha D_{ij}}$$

where $D_{ij}$ is the distance between samples $x_i$ and $x_j$,

$$D_{ij} = \left[ \sum_{k=1}^{n} \left( \frac{(x_{ik} - x_{jk})}{(\max_k - \min_k)} \right)^2 \right]^{1/2}$$

and $\alpha$ is a parameter mathematically expressed as:

$$\alpha = -\frac{\ln 0.5}{D}$$

$D$ is the average distance among samples in the data set, but in practical applications $\alpha = 0.5$.

1.2 Similarity for nominal variables is measured directly using Hamming distance:

$$S_{ij} = \frac{\sum_{k=1}^{n} \left( x_{ik} = x_{jk} \right)}{n}$$

where $\left( x_{ik} = x_{jk} \right)$ is 1 if $x_{ik} = x_{jk}$, and 0 otherwise.

2. The entropy measure is given by:

$$E = -\sum_{i=1}^{N-1} \sum_{j=i+1}^{N} (S_{ij} \times \log S_{ij} + (1 - S_{ij}) \times \log (1 - S_{ij}))$$

Sample   F1    F2    F3   R1   R2   R3   R4   R5
R1 A X 1 R1 0/3 0/3 2/3 0/3
R2 B Y 2 R2 2/3 1/3 0/3
R3 C Y 2 R3 0/3 1/3
R4 B X 1 R4 0/3
R5 C Z 3

Data set Table of similarity measures

4. Entropy measure for features ranking - Algorithm

1. Start with the initial full set of features $F$.

2. For each feature $f \in F$: Remove one feature $f$ from $F$ and obtain a subset $F_f$. Find the difference between entropy for $F$ and entropy for all $F_f$. For our example in Figure we have to compare the differences: $(E_F - E_{F-F1})$, $(E_F - E_{F-F2})$, and $(E_F - E_{F-F3})$.

3. Let $f_k$ be the feature such that the difference between entropy for $F$ and entropy for $F_{f_k}$ is minimum.

4. Update the set of features: $F = F - \{f_k\}$, where $-$ is a difference operation on sets. For our example, if the difference $(E_F - E_{F-F1})$ is minimum, then the reduced set of features is $\{F_2, F_3\}$. $F_1$ becomes bottom of the ranked list.

5. Repeat steps (2) – (4) until there is only one feature in $F$. 

56:235 Computational Intelligence http://css.engineering.uiowa.edu/~comp/
VALUES REDUCTION

Feature discretization techniques:

- What are the cut points?
- How to select representatives of intervals?

1. BINS METHOD (Example)

Given the set of values for the feature f: \( \{5, 1, 8, 2, 9, 2, 1, 8, 6\} \). Split them into three bins \((k = 3)\), where bins will be represented by its modes.

**a)** Sorted set of values for feature f is:

\[ \{1, 1, 2, 2, 2, 5, 6, 8, 8, 9\} \]

**b)** Initial bins \((k=3)\) are:

\[
\begin{align*}
\text{BIN}_1 & : \{1, 2, 5, 6, 8, 9\} \\
\text{BIN}_2 & : \{1, 2, 2, 8\} \\
\text{BIN}_3 & : \{1\}
\end{align*}
\]

**c1)** Modes for three selected bins are: \(\{1, 2, 8\}\). After initial distribution the total error, using absolute distance for modes, is:

\[ ERR = 0 + 0 + 1 + 0 + 0 + 3 + 2 + 0 + 0 + 1 = 7. \]

**c4)** After moving two elements from \(\text{BIN}_2\) into \(\text{BIN}_1\), and one element from \(\text{BIN}_3\) to \(\text{BIN}_2\) in the next three iterations, the new and final distribution of elements will be:

\[ f = \{1, 1, 2, 2, 5, 6, 8, 8, 9\} \]

Final bins ⇒ \(\text{BIN}_1: \{1, 2, 2, 5, 6, 8, 9\} \)
\(\text{BIN}_2: \{1, 2\} \)
\(\text{BIN}_3: \{1\}\)

Modes are: \(\{2, 5, 8\}\), and the total minimized error is: \(ER = 4\).
VALUES REDUCTION

2. **Number Approximation by Rounding**

1. **Integer division:** \( Y = \text{int} \left( \frac{X}{10^k} \right) \)

2. **Rounding:** If \( \text{mod} \left( X, 10^k \right) \geq \left( \frac{10^k}{2} \right) \) then \( Y = Y + 1 \)

3. **Integer multiplication:** \( X = Y \times 10^k \)

where \( k \) is the number of rightmost decimal places to round.

- For example, 1453 is rounded to 1450 with \( k = 1 \), rounded to 1500 with \( k = 2 \), and rounded to 1000 with \( k = 3 \).

- This simple rounding algorithm can be applied in iterations to reduce the number of values for selected features in large data sets.

- A maximum number of values in real-world applications is given between 50 and 100.

VALUES REDUCTION

3. **ChiMerge Technique**

The algorithm determines similarities between distributions of data in two adjacent intervals based on output classification of samples.

1. Sort the data for the given feature in ascending order,

2. Define initial intervals so that every value of the feature is in a separate interval.

3. Repeat until no \( \chi^2 \) of any two adjacent intervals is less than threshold value:

   - After each merger, \( \chi^2 \) tests for the remaining intervals are calculated, and two adjacent with smallest \( \chi^2 \) value are found. Merge these intervals if calculated \( \chi^2 \) is less than threshold.

   - If no merge is possible, and the number of intervals is greater then user-defined maximum, increase the threshold value.
VALUES REDUCTION

3. ChiMerge Technique

- A contingency table for $2 \times 2$ categorical data:

<table>
<thead>
<tr>
<th></th>
<th>Class1</th>
<th>Class2</th>
<th>∑</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interval-1</td>
<td>$A_{11}$</td>
<td>$A_{12}$</td>
<td>$R_1$</td>
</tr>
<tr>
<td></td>
<td>$A_{21}$</td>
<td>$A_{22}$</td>
<td>$R_2$</td>
</tr>
<tr>
<td>∑</td>
<td>$C_1$</td>
<td>$C_2$</td>
<td>$N$</td>
</tr>
</tbody>
</table>

- $\chi^2$ test is:

$$\chi^2 = \sum_{i=1}^{2} \sum_{j=1}^{k} \frac{(A_{ij} - E_{ij})^2}{E_{ij}}$$

where:

- $k$ = number of classes,
- $A_{ij}$ = number of instances in the $i$-th interval, $j$-th class,
- $E_{ij}$ = expected frequency of $A_{ij}$, which is computed as $(R_i \cdot C_j) / N$,
- $R_i$ = number of instances in the $i$-th interval = $\sum A_{ij}$, $j = 1, \ldots, k$,
- $C_j$ = number of instances in the $j$-th class = $\sum A_{ij}$, $i = 1, 2$,
- $N$ = total number of instances = $\sum R_i$, $i = 1, 2$.

Values Reduction (Example)

3. ChiMerge Technique

- Data Set

<table>
<thead>
<tr>
<th></th>
<th>Sample: F</th>
<th>K</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>7</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>8</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>9</td>
<td>1</td>
</tr>
<tr>
<td>6</td>
<td>11</td>
<td>2</td>
</tr>
<tr>
<td>7</td>
<td>23</td>
<td>2</td>
</tr>
<tr>
<td>8</td>
<td>37</td>
<td>1</td>
</tr>
<tr>
<td>9</td>
<td>39</td>
<td>2</td>
</tr>
<tr>
<td>10</td>
<td>45</td>
<td>1</td>
</tr>
<tr>
<td>11</td>
<td>46</td>
<td>1</td>
</tr>
<tr>
<td>12</td>
<td>59</td>
<td>1</td>
</tr>
</tbody>
</table>

- Interval points for feature F are: 0, 2, 5, 7.5, 8.5, 10, etc.
- $\chi^2$ was minimum for intervals: [7.5, 8.5] and [8.5, 10]

<table>
<thead>
<tr>
<th></th>
<th>K=1</th>
<th>K=2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interval [7.5, 8.5]</td>
<td>$A_{11}=1$</td>
<td>$A_{21}=0$</td>
</tr>
<tr>
<td></td>
<td>$A_{12}=0$</td>
<td>$A_{22}=0$</td>
</tr>
<tr>
<td>Interval [8.5, 9.5]</td>
<td>$C_1=2$</td>
<td>$C_2=0$</td>
</tr>
</tbody>
</table>

Based on the table’s values, we can calculate expected values:

$$E_{11} = \frac{2}{2} = 1,$$
$$E_{12} = \frac{0}{2} = 0.1,$$
$$E_{21} = \frac{2}{2} = 1,$$  and $$E_{22} = \frac{0}{2} = 0.1$$

and corresponding $\chi^2$ test:

$$\chi^2 = \frac{(1 - 1)^2}{1} + \frac{(0 - 0.1)^2}{0.1} + \frac{(1 - 1)^2}{1} + \frac{(0 - 0.1)^2}{0.1} = 0.2$$

- For the degree of freedom $d=1$, and $\chi^2 = 0.2 < 2.706$ (MERGE !)
VALUES REDUCTION

3. ChiMerge Technique (Example)

Additional Iterations:

<table>
<thead>
<tr>
<th>Interval</th>
<th>K=1</th>
<th>K=2</th>
<th>Σ</th>
</tr>
</thead>
<tbody>
<tr>
<td>[0, 7.5]</td>
<td>A11=2</td>
<td>A12=1</td>
<td>R1=3</td>
</tr>
<tr>
<td>Interval [7.5, 10]</td>
<td>A21=2</td>
<td>A22=0</td>
<td>R2=2</td>
</tr>
<tr>
<td>Σ</td>
<td>C1=4</td>
<td>C2=1</td>
<td>N=5</td>
</tr>
</tbody>
</table>

\[ E_{11} = \frac{12}{5} = 2.4, \]
\[ E_{12} = \frac{3}{5} = 0.6, \]
\[ E_{21} = \frac{8}{5} = 1.6, \text{ and } \]
\[ E_{22} = \frac{2}{5} = 0.4 \]

\[ \chi^2 = (2 - 2.4)^2 / 2.4 + (1 - 0.6)^2 / 0.6 + (2 - 1.6)^2 / 1.6 + (0 - 0.4)^2 / 0.4 \]

\[ \chi^2 = 0.834 \]

For the degree of freedom \( d=1 \), \( \chi^2 = 0.834 < 2.706 \) (MERGE!)

\[ : \]

<table>
<thead>
<tr>
<th>Interval</th>
<th>K=1</th>
<th>K=2</th>
<th>Σ</th>
</tr>
</thead>
<tbody>
<tr>
<td>[0, 10.0]</td>
<td>A11=4</td>
<td>A12=1</td>
<td>R1=5</td>
</tr>
<tr>
<td>Interval [10.0, 42.0]</td>
<td>A21=1</td>
<td>A22=3</td>
<td>R2=4</td>
</tr>
<tr>
<td>Σ</td>
<td>C1=5</td>
<td>C2=4</td>
<td>N=9</td>
</tr>
</tbody>
</table>

\[ E_{11} = 2.78, E_{12} = 2.22, E_{21} = 2.22, E_{22} = 1.78, \text{ and } \chi^2 = 2.72 > 2.706 \] (NO MERGE!)

Final discretization: [0, 10], [10, 42], and [42, 60]

CASES REDUCTION

- The largest and the most critical dimension.
- The most complex task in data reduction.
- **Sampling**: After the subset of data is obtained, it is used to estimate information about entire data set.
- Sampling process causes always a sampling error. Sampling error is inherent and avoidable for every approach and every strategy.
- Advantages: reduce cost, greater speed, greater scope, and sometimes even higher accuracy

Classifications for sampling methods:

1) general-purpose sampling methods, and
2) sampling methods for specific domains.
CASES REDUCTION

General-purpose sampling methods:

1. **Systematic sampling**

2. **Random sampling**
   - random sampling without replacement,
   - random sampling with replacement.
   1.1 Incremental sampling
   2.2 Average sampling

3. **Stratified sampling**

4. **Inverse sampling**

Types of inference: *induction, deduction, and transduction*

Diagram:

- A priori knowledge (Assumptions)
- Estimated model
- Induction
- Deduction
- Transduction
- Training data
- Predicted output
LEARNING MACHINE

Hypotheses for a given data set
LEARNING MACHINE

1. loss function $L(y, f(X, w))$

   where:
   - $y$ – is the output produced by the system,
   - $X$ – is a set of inputs,
   - $f(X, w)$ – is the output produced by the learning machine for a selected approximating function, and
   - $w$ – is the set of parameters in the approximating functions.

2. risk functional $R(w)$:

   \[
   R(w) = \int\int L(y, f(X, w)) p(X, y) \, dX \, dy
   \]

   1a) classification error:

   \[
   L(y, f(X, w)) = \begin{cases} 
   0 & \text{if } y = f(X, w) \\ 
   1 & \text{if } y \neq f(X, w) 
   \end{cases}
   \]

   1b) loss function for regression is the squared error measure:

   \[
   L(y, f(X, w)) = (y - f(X, w))^2
   \]

STATISTICAL LEARNING THEORY

Asymptotic consistency of the ERM:

\[
\text{Number of samples} \quad \begin{array}{c} \text{Risk Functional} \\ \text{True (expected) risk} \\ \text{Empirical risk} \end{array} \quad \begin{array}{c} \min R(w) \end{array}
\]
**GROWTH FUNCTION**  \( G(N) \)

![Graph showing growth function](image)

\[
G(n) = h \left( \ln(n/h) + 1 \right)
\]

\[
n \ln 2
\]

---

**STRUCTURAL RISK MINIMIZATION (SRM)**

Structure on a set of approximating functions

SRM approach:

1) **calculate or estimate VC-dimension** for any element \( S_k \) of the structure,

2) **minimize empirical risk** for each element of the structure.
SRM optimization strategy

![Graph showing error vs. model complexity with underfitting and overfitting]

Statistical Learning Theory

1. A wide, flexible set of *approximating functions* \( f(X, w) \).

2. *A priori knowledge* (or assumptions) used to impose constraints on a potential solution.


4. *A learning method*, namely a constructive, computational implementation of an inductive principle for a given class of approximating functions.

Optimization:

1. Stochastic approximation (or gradient descent)
2. Iterative methods
3. Greedy optimization

Empirical and true risk as a function of \( h \) (model complexity)
MAIN TYPES OF INDUCTIVE LEARNING

Supervised learning

Unsupervised learning

MODEL ESTIMATION

a) Resubstitution Method

b) Holdout Method

c) Leave-one-out Method

d) Rotation Method
   (n-fold cross validation)

e) Bootstrap method
Confusion matrix for three classes

<table>
<thead>
<tr>
<th>Classification model</th>
<th>True class</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
<td>28</td>
<td>1</td>
<td>4</td>
<td>33</td>
</tr>
<tr>
<td>1</td>
<td></td>
<td>2</td>
<td>28</td>
<td>2</td>
<td>32</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>0</td>
<td>1</td>
<td>24</td>
<td>25</td>
</tr>
<tr>
<td>Total</td>
<td></td>
<td>30</td>
<td>30</td>
<td>30</td>
<td>90</td>
</tr>
</tbody>
</table>

\[
R = \frac{E}{S} = \frac{10}{90} = 0.11
\]

\[
A = 1 - R = 1 - 0.11 = 0.89 \ (89\%)
\]

ROC curve

- ROC curve shows the trade-off between sensitivity and 1-specificity values
STATISTICAL INFERENCE

a) Estimation - one wants to come up with a plausible value or range of plausible values for the unknown system parameter.

Data set $T$ is described by ordered $n$-tuples of values for variables $X = \{X_1, X_2, \ldots, X_n\}$ (attributes of entities in population):

\[
T = \{(x_{11}, \ldots, x_{1n}), (x_{21}, \ldots, x_{2n}), \ldots, (x_{m1}, \ldots, x_{mn})\}
\]

Output: $Y \in X$, $X^* = X - Y$  → Based on given $X^*$ estimate $Y$

| Y numeric → | regression |
| Y categorical → | classification |

b) Tests of hypotheses - one wants to decide whether a hypothesis concerning the value of the population characteristic should be accepted or rejected in the light of data set.

- null hypothesis $H_0$

Assessing Data Sets

1. $mean = \frac{1}{n} \sum_{i=1}^{n} x_i$

\[
mean = \frac{\sum_{i=1}^{n} w_i x_i}{\sum_{i=1}^{n} w_i}
\]

2. $median = \begin{cases} x_{(n+1)/2} & \text{if } n \text{ is odd} \\ (x_{n/2} + x_{(n/2)+1}) / 2 & \text{if } n \text{ is even} \end{cases}$

3. $mode$ - is the value occurs most frequently in the set of data.

4. $variance = \frac{1}{(n-1)} \sum_{i=1}^{n} (x_i - mean)^2$

53
Assessing Data Sets

Given the data set : \( T = \{3, 5, 2, 9, 0, 7, 3, 6\} \)

After sorting : \( T = \{0, 2, 3, 3, 5, 6, 7, 9\} \)

1. \( \text{mean} = \frac{1}{n} \sum_{i=1}^{n} x_i \)
   \[ \frac{1}{8} (0+2+3+3+5+6+7+9) = 4.375 \]

2. \( \text{median} = \begin{cases}  x_{\frac{n+1}{2}} \text{ if } n \text{ is odd} \\  \frac{x_{\frac{n}{2}} + x_{\frac{n}{2}+1}}{2} \text{ if } n \text{ is even} \end{cases} \)
   \[ = \frac{3+5}{2} = 4 \]

3. \( \text{mode} \) - is the value occurs most frequently in the set of data.
   \( \text{mode} = 3 \)

4. \( \text{variance} - \sigma^2 = \frac{1}{n-1} \sum_{i=1}^{n} (x_i - \text{mean})^2 \)
   \[ \sigma^2 = \frac{1}{7} \cdot \sum_{i=1}^{8} (x_i - 4.375)^2 = 8.5532 \]
   \( \sigma = 2.9246 \)
Bayesian Inference

- Let \( X \) be a data sample whose class label is unknown.

- Let \( H \) be some hypothesis, such as that the data sample \( X \) belongs to a specific class \( C \).

\[
P(H/X) = [P(X/H) \cdot P(H)] / P(X)
\]

- Naïve Bayesian Classifier:

\[
P(C_i/X) = [P(X/C_i) \cdot P(C_i)] / P(X)
\]

\[
P(X/C_i) = \prod P(x_t / C_i) \quad t=1, \ldots, n
\]

Naïve Bayesian Classifier

- Prior probabilities \( P(C_i) \):

\[
P(C=1) = \frac{4}{7} = 0.5714
\]
\[
P(C=2) = \frac{3}{7} = 0.4286
\]

- Conditional probabilities \( P(x_t / C_i) \) for every attribute value:

\[
P(A_1=1 / C=1) = \frac{2}{4} = 0.50
\]
\[
P(A_1=1 / C=2) = \frac{1}{3} = 0.33
\]
\[
P(A_2=2 / C=1) = \frac{1}{4} = 0.25
\]
\[
P(A_2=2 / C=2) = \frac{2}{3} = 0.66
\]
\[
P(A_3=2 / C=1) = \frac{1}{4} = 0.25
\]
\[
P(A_3=2 / C=2) = \frac{2}{3} = 0.66
\]

- Conditional probabilities \( P(X/C_i) \):

\[
P(X/C=1) = P(A_1=1/C=1) \cdot P(A_2=2/C=1) \cdot P(A_3=2/C=1) =
\quad = 0.50 \cdot 0.25 \cdot 0.25 = 0.03125
\]
\[
P(X/C=2) = P(A_1=1/C=2) \cdot P(A_2=2/C=2) \cdot P(A_3=2/C=2) =
\quad = 0.33 \cdot 0.66 \cdot 0.66 = 0.14375
\]

\[
P(C_1 / X) \approx P(X / C=1) \cdot P(C=1) = 0.03125 \cdot 0.5714 = 0.0179
\]
\[
P(C_2 / X) \approx P(X / C=2) \cdot P(C=2) = 0.14375 \cdot 0.4286 = 0.0616
\]

\[
P(C_2 / X) = \text{Max} \{ P(C_1 / X), P(C_2 / X) \} = \text{Max} \{0.0179, 0.0616\}
\]
REGRESSION

General linear model:

\[ Y = \alpha + \beta_1 \cdot X_1 + \beta_2 \cdot X_2 + \beta_3 \cdot X_3 + \ldots + \beta_n \cdot X_n \]

- Applying this equation to each of the given samples we obtain new set of equations:

  \[ y_j = \alpha + \beta_{1j} \cdot x_{1j} + \beta_{2j} \cdot x_{2j} + \beta_{3j} \cdot x_{3j} + \ldots + \beta_{nj} \cdot x_{nj} + \varepsilon_j \quad j=1, \ldots, m \]

Nonlinear regression (examples):

- Linearization of the regression model

Estimate the quality of linear regression model:

**Correlation coefficient** \( r \) :

\[ r = \beta \cdot \sqrt{\frac{S_{xx}}{S_{yy}}} = \frac{S_{xy}}{S_{xx} \cdot S_{yy}} \]

where:

\[ S_{xx} = \sum_{i=1}^{n} (x_i - \text{mean}_x)^2 \]

\[ S_{yy} = \sum_{i=1}^{n} (y_i - \text{mean}_y)^2 \]

\[ S_{xy} = \sum_{i=1}^{n} (x_i - \text{mean}_x)(y_i - \text{mean}_y) \]
**ANOVA**

- The *analysis of variance*, or *ANOVA*, is a primarily a method of identifying which of the $\beta$’s in a linear regression model are non-zero.

$$Y = \alpha + \beta_1 \cdot X_1 + \beta_2 \cdot X_2 + \beta_3 \cdot X_3 + \ldots + \beta_n \cdot X_n$$

- The size of the residuals, for all $m$ samples in a data set, is related to the size of variance $\sigma^2$, and it can be estimated by:

$$S^2 = \frac{1}{m} \sum_{i=1}^{m} (y_i - f(x_i))^2 / (m - (n - 1))$$

- F-ratio or F-statistic test in a form:

$$F = \frac{S_{new}^2}{S_{old}^2}$$

For example:

<table>
<thead>
<tr>
<th>Case</th>
<th>Set of inputs</th>
<th>$S_i^2$</th>
<th>$F$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$X_1, X_2, X_3$</td>
<td>3.56</td>
<td>$F_{21} = 1.12$</td>
</tr>
<tr>
<td>2</td>
<td>$X_1, X_2$</td>
<td>3.98</td>
<td>$F_{11} = 1.75$</td>
</tr>
<tr>
<td>3</td>
<td>$X_1, X_3$</td>
<td>6.22</td>
<td>$F_{11} = 2.34$</td>
</tr>
<tr>
<td>4</td>
<td>$X_2, X_3$</td>
<td>8.34</td>
<td>$F_{11} = 2.27$</td>
</tr>
<tr>
<td>5</td>
<td>$X_1$</td>
<td>9.02</td>
<td>$F_{11} = 2.48$</td>
</tr>
<tr>
<td>6</td>
<td>$X_2$</td>
<td>9.89</td>
<td>$F_{11} = 2.48$</td>
</tr>
</tbody>
</table>

- **MANOVA** (multivariate ANOVA).

**LOGISTIC REGRESSION**

- **Logistic regression** - is modeling of categorical response variables using generalized linear regression model.

- Rather than predicting the value of the dependent variable, the logistic regression method tries to *estimate the probability* $p$ that the dependent variable will have a given value.

- Suppose that output $Y$ has two possible categorical values coded with 0 and 1.

- Based on available data we can compute probabilities for both values for the given input sample:

$$P(y_j=0) = 1-p_j \quad \text{and} \quad P(y_j=1) = p_j$$

- **Linear logistic model** - is the model that will fit these probabilities for accommodated linear regression:

$$\log \left( \frac{p_j}{1-p_j} \right) = \alpha + \beta_1 \cdot X_{1j} + \beta_2 \cdot X_{2j} + \beta_3 \cdot X_{3j} + \ldots + \beta_n \cdot X_{nj}$$

$$\log \left( \frac{p_j}{1-p_j} \right) \leftrightarrow \text{logit}(p)$$
LOGISTIC REGRESSION - example

Model:

\[
\text{logit}(p) = 1.5 - 0.6 \cdot x_1 + 0.4 \cdot x_2 - 0.3 \cdot x_3
\]

New sample for classification:

\[\{x_1, x_2, x_3, Y\} = \{1, 0, 1, ?\}\]

Classification:

\[
\text{logit}(p) = 1.5 - 0.6 \cdot 1 + 0.4 \cdot 0 - 0.3 \cdot 1 = 0.6
\]

\[
\log \left( \frac{p}{1-p} \right) = 0.6
\]

\[
p = p(Y=1) = \frac{e^{-0.6}}{1 + e^{-0.6}} = 0.35
\]

\[
\text{Y}=1 \text{ (with } p=0.35) \text{ is less probable than the other categorical value } \text{Y}=0 \text{ (with } p=0.65)\]

LOG-LINEAR MODELS

- Log-linear modeling is a way of analyzing the relationship between categorical (or quantitative) variables

- Because all variables are categorical, we analyze possible associations between them using incidence matrices, also called contingency tables.

Example (2 X 2):

<table>
<thead>
<tr>
<th></th>
<th>Support</th>
<th>No</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sex</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Female</td>
<td>309</td>
<td>191</td>
<td>500</td>
</tr>
<tr>
<td>Male</td>
<td>319</td>
<td>281</td>
<td>600</td>
</tr>
<tr>
<td>Total</td>
<td>628</td>
<td>472</td>
<td>1100</td>
</tr>
</tbody>
</table>

- Algorithm for feature association:

  1. Transform given contingency table into a similar table, but instead of observed values we calculate expected values for the case when the variables are independent.

  2. Compare this two matrices using squared distance measure, and chi-square test as a criteria of association for two categorical variables.
Linear regression with one input variable

\[ Y = \alpha + \beta \cdot X \]

- Sum of squares of the errors

\[
SSE = \sum_{i=1}^{n} e_i^2 = \sum_{i=1}^{n} (y_i - y_i')^2 = \sum_{i=1}^{n} (y_i - \alpha - \beta x_i)^2
\]

- Differentiating SSE with respect to \( \alpha \) and \( \beta \), we have:

\[
\frac{\partial (SSE)}{\partial \alpha} = -2 \sum_{i=1}^{n} (y_i - \alpha - \beta x_i)
\]

\[
\frac{\partial (SSE)}{\partial \beta} = -2 \sum_{i=1}^{n} ((y_i - \alpha - \beta x_i) \cdot x_i)
\]

- Setting the partial derivatives equal to zero (minimization of the total error):

\[
\alpha \cdot n + \beta \sum_{i=1}^{n} x_i = \sum_{i=1}^{n} y_i
\]

\[
\alpha \sum_{i=1}^{n} x_i + \beta \sum_{i=1}^{n} x_i^2 = \sum_{i=1}^{n} x_i y_i
\]

- Solving for \( \alpha \) and \( \beta \):

\[
\beta = \left[ \frac{\sum_{i=1}^{n} (x_i - \text{mean}_x) \cdot (y_i - \text{mean}_y)}{\sum_{i=1}^{n} (x_i - \text{mean}_x)^2} \right]
\]

\[
\alpha = \text{mean}_y - \beta \cdot \text{mean}_x
\]

(Example)

\[
\begin{array}{c|c}
A & B \\
\hline
1 & 3 \\
7 & 9 \\
10 & 11 \\
4 & 5 \\
3 & 2 \\
\end{array}
\]

\[
B = \alpha + \beta \cdot A
\]

\[
\text{mean}_A = 5 \quad \text{mean}_B = 6
\]

\[
\alpha = 0.8 \quad \beta = 1.04
\]

\[
B = 0.8 + 1.04 \cdot A
\]
LOG-LINEAR MODELS

Algorithm for feature association steps:

1.1 Denote the contingency table as $X_{m \times n}$.

   The row totals for the table are:
   $$n \sum_{i=1}^{n} X_{ji}$$

   The column totals are:
   $$m \sum_{j=1}^{m} X_{j}i$$

   The grand total is defined as a sum of row totals:
   $$X_{++} = \sum_{j=1}^{m} X_j+$$

   or as a sum of column totals:
   $$X_{++} = \sum_{i=1}^{n} X_{+i}$$

1.2 Using these totals we can calculate the contingency table of expected values under the assumption that there is no association between the row variable and the column variable. The expected values are:

   $$E_{ji} = \frac{(X_j+ \cdot X_{+i})}{X_{++}}$$ for $j = 1, \ldots, m, i = 1, \ldots, n$

2. Calculate $\chi^2$ test.

   The bigger $\chi^2$, the greater the evidence against $H_0$ hypothesis.

   $$\chi^2 = \sum_{j=1}^{m} \sum_{i=1}^{n} \frac{(X_{ji} - E_{ji})^2}{E_{ji}}$$

   d.f.(degrees of freedom) = $(m-1)(n-1) = (2-1)(2-1) = 1$

   $T(0.05) = \chi^2 (1-\alpha, d.f.) = \chi^2 (0.95, 1) = 3.84.$

   $\chi^2 = 8.2816 \geq T(0.05) = 3.84$

   Hypothesis $H_0$ is rejected; the attributes, analyzed in the survey, are with high level of dependency.

---

Feature’s association - example

2×2 contingency table for the 1100 samples survey

<table>
<thead>
<tr>
<th></th>
<th>Support</th>
<th>No</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sex</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Female</td>
<td>309</td>
<td>191</td>
<td>500</td>
</tr>
<tr>
<td>Male</td>
<td>319</td>
<td>281</td>
<td>600</td>
</tr>
<tr>
<td>Total</td>
<td>628</td>
<td>472</td>
<td>1100</td>
</tr>
</tbody>
</table>

2×2 contingency table of expected values

<table>
<thead>
<tr>
<th></th>
<th>Support</th>
<th>No</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sex</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Female</td>
<td>285.5</td>
<td>214.5</td>
<td>500</td>
</tr>
<tr>
<td>Male</td>
<td>342.5</td>
<td>257.5</td>
<td>600</td>
</tr>
<tr>
<td>Total</td>
<td>628</td>
<td>472</td>
<td>1100</td>
</tr>
</tbody>
</table>

$$\chi^2 = \sum_{j=1}^{m} \sum_{i=1}^{n} \frac{(X_{ji} - E_{ji})^2}{E_{ji}} = 8.2816$$
LINEAR DISCRIMINANT ANALYSIS (LDA)

- The dependent variable (output) is categorical (nominal or ordinal), and the independent variables (inputs) are metric.

- LDA - construct a discriminant function that yields different scores \((z)\) when computed with data from different output classes. A linear discriminant function has the following form:

\[
z = w_1 x_1 + w_2 x_2 + \ldots + w_k x_k
\]

\[
z_{\text{cut-ab}} = \frac{(z_a + z_b)}{2}
\]

Geometric interpretation of the discriminant score

Multiple discriminant analysis

![Diagram of discriminant analysis](image.png)
### SIMILARITY MEASURES

- The word “similarity” in clustering means that the value of \( s(x, x') \) is large when \( x \) and \( x' \) are two similar samples; the value of \( s(x, x') \) is small when \( x \) and \( x' \) are not similar.

- Similarity measure \( s \) is symmetric:
  \[
  s(x, x') = s(x', x), \quad \forall x, x' \in X
  \]

- Similarity measure is normalized:
  \[
  0 \leq s(x, x') \leq 1, \quad \forall x, x' \in X
  \]

### SIMILARITY/DISTANCE MEASURES

1. **Euclidean distance** in \( m \)-dimensional feature space:
   \[
   d_2(x_i, x_j) = \left( \sum_{k=1}^{m} (x_{ik} - x_{jk})^2 \right)^{1/2}
   \]

2. **L1 metric or city block distance**:
   \[
   d_1(x_i, x_j) = \sum_{k=1}^{m} |x_{ik} - x_{jk}|
   \]

3. **Minkowski metric** (includes the Euclidean distance and city block distance as special cases):
   \[
   d_p(x_i, x_j) = \left( \sum_{k=1}^{m} (x_{ik} - x_{jk})^p \right)^{1/p}
   \]

4. **Cosine-correlation**:
   \[
   s_{\text{cos}}(x_i, x_j) = \frac{\sum_{k=1}^{m} (x_{ik} \cdot x_{jk})}{\sqrt{\sum_{k=1}^{m} x_{ik}^2 \cdot \sum_{k=1}^{m} x_{jk}^2}}
   \]
   \[
   s_{\text{cos}}(x_i, x_j) = 1 \iff \forall i, j \text{ and } \lambda > 0 \text{ where } x_i = \lambda \cdot x_j
   \]
   \[
   s_{\text{cos}}(x_i, x_j) = -1 \iff \forall i, j \text{ and } \lambda < 0 \text{ where } x_i = \lambda \cdot x_j
   \]
SIMILARITY/DISTANCE MEASURES (example)

For 4-dimensional vectors

\[ x_1 = \{1, 0, 1, 0\} \] and \[ x_2 = \{2, 1, -3, -1\} \]

these distance measures are:

1. **Euclidean distance** in m-dimensional feature space:
   \[
d_2(x_i, x_j) = (1 + 1 + 16 + 1)^{1/2} = 4.36
   \]

2. **L_1 metric or city block distance**:
   \[
d_1(x_i, x_j) = 1 + 1 + 4 + 1 = 7
   \]

3. **Minkowski metric** (for p=3):
   \[
d_3(x_i, x_j) = (1 + 1 + 64 + 1)^{1/3} = 4.06
   \]

4. **Cosine-correlation**:
   \[
s_{\cos}(x_i, x_j) = (2 + 0 - 3 + 0) / (2^{1/2} \cdot 15^{1/2}) = -0.18
   \]

5. **Distance measure for n-dimensional vectors with binary data**:
   - Use the 2 x 2 contingency table for samples \( x_i \) and \( x_j \):
     \[
     \begin{array}{cc}
     & 1 & 0 \\
     1 & a & b \\
     0 & c & d \\
     \end{array}
     \]
   - a) **Simple Matching Coefficient (SMC)**:
     \[
s_{\text{smc}}(x_i, x_j) = (a + d) / (a + b + c + d)
     \]
   - b) **Jaccard Coefficient**:
     \[
s_{\text{jacc}}(x_i, x_j) = a / (a + b + c)
     \]
   - c) **Rao’s Coefficient**:
     \[
s_{\text{rc}}(x_i, x_j) = a / (a + b + c + d)
     \]

Example: For \( x_i = \{0,0,1,0,1,0,1,0\} \) and \( x_j = \{0,1,1,0,0,1,0,0\} \) parameters are
\[ a = 2, \ b = 2, \ c = 1, \text{and} \ d = 3 \]
and similarity measures:
\[ s_{\text{smc}}(x_i, x_j) = 5/8, \ s_{\text{jacc}}(x_i, x_j) = 2/5, \text{and} \ s_{\text{rc}}(x_i, x_j) = 2/8 \]
SIMILARITY/DISTANCE MEASURES

6. MUTUAL NEIGHBOR DISTANCE (MND):

The similarity between two points $x_i$ and $x_j$ is:

$$MND(x_i, x_j) = NN(x_i, x_j) + NN(x_j, x_i)$$

where:

$NN(x_i, x_j)$ is the neighbor number of $x_j$ with respect to $x_i$.

- If $x_i$ is the closest point to $x_j$, then $NN(x_i, x_j)$ is equal to 1,
- if it is the second closest point to $x_j$, then $NN(x_i, x_j)$ is equal to 2,
- etc.

$NN=3$ $NN=1$ $NN=2$

$\Rightarrow$ $MND(A, B) = 2$

$NN(A, B) = NN(B, A) = 1$

$NN(B, C) = 1, NN(C, B) = 2$ $\Rightarrow$ $MND(B, C) = 3$

A and B are more similar than B and C using MND measure.

$NN(A, B) = 1, \quad NN(B, A) = 4 \quad \Rightarrow \quad MND(A, B) = 5$

$NN(B, C) = 1, \quad NN(C, B) = 2 \quad \Rightarrow \quad MND(B, C) = 3$

After changes in the context, B and C are more similar than A and B using MND measure.
DISTANCE MEASURE BETWEEN CLUSTERS (set of samples)

1) \( D_{\text{min}} (C_i, C_j) = \min \left| p_i - p_j \right| \)
where \( p_i \in C_i \) and \( p_j \in C_j \).

2) \( D_{\text{mean}} (C_i, C_j) = \left| m_i - m_j \right| \)
where \( m_i \) and \( m_j \) are centroids of \( C_i \) and \( C_j \).

3) \( D_{\text{avg}} (C_i, C_j) = \frac{1}{(n_i n_j)} \sum \sum \left| p_i - p_j \right| \)
where \( p_i \in C_i \) and \( p_j \in C_j \), and \( n_i \) and \( n_j \) are the numbers of samples in clusters \( C_i \) and \( C_j \).

4) \( D_{\text{max}} (C_i, C_j) = \max \left| p_i - p_j \right| \)
where \( p_i \in C_i \) and \( p_j \in C_j \).

CLUSTERING ALGORITHMS

1. HIERARCHICAL CLUSTERING
   1.1 Divisible algorithms
   1.2 Agglomerative algorithms
      - single-link or
      - complete-link

![Single-link distance](Cluster 1) ![Complete-link distance](Cluster 2)

2. PARTITIONAL CLUSTERING
   (ITERATIVE SQUARE-ERROR BASED)
   2.1 K-means partitional clustering algorithm
      2.1.1 K-nearest neighbor
   2.2 Incremental clustering
AGGLOMERATIVE CLUSTERING ALGORITHM

1. Place each sample in its own cluster. Construct the list of inter-cluster distances for all distinct unordered pairs of samples, and sort this list in ascending order.

2. Step through the sorted list of distances, forming for each distinct threshold value $d_k$ a graph of the samples where pairs of samples closer than $d_k$ are connected into a new cluster by a graph edge. If all the samples are members of a connected graph, stop. Otherwise, repeat this step.

3. The output of the algorithm is a nested hierarchy of graphs, which can be cut at a desired dissimilarity level forming a partition (clusters) identified by simple connected components in the corresponding sub-graph.

Example:

$x_1 = (0, 2), \ x_2 = (0, 0), \ x_3 = (1.5, 0), \ x_4 = (5, 0), \ x_5 = (5, 2)$.

K-means partitional clustering algorithm

Suppose that the given set of $N$ samples in $n$-dimensional space has to be partitioned into $K$ clusters {$C_1, C_2, ..., C_k$}.

1. Select an initial partition with $K$ clusters containing randomly chosen samples, and compute centroids of the clusters:

   
   
   $M_k = (1/n_k) \sum_{i=1}^{n_k} x_{ik}$  \quad (centroid of each cluster)

   
   
   $e_k^2 = \sum_{i=1}^{n_k} (x_{ik} - M_k)^2$  \quad (within-cluster variation)

   
   
   $E_k^2 = \sum_{k=1}^{K} e_k^2$  \quad (the total square-error)

2. Generate a new partition by assigning each sample to the closest cluster center.

3. Compute new cluster centers as the centroids of the clusters.

4. Repeat steps 2 and 3 until an optimum value of the criterion function is found (or until the cluster membership stabilizes).
K-means partitional clustering algorithm
(Example)

Data: \( x_1 = (0, 2), \ x_2 = (0, 0), \ x_3 = (1.5, 0), \ x_4 = (5, 0), \ x_5 = (5, 2). \)

2. Random distribution of samples:

\[ C_1 = \{x_1, x_2, x_4\} \text{ and } C_2 = \{x_3, x_5\} \]

3. Centroids:

\[ M_1 = \{(0+0+5)/3, (2+0+0)/3\} = \{1.66, 0.66\} \]
\[ M_2 = \{(1.5+5)/2, (0+2)/2\} = \{3.25, 1.00\} \]

Within-cluster variations:

\[ e_1^2 = [(0-1.66)^2+(2-0.66)^2] + [(0-1.66)^2+(0-0.66)^2] + [(5-1.66)^2+(0-0.66)^2] \]
\[ = 19.36 \]
\[ e_2^2 = [(1.5-3.25)^2+(0-1)^2] + [(5-3.25)^2+(2-1)^2] \]
\[ = 8.12 \]

Total square error:

\[ E^2 = e_1^2 + e_2^2 = 19.36 + 8.12 = 27.48 \]

K-means partitional clustering algorithm
(Example)

Reassign all samples:

\[ d(M_1, x_1) = 2.14 \text{ and } d(M_2, x_1) = 3.40 \Rightarrow x_1 \in C_1 \]
\[ d(M_1, x_2) = 1.79 \text{ and } d(M_2, x_2) = 3.40 \Rightarrow x_2 \in C_1 \]
\[ d(M_1, x_3) = 0.83 \text{ and } d(M_2, x_3) = 2.01 \Rightarrow x_3 \in C_1 \]
\[ d(M_1, x_4) = 3.41 \text{ and } d(M_2, x_4) = 2.01 \Rightarrow x_4 \in C_2 \]
\[ d(M_1, x_5) = 3.60 \text{ and } d(M_2, x_5) = 2.01 \Rightarrow x_5 \in C_2 \]

New clusters:

\[ C_1 = \{x_1, x_2, x_3\} \text{ and } C_2 = \{x_4, x_5\} \]

New centroids:

\[ M_1 = \{0.5, 0.67\} \]
\[ M_2 = \{5.0, 1.0\} \]

Errors:

\[ e_1^2 = 4.17 \]
\[ e_2^2 = 2.00 \]
\[ E^2 = 6.17 \]
INCREMENTAL CLUSTERING

2. Assign the first data item to the first cluster.

3. Consider the next data item. Either assign this item to one of the existing clusters or assign it to a new cluster. This assignment is done based on some criterion, e.g. the distance between the new item and the existing cluster centroids. In that case, after every addition of a new item to an existing cluster recompute a new value for the centroid.

4. Repeat previous step 2 till all the data samples are clustered.

- The space requirements of incremental algorithm are very small; only for centroids of the clusters.
- If these algorithms are noniterative, their time requirements are also small.
- Even if we introduce iterations into incremental clustering algorithm, computational complexity and corresponding time requirements do not increase significantly.
- Do not satisfy: order-independence.

(INCREMENTAL CLUSTERING (Example))

Data: \( x_1 = (0, 2), x_2 = (0, 0), x_3 = (1.5, 0), x_4 = (5, 0), x_5 = (5, 2). \)
Threshold: \( \delta = 3 \)

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>First sample ( x_1 ) will become first cluster ( C_1 = {x_1} ) -&gt; centroid ( M_1 = {0, 2} ).</td>
</tr>
<tr>
<td>2</td>
<td>Start analysis of other samples:</td>
</tr>
<tr>
<td>2.1</td>
<td>Second sample ( x_2 ) is compared with ( M_1 ):</td>
</tr>
<tr>
<td></td>
<td>( d(x_2, M_1) = (0^2 + 2^2)^{1/2} = 2.0 &lt; 3 )</td>
</tr>
<tr>
<td></td>
<td>Therefore, ( x_2 \in C_1 ). New centroid will be: ( M_1 = {0, 1} )</td>
</tr>
<tr>
<td>2.2</td>
<td>Third sample ( x_3 ) is compared with the centroid ( M_1 ) (still the only centroid!):</td>
</tr>
<tr>
<td></td>
<td>( d(x_3, M_1) = (1.5^2 + 1^2)^{1/2} = 1.8 &lt; 3 )</td>
</tr>
<tr>
<td></td>
<td>( x_3 \in C_1 \Rightarrow C_1 = {x_1, x_2, x_3} \Rightarrow M_1 = {0.5, 0.66} )</td>
</tr>
<tr>
<td>2.3</td>
<td>Fourth sample ( x_4 ) is compared with the centroid ( M_1 ):</td>
</tr>
<tr>
<td></td>
<td>( d(x_4, M_1) = (4.5^2 + 0.66^2)^{1/2} = 4.55 &gt; 3 )</td>
</tr>
<tr>
<td></td>
<td>( C_2 = {x_4} ) with the new centroid ( M_2 = {5, 0} ).</td>
</tr>
<tr>
<td>2.4</td>
<td>Fifth example ( x_5 ) is comparing with both cluster centroids:</td>
</tr>
<tr>
<td></td>
<td>( d(x_5, M_1) = (4.5^2 + 1.44^2)^{1/2} = 4.72 &gt; 3 )</td>
</tr>
<tr>
<td></td>
<td>( d(x_5, M_2) = (0^2 + 2^2)^{1/2} = 2 &lt; 3 )</td>
</tr>
<tr>
<td></td>
<td>( C_2 = {x_4, x_5} \Rightarrow M_2 = {5, 1} )</td>
</tr>
<tr>
<td>3</td>
<td>All samples are analyzed and a final clustering solution:</td>
</tr>
<tr>
<td></td>
<td>( C_1 = {x_1, x_2, x_3} ) and ( C_2 = {x_4, x_5} )</td>
</tr>
</tbody>
</table>
K-nearest neighbor

• APPLIED FOR CATEGORICAL DATA

1. The distances between new sample and all previous samples, already classified into clusters, are computed.

2. The distances are sorted in increasing order, and K samples with smallest distance values are selected.

3. Voting principle is applied:
   New sample will be added (classified) to the cluster that belongs the largest number out of K selected samples.

K-nearest neighbor
(Example)

• Given six 6-dimensional categorical samples:
  
  \[ X_1 = \{A, B, A, B, C, B\} \]
  \[ X_2 = \{A, A, A, B, A, B\} \]
  \[ X_3 = \{B, B, A, B, A, B\} \]
  \[ X_4 = \{B, C, A, B, B, A\} \]
  \[ X_5 = \{B, A, B, A, C, A\} \]
  \[ X_6 = \{A, C, B, A, B, B\} \]

  clustered into two clusters \(C_1 = \{X_1, X_2, X_3\}\) and \(C_2 = \{X_4, X_5, X_6\}\).

• Classify the new sample: \(Y = \{A, C, A, B, C, A\}\)

1. Using SMC measure:

   - Similarities with elements in \(C_1\):
     - \(SMC(Y, X_1) = 4/6 = 0.66\)
     - \(SMC(Y, X_2) = 3/6 = 0.50\)
     - \(SMC(Y, X_3) = 2/6 = 0.33\)

   - Similarities with elements in \(C_2\):
     - \(SMC(Y, X_4) = 4/6 = 0.66\)
     - \(SMC(Y, X_5) = 2/6 = 0.33\)

2. Using 1-nearest neighbor rule (K = 1) new sample cannot be classified because there are two samples \((X_1\) and \(X_4\)) with the same, highest similarity (smallest distances), and one of them is in class \(C_1\) and the other in the class \(C_2\).

2a. Using 3-nearest neighbor rule (K = 3), and selecting three largest similarities in the set, we can see that two samples \((X_1\) and \(X_3\)) belongs to class \(C_1\), and only one sample to class \(C_2\).

3. Using simple voting system: \(Y \in C_1\) class.
DECISION TREES

- Efficient method for producing classifiers from data (logic method).
- Supervised learning methods that construct decision trees from a set of input-output samples
- It guarantees that a simple, but not necessary the simplest, tree is found

A decision tree consists of nodes that are tests on the attributes.
The outgoing branches of a node correspond to all the possible outcomes of the test at the node.
Leaves are sets of samples belong to the same class

DECISION TREE ALGORITHMS

Requirements:
1. Attribute-value description:
2. Predefined classes:
3. Discrete classes
4. Sufficient data
5. “Logical” classification models

Examples:
- CLS
- ID3
- C4.5 (C5.0)
C4.5 algorithm

- Let the classes be denoted \( \{C_1, C_2, ..., C_k\} \). There are three possibilities for the content of the set of training samples \( T \) in the given node of decision tree:

1. \( T \) contains one or more samples, all belonging to a single class \( C_j \). The decision tree for \( T \) is a leaf identifying class \( C_j \).

2. \( T \) contains no samples. The decision tree is again a leaf, but the class to be associated with the leaf must be determined from information other than \( T \), such as the overall majority class in \( T \). C4.5 algorithm uses as a criterion the most frequent class at the parent of the given node.

3. \( T \) contains samples that belong to a mixture of classes. In this situation, the idea is to refine \( T \) into subsets of samples that are heading towards single-class collections of samples. An appropriate test is chosen, based on single attribute, that has one or more mutually exclusive outcomes \( \{O_1, O_2, ..., O_n\} \):

   \( T \) is partitioned into subsets \( T_1, T_2, ..., T_n \) where \( T_i \) contains all the samples in \( T \) that have outcome \( O_i \) of the chosen test. The decision tree for \( T \) consists of a decision node identifying the test and one branch for each possible outcome.

TEST - entropy:

- If \( S \) is any set of samples, let \( freq(C_i, S) \) stand for the number of samples in \( S \) that belong to class \( C_i \) (out of \( k \) possible classes), and \( |S| \) denotes the number of samples in the set \( S \). Then the entropy of the set \( S \):

\[
\text{Info}(S) = - \sum_{i=1}^{k} \left( \frac{\text{freq}(C_i, S)}{|S|} \right) \cdot \log_2 \left( \frac{\text{freq}(C_i, S)}{|S|} \right)
\]

- After set \( T \) has been partitioned in accordance with \( n \) outcomes of one attribute test \( X \):

\[
\text{Info}_x(T) = - \sum_{i=1}^{n} \left( \frac{|T_i|}{|T|} \right) \cdot \text{Info}(T_i)
\]

\[
\text{Gain}(X) = \text{Info}(T) - \text{Info}_x(T)
\]

- Criterion: select an attribute with the highest Gain value.
### C4.5 algorithm

#### (Example)

**Database T:**

<table>
<thead>
<tr>
<th>Attribute1</th>
<th>Attribute2</th>
<th>Attribute3</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>70</td>
<td>True</td>
<td>CLASS1</td>
</tr>
<tr>
<td>A</td>
<td>90</td>
<td>True</td>
<td>CLASS2</td>
</tr>
<tr>
<td>A</td>
<td>85</td>
<td>False</td>
<td>CLASS2</td>
</tr>
<tr>
<td>A</td>
<td>95</td>
<td>False</td>
<td>CLASS2</td>
</tr>
<tr>
<td>A</td>
<td>70</td>
<td>False</td>
<td>CLASS1</td>
</tr>
<tr>
<td>B</td>
<td>90</td>
<td>True</td>
<td>CLASS1</td>
</tr>
<tr>
<td>B</td>
<td>78</td>
<td>False</td>
<td>CLASS1</td>
</tr>
<tr>
<td>B</td>
<td>65</td>
<td>True</td>
<td>CLASS1</td>
</tr>
<tr>
<td>B</td>
<td>75</td>
<td>False</td>
<td>CLASS1</td>
</tr>
<tr>
<td>C</td>
<td>80</td>
<td>True</td>
<td>CLASS2</td>
</tr>
<tr>
<td>C</td>
<td>70</td>
<td>True</td>
<td>CLASS2</td>
</tr>
<tr>
<td>C</td>
<td>80</td>
<td>False</td>
<td>CLASS1</td>
</tr>
<tr>
<td>C</td>
<td>80</td>
<td>False</td>
<td>CLASS1</td>
</tr>
<tr>
<td>C</td>
<td>96</td>
<td>False</td>
<td>CLASS1</td>
</tr>
</tbody>
</table>

**Info(T) = -9/14 \log_2 (9/14) - 5/14 \log_2 (5/14) = 0.940 bits**

**Attribute1:** $\text{Info}_x(T) = \frac{5}{14} (-2/5 \log_2 (2/5) - 3/5 \log_2 (3/5))$

$\quad + \frac{4}{14} (-4/4 \log_2 (4/4) - 0/4 \log_2 (0/4))$

$\quad + \frac{5}{14} (-3/5 \log_2 (3/5) - 2/5 \log_2 (2/5))$

$= 0.694 \text{ bits}$

**Gain (x₁) = 0.940 – 0.694 = 0.246 bits**

**Attribute3:** $\text{Info}_x(T) = \frac{6}{14} (-3/6 \log_2 (3/6) - 3/6 \log_2 (3/6))$

$\quad + \frac{8}{14} (-6/8 \log_2 (6/8) - 2/8 \log_2 (2/8))$

$= 0.892 \text{ bits}$

**Gain (x₂) = 0.940 – 0.892 = 0.048 bits**

#### C4.5 algorithm

C4.5 contains mechanisms for proposing three types of tests:

a) The “standard” test on a discrete attribute, with one outcome and branch for each possible value of that attribute.

b) If attribute Y has continuous numeric values, a binary test with outcomes $Y \leq Z$ and $Y > Z$ could be defined, based on comparing the value of attribute against a threshold value $Z$.

c) A more complex test based also on a discrete attribute, in which the possible values are allocated to a variable number of groups with one outcome and branch for each group.

---

b*) **Threshold value $Z$:**

- The training samples are first sorted on the values of the attribute $Y$ being considered. There are only a finite number of these values, so let us denote them in sorted order as $\{v₁, v₂, ..., vₘ\}$.

- Any threshold value lying between $v_i$ and $v_{i+1}$ will have the same effect of dividing the cases into those whose value of the attribute $Y$ lies in $\{v₁, v₂, ..., v_i\}$ and those whose value is in $\{v_{i+1}, v_{i+2}, ..., vₘ\}$. There are thus only $m-1$ possible splits on $Y$, all of which should be examined systematically to obtain an optimal split.

- It is usual to choose the midpoint of each interval: $(v_i + v_{i+1})/2$ as the representative threshold. C4.5 chooses as the threshold a smaller value $v_i$ for every interval $\{v_i, v_{i+1}\}$, rather than the midpoint itself.
C4.5 algorithm

*Example*

Attribute2: After a sorting process, the set of values is:

\{65, 70, 75, 78, 80, 85, 90, 95, 96\},

and the set of potential threshold values Z is:

\{65, 70, 75, 78, 80, 85, 90, 95\}.

The optimal Z value is Z=80 and the corresponding process of information gain computation for the test $x_3$ (Attribute2 ≤ 80 or Attribute2 > 80).

$$\text{Info}_{x_3}(T) = \frac{9}{14} \left( -\frac{7}{9} \log_2 \left( \frac{7}{9} \right) - \frac{2}{9} \log_2 \left( \frac{2}{9} \right) \right) + \frac{5}{14} \left( -\frac{2}{5} \log_2 \left( \frac{2}{5} \right) - \frac{3}{5} \log_2 \left( \frac{3}{5} \right) \right) = 0.837 \text{ bits}$$

Gain($x_3$) = 0.940 - 0.837 = 0.103 bits

Attribute1 gives the highest gain of 0.246 bits, and therefore this attribute will be selected for the first splitting.

C4.5 algorithm

- Gain criterion has a serious deficiency: a strong bias in favor of tests with many outcomes.
- Solution was found in some kind of normalization:

\[ \text{Gain-ratio}(X) = \frac{\text{Gain}(X)}{\text{Split-info}(X)} \]

Example:

\[ \text{Split-info}(x_1) = - \sum_{i=1}^{n} \left( \frac{|T_i|}{|T|} \log_2 \left( \frac{|T_i|}{|T|} \right) \right) \]

Gain-ratio($x_1$) = 0.246 / 1.557 = 0.156
C4.5 algorithm
(UNKNOWN VALUES)

In C4.5 it is accepted a principle that samples with the unknown values are distributed probabilistically according to the relative frequency of known values.

The new gain criterion will have the form:
\[
\text{Gain}(x) = F \cdot (\text{Info}(T) - \text{Info}_x(T))
\]
where
\[
F = \frac{\text{number of samples in database with known value for a given attribute}}{\text{total number of samples in a data set}}
\]

Database T:

<table>
<thead>
<tr>
<th>Attribute1</th>
<th>Attribute2</th>
<th>Attribute3</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>70</td>
<td>True</td>
<td>CLASS1</td>
</tr>
<tr>
<td>A</td>
<td>90</td>
<td>True</td>
<td>CLASS2</td>
</tr>
<tr>
<td>A</td>
<td>85</td>
<td>False</td>
<td>CLASS2</td>
</tr>
<tr>
<td>A</td>
<td>95</td>
<td>False</td>
<td>CLASS1</td>
</tr>
<tr>
<td>?</td>
<td>90</td>
<td>True</td>
<td>CLASS1</td>
</tr>
<tr>
<td>B</td>
<td>78</td>
<td>False</td>
<td>CLASS1</td>
</tr>
<tr>
<td>B</td>
<td>65</td>
<td>True</td>
<td>CLASS1</td>
</tr>
<tr>
<td>C</td>
<td>80</td>
<td>True</td>
<td>CLASS2</td>
</tr>
<tr>
<td>C</td>
<td>70</td>
<td>True</td>
<td>CLASS2</td>
</tr>
<tr>
<td>C</td>
<td>80</td>
<td>False</td>
<td>CLASS1</td>
</tr>
<tr>
<td>C</td>
<td>96</td>
<td>False</td>
<td>CLASS1</td>
</tr>
</tbody>
</table>

\[
\text{Info}(T) = -\frac{8}{13} \log_2 \left(\frac{8}{13}\right) -\frac{5}{13} \log_2 \left(\frac{5}{13}\right) = 0.961 \text{ bits}
\]

\[
\text{Info}_x(T) = \frac{5}{13} (-\frac{2}{5} \log_2 \left(\frac{2}{5}\right) - \frac{3}{5} \log_2 \left(\frac{3}{5}\right)) + \frac{3}{13} (-\frac{3}{3} \log_2 \left(\frac{3}{3}\right) - \frac{0}{3} \log_2 \left(\frac{0}{3}\right)) + \frac{1}{14} (-\frac{3}{5} \log_2 \left(\frac{3}{5}\right) - \frac{2}{5} \log_2 \left(\frac{2}{5}\right)) = 0.747 \text{ bits}
\]

\[
\text{Gain}(x_1) = \frac{13}{14} (0.961 - 0.747) = 0.199 \text{ bits}
\]

\[
\text{Split-info}(x_1) = -\left(\frac{5}{13} \log_2 (5/13) + \frac{3}{13} \log_2 (3/13) + \frac{5}{13} \log_2 (5/13) + \frac{1}{13} \log_2 (1/3)\right) = 1.876
\]
PRUNING DECISION TREE

- Discarding one or more subtrees and replacing them with leaves simplify decision tree and that is the main task in decision tree pruning:
  - Prepruning
  - Postpruning

- C4.5 follows a postpruning approach (pessimistic pruning).

\[ \text{If} \ A=1 \ \text{and} \ B=1 \ \text{Then} \ \text{CLASS1} \]
\[ \text{If} \ A=1 \ \text{and} \ B=2 \ \text{Then} \ \text{CLASS1} \]
\[ \text{If} \ A=2 \ \text{Then} \ \text{CLASS2} \]

Using default confidence of 25%, upper confidence limits for all nodes are collected from statistical tables:

- \[ U_{25\%}(6,0) = 0.206 \]
- \[ U_{25\%}(9,0) = 0.143 \]
- \[ U_{25\%}(1,0) = 0.750 \]
- \[ U_{25\%}(16,1) = 0.157 \]

Predicted errors for the initial tree and replaced node are:

- \[ P_{\text{tree}} = 6 \cdot 0.206 + 9 \cdot 0.143 + 1 \cdot 0.750 = 3.257 \]
- \[ P_{\text{node}} = 16 \cdot 0.157 = 2.512 \]

Since \( P_{\text{tree}} > P_{\text{node}} \), replace the subtree with the new leaf node.

C4.5 ALGORITHM: GENERATING DECISION RULES

Decision rules for database T:

- If \( \text{Attribute1} = A \) and \( \text{Attribute2} \leq 70 \) Then Classification = CLASS1 (2.0 / 0);
- If \( \text{Attribute1} = A \) and \( \text{Attribute2} > 70 \) Then Classification = CLASS2 (3.4 / 0.4);
- If \( \text{Attribute1} = B \) Then Classification = CLASS1 (3.2 / 0);
- If \( \text{Attribute1} = C \) and \( \text{Attribute3} = \text{True} \) Then Classification = CLASS2 (2.4 / 0);
- If \( \text{Attribute1} = C \) and \( \text{Attribute3} = \text{False} \) Then Classification = CLASS1 (3.0 / 0).
Limitations of decision trees and decision rules

- Given class is supported if \( n \) out of \( m \) conditions are presented. To represent this classifier with rules, it would be necessary to define \( \binom{m}{n} \) regions only for one class:

  Medical diagnostic: If 4 out of 11 symptoms support diagnosis of a given disease, then the corresponding classifier will generate 330 regions in 11-dimensional space for positive diagnosis only. That corresponds to 330 decision rules.

- Introducing new attributes, rather than removing old ones, can avoid sometimes-intensive fragmentation of the \( n \)-dimensional space.

CMAR (Classification based on Multiple Association Rules)

- Adopted from the FP-growth

**Phases:**

4. rule generation or training (\( R: P \rightarrow c \), such that \( sup(R) \) and \( conf(R) \) pass the given thresholds), and

5. classification or testing phase (predict the classification of the new sample).

- Training database \( T \) for CMAR algorithm (the support threshold is 2 and the confidence threshold is 70%).

<table>
<thead>
<tr>
<th>ID</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>01</td>
<td>a_1</td>
<td>b_1</td>
<td>c_1</td>
<td>d_1</td>
<td>A</td>
</tr>
<tr>
<td>02</td>
<td>a_1</td>
<td>b_2</td>
<td>c_1</td>
<td>d_2</td>
<td>B</td>
</tr>
<tr>
<td>03</td>
<td>a_2</td>
<td>b_3</td>
<td>c_2</td>
<td>d_1</td>
<td>A</td>
</tr>
<tr>
<td>04</td>
<td>a_1</td>
<td>b_2</td>
<td>c_1</td>
<td>d_1</td>
<td>C</td>
</tr>
<tr>
<td>05</td>
<td>a_1</td>
<td>b_2</td>
<td>c_1</td>
<td>d_3</td>
<td>C</td>
</tr>
</tbody>
</table>

- F-list = (\( a_1, b_2, c_1, d_3 \))

- FP-tree is a prefix tree with respect to F-list
**CMAR (Classification based on Multiple Association Rules)**

- **Rules subsets:**
  a) The rules having $d_3$ value;
  b) The rules having $c_1$ but no $d_3$;
  c) The rules having $b_2$ but no $d_3$ nor $c_1$; and
  d) The rules having only $a_1$.

- $d_3$-projected samples: $(a_1, b_2, c_1, d_3):C$, $(a_1, b_2, d_3):C$, and $(d_3)$:

  Rule: $A(a_1, b_2) \rightarrow C$ (support $= 2$, confidence $= 100\%$)

- $(a_1, c_1)$ is a frequent pattern with support 3, but all rules are with confidence less than threshold value. Similar conclusions are for pattern $(a_1, b_2)$, and finally for $(a_1)$.

**MARKET BASKET ANALYSIS**

- **A market basket** is a collection of items purchased by a customer in an individual customer transaction.

<table>
<thead>
<tr>
<th>TID</th>
<th>Items</th>
</tr>
</thead>
<tbody>
<tr>
<td>001</td>
<td>A C D</td>
</tr>
<tr>
<td>002</td>
<td>B C E</td>
</tr>
<tr>
<td>003</td>
<td>A B C E</td>
</tr>
<tr>
<td>004</td>
<td>B E</td>
</tr>
</tbody>
</table>

**RULES & MEASURES**

The association rule $X \Rightarrow Y$ holds in the transaction set $DB$ with confidence $c$ if $c\%$ of transactions in $D$ that contain $X$ also contain $Y$.

The association rule $X \Rightarrow Y$ has support $s$ in the transaction set $D$ if $s\%$ of transactions in $DB$ contain $X \cup Y$.

**ASSOCIATION RULES MINING PROCES:**

1. Discover the large itemsets, i.e., the sets of items that have transaction support $s$ above a pre-determined minimum threshold.
2. Use the large itemsets to generate the association rules for the database that have confidence $c$ above pre-determined minimum threshold.
Algorithm Apriori

- Finds all frequent itemsets in DB.
  - Works iteratively: 1-itemsets, 2-itemsets, etc.
  - Iteration $i$ computes all frequent $i$-itemsets (itemsets with $i$ elements). $i$th iteration is the last one if there is no frequent $i+1$ itemsets.
  - Each of iterations has two steps:
    - candidate generation, and
    - candidate counting and selection.

From frequent itemsets to association rules

Process:

1. For the rule in a form of implication: \( \{x_1, x_2, x_3\} \rightarrow x_4 \), it is necessary that both, itemsets \( \{x_1, x_2, x_3, x_4\} \) and \( \{x_1, x_2, x_3\} \) are frequent.
2. The confidence $c$ of the rule is computed as quotient of supports for itemsets: $c = \frac{s(x_1, x_2, x_3, x_4)}{s(x_1, x_2, x_3)}$.

Example:

Check is the association rule: \{B, C\} \rightarrow E a strong rule?

We select corresponding supports from tables L2 and L3:

\[
\begin{align*}
  s(B, C) &= 2, \\
  s(B, C, E) &= 2
\end{align*}
\]

and using these supports we compute the confidence of the rule:

\[
c(\{B, C\} \rightarrow E) = \frac{s(B, C, E)}{s(B, C)} = \frac{2}{2} = 1 \text{ (or 100%)}
\]

- To filter out misleading associations:

\[
s(A, B) / s(A) - s(B) > d
\]
Improving the efficiency of the Apriori algorithm

1. **Partition** based Apriori

2. **Sampling** based Apriori

3. **Incremental updating** techniques

4. **Concept level** associations
   - *is-a* hierarchy on database items

**FREQUENT PATTERN GROWTH METHOD (FTP-growth)**

*Example:*
Given minimum support threshold of 3 and transactional database T:

<table>
<thead>
<tr>
<th>TID</th>
<th>Itemset</th>
</tr>
</thead>
<tbody>
<tr>
<td>01</td>
<td>c, d, g, i, m, p</td>
</tr>
<tr>
<td>02</td>
<td>c, f, l, m, o</td>
</tr>
<tr>
<td>03</td>
<td>h, j, o</td>
</tr>
<tr>
<td>04</td>
<td>k, s, p</td>
</tr>
<tr>
<td>05</td>
<td>f, e, l, p, m, n</td>
</tr>
</tbody>
</table>

- First scan of database T derives a list L:
  \[ L = \{(f,4), (c,4), (a,3), (b,3), (m,3), (p,3)\}. \]
- The root of the tree, labeled with ROOT is created.
- Scan the database T second time.
  - The scan of the first transaction leads to the construction of the first branch of the FP-tree:
    \[ \{(f,1), (c,1), (a,1), (m,1), (p,1)\}. \]
  - For the second transaction: since it shares common items, f, c, and a, it shares the common prefix \{f, c, a\} with the previous branch, and extend to the new branch:
    \[ \{(f, 2), (c, 2), (a, 2), (m, 1), (p, 1)\}. \]
FREQUENT PATTERN GROWTH METHOD

To facilitate tree traversal, an *item header table* is built, in which each item in L list connects nodes in FP-tree with the same item value through node-links.

According to the list of frequent items L, the complete set of frequent itemsets can be divided into subsets (6 for our example) without overlap:

1) frequent itemsets having item \( p \) (the end of L list);
2) the itemsets having item \( m \) but no \( p \);
3) the frequent itemsets with \( b \) and without both \( m \) and \( p \);
   ...
6) the large itemsets only with \( f \).

Repeating the same process for subsets 3) to 6) in our example, additional frequent itemsets could be mined. In our example these are itemsets \{f, c, a\} and \{f, c\}, but they are already subsets of the frequent itemset \{f, c, a, m\}. Therefore the final solution of the FP-growth method is the set of frequent itemsets, which is in our example:

\{ \{c, p\}, \{f, c, a, m\} \}

- FP-growth algorithm is about an order of magnitude faster than Apriori algorithm.
MULTIDIMENSIONAL ASSOCIATION RULES MINING

DB schema: \((ID, A_1, A_2, ..., A_n, items)\)

<table>
<thead>
<tr>
<th>ID</th>
<th>A_1</th>
<th>A_2</th>
<th>A_3</th>
<th>items</th>
</tr>
</thead>
<tbody>
<tr>
<td>01</td>
<td>a</td>
<td>1</td>
<td>m</td>
<td>x, y, z</td>
</tr>
<tr>
<td>02</td>
<td>b</td>
<td>2</td>
<td>n</td>
<td>z, w</td>
</tr>
<tr>
<td>03</td>
<td>a</td>
<td>2</td>
<td>m</td>
<td>x, z, w</td>
</tr>
<tr>
<td>04</td>
<td>c</td>
<td>3</td>
<td>p</td>
<td>x, w</td>
</tr>
</tbody>
</table>

Multi-dimensional transactional database DB

1.1 Sort all tuples in the database in alphabetical order of values for the first dimension \((A_1)\). The only MD-pattern found for this dimension is \((a, *, *)\).

1.2 Select tuples in a database with found MD-pattern (or patterns). In our database there are samples with ID values 01 and 03.

1.3 Reduced database sort again with respect to the second dimension \((A_2)\), where values are 1 and 2, and there is no MD-patterns above the threshold.

1.4 Selected tuples in a database are sorted in alphabetic order of values for the third dimension (in our example \(A_3\)). A sub-group \((a, *, m)\) is contained in two tuples and it is MD-pattern.

2. Repeat the processes in step 1 only starting with the second dimension, and then third dimension, etc.

MULTIDIMENSIONAL ASSOCIATION RULES MINING

ROOT

\((A_1, *, *)\) \quad \((*, A_2, *)\) \quad \((*, *, A_3)\)

\((A_1, A_2, *)\) \quad \((A_1, *, A_3)\) \quad \((*, A_2, A_3)\)

\((A_1, A_2, A_3)\)

MD-patterns: \((a, *, m)\), \((*, 2, *)\)

- SECOND PHASE:
  ONE OF STANDARD ASSOCIATION RULES MINING ALGORITHMS APPLIED ON SUBSETS OF DATABASE CONTAINING MD-PATTERNS.
WEB MINING

Four subtasks of Web mining:

1. **Resource finding** – is the process of retrieving the data.

2. **Information selection and pre-processing** – is the process of different kind of transformation of the original data retrieved.

3. **Generalization** – is a process of automatically discovering general patterns at individual Web sites as well as across multiple sites.

4. **Analysis** - this is a phase in which validation and/or interpretation of the mined patterns is performed.

Factors affecting Web mining process:

i. Web page content,

ii. Web page design, and

iii. Overall site design including its structure.

WEB MINING

Three areas of Web mining:

1. **Web content mining** - describes the discovery of useful information from the Web documents.

2. **Web structure mining** - tries to discover the model underlying the link structure on the Web.

3. **Web usage mining** - tries to make sense of the data generated by the Web surfer’s sessions or behaviors.
HITS algorithm

Two types of important pages:

- **authorities**, which provide the best source of information on a given topic, and
- **hubs**, which provide collections of links to authorities.

Two main steps of the HITS algorithm are:

1. **sampling component**, which constructs a focused collection of Web pages likely to be rich in relevant information, and
2. **weight-propagation component**, which determines estimates of hubs and authorities by an iterative procedure, and obtain the subset of most relevant and authoritative Web pages.

**BASIC IDEAS:**

- We associate a nonnegative authority weight \( a_p \) and a nonnegative hub weight \( h_p \) with each page \( p \in V \).

- If page is pointed to by many good hubs, we update the value of \( a_p \) for the page \( p \), to be the sum of \( h_q \) over all pages \( q \) that link to \( p \):
  \[
  a_p = \sum h_q, \quad \forall q \text{ such that } q \rightarrow p
  \]
  where the notation \( q \rightarrow p \) indicates that page \( q \) links to page \( p \).

- In a strictly dual fashion, if a page points to many good authorities, we increase its hub weight:
  \[
  h_p = \sum a_q, \quad \forall q \text{ such that } p \rightarrow q.
  \]
HITS algorithm

FORMALIZATION:

- Let us number the pages \{1,2,\ldots, n\} and define their adjacency matrix \(A\) to be \(n \times n\) matrix whose \((i, j)\)\(^{th}\) element is equal to 1 if page \(i\) links to page \(j\), and is 0 otherwise.

- All pages at the beginning of the computation are both hubs and authorities:
  
  \[ a = \{a_1, a_2, \ldots, a_n\} \text{ and } h = \{h_1, h_2, \ldots, h_n\} \]

- Update rules for authorities and hubs can be written in a form:
  
  \[ a = A^T h \]
  
  \[ h = A a \]

  or substituting one into another relation:

  \[ a = A^T h = A^T A a = (A^T A) a \]
  
  \[ h = A a = A A^T h = (A A^T) h \]

- These are relations for iterative computation of vectors \(a\) and \(h\).

Initialization of HITS algorithm:

\[
\begin{bmatrix}
0 & 0 & 0 & 1 & 1 & 1
\end{bmatrix}
\begin{bmatrix}
0.1 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}
\]

Subgraph of the linked pages

Adjacency matrix \(A\)

and weight vectors

First iteration:

\[
\begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 1 & 1 \\
1 & 1 & 0 & 0 & 0 & 0 \\
1 & 1 & 1 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 
\end{bmatrix}
\begin{bmatrix}
0.1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0.1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 
\end{bmatrix}
\]

\[
\begin{bmatrix}
0.6 & 0.5 & 0.3 & 0.0 & 0.0 & 0.1
\end{bmatrix}
\begin{bmatrix}
0.6 & 0.5 & 0.3 & 0.0 & 0.0 & 0.1
\end{bmatrix}
\]

Document-5 has the most authority, and Document-1 is the best hub.
LOGSOM algorithm

- We assume that there is a finite set of unique URLs:
  \[ U = \{ \text{url}_1, \text{url}_2, \ldots, \text{url}_n \} \]

  and a finite set of \( m \) user transactions:
  \[ T = \{ t_1, t_2, \ldots, t_m \} \].

- Transactions are represented as a vector with binary values \( u_i \):
  \[ t = [u_1, u_2, \ldots, u_n] \]

  where
  \[ u_i = \begin{cases} 
  1 & \text{if } \text{url}_i \in t \\
  0 & \text{otherwise} 
  \end{cases} \]

- Pre-processed log file can be represented as a binary matrix:

  \[
  \begin{array}{cccc}
  \text{url}_1 & \text{url}_2 & \ldots & \text{url}_n \\
  t_1 & 0 & 1 & 1 \\
  t_2 & 1 & 1 & 0 \\
  \vdots & \vdots & \ddots & \vdots \\
  t_m & 0 & 0 & 0 \\
  \end{array}
  \]

TRANSFORMATIONS:

1. Representing URLs as vectors of transaction group activity:

   | Transaction Groups | 1 | 2 | \ldots | k |
   |-------------------|---|---|--------|
   | \text{url}_1      | 15| 0 | \ldots | 2  |
   | \text{url}_2      | 2 | 1 | \ldots | 10 |
   | \ldots            | \text{url}_n | 0 | 1     | 2  |

2. SOM generated by URLs descriptions:

   | 1 | 2 | 3 | \ldots | p |
   |---|---|---|--------|
   | 1 | 2 | 1 | \ldots | 15|
   | 2 | 3 | 1 | \ldots | 10|
   | \ldots | \text{...} | 54 | \ldots | 11|

- The SOM organize web pages into similar classes based on users’ navigation patterns.
MINING PATH TRAVERSAL PATTERNS

- Capturing user access patterns in Web environment is referred to as mining path traversal patterns.

a) Convert the original sequence of log data into a set of traversal subsequences.
   - Filter out the effect of backward references, which are mainly made for easy of traveling.
   - The new reduced set of user-defined forward paths enables us to concentrate on mining meaningful user access sequences.
   - Each traversal subsequence represents a maximum forward reference from the starting point of a user access.

b) Determine the frequent traversal patterns, termed large reference sequences.
   - A large reference sequence is a sequence that appeared a sufficient number of times in the log database.
   - Form the maximal references obtained from large reference sequences:
     A maximal large sequence is a large reference sequence that is not contained in any other maximal reference sequence.

EXAMPLE:

Path = {A B C D C B E G H G W A O U O V}

\[ \downarrow \]

1. MFR = \{ABCD, ABEGH, ABEGW, AOU, AOV\}.

   If the threshold value is 0.4 (or 40%), large reference sequences LRS with lengths 2, 3, and 4 are:

1. LRS = \{AB, BE, EG, AO, BEG, ABEG\}

3. Maximal reference sequences are:
   \[ \text{MRS} = \{\text{ABEG, AO}\} \].
TEXT MINING

Two key technologies:

1. Internet searching capabilities, and

2. Text analysis methodology.

1. In the information retrieval (IR) domain, documents have been traditionally represented in the vector space model using keywords in text documents. Documents are vectors in this n-dimensional space.

   • If a keyword $t$ occurs $n$ times in document $d$, then the $t$-th coordinate of $d$ is simply $n$. One may choose to normalize the length of the document to 1, using the $L_1$, $L_2$, or $L_\infty$ norms:

     $||d_1|| = \sum_t n(d,t)$,

     $||d_2|| = \sqrt{\sum_t n(d,t)^2}$,

     $||d_\infty|| = \max_t n(d,t)$

   • If $t$ occurs in $n_t$ out of $N$ documents, $n_t/N$ gives sense of rarity, hence, importance of the term.

   • The inverse document frequency:

     $\text{IDF} = 1 + \log \left(\frac{n_t}{N}\right)$

     is used to stretch the axes of the vector space differentially. Thus the $t$-th coordinate of document $d$ may be represented with the value

     $\left( \frac{n(d,t)}{||d_1||} \right) \times \text{IDF}(t)$

     in the weighted vector space model.

2. Text analysis methodology

   • Content-based analysis and partition of documents is a more complicated problem. No standards or common theoretical background have been established in the domain.

   • Generally, you can think of text categorization as comparing a document to other documents or to some predefined set of terms or definitions. This idea is analogous to the type of approach used to construct Kohonen feature maps.

   • The automatic analysis of text information can be used:

     1. To provide an overview of the contents of a large document collection, and to organize them in the most efficient way,

     2. To identify hidden structures between documents or groups of documents,

     3. To increase the efficiency and effectiveness of a search process to find similar or related information, and

     4. To detect duplicate information or documents in an archive.
TEXT MINING PROCESS

Two phases:

- **Text refining** that transforms free-form text documents into a chosen intermediate form, and
- **Knowledge distillation** that deduces patterns or knowledge from an intermediate form.

A text-mining framework:

```
+-----------------------------------------------------------+
| Text refining                                            |
|                                                          |
|                                                          |
| Document-based                                          |
| intermediate form                                       |
|                                                          |
| IF                                                       |
|                                                          |
| Knowledge distillation                                   |
|                                                          |
| Predictive                                              |
| modeling                                               |
| Associative                                              |
| discovery                                              |
|                                                          |
| Concept-based                                           |
| intermediate form                                       |
|                                                          |
| Clustering                                              |
| Visualization                                          |
+-----------------------------------------------------------+
```

ARTIFICIAL NEURAL NETWORKS

- An artificial neural network is a massively parallel distributed processor made up of simple processing units, which has a property of learning experiential knowledge expressed through inter-unit connection strengths, and making it available for use.

- **Properties:**
  - Nonlinearity
  - Learning from examples
  - Adaptivity
  - Evidential Response
  - Fault Tolerance
  - Uniformity of Analysis and Design
MODEL OF AN ARTIFICIAL NEURON

Three basic elements:

1. A set of connecting links from different inputs $x_i$ (or synapses), each of which is characterized by a weight or strength $w_{ki}$.

2. An adder for summing the input signals $x_i$ weighted by the respective synaptic strengths $w_{ki}$.

3. An activation function $f(\text{net})$.

Using adopted notation for $w_{k0} = b_k$ and default input $x_0 = 1$:

$$\text{net}_k = x_0 w_{k0} + x_1 w_{k1} + x_2 w_{k2} + \ldots + x_m w_{km} = \sum_{i=1}^{m} x_i w_{ki}$$

Vector notation:

$$\text{net}_k = X \cdot W$$

where:

- $X = \{x_0, x_1, x_2, \ldots, x_m\}$
- $W = \{w_{k0}, w_{k1}, w_{k2}, \ldots, w_{km}\}$

Finally, an artificial neuron computes the output $y_k$ as a certain function of $\text{net}_k$ value:

$$y_k = f(\text{net}_k)$$
MODEL OF ARTIFICIAL NEURONS
(EXAMPLES)

a) A single node

\[ \begin{align*}
    & x_1 = 0.5, \\ & x_2 = 0.5, \\ & x_3 = 0.2
\end{align*} \]

[Diagram showing a single node with inputs and a single output.]

\[ \text{\( \Sigma f \) } \]

\[ y_1 = f(0.5) = 0.5 \]

b) Three interconnected nodes

\[ \begin{align*}
    & x_1 = 1, \\ & x_2 = 0.5, \\ & x_3 = 0.2
\end{align*} \]

[Diagram showing three interconnected nodes with inputs and outputs.]

\[ \begin{align*}
    & y_1 = 0.15, \\ & y_2 = -1, \\ & y_3 = 0.95
\end{align*} \]

---

a1) Symmetrical Hard Limit

\[ \begin{align*}
    \text{net} &= 0.5 \cdot 0.3 + 0.5 \cdot 0.2 + 0.5 \cdot (-0.2) - 1 = 0.15 \\
    y &= f(\text{net}) = f(0.15) = 1
\end{align*} \]


a2) Saturating Linear

\[ \begin{align*}
    \text{net} &= 0.15 \text{ (computation is the same as for case a) } \\
    y &= f(\text{net}) = f(0.15) = 0.15
\end{align*} \]


a3) Log-Sigmoid

\[ \begin{align*}
    \text{net} &= 0.15 \text{ (computation is the same as for case a) } \\
    y &= f(\text{net}) = f(0.15) = \frac{1}{1+e^{-0.15}} = 0.54
\end{align*} \]

---

b) Symmetric saturating linear

\[ \begin{align*}
    \text{net}_1 &= 1 \cdot 0.2 + 0.5 \cdot 0.5 = 0.45 \\
    \text{net}_2 &= 1 \cdot (-0.6) + 0.5 \cdot (-1) = -1.1
\end{align*} \]

\[ \begin{align*}
    y_1 &= f(0.45) = 0.45 \\
    y_2 &= f(-1.1) = -1
\end{align*} \]

\[ \begin{align*}
    \text{net}_3 &= y_1 \cdot 1 + y_2 \cdot (-0.5) = 0.45 \cdot 1 + (-1) \cdot (-0.5) = 0.95 \\
    y_3 &= f(0.95) = 0.95
\end{align*} \]

---

LEARNING PROCESS

- Learning is a process by which the free parameters of a neural network are adapted through a process of stimulation by the environment in which the network is embedded.

- The type of learning is determined by the manner in which the parameter changes take place.

- Every data sample for ANN training (learning) consists of input vector \( X(n) \) and corresponding output \( d(n) \).

\[ \begin{array}{|c|c|}
\hline
\text{Inputs} & \text{Output} \\
\hline x_1, x_2, \ldots, x_m & d_k \\
\hline
\end{array} \]

- Processing the input vector \( X(n) \), a neuron \( k \) produce the output that is denoted by \( y_k(n) \):

\[ y_k(n) = f \left( \sum_{i=1}^{m} x_i w_{ki} \right) \]

- \( y_k(n) \) is compared to a desired response or target output \( d_k(n) \) given in the sample. An error \( e_k(n) \) produced at the output is by definition:

\[ e_k(n) = d_k(n) - y_k(n) \]

- The corrective adjustments are designed to make the output signal \( y_k(n) \) come closer to the desired response \( d_k(n) \) in a step-by-step manner.

- This objective is achieved by minimizing a cost function \( E(n) \) (error energy):

\[ E(n) = \frac{1}{2} e_k^2(n) \]
LEARNING PROCESS

- The learning process based on minimization of a cost function is referred to as error-correction learning.

- In particular, minimization of $E(n)$ leads to a learning rule commonly referred to as the delta rule or Widrow-Hoff rule.

DELTA RULE:

- The adjustment $\Delta w_{kj}(n)$ is defined by:
  $$\Delta w_{kj}(n) = \eta \cdot e(n) \cdot x_j(n)$$
  where $\eta$ is a positive constant that determines the rate of learning.

- The adjustment made to a weight factor of an input neuron connection is proportional to the product of the error signal and the input value of the connection in question.

  $w_{kj}(n+1) = w_{kj}(n) + \Delta w_{kj}(n)$

- In effect, $w_{kj}(n)$ and $w_{kj}(n+1)$ may be viewed as the old and new values of synaptic weight $w_{kj}$, respectively.

(AN EXAMPLE)

DATA:

<table>
<thead>
<tr>
<th>n (sample)</th>
<th>x1</th>
<th>x2</th>
<th>x3</th>
<th>d</th>
<th>y</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>-0.3</td>
<td>0.5</td>
<td>0.7</td>
</tr>
<tr>
<td>2</td>
<td>-1</td>
<td>0.7</td>
<td>0.2</td>
<td>-0.5</td>
<td>0.2</td>
</tr>
<tr>
<td>3</td>
<td>0.3</td>
<td>0.3</td>
<td>-0.3</td>
<td>0.5</td>
<td></td>
</tr>
</tbody>
</table>

$\eta = 0.1$

LEARNING:

- First sample:
  $$\text{net}(1) = 0.5 \cdot 1 + (-0.3) \cdot 1 + 0.8 \cdot 0.5 = 0.6$$
  $$\Downarrow$$
  $$y(1) = f(\text{net}(1)) = f(0.6) = 0.6$$
  $$\Downarrow$$
  $$e(1) = d(1) - y(1) = 0.7 - 0.6 = 0.1$$
  $$\Downarrow$$
  $$\Delta w_1(1) = 0.1 \cdot 0.1 \cdot 1 = 0.01 \Rightarrow w_1(2) = w_1(1) + \Delta w_1(1) = 0.5 + 0.01 = 0.51$$
  $$\Delta w_2(1) = 0.1 \cdot 0.1 \cdot 1 = 0.01 \Rightarrow w_2(2) = w_2(1) + \Delta w_2(1) = -0.3 + 0.01 = -0.29$$
  $$\Delta w_3(1) = 0.1 \cdot 0.1 \cdot 0.5 = 0.005 \Rightarrow w_3(2) = w_3(1) + \Delta w_3(1) = 0.8 + 0.005 = 0.805$$

- Continue the learning process with second and third samples ($n=2$ and $n=3$).
LEARNING TASKS USING ANN’S

- Pattern Association
  
  autoassociation

  heteroassociation

- Pattern Recognition

- Function Approximation

- Control

  Indirect learning

  Direct learning

  Error back-propagation learning consists of two phases:

  - **Forward pass** – computing \( y_j(n) \), and
  
  - **Backward pass** – correct weights \( w_{ji} \) based on error \( e_j(n) \).

  
  \[
  e_j(n) = d_j(n) - y_j(n)
  \]

  \[
  E(n) = \frac{1}{2} \sum_{j \in C} e_j^2(n)
  \]

  Set \( C \) includes all neurons in the output layer of the network.

  
  For \( N \) samples:

  \[
  E_{av} = \frac{1}{N} \sum_{n=1}^{N} E(n)
  \]

  The backpropagation algorithm applies a correction \( \Delta w_{ji}(n) \) to the synaptic weight \( w_{ji}(n) \), which is proportional to the partial derivative \( \frac{\delta E(n)}{\delta w_{ji}(n)} \).

MULTILAYER PERCEPTRON

- Multilayer perceptrons use error back-propagation algorithm for learning.

  Error back-propagation learning consists of two phases:

  - **Forward pass** – computing \( y_j(n) \), and
  
  - **Backward pass** – correct weights \( w_{ji} \) based on error \( e_j(n) \).

  
  \[
  e_j(n) = d_j(n) - y_j(n)
  \]

  \[
  E(n) = \frac{1}{2} \sum_{j \in C} e_j^2(n)
  \]

  Set \( C \) includes all neurons in the output layer of the network.

  
  For \( N \) samples:

  \[
  E_{av} = \frac{1}{N} \sum_{n=1}^{N} E(n)
  \]

  The backpropagation algorithm applies a correction \( \Delta w_{ji}(n) \) to the synaptic weight \( w_{ji}(n) \), which is proportional to the partial derivative \( \frac{\delta E(n)}{\delta w_{ji}(n)} \).
MULTILAYER PERCEPTRON
(Learning)

a) NEURON $j$ IS A HIDDEN NODE

Partial derivatives for $E(n)$ are:

$$\frac{\partial E(n)}{\partial w_{ji}(n)} = \frac{\partial E(n)}{\partial e_j(n)} \cdot \frac{\partial e_j(n)}{\partial y_j(n)} \cdot \frac{\partial y_j(n)}{\partial v_j(n)} \cdot \frac{\partial v_j(n)}{\partial w_{ji}(n)}$$

The partial derivative $\partial E(n)/\partial w_j(n)$ represents a sensitive factor, determining the direction of search in weight space. Knowing:

$$\frac{\partial E(n)}{\partial e_j(n)} = e_j(n) \quad \text{(from } E(n) = \frac{1}{2} \sum e_j^2(n) \text{)}$$
$$\frac{\partial e_j(n)}{\partial y_j(n)} = -1 \quad \text{(from } e_j(n) = d_j(n) - y_j(n) \text{)}$$
$$\frac{\partial y_j(n)}{\partial v_j(n)} = \phi'(v_j(n)) \quad \text{(from } y_j(n) = \phi(v_j(n)) \text{)}$$
$$\frac{\partial v_j(n)}{\partial w_{ji}(n)} = x_i(n) \quad \text{(from } v_j(n) = \sum w_{ji}(n) x_i(n) \text{)}$$

Partial derivative $\partial E(n)/\partial w_j(n)$ we can express in the form:

$$\frac{\partial E(n)}{\partial w_j(n)} = - e_j(n) \cdot \phi'(v_j(n)) \cdot x_i(n)$$

The correction $\Delta w_j(n)$ applied to $w_{ji}(n)$ is defined by delta rule:

$$\Delta w_{ji}(n) = - \eta \cdot \frac{\partial E(n)}{\partial w_{ji}(n)} = \eta \cdot e_j(n) \cdot \phi'(v_j(n)) \cdot x_i(n)$$

or

$$\Delta w_j(n) = \eta \cdot \delta_j(n) \cdot x_i(n)$$

where

$$\delta_j(n) = e_j(n) \cdot \phi'(v_j(n))$$

is the local gradient.

b) NEURON $j$ IS A HIDDEN NODE

- If the activation function is sigmoid:

$$y_j(n) = \phi(v_j(n)) = \frac{1}{1 + e^{-v_j(n)}}$$

then the first derivative is:

$$\phi'(v_j(n)) = \frac{e^{-v_j(n)}}{1 + e^{-v_j(n)}}^2 = y_j(n) (1 - y_j(n))$$

and a final weight correction is:

$$\Delta w_{ji}(n) = \eta \cdot e_j(n) \cdot y_j(n) (1 - y_j(n)) \cdot x_i(n)$$

- If the activation function is hyperbolic tangent:

$$\phi'(v_j(n)) = (1 - y_j(n)) \cdot (1 + y_j(n))$$

$$\Delta w_{ji}(n) = \eta \cdot e_j(n) \cdot (1 - y_j(n)) \cdot (1 + y_j(n)) \cdot x_i(n)$$

b) NEURON $j$ IS A HIDDEN NODE

Going backward, all $\delta_k(n)$ for the nodes in a next layer are known before computation of local gradient $\delta_j(n)$ for a given node on a layer closer to the inputs.
MULTILAYER PERCEPTRON
(Learning)

Momentum

- A simple method of increasing the rate of learning yet avoiding the danger of instability is to modify the delta rule by including so called a momentum term:

\[
\Delta w_{ji}(n) = \eta \cdot \delta_j(n) \cdot x_i(n) + \alpha \cdot \Delta w_{ji}(n-1)
\]

where

\( \alpha \) - is usually a positive number called momentum constant and

\( \Delta w_{ji}(n-1) \) - is the correction of the weight factor for a previous \((n-1)\)th sample (in practice: \( \alpha \in [0.1, 1] \)).

- The addition of the momentum term smooths weight updating and tends to resist erratic weight changes due to gradient noise (a stabilizing effect).

ANN overfitting / overtraining

Generalization as a curve-fitting problem:

Properly fitting curve with good generalization

Overfitted curve
COMPETITIVE NETWORKS

- A set of neurons that have the same structure, and they are connected with initially randomly selected weights. Therefore, the neurons respond differently to a given set of input samples.

- A limit value is determined on the strength of each neuron.

- A mechanism that permits the neurons to compete for the right to respond to a given subset of inputs, such that only one output neuron is active at the time. The neuron that wins the competition is called winner-takes-all neuron.

Connections:

- lateral inhibitory,
- feedforward excitatory

Architecture:

<table>
<thead>
<tr>
<th>Layer of inputs</th>
<th>Single layer of output nodes</th>
<th>Competitive outputs</th>
</tr>
</thead>
<tbody>
<tr>
<td>( x_1 )</td>
<td>( y_1 )</td>
<td>( y_k )</td>
</tr>
<tr>
<td>( x_2 )</td>
<td>( y_2 )</td>
<td></td>
</tr>
<tr>
<td>\ldots</td>
<td>\ldots</td>
<td></td>
</tr>
<tr>
<td>( x_n )</td>
<td>( y_k )</td>
<td></td>
</tr>
</tbody>
</table>

\[ y_k = \begin{cases} 
1 & \text{if net}_k > \text{net}_j \quad \text{for all} \ j, \ j \neq k \\
0 & \text{otherwise} 
\end{cases} \]

\[ \Delta w_{kj} = \begin{cases} 
\eta (x_j - w_{kj}) & \text{if neuron k wins the competition} \\
0 & \text{if neuron k loses the competition} 
\end{cases} \]

---

Geometric interpretation of the competitive learning

Initial state of the network

Final state of the network
**COMPETITIVE NETWORKS**  
*(EXAMPLE)*

New sample: \( X = \{ x_1, x_2, x_3 \} = \{ 1, 0, 1 \} \)

Network:

Learning:

\[
\begin{align*}
\text{net}_1^* &= 0.5 \cdot x_1 + (-0.5) \cdot x_3 = 0.5 \cdot 1 - 0.5 \cdot 1 = 0 \\
\text{net}_2^* &= 0.3 \cdot x_1 + 0.7 \cdot x_2 = 0.3 \cdot 1 + 0.7 \cdot 0 = 0.3 \\
\text{net}_3^* &= 0.2 \cdot x_2 + (-0.2) \cdot x_3 = 0.2 \cdot 0 - 0.2 \cdot 1 = -0.2
\end{align*}
\]

After including lateral inhibitory connections:

\[
\begin{align*}
\text{net}_1 &= \text{net}_1^* + 0.5 \cdot 0.3 + 0.6 \cdot (-0.2) = 0.03 \\
\text{net}_2 &= \text{net}_2^* + 0.2 \cdot 0 + 0.1 \cdot (-0.2) = 0.28 \quad \text{(maximum !\textrightarrow Vinner)} \\
\text{net}_3 &= \text{net}_3^* + 0.4 \cdot 0 + 0.2 \cdot 0.3 = -0.14
\end{align*}
\]

The final outputs from the network for a given sample will be:

\[\mathbf{Y} = \{ y_1, y_2, y_3 \} = \{ 0, 1, 0 \} \]

New weight factors (only for the winning node \( y_2 \)) based on learning rate \( \eta = 0.2 \):

\[
\begin{align*}
w_{12} &= 0.3 + 0.2 \cdot (1 - 0.3) = 0.44 \\
w_{22} &= 0.7 + 0.2 \cdot (0 - 0.7) = 0.56 \\
w_{32} &= 0.0 + 0.2 \cdot (1 - 0.0) = 0.20
\end{align*}
\]

---

**VISUALIZATION METHODS**

- Human vision is still the most powerful means of shifting out irrelevant information and detecting significant patterns.
- Seeing and understanding together enable humans to discover a new knowledge, having deeper insight from large amount of data.
- The power of visualization to exploit human perception offers both a challenge and an opportunity.
- Computer visualization, in particular, is about using computer graphics and other techniques to think about more cases, more variables, and more relations.
- Visual data mining uses computer visualization techniques to augment data mining process.

---

**Visualizations can be used:**

1. To explore data, - *exploratory visualizations*
2. To confirm a hypothesis, - *confirmatory visualizations*
3. To manipulate a view - *manipulative visualizations*.
VISUALIZATION METHODS

Second taxonomy:

- **Scientific visualization**:
  - Focuses primarily on *physical data* such as human body, the earth, molecules.
  - Deals with multi-dimensional data, but most of the data sets used in this field use the *spatial attributes*.

- **Information visualization**:
  - Focuses on abstract, *nonphysical data* such as text, hierarchies, and statistical data.
  - Design a visual representation in a case of *multidimensional samples* (where the number of dimensions is greater than three).

- Data mining techniques are primary oriented toward information visualization.

Information Visualization

Categories:

- **Traditional IVT** (based on 2D and 3D graphics)
  - Small number of dimensions.
  - Include:
    - *Bar chart* that show aggregations and frequencies.
    - *Histogram* shows the distribution of variables values.
    - *Line chart* for understanding trends in ordered.
    - *Pie charts* for visualizing fractions of a total.
    - *Scatter plot* for bivariate analysis.
    - *Color coding*
    - *Data cubes*

- **New IVT**
  - With large number of dimensions at the same time on one screen.
  - Classification:
    1. Geometric projection techniques,
    2. Icon-based techniques,
    3. Pixel-oriented techniques, and
    4. Hierarchical techniques.
New Information Visualization Techniques

1. Geometric projection techniques

- Aim to find interesting projections of multidimensional data sets.
- Includes techniques of exploratory statistics such as
  i. principal component analysis,
  ii. factor analysis, and
  iii. multidimensional scaling.
- Parallel coordinate visualization technique and radial visualization technique belong in this category of visualizations.

- Scatter-Plot Matrix Technique is an approach that is very often available in new data mining software tools. User interactions of zooming and panning are needed to interpret the scatter plots effectively.

- Survey Plot is a simple technique of extending an n-dimensional point (sample) in a line graph.

1. Geometric projection techniques

- 4-dimensional Survey Plot

<table>
<thead>
<tr>
<th>Dimension1</th>
<th>Dimension 2</th>
<th>Dimension 3</th>
<th>Dimension 4</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sample</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- Andrews’s curves technique plots each n-dimensional sample as a curved line transforming the n-dimensional point

\[ X = (x_1, x_2, x_3, \ldots, x_n) \]

into continuous plot:

\[ f(t) = x_1/1.41 + x_2\sin(t) + x_3\cos(t) + x_4\sin(2t) + x_5\cos(2t) + \ldots \]
New Information Visualization Techniques

2. **Icon –based techniques** (or iconic display techniques).
   - The idea is to map each multidimensional data item to an icon.
   - An example is the *stick figure technique*. It maps two dimensions to the display dimensions and the remaining dimensions are mapped to the angles and/or limb lengths of the stick figure icon. This technique limits the number of dimensions that can be visualized.

   - A *star display* for data on seven quality of life measures for three states

3. **Pixel-oriented techniques**
   - They use only one pixel per data value.
   - The techniques allow a visualization of the largest amount of data, which is possible on current displays (up to about 1,000,000 data values).
   - Map each data value to a colored pixel and present the data values belonging to one attribute in separate windows.

4. **Hierarchical techniques**
   - These types of visualization subdivide the k-dimensional space and present the subspaces in a hierarchical fashion. For example, at the lowest level are 2D subspaces.
   - Common representative of these techniques is *dimensional stacking* representation.

   - *Dimensional stacking* is a recursive visualization technique for displaying high-dimensional data.
     - Each dimension is discretized into a small number of bins, and the display area is broken into a grid of sub-images.
     - The sub-images are decomposed further based on the number of bins for two more dimensions. This decomposition process continues recursively until all dimensions have been assigned.
Parallel Coordinates Visualization

- Map the k-dimensional space onto the two display dimensions by using k equidistant axes parallel to one of the display axes.

- The axes correspond to the dimensions and are linearly scaled from the minimum to the maximum value of the corresponding dimension.

- Each data item is presented as a polygonal line.

- **Database with 6 numerical attributes:**

<table>
<thead>
<tr>
<th>Sample#</th>
<th>Dimensions</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td>1</td>
<td>5</td>
<td>10</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>3</td>
<td>1</td>
<td>3</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>4</td>
<td></td>
<td>4</td>
<td>2</td>
<td>1</td>
<td>3</td>
<td>1</td>
</tr>
</tbody>
</table>

- **Parallel Coordinates Visualization**

- **The database for visualization:**

<table>
<thead>
<tr>
<th>Sample#</th>
<th>Dimensions</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>A</td>
</tr>
<tr>
<td>1</td>
<td>low</td>
</tr>
<tr>
<td>2</td>
<td>med.</td>
</tr>
<tr>
<td>3</td>
<td>high</td>
</tr>
<tr>
<td>4</td>
<td>med.</td>
</tr>
<tr>
<td>5</td>
<td>low</td>
</tr>
<tr>
<td>6</td>
<td>low</td>
</tr>
</tbody>
</table>

- **Parallel Coordinates Visualization (Parabox)**

- Circular coordinates method is a simple variation of parallel coordinates, in which the axes radiate from the center of a circle and extend to the perimeter.
Radial Visualization

Radial visualization represents a non-linear transformation of the data, which preserves certain symmetries.

- Data dimensions are laid out as points equally spaced around the perimeter of a circle.
  - For example, in a case of an eight-dimensional space:

- Model of springs is used for point representation.
  - One end of n springs (one spring for each of n dimensions) is attached to the n perimeter points, and the other ends of the springs are attached to a data point.
  - Spring constants can be used to represent values of dimensions for a given point.
  - Values for all dimensions are normalized to the interval between 0 and 1.

Radial Visualization of 4-dimensional point $P(K_1, K_2, K_3, K_4)$ with corresponding spring force:

\[ F = K \cdot d \] (For example: $F_1 = K_1 \cdot [(x -1)i + yj]$)

\[ K_1(x -1)i + K_2x + K_3(x +1)i + K_4x = 0 \]
\[ K_1y + K_2(y + 1) + K_3y + K_4(y -1) = 0 \]

or

\[ x = \frac{(K_1 - K_3)}{(K_1 + K_2 + K_3 + K_4)} \]
\[ y = \frac{(K_4 - K_2)}{(K_1 + K_2 + K_3 + K_4)} \]

4-dimensional point $P^*(K_1, K_2, K_3, K_4) \Rightarrow 2D$ space point $P(x, y)$
Radial Visualization

- If all n coordinates have the same value, the data point will lie exactly in the center of the circle.
  \[ P_1^*(0.6, 0.6, 0.6, 0.6) \text{ will be represented as } P_1(0, 0) \]
- If the n-dimensional point is unit vector for one dimension, then the projected point will lie exactly at the fixed point on the edge of the circle (where the spring for that dimension is fixed):
  Point \( P_2^*(0, 0, 1, 0) \) will be represented as \( P_2(-1, 0) \).
- Points with approximately equal coordinate values will lie close to the center of the representational circle:
  \[ P_3^*(0.5, 0.6, 0.4, 0.5) \text{ will have 2D coordinates: } P_3(0.05, -0.05) \]
- Points, which have one or two coordinate values greater than the others, lie closer to origins of those dimensions:
  \[ P_4^*(0.1, 0.8, 0.6, -0.1) \text{ will have 2D representation } P_4(-0.36, -0.64) \]
- An n-dimensional line will map to the line or in some cases to the point:
  \[ P_5^*(0.3, 0.3, 0.3, 0.3), \ P_6^*(0.6, 0.6, 0.6, 0.6), \ P_7^*(0.9, 0.9, 0.9, 0.9) \]
  \[ P_{567}(0, 0) \]
- A sphere will map to an ellipse.
- An n-dimensional plane maps to a bounded polygon.

Gradviz method is a simple extension of a radial visualization that places the dimensional anchors on a rectangular grid instead of the perimeter of a circle.

Kohonen Self-Organized Maps

- Kohonen neural networks, also called self-organized maps (SOM), are used as a clustering technique based on n-dimensional visualization.

- The Kohonen neural network can be considered as a type of nonlinear data projection onto two dimensions, while trying to preserve the distances between all initially given n-dimensional samples

- The output nodes of the network generate x and y coordinates for each input vector and they can be used for a 2D display: similar data samples are mapped to similar x and y coordinates.

- The n dimensions of samples are used as n input nodes to the network.

- The output nodes of the net are arranged in a rectangular grid (for example a 5×6 grid would contain 30 output nodes).

- The weights of the network are adjusted through the training process so that only one output node is turned on for a given input vector (n-dimensional sample). Input vectors that are “closer” to an output node will reinforce the weights to the given output.
Kohonen Self-Organized Maps

EXAMPLE:

- Trained Kohonen SOM (119 samples) with 9 outputs, represented as a $3 \times 3$ grid.

- Values in nodes are the number of input samples triggered the given output.

Visualization Systems for Data Mining

Principles:

- Simplicity,
- Visibility,
- User autonomy,
- Reliability,
- Reusability,
- Availability, and
- Security.
Visualization Systems for Data Mining

Requirements for a new generation of visualization systems:

1. **Presentation graphics:**
   The next generation of presentation graphics enriches the static displays with 3D or projected n-dimensional information landscape. User can then navigate through the landscape and animate it to display time-oriented information.

2. **Visual interfaces for information access:**
   They are focused on enabling users to navigate through complex information spaces, to locate and retrieve information. Supported user tasks involve searching, back tracking, and history logging. User interface techniques attempt to preserve user context and support smooth transitions between locations.

3. **Full visual discovery and analysis:**
   Manipulate the display to answer the “why” question as well as “what” question. The difference between answering a “what” and a ”why” question involves an interactive operation. Therefore, it requires using:
   - *interaction* techniques (interactive mapping, projection, filtering, zooming, and interactive linking and brushing) and
   - *distortion* techniques (focusing while preserving an overview of the data).

Data Exploration Tasks with New Visualization Tools:

1. **Finding Gestalt:** Local and global linearities and nonlinearities, discontinuities, clusters, outliers, unusual groups, and so on, are examples of Gestalt features that can be of interest.
   - Focusing through individual views is the basic requirement to obtain qualitative data exploration using visualization of data.
   - Focusing determines what gestalt of the data is seen.

2. **Posing queries:** This is a natural task after initial Gestalt features have been found, and query identification and characterization is desired.
   - Queries can concern individual cases as well as subsets of cases. In graphical data analysis it is natural to pose queries graphically.
   - For example, familiar brushing techniques such as coloring or otherwise highlighting a subset of data means issuing a query about this subset. Ideally, responses to queries are instantaneous.

3. **Making comparison:**
   a) *Comparison of variables or projections*, where one compares views “from different angles”, and
   b) *Comparison of subsets of data*. It is based on views “of different slices” of the data.
   - In either case it is likely that large number of plots are generated, and therefore it is challenge to organize the plots in such a way that meaningful comparisons are possible.
VISUALIZATION SYSTEMS FOR DATA MINING

- A stronger strategy lies in tightly coupling the visualization and analytical processes into one data-mining tool.

- Letting human visualization participate in an analytical process’ decision-making remains a major challenge.

An example of the need for general projections, which are not parallel to axes, to improve clustering process:

- Based on the visual representation of the projections, it is possible to find clusters with unexpected characteristics (shapes, dependencies) very difficult or impossible to find by tuning the parameter settings of automatic clustering algorithms.