Friedrich-Alexander-Universität
Technische Fakultät

4. Data Preprocessing

Knowledge Discovery in Databases

Dominik Probst, Dominik.probst@fau.de
Chair of Computer Science 6 (Data Management), Friedrich-Alexander-University Erlangen-Nürnberg
Summer semester 2024

Outline

1. Overview
2. Data cleaning
3. Data integration
4. Data reduction
5. Data transformation and data discretization
6. Summary

Overview

Data Quality: Why Preprocess the Data?

- **Measures for data quality: A multidimensional view:**
  - **Accuracy:** correct or wrong, accurate or not.
  - **Completeness:** not recorded, unavailable.
  - **Consistency:** some modified but some not, dangling refs, etc.
  - **Timeliness:** timely updated?
  - **Believability:** how trustworthy is it, that the data is correct?
  - **Interpretability:** how easily can the data be understood?
  - And even many more!

Major Tasks in Data Preprocessing (I)

- **Data cleaning:**
  - Fill in missing values.
  - Smooth noisy data.
  - Identify or remove outliers.
  - Resolve inconsistencies.
- **Data integration:**
  - Integration of multiple databases.
  - Data cubes or files.

Major Tasks in Data Preprocessing (II)

- **Data reduction:**
  - Dimensionality reduction.
  - Numerosity reduction.
  - Data compression.
- **Data transformation and data discretization:**
  - Normalization.
  - Concept-hierarchy generation.

Data cleaning

Data Cleaning

Data in the real world is dirty. Lots of potentially incorrect data:

- E.g. instrument faulty, human or computer error, transmission error.
- **Incomplete:** lacking attributes, lacking certain attributes of interest or containing aggregate data.
  - E.g. occupation = "" (missing data).
- **Noisy:** containing noise.
  - E.g. small measurement inaccuracies with a sensor (noise)
- **Errors/Outliers:** containing errors or outliers.
  - E.g. scores = "2,3,0,6,1,9,95" (outlier = "95")
  - E.g. salary = "-10" (error)
- **Inconsistencies:** containing discrepancies in codes or names.
  - E.g. age = "42", birthday = "03/07/2010".
  - E.g. old rating = "1,2,3", new rating = "A,B,C".
  - E.g. discrepancy between duplicate records (e.g. address).
- **Intentional** (only default value, e.g. disguised missing data):
  - E.g. "Doe" as everyone's surname

Incomplete (Missing) Data

- **Data is not always available.**
  - E.g. many tuples have no recorded value for several attributes.
  - Examples are customer income in sales data.
- **Missing data may be due to:**
  - Equipment malfunction.
  - Inconsistency 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 registered history or changes of the data.
- **Missing data may need to be inferred.**

How to Handle Missing Data?

- **Ignore the tuple:**
  - Usually done when class label is missing (when doing classification).
  - Not effective when the percentage of missing values per attribute varies considerably.
- **Fill in the missing value manually.**
  - Tedious or infeasible.
- **Fill in automatically with:**
  - A global constant, e.g. "unknown", maybe a new class.
  - The attribute mean.
  - The attribute mean for all samples belonging to the same class.
  - **The most probable value:** Inference-based such as Bayesian formula or decision tree.

Noisy Data

- **Noise:**
  - Random error or variance in a measured variable.
  - Stored value a little bit off the real value, up or down.
  - Leads to (slightly) incorrect attribute values.
- **May be due to:**
  - Faulty or imprecise data-collection instruments.
  - Data-entry problems.
  - Data-transmission problems.
  - Technology limitation.
  - Inconsistency in naming conventions.

How to Handle Noisy Data?

- **Binning:**
  - First sort data and partition into (equal-frequency) bins.
  - Then smooth by bin mean, by bin median or by bin boundaries.
- **Regression:**
  - Smooth by fitting the data to regression functions.
- **Clustering:**
  - Detect and remove outliers.
- **Combined computer and human inspection:**
  - Detect suspicious values and check by human.
  - E.g. deal with possible outliers.

Data Cleaning as a Process (I)

- **Data discrepancy detection:**
  - Use **metadata** (e.g. domain, range, dependency, distribution).
  - Check field overloading.
  - Check uniqueness rule, consecutive rule and null rule.
  - Use commercial tools:
    - **Data scrubbing:** use simple domain knowledge (e.g. postal code, spell-check) to detect errors and make corrections.
    - **Data auditing:** by analyzing data to discover rules and relationships to detect violators (e.g. correlation and clustering to find outliers).

Data Cleaning as a Process (II)

- **Data migration and integration:**
  - Data-migration tools: allow transformations to be specified.
  - ETL (Extraction/Transformation/Loading) tools: allow users to specify transformations through a graphical user interface.
- **Integration of the two processes.**
  - Iterative and interactive (e.g. the Potter's Wheel tool).

Data integration

Data Integration

- **Data integration:**
  - Combine data from multiple sources into a coherent store.
- **Schema integration:**
  - E.g. `A.cust-id` $\equiv$ `B.cust-#`.
  - Integrate metadata from different sources.
- **Entity-identification problem:**
  - Identify the same real-world entities from multiple data sources.
  - E.g. Bill Clinton = William Clinton.
- **Detecting and resolving data-value conflicts:**
  - For the same real world entity, attribute values from different sources are different.
  - Possible reasons:
    - Different representations (coding).
    - Different scales, e.g. metric vs. British units.

Handling Redundancy in Data Integration

- **Redundant data often occur when integrating multiple databases.**
  - **Object (entity) identification:**
    The same attribute or object may have different names in different databases.
  - **Derivable data:**
    One attribute may be a "derived" attribute in another table. E.g. annual revenue.
- **Redundant attributes:**
  - Can be detected by **correlation analysis** and **covariance analysis**.
- **Careful integration of the data from multiple sources:**
  - Helps to reduce/avoid redundancies and inconsistencies and improve mining speed and quality.

Correlation Analysis for Nominal Data (I)

- **Example:**
  We want to determine if the interests "Reads Books" and "Plays Chess" in the following table correlate with each other:

ID	Reads Books	Plays Chess
1	Y	Y
2	Y	Y
3	Y	N
...	...	...
1499	N	Y
1500	N	N

Correlation Analysis for Nominal Data (II)

- **General starting point:**
  - **The attributes A and B to be analyzed:**
    - A has n distinct values:
      $A := \{a_1, a_2, \ldots, a_n\}$, where $n \in \mathbb{N}_{>1}$.
    - B has m distinct values:
      $B := \{b_1, b_2, \ldots, b_m\}$, where $m \in \mathbb{N}_{>1}$.
  - **The set X of all distinct combinations:**
    - X is defined as follows:
      $X := \{(a, b) \mid a \in A \text{ and } b \in B\}$.
  - **The multi set Y of all tuples:**
    - The multiset Y over the set X is a mapping of X to the set of natural numbers $\mathbb{N}_0$. The number $Y(x), x \in X$ tells how often x is contained in the multiset Y.

- **Starting point in the example:**
  - **The attributes A and B to be analyzed:**
    - A ("Reads Books") has 2 distinct values:
      $A := \{Y, N\}$
    - B ("Plays Chess") has 2 distinct values:
      $B := \{Y, N\}$
  - **The set X of all distinct combinations:**
    - X contains 4 distinct combinations:
      $X := \{(Y, Y), (Y, N), (N, Y), (N, N)\}$.
  - **The multi set Y of all tuples:**
    - Y contains 1500 tuples:
      $Y := \{(Y, Y), (Y, Y), \ldots, (N, N)\}$.

Correlation Analysis for Nominal Data (III)

- **Actual quantity in Y:**

$$c_{ij} = \#\{(a,b) \in Y \mid a = a_i, b = b_j\} = Y((a_i, b_j))$$

- **Expected quantity (value of c_{ij}) in case of independence, i. e. no correlation:**

$$e_{ij} = \frac{\sum_{k=1}^{m} c_{ik}}{\#Y} \cdot \frac{\sum_{l=1}^{n} c_{lj}}{\#Y} \cdot \#Y = \frac{\sum_{k=1}^{m} c_{ik} \cdot \sum_{l=1}^{n} c_{lj}}{\#Y}$$

Please note that:

- The sum of all c_{ij} over an attribute a_i (or b_j) is identical to the sum of all e_{ij} over a_i (or b_j):

$$\sum_{k=1}^{m} e_{ik} = \sum_{k=1}^{m} c_{ik} \text{ and } \sum_{l=1}^{n} e_{lj} = \sum_{l=1}^{n} c_{lj}$$

Correlation Analysis for Nominal Data (IV)

- **The values c_{ij} and e_{ij} are often presented in a contingency table:**

	a_1	$\dots$	a_n	
b_1	$c_{11}(e_{11})$	$\dots$	$c_{n1}(e_{n1})$	$\sum_{i=1}^{n} e_{i1}$
$\dots$	$\dots$	$\dots$	$\dots$	$\dots$
b_m	$c_{1m}(e_{1m})$	$\dots$	$c_{nm}(e_{nm})$	$\sum_{i=1}^{n} e_{im}$
	$\sum_{j=1}^{m} e_{1j}$	$\dots$	$\sum_{j=1}^{m} e_{nj}$	$\sum_{i=1}^{n} \sum_{j=1}^{m} e_{ij}$

- **In our example it would look like this:**

	Plays chess	Doesn't play chess	Sum (row)
Reads books	250(90)	200(360)	450
Doesn't read books	50(210)	1000(840)	1050
Sum (column)	300	1200	1500

Correlation Analysis for Nominal Data (V)

- **To determine the correlation the χ^2-test (Chi-squared test) is applied:**

$$\chi^2 = \sum_{i=1}^{n} \sum_{j=1}^{m} \frac{(c_{ij} - e_{ij})^2}{e_{ij}}.$$

Properties of the χ^2-test

- No correlation (i.e. independence of attributes) yields χ^2 value of zero.
- The larger the χ^2 value, the more likely the variables are related.
- The cells that contribute the most to the χ^2 value are those whose actual count is very different from the expected count e_{ij}.

Correlation Analysis for Nominal Data (VI)

- **Calculation of χ^2 in our example:**

$$\chi^2 = \frac{(250-90)^2}{90} + \frac{(50-210)^2}{210} + \frac{(200-360)^2}{360} + \frac{(1000-840)^2}{840} = 507.93.$$

- **It shows that "Reads Books" and "Plays Chess" are correlated (in our example)**

Important: Correlation does not imply causality!

- E.g. $\#$ of hospitals and $\#$ of car-thefts in a city are correlated.
- Both are causally linked to the third variable: population.

Correlation Analysis of Numerical Data (I)

- **Numerical correlation can be determined with Pearson's product-moment coefficient:**

$$\mathrm{Cor}(A, B) = \frac{\sum_{i=1}^{n}(a_i - \mu_A)(b_i - \mu_B)}{n \cdot \sigma_A \sigma_B} = \frac{\sum_{i=1}^{n} a_i b_i - n \cdot \mu_A \mu_B}{n \cdot \sigma_A \sigma_B}.$$

where n is the number of tuples, a_i and b_i are the respective values of A and B in tuple i, μ_A and μ_B are the respective mean values of A and B, σ_A and σ_BB are the respective standard deviations of A and B

Properties of Pearson's product-moment coefficient

- If $\mathrm{Cor}(A, B) > 0$: A and B are positively correlated.
- If $\mathrm{Cor}(A, B) = 0$: A and B are independent.
- If $\mathrm{Cor}(A, B) < 0$: A and B are negatively correlated.

Correlation Analysis of Numerical Data (II)

- **It is also possible to visually detect numerical correlation:**

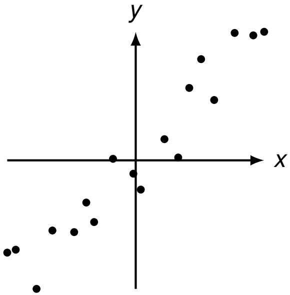


Figure: a) Positive correlation.

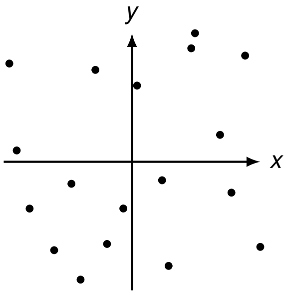


Figure: b) Uncorrelated/no correlation.

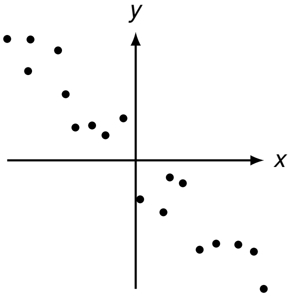


Figure: c) Negative correlation.

Covariance of Numerical Data (I)

- **Covariance is similar to correlation:**

$$\mathrm{Cov}(A, B) = \frac{\sum_{i=1}^{n}(a_i - \mu_A)(b_i - \mu_B)}{n} = \frac{\sum_{i=1}^{n} a_i b_i}{n} - \mu_A \mu_B$$

- **It is possible to compute the correlation based on the covariance:**

$$\mathrm{Cor}(A, B) = \frac{\mathrm{Cov}(A, B)}{\sigma_A \sigma_B}$$

Properties of the covariance

- If $\mathrm{Cov}(A, B) > 0$: A and B tend to be either both larger or both smaller than their expected values.
- If $\mathrm{Cov}(A, B) < 0$: If A is larger than its expected value, B is likely to be smaller than its expected value and vice versa.

Covariance of Numerical Data (II)

- **Example:**
  - We examine a table containing the history of two stock prices:

Date	Stock 1	Stock 2
21.06	2	5
22.06	3	8
23.06	5	10
24.06	4	11
25.06	6	14

  - If the stocks are affected by the same industry trends, will their prices rise or fall together?

$$\mathrm{Cov}(A, B) = \frac{2 \cdot 5 + 3 \cdot 8 + 5 \cdot 10 + 4 \cdot 11 + 6 \cdot 14}{5} - 4 \cdot 9.6 = 4.$$

  - Thus, A and B rise together since $\mathrm{Cov}(A, B) > 0$.

Data reduction

Data Reduction (I)

- **What is data reduction?**
  - Obtain a reduced representation of the data set that is much smaller in volume but yet produces the same (or almost the same) results.
- **Why data reduction?**
  - A database/data warehouse may store terabytes of data.
  - Complex data analysis may take a very long time to run on the complete data set.
- **Data reduction strategies:**
  - Dimensionality reduction, i.e. remove unimportant attributes.
    - Wavelet transforms.
    - Principal component analysis.
    - Attribute subset selection or attribute creation.

Data Reduction (II)

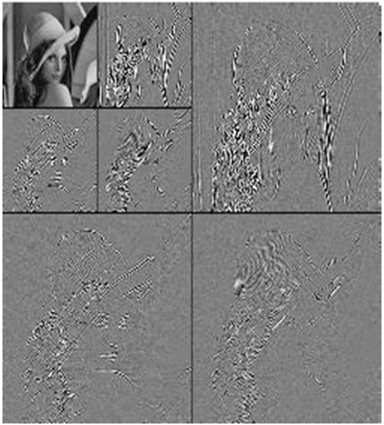
- **Data reduction strategies (continued):**
  - Numerosity reduction:
    - Regression and log-linear models.
    - Histograms, clustering and sampling.
    - Data cube aggregation.
  - Data compression.

Data Reduction (I): Dimensionality Reduction

- **Curse of dimensionality:**
  - When dimensionality increases data becomes increasingly sparse.
  - Density and distance between points become less meaningful.
  - The possible combinations of subspaces will grow exponentially.
- **Dimensionality reduction:**
  - Avoid the curse of dimensionality.
  - Help eliminate irrelevant features and reduce noise.
  - Reduce time and space required in data mining.
  - Allow easier visualization.
- **Dimensionality-reduction techniques:**
  - Wavelet transforms.
  - Principal component analysis.
  - Supervised and nonlinear techniques (e.g. feature selection).

Wavelet Transform (I)

- **Decomposes a signal into different frequency subbands.**
  Applicable to *n*-dimensional signals.
- Data transformed to preserve relative distance between objects at different levels of resolution.
- Allow natural clusters to become more distinguishable.
- Used for image compression.



Wavelet Transform (II)

- **Discrete wavelet transform:**
  Transforms a vector X into a different vector X' of wavelet coefficients with the same length.
- **Compressed approximation:**
  Store only a small fraction of the strongest of the wavelet coefficients.
- **Similar to discrete fourier transform, but better lossy compression, localized in space.**
- **Method:**
  - The length of the vector must be an integer power of 2 (padding with 0's if necessary).
  - Each transform has two functions: smoothing and difference.
  - Applied to pairs of data, resulting in two sets of data with half the length.
  - The two functions are applied recursively until reaching the desired length.

Example: Wavelet Transform (I)

- **Initial vector:**
  - $X = (2, 2, 0, 2, 3, 5, 4, 4)$
- **First step:**
  - $(2, 2) \rightarrow$ Average: 2, Weighted difference: 0
  - $(0, 2) \rightarrow$ Average: 1, Weighted difference: -1
  - $(3, 5) \rightarrow$ Average: 4, Weighted difference: -1
  - $(4, 4) \rightarrow$ Average: 4, Weighted difference: 0
  - $A_1 = (2, 1, 4, 4)$, $D_1 = (0, -1, -1, 0)$
- **Second step:**
  - $(2, 1) \rightarrow$ Average: 1.5, Weighted difference: 0.5
  - $(4, 4) \rightarrow$ Average: 4, Weighted difference: 0
  - $A_2 = (1.5, 4)$, $D_2 = (0.5, 0)$

Example: Wavelet Transform (II)

- **Third step:**
  - $(1.5, 4) \rightarrow$ Average: 2.75, Weighted difference: -1.25
  - $A_3 = (2.75)$, $D_3 = (-1.25)$
- **Resulting vector:**
  - $X' = (2.75, -1.25, 0.5, 0, 0, -1, -1, 0)$
- **Possible compression:**
  - Small detail coefficients ($D_{1,2,3}$) can be replaced by 0's, while retaining significant coefficients.

Resolution	Averages	Detail coefficients
8	$(2, 2, 0, 2, 3, 5, 4, 4)$	-
4	$(2, 1, 4, 4)$	$(0, -1, -1, 0)$
2	$(1.5, 4)$	$(0.5, 0)$
1	(2.75)	(-1.25)

Why Wavelet Transform?

- **Hat-shaped filters:**
  - Emphasize region where points cluster.
  - Suppress weaker information in their boundaries.
- **Effective removal of outliers:**
  - Insensitive to noise, insensitive to input order.
- **Multi-resolution:**
  - Detect arbitrary shaped clusters at different scales.
- **Efficient:** Complexity $\mathcal{O}(N)$.

Principal Component Analysis (PCA)

- **Main idea:**
  - Given a data set with n dimensions.
  - Find $k \leq n$ orthogonal vectors that capture the largest amount of data.
  - Works only for numeric data.
- **Example data set:**
  - Used on the next few slides to explain the steps of a PCA:

d_1	d_2	d_3
23	6	1
9	9	5
17	5	1
3	6	1

Principal Component Analysis - 1. Step: Standardization (I)

- **Procedure:**
  - Each value x within a dimension d_n is standardized with the help of the mean (μ_{d_n}) and standard deviation (σ_{d_n}) of d_n:

$$x' = \frac{x - \mu_{d_n}}{\sigma_{d_n}}$$

- **Reason:**
  - Each dimension should be considered equally in the analysis.
  - Dimensions with a wider range of values would dominate without this step.

Principal Component Analysis - 1. Step: Standardization (II)

- **Example:**
  - Mean and standard deviation per dimension:

	d_1	d_2	d_3
μ	13.000000	6.500000	2.0
σ	8.793937	1.732051	2.0

  - Standardized data set:

d_1	d_2	d_3
$+1.137147$	-0.288675	-0.5
-0.454859	$+1.443376$	$+1.5$
$+0.454859$	-0.866025	-0.5
-1.137147	-0.288675	-0.5

Principal Component Analysis - 2. Step: Covariance Matrix (I)

- **Procedure:**
  - A n x n covariance matrix is generated that contains the covariance between each possible attribute pairing. When the dimensions are compared with themselves, the variance always replaces the covariance:

$$\begin{bmatrix} \mathrm{Var}(d_1) & \dots & \mathrm{Cov}(d_1, d_n) \\ \dots & \dots & \dots \\ \mathrm{Cov}(d_n, d_1) & \dots & \mathrm{Var}(d_n) \end{bmatrix}$$

- **Reason:**
  - Dimensions that are highly correlated contain redundant information.
  - This step helps to identify these correlations.

Principal Component Analysis - 2. Step: Covariance Matrix (II)

- **Example:**
  - The 3 x 3 covariance matrix of our example:

	d_1	d_2	d_3
d_1	$+1.000000$	-0.350150	-0.303239
d_2	-0.350150	$+1.000000$	$+0.962250$
d_3	-0.303239	$+0.962250$	$+1.000000$

Principal Component Analysis - 3. Step: Eigenvalues (I)

- **Procedure:**
  - The eigenvectors and eigenvalues of the covariance matrix (C) are computed by solving the following equation:

$$C\nu = \lambda\nu$$

  - If an n digit vector ν satisfies this equation for a $\lambda \in \mathbb{R}$, then ν is called an eigenvector with associated eigenvalue λ
- **Reason:**
  - The determined eigenvectors are called **principal components** of the dataset. The eigenvalues indicate which of these principal components has which importance for the significance of the dataset.
  - By sorting the eigenvectors in descending order according to their eigenvalues, the principal components that contain the most information can be identified.

Principal Component Analysis - 3. Step: Eigenvalues (II)

- **Example:**
  - Eigenvalues and eigenvectors in our example:

$$\lambda_1 = +2.14823654, \nu_1 = \begin{bmatrix} +0.37342507 \\ -0.92684562 \\ -0.03887043 \end{bmatrix}$$

$$\lambda_2 = +0.81530433, \nu_2 = \begin{bmatrix} -0.66009198 \\ -0.23604255 \\ -0.71313568 \end{bmatrix}$$

$$\lambda_3 = +0.03645914, \nu_3 = \begin{bmatrix} -0.6517916 \\ -0.2919608 \\ +0.69994757 \end{bmatrix}$$

  - Sorting these three eigenvectors by their significance, we arrive at the order ν_1, ν_2, ν_3

Principal Component Analysis - 4. Step: Feature matrix (I)

- **Procedure:**
  - The top N eigenvectors are selected to create a feature matrix from them.
  - There is no fixed rule exactly how many eigenvectors should be selected.
  - The dimensionality reduction is larger the fewer eigenvectors are chosen.
  - The information loss increases with each eigenvector that is discarded.
- **Reason:**
  - It must be considered carefully how much information can be given up in favor of dimensionality reduction.

Principal Component Analysis - 4. Step: Feature matrix (II)

- **Example:**
  - In our example ν_1 carries approx. 72$\%$ of the information:

$$\frac{2.14823654}{2,14823654 + 0,81530433 + 0,03645914} = 0.71607885$$

  - It might be interesting to keep only the eigenvector ν_1 and discard the other two eigenvectors. Our feature matrix therefore looks as follows:

$$\begin{bmatrix} +0.37342507 \\ -0.92684562 \\ -0.03887043 \end{bmatrix}$$

Principal Component Analysis - 5. Step: Transformation (I)

- **Procedure:**
  - The original data set (D) gets multiplied with the feature matrix (F), to create a new data set (N) with lower dimensionality:

$$N = D \cdot F$$

- **Reason:**
  - This step applies the dimensionality reduction to each tuple.
  - The PCA is completed with this step.

Principal Component Analysis - 5. Step: Transformation (II)

- **Example:**
  - Our dataset after the transformation and with the PCA completed looks like this:

$$\begin{bmatrix} +0.711632 \\ -1.565948 \\ +0.991963 \\ -0.137647 \end{bmatrix}$$

  - It is to be expected that this dataset still contains about 72% of its original information, which can be further used for data mining, while having to deal with a lot less dimensions.

Attribute-subset Selection

- **Another way to reduce dimensionality of data.**
- **Redundant attributes:**
  - Duplicate much or all of the information contained in other attributes.
    - E.g. purchase price of a product and the amount of sales tax paid.
- **Irrelevant attributes:**
  - contain no information that is useful for the data-mining task at hand.
    - E.g. students' ID is often irrelevant to the task of predicting students' GPA.

Heuristic Search in Attribute Selection

- **There are 2^d possible attribute combinations of d attributes.**
- **Typical heuristic attribute-selection methods:**
  - Best single attribute under the attribute-independence assumption: choose by significance tests (e.g. t-test, see Chapter 7 “Classification”).
  - Best step-wise feature selection:
    - The best single attribute is picked first.
    - Then next best attribute condition to the first . . .
- **Step-wise attribute elimination:**
  - Repeatedly eliminate the worst attribute.
- Best combined attribute selection and elimination.
- Optimal branch and bound:
  - Use attribute elimination and backtracking.

Attribute Creation (Feature Generation)

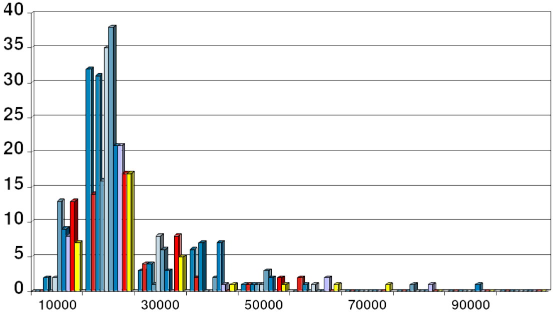
- **Create new attributes (features) that can capture the important information in a data set more effectively than the original ones.**
- **Three general methodologies:**
  - Attribute extraction.
    - Domain-specific.
  - Mapping data to new space (see: data reduction).
    - E.g. Fourier transformation, wavelet transformation, manifold approaches (not covered).
  - Attribute construction:
    - Combining features (see: discriminative frequent patterns in Chapter 5).
    - Data discretization.

Data Reduction (II): Numerosity Reduction

- **Reduce data volume by choosing alternative, smaller forms of data representation.**
- **Parametric methods (e.g., regression):**
  - Assume the data fits some **model** (e.g. a function).
  - Estimate model parameters.
  - Store only the parameters.
  - Discard the data (except possible outliers):
    - Ex. log-linear models obtain value at a point in *m*-dimensional space as the product of appropriate marginal subspaces.
- **Non-parametric methods:**
  - Do not assume models.
  - Major families: histograms, clustering, sampling, ...

Histogram Analysis

- **Divide data into buckets and store average (sum) of each bucket.**
- **Partitioning rules:**
  - Equal-width: equal bucket range.
  - Equal-frequency (or equal-depth).



Clustering

- **Partition data set into clusters based on similarity and store cluster representation (e.g., centroid and diameter) only.**
  - Can be very effective if data points are close to each other under a certain norm and choice of space.
  - Can have hierarchical clustering and be stored in multidimensional index-tree structures.
  - There are many choices of clustering algorithms.
  - Cluster analysis will be studied in depth in Chapter 7.

Sampling

- **Obtain a small sample x to represent the whole data set X.**
- **Allow a mining algorithm to run in complexity that is potentially sub-linear to the size of the data.**
- **Key principle: Choose a representative subset of the data.**
  - Simple random sampling may have very poor performance in the presence of skew.
  - Develop adaptive sampling methods, e.g. stratified sampling.
- **Note: Sampling may not reduce database I/Os.**
  - One page at a time.

Types of Sampling

- **Simple random sampling.**
  - There is an equal probability of selecting any particular item.
- **Sampling without replacement.**
  - Once an object is selected, it is removed from the population.
- **Sampling with replacement.**
  - A selected object is not removed from the population.
- **Stratified sampling:**
  - Partition the data set and draw samples from each partition: Proportionally, i.e. approximately the same percentage of the data.
  - Used in conjunction with skewed data.

Data-cube Aggregation

- **The lowest level of a data cube (base cuboid).**
  - The aggregated data for an **individual entity of interest**.
  - E.g. a customer in a phone-calling data warehouse.
  - Number of calls per hour, day, or week.
- **Multiple levels of aggregation in data cubes.**
  - Further reduce the size of data to deal with.
- **Reference appropriate levels.**
  - Use the smallest representation that is enough to solve the task.
- **Queries regarding aggregated information should be answered using the data cube, if possible.**

Data Reduction (III): Data Compression

- **String compression.**
  - There are extensive theories and well-tuned algorithms.
  - Typically lossless, but only limited manipulation is possible without expansion.
- **Audio/video compression.**
  - Typically lossy compression, with progressive refinement.
  - Sometimes small fragments of signal can be reconstructed without reconstructing the whole.
- **Time sequence is not audio.**
  - Typically short and varies slowly with time.
- **Dimensionality and numerosity reduction may also be considered as forms of data compression.**

Data transformation and data discretization

Data Transformations

- Functions applied to a finite set of samples.
- **Methods:**
  - Smoothing: Remove noise from data.
  - Attribute/feature construction: New attributes constructed from the given ones.
  - Aggregation: Summarization, data-cube construction.
  - Normalization: Scaled to fall within a smaller, specified range.
    - Min-max normalization
    - Z-score normalization.
    - Normalization by decimal scaling.
  - Discretization: concept-hierarchy climbing.

Normalization

- **Min-max normalization (to some interval $[\min, \max]$):**

$$a_{\text{new}} = \frac{a - \min_A}{\max_A - \min_A}(\max - \min) + \min .$$

Example: let income range from $\$12,000$ to $\$98,000$ normalized to $[0, 1]$.
Then $\$73,600$ is mapped to $\frac{73{,}600 - 12{,}000}{98{,}000 - 12{,}000}(1 - 0) + 0 = 0.716$.

- **Z-score normalization:**

$$a_{\text{new}} := z(a) = \frac{a - \mu_A}{\sigma_A}, \text{ with } \mu \text{ being the mean and } \sigma \text{ the standard deviation.}$$

Example: let $\mu = 54,000$ and $\sigma = 16,000$. Then $\frac{73{,}000 - 54{,}000}{16{,}000} = 1.188$.

- **Normalization by decimal scaling:**

$$a_{\text{new}} = \frac{a}{10^k}, \text{ where } k \text{ is the smallest integer such that } \max(|a_{\text{new}}|) < 1.$$

Discretization

- **Three types of attributes:**
  - Nominal – values from an unordered set, e.g. color, profession.
  - Ordinal – values from an ordered set, e.g. military or academic rank.
  - Numerical – numbers, e.g. integer or real numbers.
- **Divide the value range of a continuous attribute into intervals:**
  - **Interval labels** can then be used to replace actual data values.
  - Reduce data size by discretization.
  - Supervised vs. unsupervised.
  - Split (top-down) vs. merge (bottom-up).
  - Discretization can be performed recursively on an attribute.
  - Prepare for further analysis, e.g. classification.

Data-discretization Methods

- **Typical methods:**
  - All the methods can be applied recursively.
  - **Binning:**
    - Unsupervised, top-down split.
  - **Histogram analysis:**
    - Unsupervised, top-down split.
  - **Clustering analysis:**
    - Unsupervised, top-down split or bottom-up merge.
  - **Decision-tree analysis:**
    - Supervised, top-down split.
  - **Correlation (e.g. χ^2) analysis:**
    - Unsupervised, bottom-up merge.

Simple Discretization: Binning

- **Equal-width (distance) partitioning:**
  - Divides the range into N intervals of equal size: uniform grid.
  - If A and B are the lowest and highest values of the attribute, the width of intervals will be: $W = \frac{(B-A)}{N}$.
  - The most straightforward, but outliers may dominate presentation.
  - Skewed data is not handled well.
- **Equal-depth (frequency) partitioning:**
  - Divides the range into N intervals, each containing approximately the same number of samples.
  - Good data scaling.
  - Managing categorical attributes can be tricky.

Binning Methods for Data Smoothing

- **Sorted data for price (in dollars):**
  4, 8, 9, 15, 21, 21, 24, 25, 26, 28, 29, 34.
- **Partition into equal-frequency (equal-depth) bins:**
  Bin 1: 4, 8, 9, 15,
  Bin 2: 21, 21, 24, 25,
  Bin 3: 26, 28, 29, 34.
- **Smoothing by bin means:**
  Bin 1: 9, 9, 9, 9,
  Bin 2: 23, 23, 23, 23,
  Bin 3: 29, 29, 29, 29.
- **Smoothing by bin boundaries:**
  Bin 1: 4, 4, 4, 15,
  Bin 2: 21, 21, 25, 25,
  Bin 3: 26, 26, 26, 34.

Discretization without using Class Labels (Binning vs. Clustering)

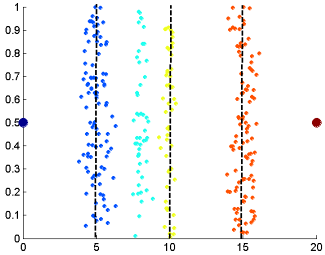


Figure: a) Equal interval width (binning).

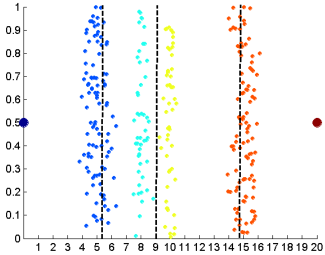


Figure: b) Equal frequency (binning).

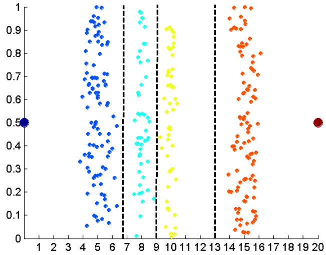


Figure: c) K-means clustering.

Discretization by Classification & Correlation Analysis

- **Classification:**
  - E.g. decision-tree analysis.
  - Supervised: Class labels given for training set e.g. cancerous vs. benign.
  - Using **entropy** to determine split point (discretization point).
  - Top-down, recursive split.
  - Details will be covered in Chapter 6.
- **Correlation analysis:**
  - E.g. χ^2-merge: χ^2-based discretization.
  - Supervised: use class information.
  - Bottom-up merge: find the best neighboring intervals (those having similar distributions of classes, i.e., low χ^2 values) to merge.
  - Merge performed recursively, until a predefined stopping condition.

Concept-hierarchy Generation

- **Concept hierarchy:**
  - Organizes concepts (i.e. attribute values) hierarchically.
  - Usually associated with each dimension in a data warehouse.
  - Facilitates **drilling and rolling** in data warehouses to view data at multiple granularity.
- **Concept-hierarchy formation:**
  - Recursively reduce the data by collecting and replacing **low-level concepts** (such as numerical values for age) by **higher-level concepts** (such as youth, adult, or senior).
  - Can be explicitly specified by domain experts and/or data-warehouse designers.
  - Can be automatically formed for both numerical and nominal data.
  - For numerical data, use discretization methods shown.

Concept-hierarchy Generation for Nominal Data

- **Specification of a partial/total ordering of attributes explicitly at the schema level by users or experts.**
  - $\#(\text{streets}) \prec \#(\text{city}) \prec \#(\text{state}) \prec \#(\text{country})$.
- **Specification of a hierarchy for a set of values by explicit data grouping.**
  - $\#(\{''Urbana'', ''Champaign'', ''Chicago''\}) \prec \#(\text{Illinois})$.
- **Specification of only a partial set of attributes.**
  - Only $\#(\text{street}) \prec \#(\text{city})$, not others.
- **Automatic generation of hierarchies (or attribute levels) by the analysis of the number of distinct values.**
  - E.g. for a set of attributes: $\{\text{street}, \text{city}, \text{state}, \text{country}\}$.
  - See on the next slides.

Automatic Concept-hierarchy Generation

- **Some hierarchies can be automatically generated based on the analysis of the number of distinct values per attribute.**
  - The attribute with the most distinct values is placed at the lowest level of the hierarchy.
  - Exceptions, e.g. weekday, month, quarter, year.
- Example:

$$\begin{aligned} \#(\text{streets}) &= 674.339 > \#(\text{city}) = 3567, \\ \#(\text{city}) &= 3567 > \#(\text{province or state}) = 356, \\ \#(\text{province or state}) &= 356 > \#(\text{country}) = 15. \end{aligned}$$

Friedrich-Alexander-Universität
Technische Fakultät

Summary

Summary

- **Data quality:** Accuracy, completeness, consistency, timeliness, believability, interpretability.
- **Data cleaning:** E.g. missing/noisy values, outliers.
- **Data integration from multiple sources:**
  - Entity identification problem.
  - Remove redundancies.
  - Detect inconsistencies.
- **Data reduction:**
  - Dimensionality reduction.
  - Numerosity reduction.
  - Data compression.
- **Data transformation and data discretization:**
  - Normalization.
  - Concept-hierarchy generation.

Any questions about this chapter?

Ask them now or ask them later in our forum:

StudOn Forum
https://www.studon.fau.de/frm5699567.html