Data Preparation
(Data pre-processing)

INTRODUCTION TO DATA PREPARATION

Why Prepare Data?

- Some data preparation is needed for all mining tools
- The purpose of preparation is to transform data sets so that their information content is best exposed to the mining tool
- Error prediction rate should be lower (or the same) after the preparation as before it

Data Preparation

- Introduction to Data Preparation
- Types of Data and Basic statistics
- Discretization of Continuous Variables
- Working in the R environment
- Outliers
- Data Transformation
- Missing Data
- Data Integration
- Data Reduction
Why Prepare Data?

- Preparing data also prepares the miner so that when using prepared data the miner produces better models, faster.

- GIGO - good data is a prerequisite for producing effective models of any type.

Data Preparation as a step in the Knowledge Discovery Process

- Data need to be formatted for a given software tool.
- Data need to be made adequate for a given method.
- Data in the real world is dirty:
  - Incomplete: lacking attribute values, lacking certain attributes of interest, or containing only aggregate data.
    - e.g., occupation=“”
  - Noisy: containing errors or outliers.
    - e.g., Salary=-10, Age=222
  - Inconsistent: containing discrepancies in codes or names.
    - e.g., Age=42, Birthday=03/07/1997
    - e.g., Was rating "1,2,3", now rating "A, B, C"
    - e.g., discrepancy between duplicate records
      - e.g., Endereço: travessa da Igreja de Nevogilde, Freguesia: Paranhos

Major Tasks in Data Preparation

- Data discretization:
  - Part of data reduction but with particular importance, especially for numerical data.
- Data cleaning:
  - Fill in missing values, smooth noisy data, identify or remove outliers, and resolve inconsistencies.
- Data integration:
  - Integration of multiple databases, data cubes, or files.
- Data transformation:
  - Normalization and aggregation.
- Data reduction:
  - Obtains reduced representation in volume but produces the same or similar analytical results.
CRISP-DM is a comprehensive data mining methodology and process model that provides anyone—from novices to data mining experts—with a complete blueprint for conducting a data mining project.

CRISP-DM breaks down the life cycle of a data mining project into six phases.

### CRISP-DM Phases and Tasks

**Business Understanding**
- Determine Business Objectives
- Assess Situation

**Data Understanding**
- Collect Initial Data
- Describe Data
- Explore Data
- Verify Data Quality

**Data Preparation**
- Clean Data
- Construct Data
- Integrate Data
- Format Data

**Modelling**
- Select Data
- Select Modeling Technique
- Generate Test Design
- Build Model
- Determine Next Steps

**Evaluation**
- Evaluate Results
- Review Process
- Plan Monitoring & Maintenance

**Deployment**
- Produce Final Report
- Review Project

### CRISP-DM: Data Understanding

- **Collect data**
  - List the datasets acquired (locations, methods used to acquire, problems encountered and solutions achieved).

- **Describe data**
  - Check data volume and examine its gross properties.
  - Accessibility and availability of attributes. Attribute types, range, correlations, the identities.
  - Understand the meaning of each attribute and attribute value in business terms.
  - For each attribute, compute basic statistics (e.g., distribution, average, max, min, standard deviation, variance, mode, skewness).

- **Explore data**
  - Identify patterns and relationships among attributes.
  - Test hypotheses and theories.

- **Significance test**
  - Use statistical tests to determine if the observed patterns are statistically significant.

- **Data limitation**
  - Identify potential limitations of the data.
  - Consider the quality and completeness of the data.

- **Data analysis**
  - Apply various data mining techniques to extract useful information from the data.

- **Data visualization**
  - Use graphs, charts, and other visual tools to present the data in a more understandable format.

- **Data selection and sampling**
  - Select the appropriate data for analysis.
  - Determine the sample size and sampling method.

- **Data cleaning**
  - Remove or correct errors and inconsistencies in the data.

- **Data transformation**
  - Apply transformations to normalize or standardize the data.

- **Data mining**
  - Apply data mining algorithms and models to discover patterns and insights.

- **Data mining report**
  - Prepare a report summarizing the findings and recommendations.

- **Data mining model**
  - Develop a predictive model based on the data mining results.

- **Data mining scenario**
  - Create a scenario that illustrates the use of the data mining model.

- **Data mining selection**
  - Select the appropriate data mining techniques and models.

- **Data mining validation**
  - Validate the data mining results using cross-validation or other validation techniques.

- **Data mining implementation**
  - Implement the data mining model in a real-world application.

- **Data mining maintenance**
  - Maintain the data mining infrastructure and models over time.

- **Data mining monitoring**
  - Monitor the performance of the data mining model and adjust as necessary.

- **Data mining user**
  - Engage stakeholders in the data mining process and communicate the results.
CRISP-DM: Data Understanding

• Explore data
  • Analyze properties of interesting attributes in detail.
    • Distribution, relations between pairs or small numbers of attributes, properties of significant sub-populations, simple statistical analyses.
• Verify data quality
  • Identify special values and catalogue their meaning.
  • Does it cover all the cases required? Does it contain errors and how common are they?
  • Identify missing attributes and blank fields. Meaning of missing data.
  • Do the meanings of attributes and contained values fit together?
  • Check spelling of values (e.g., same value but sometime beginning with a lower case letter, sometimes with an upper case letter).
  • Check for plausibility of values, e.g. all fields have the same or nearly the same values.

CRISP-DM: Data Preparation

• Select data
  • Reconsider data selection criteria.
  • Decide which dataset will be used.
  • Collect appropriate additional data (internal or external).
  • Consider use of sampling techniques.
  • Explain why certain data was included or excluded.
• Clean data
  • Correct, remove or ignore noise.
  • Decide how to deal with special values and their meaning (99 for marital status).
  • Aggregation level, missing values, etc.
  • Outliers?

CRISP-DM: Data Preparation

• Construct data
  • Derived attributes.
  • Background knowledge.
  • How can missing attributes be constructed or imputed?
• Integrate data
  • Integrate sources and store result (new tables and records).
• Format Data
  • Rearranging attributes. (Some tools have requirements on the order of the attributes, e.g. first field being a unique identifier for each record or last field being the outcome field the model is to predict).
  • Reordering records. (Perhaps the modelling tool requires that the records be sorted according to the value of the outcome attribute).
  • Reformatted within-value. (These are purely syntactic changes made to satisfy the requirements of the specific modelling tool, remove illegal characters, uppercase lowercase).

TYPES OF DATA AND BASIC STATISTICS
Types of Data Measurements

- Measurements differ in their nature and the amount of information they give
- Qualitative vs. Quantitative

Types of Measurements: Examples

- Nominal:
  - ID numbers, Names of people
- Categorical:
  - eye color, zip codes
- Ordinal:
  - rankings (e.g., taste of potato chips on a scale from 1-10), grades, height in (tall, medium, short)
- Interval:
  - calendar dates, temperatures in Celsius or Fahrenheit, GRE score
- Ratio:
  - temperature in Kelvin, length, time, counts

Types of Measurements

- Nominal scale
- Categorical scale
- Ordinal scale
- Interval scale
- Ratio scale

More information content

Discrete or Continuous

Types of Measurements: Examples

<table>
<thead>
<tr>
<th>Day</th>
<th>Outlook</th>
<th>Temperature</th>
<th>Humidity</th>
<th>Wind</th>
<th>PlayTennis?</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Sunny</td>
<td>85</td>
<td>85</td>
<td>Light</td>
<td>No</td>
</tr>
<tr>
<td>2</td>
<td>Sunny</td>
<td>80</td>
<td>90</td>
<td>Strong</td>
<td>No</td>
</tr>
<tr>
<td>3</td>
<td>Overcast</td>
<td>83</td>
<td>86</td>
<td>Light</td>
<td>Yes</td>
</tr>
<tr>
<td>4</td>
<td>Rain</td>
<td>70</td>
<td>96</td>
<td>Light</td>
<td>Yes</td>
</tr>
<tr>
<td>5</td>
<td>Rain</td>
<td>85</td>
<td>85</td>
<td>Light</td>
<td>No</td>
</tr>
<tr>
<td>6</td>
<td>Rain</td>
<td>100</td>
<td>100</td>
<td>Light</td>
<td>Yes</td>
</tr>
<tr>
<td>7</td>
<td>Overcast</td>
<td>80</td>
<td>80</td>
<td>Strong</td>
<td>No</td>
</tr>
<tr>
<td>8</td>
<td>Sunny</td>
<td>85</td>
<td>85</td>
<td>Strong</td>
<td>No</td>
</tr>
<tr>
<td>9</td>
<td>Sunny</td>
<td>85</td>
<td>85</td>
<td>Strong</td>
<td>No</td>
</tr>
<tr>
<td>10</td>
<td>Rain</td>
<td>70</td>
<td>70</td>
<td>Light</td>
<td>Yes</td>
</tr>
<tr>
<td>11</td>
<td>Rain</td>
<td>70</td>
<td>70</td>
<td>Light</td>
<td>Yes</td>
</tr>
<tr>
<td>12</td>
<td>Overcast</td>
<td>80</td>
<td>80</td>
<td>Strong</td>
<td>No</td>
</tr>
<tr>
<td>13</td>
<td>Overcast</td>
<td>80</td>
<td>80</td>
<td>Strong</td>
<td>No</td>
</tr>
<tr>
<td>14</td>
<td>Rain</td>
<td>80</td>
<td>80</td>
<td>Strong</td>
<td>No</td>
</tr>
</tbody>
</table>
Data Conversion

- Some tools can deal with nominal values but other need fields to be numeric.

- Convert ordinal fields to numeric to be able to use "\(>\)" and "\(<\)" comparisons on such fields.
  - \(A \rightarrow 4.0\)
  - \(A- \rightarrow 3.7\)
  - \(B+ \rightarrow 3.3\)
  - \(B \rightarrow 3.0\)

- Multi-valued, unordered attributes with small no. of values
  - e.g. Color=Red, Orange, Yellow, …, Violet
  - for each value \(v\) create a binary "flag" variable \(C_v\), which is 1 if Color=\(v\), 0 otherwise

Conversion: Nominal, Many Values

- Examples:
  - US State Code (50 values)
  - Profession Code (7,000 values, but only few frequent)

- Ignore ID-like fields whose values are unique for each record

- For other fields, group values "naturally":
  - e.g. 50 US States → 3 or 5 regions
  - Profession - select most frequent ones, group the rest

- Create binary flag-fields for selected values
**Discretization**

- Divide the range of a continuous attribute into intervals
  - Some methods require discrete values, e.g., most versions of Naïve Bayes, CHAID
  - Reduce data size by discretization
  - Prepare for further analysis

- Discretization is very useful for generating a summary of data
- Also called “binning”

**Equal-width Binning**

- It divides the range into \( N \) intervals of equal size (range): uniform grid
- If \( A \) and \( B \) are the lowest and highest values of the attribute, the width of intervals will be: \( W = \frac{B - A}{N} \)

**Temperature values:**

- [64, 67)  
- [67, 70)  
- [70, 73)  
- [73, 76)  
- [76, 79)  
- [79, 82)  
- [82, 85]

**Count:**

- 2  
- 2  
- 4  
- 2  
- 0  
- 2  
- 2

**Salary in a corporation**

- [0 - 200,000)  
- [200,000 - 1,800,000]  
- [1,800,000 - 2,000,000]

**Disadvantage**

(a) Unsupervised  
(b) Where does \( N \) come from?  
(c) Sensitive to outliers

**Advantage**

(a) Simple and easy to implement  
(b) Produce a reasonable abstraction of data

**Equal-depth (or height) Binning**

- It divides the range into \( N \) intervals, each containing approximately the same number of samples
  - Generally preferred because avoids clumping
  - In practice, “almost-equal” height binning is used to give more intuitive breakpoints

**Additional considerations:**

- Don’t split frequent values across bins  
- Create separate bins for special values (e.g., 0)  
- Readable breakpoints (e.g., round breakpoints)
### Equal-depth (or height) Binning

Temperature values:
64 65 68 69 70 71 72 75 80 81 83 85

Count

Equal Height = 4, except for the last bin

### Discretization considerations

- **Class-independent methods**
  - Equal Width is simpler, good for many classes
  - can fail miserably for unequal distributions
  - Equal Height gives better results
- **Class-dependent methods** can be better for classification
  - Decision tree methods build discretization on the fly
  - Naive Bayes requires initial discretization
- Many other methods exist...

### Method 1R

- Developed by Holte (1993).
- It is a supervised discretization method using binning.
- After sorting the data, the range of continuous values is divided into a number of disjoint intervals and the boundaries of those intervals are adjusted based on the class labels associated with the values of the feature.
- Each interval should contain a given minimum of instances (6 by default) with the exception of the last one.
- The adjustment of the boundary continues until the next values belongs to a class different to the majority class in the adjacent interval.

### 1R Example

Interval contains at least 6 elements
Adjustment of the boundary continues until the next values belongs to a class different to the majority class in the adjacent interval.
Exercise

- Discretize the following values using EW and ED binning
- 13, 15, 16, 19, 20, 21, 22, 22, 25, 30, 33, 35, 35, 36, 40, 45

Entropy Based Discretization

Class dependent (classification)

1. Sort examples in increasing order
2. Each value forms an interval (‘m’ intervals)
3. Calculate the entropy measure of this discretization
4. Find the binary split boundary that minimizes the entropy function over all possible boundaries. The split is selected as a binary discretization.
5. Apply the process recursively until some stopping criterion is met, e.g.,
   \[ \text{Ent}(S) - E(T, S) > \delta \]

Entropy

\[
\text{Ent} = - \sum_{c=1}^{N} p_c \cdot \log_2 p_c
\]

Entropy/Impurity

- \( S \) - training set, \( C_1, \ldots, C_N \) classes
- Entropy \( E(S) \) - measure of the impurity in a group of examples

\[
\text{Impurity}(S) = - \sum_{c=1}^{N} p_c \cdot \log_2 p_c
\]
Impurity

Very impure group

Less impure

Minimum impurity

An example of entropy disc.

Test split temp < 71.5

<table>
<thead>
<tr>
<th>Temp</th>
<th>Play?</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt; 71.5</td>
<td>yes</td>
</tr>
<tr>
<td>&gt; 71.5</td>
<td>no</td>
</tr>
</tbody>
</table>

\[
\text{Ent(split 71.5)} = \frac{6}{14} \left( \frac{4}{6} \log_2 \frac{4}{6} + \frac{2}{6} \log_2 \frac{2}{6} \right) + \frac{8}{14} \left( \frac{5}{8} \log_2 \frac{5}{8} + \frac{3}{8} \log_2 \frac{3}{8} \right) = 0.939
\]

An example (cont.)

The method tests all split possibilities and chooses the split with smallest entropy.

In the first iteration a split at 84 is chosen.

The two resulting branches are processed recursively.

The stopping criterion

Previous slide did not take into account the stopping criterion.

\[
\text{Ent}(S) - E(T, S) > \delta
\]

\[
\delta > \frac{\log(N - 1)}{N} + \frac{\Delta(T, S)}{N}
\]

\[
\Delta(S, T) = \log_2(3^c - 2) - [c \text{Ent}(S) - c_1 \text{Ent}(S_1) - c_2 \text{Ent}(S_2)]
\]

c is the number of classes in S

c_1 is the number of classes in S_1

c_2 is the number of classes in S_2.

This is called the Minimum Description Length Principle (MDLP)
Exercise

• Compute the gain of splitting this data in half

<table>
<thead>
<tr>
<th>Humidity</th>
<th>Play</th>
</tr>
</thead>
<tbody>
<tr>
<td>65</td>
<td>Yes</td>
</tr>
<tr>
<td>70</td>
<td>No</td>
</tr>
<tr>
<td>70</td>
<td>Yes</td>
</tr>
<tr>
<td>70</td>
<td>Yes</td>
</tr>
<tr>
<td>75</td>
<td>Yes</td>
</tr>
<tr>
<td>80</td>
<td>Yes</td>
</tr>
<tr>
<td>80</td>
<td>Yes</td>
</tr>
<tr>
<td>85</td>
<td>No</td>
</tr>
<tr>
<td>85</td>
<td>Yes</td>
</tr>
<tr>
<td>86</td>
<td>Yes</td>
</tr>
<tr>
<td>90</td>
<td>No</td>
</tr>
<tr>
<td>90</td>
<td>Yes</td>
</tr>
<tr>
<td>91</td>
<td>No</td>
</tr>
<tr>
<td>95</td>
<td>No</td>
</tr>
<tr>
<td>96</td>
<td>Yes</td>
</tr>
</tbody>
</table>

more R examples

• ?log – help on a function
• help.search("clustering")
• objects() – lists existing objects
• rm(obj1, obj2,...) – removes existing objects
• str(obj) – displays the internal structure of an object
• Menu “File; Change dir…”
• dir()

• v <- c(1,2,3,4,5) – defines a vector
• m <- matrix(c(1,2,3,4),2,2) - defines 2x2 matrix de 2x2
• a <- array(1:8, c(2,2,2)) – defines 2x2x2 array
• m*2
• m[1,1]
• m[1,]
The california housing dataset in R

- File/change dir - to the directory with the dataset
- `cal_housing <- read.table("aula_02_dataset_california.txt")`
- `cal_housing[1:10,]` - first 10 rows
- `summary(cal_housing["aula_02_dataset_california.txt", header = TRUE])` - with headers
- `hist(cal_housing$TotalRooms)` - histogram
- `hist(cal_housing[,4:4])` - for pairs of variables
- `pairs(cal_housing[,3:8])` - scatters for pairs of variables
- `cor(cal_housing[,3:8])` - correlation matrix
- `boxplot(cal_housing[,3:8])` - boxplots

Discritization with R

- Load Dataset
  - `data <- read.table("aula_02_1R_exemplo.txt")`
- Load Data Preparation Package
  - `library(dprep)`
- Equal Width
  - `disc_data_ew <- disc.ew(data,1:1)`
- Equal Depth
  - `disc_data_ef <- disc.ef(data,1:1,3)`
- Holte 1R
  - `disc_data_1r <- disc.1r(data,1:1,6)`
- Entropy
  - `disc_data_ent <- disc.mentr(data,1:2)`

Outliers

- Outliers are values thought to be out of range.
- “An outlier is an observation that deviates so much from other observations as to arouse suspicion that it was generated by a different mechanism”
- Can be detected by standardizing observations and labeling the standardized values outside a predetermined bound as outliers
- Outlier detection can be used for fraud detection or data cleaning

Approaches:
- do nothing
- enforce upper and lower bounds
- let binning handle the problem
Outlier detection

- **Univariate**
  - Compute mean and std. deviation. For \( k=2 \) or \( 3 \), \( x \) is an outlier if outside limits (normal distribution assumed)
    
    \[
    (\bar{x} - ks, \bar{x} + ks)
    \]
  - Boxplot: An observation is an extreme outlier if
    
    \((Q_1-3\times IQR, Q_3+3\times IQR)\), where \( IQR = Q_3 - Q_1 \)
    
    \[(IQR = \text{Inter Quartile Range})\]
    
    and declared a mild outlier if it lies outside of the interval
    
    \((Q_1-1.5\times IQR, Q_3+1.5\times IQR)\).

- **Multivariate**
  - Clustering
    - Very small clusters are outliers
  - Distance based
    - An instance with very few neighbors within \( \lambda \) is regarded as an outlier
A bi-dimensional outlier that is not an outlier in either of its projections.
Data Transformation

- **Smoothing**: remove noise from data (binning, regression, clustering)
- **Aggregation**: summarization, data cube construction
- **Generalization**: concept hierarchy climbing
- **Attribute/feature construction**
  - New attributes constructed from the given ones (add att. area which is based on height and width)
- **Normalization**
  - Scale values to fall within a smaller, specified range

Data Cube Aggregation

- Data can be aggregated so that the resulting data summarize, for example, sales per year instead of sales per quarter.

<table>
<thead>
<tr>
<th>Year</th>
<th>Quarter</th>
<th>Sales</th>
</tr>
</thead>
<tbody>
<tr>
<td>1997</td>
<td>Q1</td>
<td>$214,000</td>
</tr>
<tr>
<td>1998</td>
<td>Q2</td>
<td>$408,000</td>
</tr>
<tr>
<td>1999</td>
<td>Q3</td>
<td>$320,000</td>
</tr>
<tr>
<td>1999</td>
<td>Q4</td>
<td>$585,000</td>
</tr>
</tbody>
</table>

- Reduced representation which contains all the relevant information if we are concerned with the analysis of yearly sales

Concept Hierarchies

- Country → State → County → City

- Jobs, food classification, time measures...

Normalization

- For distance-based methods, normalization helps to prevent that attributes with large ranges out-weight attributes with small ranges
  - min-max normalization
  - z-score normalization
  - normalization by decimal scaling
Normalization

• min-max normalization

\[ v' = \frac{v - \text{min}}{\text{max} - \text{min}} \times (\text{new_max} - \text{new_min}) + \text{new_min} \]

In R
\[
\text{mmnorm(data, minval=0, maxval=1)}
\]

• z-score normalization

\[ v' = \frac{v - \mu}{\sigma} \]

In R
\[
\text{boxplot(znorm(cal_housing[3:8]))}
\]

• normalization by decimal scaling

\[ v' = \frac{v}{10^j} \]

Where \( j \) is the smallest integer such that \( \max(|v'|) \leq 1 \)

range: -986 to 917 \( \Rightarrow j=3 \)

-986 -> -0.986  
917 -> 0.917

MISSING DATA

Missing Data

• Data is not always available
  • E.g., many tuples have no recorded value for several attributes, such as customer income in sales data

• Missing data may be due to
  • equipment malfunction
  • inconsistent with other recorded data and thus deleted
  • data not entered due to misunderstanding
  • certain data may not be considered important at the time of entry
  • not register history or changes of the data

• Missing data may need to be inferred.

• Missing values may carry some information content: e.g. a credit application may carry information by noting which field the applicant did not complete

Missing Values

• There are always MVs in a real dataset

• MVs may have an impact on modelling, in fact, they can destroy it!

• Some tools ignore missing values, others use some metric to fill in replacements
  • The modeller should avoid default automated replacement techniques
    • Difficult to know limitations, problems and introduced bias

• Replacing missing values without elsewhere capturing that information removes information from the dataset
How to Handle Missing Data?

- Ignore records (use only cases with all values)
  - Usually done when class label is missing as most prediction methods do not handle missing data well
  - Not effective when the percentage of missing values per attribute varies considerably as it can lead to insufficient and/or biased sample sizes
- Ignore attributes with missing values
  - Use only features (attributes) with all values (may leave out important features)
- Fill in the missing value manually
  - Tedious + infeasible?
- Use a global constant to fill in the missing value
  - E.g., "unknown". (May create a new class!)
- Use the attribute mean to fill in the missing value
  - It will do the least harm to the mean of existing data
  - If the mean is to be unbiased?
  - What if the standard deviation is to be unbiased?
  - Use the attribute mean for all samples belonging to the same class to fill in the missing value
- Use the most probable value to fill in the missing value
  - Inference-based such as Bayesian formula or decision tree
  - Identify relationships among variables
    - Linear regression, Multiple linear regression, Nonlinear regression
  - Nearest-Neighbour estimator
    - Finding the k neighbours nearest to the point and fill in the most frequent value or the average value
    - Finding neighbours in a large dataset may be slow

Nearest-Neighbour
How to Handle Missing Data?

• Note that, it is as important to avoid adding bias and distortion to the data as it is to make the information available.
  • bias is added when a wrong value is filled-in

• No matter what techniques you use to conquer the problem, it comes at a price. The more guessing you have to do, the further away from the real data the database becomes. Thus, in turn, it can affect the accuracy and validation of the mining results.

Missing Data with R

• library(dprep)
  • data(hepatitis) - loads dataset
  • str(hepatitis) - gives dataset structure
  • summary(hepatitis)
  • short_hep <- hepatitis[1:15,]
  • ?ce.impute - gives information about the fill missing values method
  • res <- ce.impute(short_hep,"median",19)
  • ?clean()ce.impute(hepatitis,"median",1:19)
  • ce.impute(hepatitis,"knn",k1=10)
  • clean() – eliminates rows and columns that have more than the set limit missings
  • clean(res,0.3,0.2)
  • imagmiss(hepatitis) – gives the percentage of missing values

Data Integration

• Turn a collection of pieces of information into an integrated and consistent whole

• Detecting and resolving data value conflicts
  • For the same real world entity, attribute values from different sources may be different
    • Which source is more reliable?
    • Is it possible to induce the correct value?

• Possible reasons: different representations, different scales, e.g., metric vs. British units

Data integration requires knowledge of the "business"
Types of Inter-schema Conflicts

- Classification conflicts
  - Corresponding types describe different sets of real world elements.
    - DB1: authors of journal and conference papers;
    - DB2: authors of conference papers only.
  - Generalization / specialization hierarchy

- Descriptive conflicts
  - Naming conflicts: synonyms, homonyms
  - Cardinalities: first name - one, two, N values
  - Domains: salary: $, Euro ...; student grade: [0:20], [1:5]
  - Solution depends upon the type of the descriptive conflict

Data type inconsistency example

- 1999 Sep 23
  The $125 million Mars Climate Orbiter was presumed lost after it hit the Martian atmosphere. The crash was later blamed on navigation confusion due to 2 teams using conflicting English and metric units.

- http://en.wikipedia.org/wiki/Mars_Climate_Orbiter

Types of Inter-schema Conflicts

- Structural conflicts
  - DB1: Book is a class; DB2: books is an attribute of Author
    - Choose the less constrained structure (Book is a class)

- Fragmentation conflicts
  - DB1: Class Road_segment; DB2: Classes Way_segment, Separator
    - Aggregation relationship

Handling Redundancy in Data Integration

- Redundant data occur often when integrating databases
  - The same attribute may have different names in different databases
  - False predictors are fields correlated to target behavior, which describe events that happen at the same time or after the target behavior
    - Example: Service cancellation date is a leaker when predicting attriters
  - One attribute may be a "derived" attribute in another table, e.g., annual revenue
  - For numerical attributes, redundancy may be detected by correlation analysis

\[
\rho = \frac{1}{N-1} \sum_{i=1}^{N} (x_i - \bar{x})(y_i - \bar{y}) \quad \{1 \leq i \leq N\}
\]
Scatter Matrix

(Almost) Automated False Predictor Detection

- For each field
  - Build 1-field decision trees for each field
  - (or compute correlation with the target field)
- Rank all suspects by 1-field prediction accuracy (or correlation)
- Remove suspects whose accuracy is close to 100% (Note: the threshold is domain dependent)
- Verify top "suspects" with domain expert

Data Reduction

- Selecting Most Relevant Attributes
  - If there are too many attributes, select a subset that is most relevant (according to your knowledge of the business).
  - Select top $N$ fields using 1-field predictive accuracy as computed for detecting false predictors.
- Attribute Numerosity Reduction
  - Parametric methods
    - Assume the data fits some model, estimate model parameters, store only the parameters, and discard the data (except possible outliers), Regression
  - Non-parametric methods
    - Do not assume models
    - Major families: histograms, clustering, sampling
Clustering

- Partition a data set into clusters makes it possible to store cluster representation only
- Can be very effective if data is clustered but not if data is "smeared"
- There are many choices of clustering definitions and clustering algorithms, further detailed in next lessons

Increasing Dimensionality

- In some circumstances the dimensionality of a variable need to be increased:
  - Color from a category list to the RGB values
  - ZIP codes from category list to latitude and longitude

Histograms

- A popular data reduction technique
- Divide data into buckets and store average (sum) for each bucket
- Can be constructed optimally in one dimension using dynamic programming:
  - Optimal histogram has minimum variance. Hist. variance is a weighted sum of the variance of the source values in each bucket.

Sampling

- The cost of sampling is proportional to the sample size and not to the original dataset size, therefore, a mining algorithm's complexity is potentially sub-linear to the size of the data
- Choose a representative subset of the data
  - Simple random sampling (SRS) (with or without reposition)
  - Stratified sampling:
    - Approximate the percentage of each class (or subpopulation of interest) in the overall database
    - Used in conjunction with skewed data
Unbalanced Target Distribution

- Sometimes, classes have very unequal frequency
  - Attrition prediction: 97% stay, 3% attrite (in a month)
  - Medical diagnosis: 90% healthy, 10% disease
  - eCommerce: 99% don't buy, 1% buy
  - Security: >99.99% of Americans are not terrorists
- Similar situation with multiple classes
  - Majority class classifier can be 97% correct, but useless

Handling Unbalanced Data

- With two classes: let positive targets be a minority
  - Separate raw held-aside set (e.g. 30% of data) and raw train
    - Put aside raw held-aside and don't use it till the final model
  - Select remaining positive targets (e.g. 70% of all targets) from raw train
    - Join with equal number of negative targets from raw train, and randomly sort it.
- Separate randomized balanced set into balanced train and balanced test

Building Balanced Train Sets

- Same % of Y and N
- SRS
- 70/30
- Raw set for estimating accuracy of final model

Summary

- Every real world data set needs some kind of data pre-processing
  - Deal with missing values
  - Correct erroneous values
  - Select relevant attributes
  - Adapt data set format to the software tool to be used
- In general, data pre-processing consumes more than 60% of a data mining project effort
References

- 'Data preparation for data mining', Dorian Pyle, 1999
- 'Data Mining: Concepts and Techniques', Jiawei Han and Micheline Kamber, 2000
- 'Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations', Ian H. Witten and Eibe Frank, 1999
- ESMA 6835 Minería de Datos (http://math.uprm.edu/~edgar/dm8.ppt)