Chapter III: Preprocessing

Knowledge Discovery in Databases

Luciano Melodia M.A.
Evolutionary Data Management, Friedrich-Alexander University Erlangen-Nürnberg
Summer semester 2021

Chapter III: Preprocessing

This is our agenda for this lecture:

- **Data preprocessing: an overview.**
  - Data quality.
  - Major tasks in data preprocessing.
- Data cleaning.
- Data integration.
- Data reduction.
- Data transformation and data discretization.
- Summary.

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

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.

Data reduction:

- Dimensionality reduction.
- Numerosity reduction.
- Data compression.

Data transformation and data discretization:

- Normalization.
- Concept-hierarchy generation.

Chapter III: Preprocessing

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, errors or outliers.
  - Stochastic deviation, imprecision.
  - E.g. measurements.
- **Inconsistencies:** containing discrepancies in codes or names.
  - E.g. age = "42", birthday = "03/07/2010".
  - Was rating "1,2,3" and now it is "A,B,C".
  - Discrepancy between duplicate records (e.g. address old and new).
- **Intentional** (only default value, e.g. disguised missing data):
  - Jan. 1 as everyone's birthday?

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. "unkown", 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

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 relationsships to detect violators (e.g. correlation and clustering to find outliers).

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).

Chapter III: Preprocessing

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)

Two categories:

A has n distinct values: $A := \{a_1, a_2, \ldots, a_n\}$.
B has m distinct values: $B := \{b_1, b_2, \ldots, b_m\}$.

Contingency table:

Given a multiset of paired data points: $X = \{(a_i, b_j) \mid a_i \in A \text{ and } b_j \in B\}$.
Columns: the a_i values of A.
Rows: the b_j values of B.
Cells: counts of data points with
$c_{ij} = \#(\{(a_i, b_j) \in X\})$.

Expected count in cell c_{ij}:

$$e_{ij} = \frac{\sum_{k=1}^{m} c_{ik} \sum_{l=1}^{n} c_{lj}}{\#(X)} \tag{1}$$

where $\#(X)$ is the total count of data points.

Correlation Analysis for Nominal Data (II)

χ^2-test:

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

Summing over all cells of the contingency table.

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 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.

χ^2 Calculation: An Example

	Play chess	Not play chess	Sum (row)
Like Science fiction	250(90)	200(360)	450
Not like science fiction	50(210)	1000(840)	1050
Sum (column)	300	1200	1500

Numbers in parenthesis are expected counts calculated based on the data distribution in the two categories.

χ^2 calculation:

$$\chi^2 = \frac{(250-90)^2}{90} + \frac{(50-210)^2}{210} + \frac{(200-360)^2}{360} + \frac{(1000-840)^2}{840} = 507.93. \tag{3}$$

Degrees of freedom are $(n-1)\cdot(m-1) = (2-1)(2-1) = 1$.

The χ^2 value for a significance level of 0.001 is 10.828.

It shows that "like science fiction" and "play chess" are correlated in the group.

Correlation Analysis of Numerical Data

Correlation coefficient:

Also called Pearson's product-moment coefficient given a paired multiset of data points $\{(a_i, b_i) \mid a_i \in A \text{ and } b_i \in B\}$:

$$r_{A,B} = \frac{\sum_{i=1}^{N}(a_i - \mu_A)(b_i - \mu_B)}{N\sigma_A\sigma_B} = \frac{\sum_{i=1}^{N}(a_i b_i) - N\mu_A\mu_B}{N\sigma_A\sigma_B}. \tag{4}$$

where $N = \#A = \#B$, μ_A and μ_B are the means of A and B, respectively. σ_A and σ_B denote the corresponding standard deviations.

If $r_{A,B} > 0$, A and B are positively correlated (A's values increase with B's).
The higher, the stronger the correlation.

$r_{A,B} = 0$: independent.

$r_{A,B} < 0$: negatively correlated.

Visually Evaluating Correlation

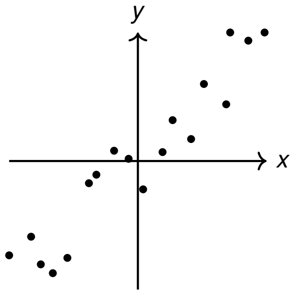


Figure: a) Positive correlation.

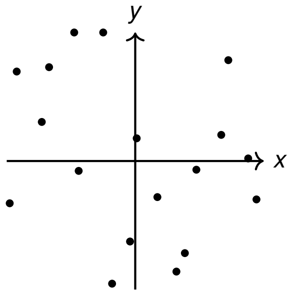


Figure: b) Uncorrelated/no correlation.

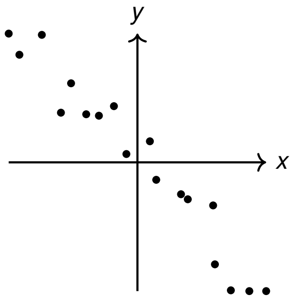


Figure: c) Negative correlation.

Covariance of Numerical Data (I)

Covariance is similar to correlation:

$$\mathrm{Cov}(A,B) = \frac{\sum_{i=1}^{n}(a_i - \overline{A})(b_i - \overline{B})}{N} \tag{5}$$

Pearson's correlation coefficient:

$$r_{A,B} = \frac{\sum_{i=1}^{N}(a_i - \mu_A)(b_i - \mu_B)}{N\sigma_A\sigma_B} = \frac{\sum_{i=1}^{N}(a_i b_i) - N\mu_A\mu_B}{N\sigma_A\sigma_B}, \tag{6}$$

where N is the number of tuples.

Covariance of Numerical Data (II)

Positive covariance:
If $\mathrm{Cov}(A, B) > 0$, then A and B tend to be either both larger or both smaller than their expected values.

Negative covariance:
If $\mathrm{Cov}(A, B) < 0$, then if A is larger than its expected value, B is likely to be smaller than its expected value and vice versa.

Independence:

$\mathrm{Cov}(A, B) = 0$.

But the converse is not true: Some pairs of random variables may have a covariance of 0 but are not independent. Only under some additional assumptions (e.g., the data follow multivariate normal distributions) does a covariance of 0 imply independence.

Covariance: An Example (I)

Can be simplified in computation as:

$$\begin{aligned}
\mathrm{Cov}(A, B) &= E((A - E(A))(B - E(B))) \qquad (7)\\
&= E(AB - AE(B) - E(A)B + E(A)E(B)) \qquad (8)\\
&= E(AB) - E(A)E(B) - E(A)E(B) + E(A)E(B) \qquad (9)\\
&= E(AB) - E(A)E(B). \qquad (10)
\end{aligned}$$

Covariance: An Example (II)

Suppose two stocks A and B have the following values within some time:
$(2, 5), (3, 8), (5, 10), (4, 11), (6, 14)$.

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

$$E(A) = \frac{2+3+5+4+6}{5} = \frac{20}{5} = 4. \tag{11}$$

$$E(B) = \frac{5+8+10+11+14}{5} = \frac{48}{5} = 9.6. \tag{12}$$

$$\mathrm{Cov}(A, B) = \frac{2\cdot 5 + 3\cdot 8 + 5\cdot 10 + 4\cdot 11 + 6\cdot 14}{5} - 4\cdot 9.6 = 4. \tag{13}$$

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

Chapter III: Preprocessing

Data Reduction (I): Dimensionality Reduction

Curse of dimensionality:

- When dimensionality increases data becomes increasingly sparse.
- Density and distance between points, which are critical to clustering and outlier analysis 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).

Data Reduction Strategies

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.
- Numerosity reduction:
    - Regression and log-linear models.
    - Histograms, clustering and sampling.
    - Data cube aggregation.
- Data compression.

Mapping Data to a New Space

Fourier transform.

Wavelet transform.

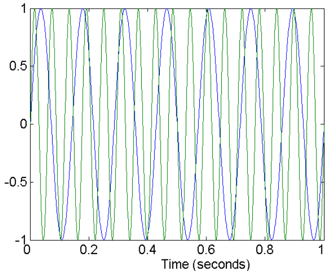


Figure: Two sine waves.

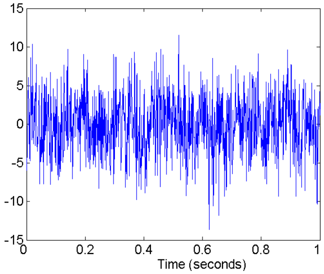


Figure: Two sine waves with noise.

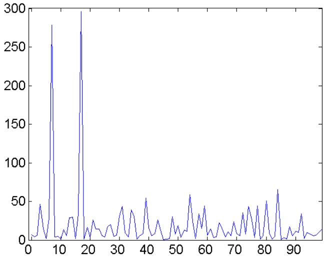


Figure: Frequencies.

What is Wavelet Transform?

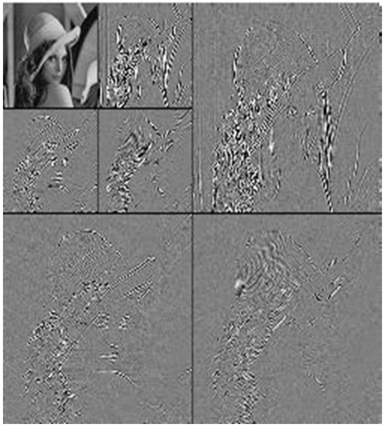
Decomposes a signal into different frequency subbands.

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

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.

Wavelet Decomposition

Example:

$$X = \big(2, 2, 0, 2, 3, 5, 4, 4\big), \text{ can be transformed to} \tag{14}$$

$$X' = \big(2.75, -1.25, 0.5, 0, 0, -1, -1, 0\big). \tag{15}$$

Compression:
Many small detail coefficients can be replaced by 0's,
and only the significant coefficients are retained.

Resolution	Averages	Detail coefficients
8	$\big(2, 2, 0, 2, 3, 5, 4, 4\big)$	-
4	$\big(2, 1, 4, 4\big)$	$\big(0, -1, -1, 0\big)$
2	$\big(1\frac{1}{2}, 4\big)$	$\big(\frac{1}{2}, 0\big)$
1	$\big(2\frac{3}{4}\big)$	$\big(-1\frac{1}{4}\big)$

Why Wavelet Transform?

Use 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 (I)

Principal component analysis is a method of summarizing the properties of a set of multivariate data samples.

It is a **linear transformation** method that is often used for data analysis or data compression.

Principal component analysis is often also called **Karhunen-Loeve transformation**.

PCA is equivalent to maximization of the information at the output of a neural network with linear neurons.

The goal of the principal component analysis (PCA) is the identification of m **normed orthogonal vectors** $\{x_i \in \mathbb{R}^n \mid i = 1, 2, \ldots, m\}$ within the input space, which **represent most of the variance** of the data.

Principal Component Analysis: The Problem (II)

Consider a set of data points $\{\mathbf{x}_i = (x_{i1}, x_{i2}, \ldots, x_{in}) \mid \mathbf{x}_i \in \mathbb{R}^n\}$, where each dimension measures some physical quantity.

Consider an orthonormal system of n-vectors with dimension n:
$\{\mathbf{b}_i = (b_{i1}, b_{i2}, \ldots, b_{in}) \mid \mathbf{b}_i \in \mathbb{R}^n\}$.

> The **orthonormal system** forms a basis of the vector space, where we have recorded our data. Thus our data can be expressed as a linear combination of $\{\mathbf{b}_i\}$.

$$\mathbf{B} = \begin{bmatrix} \mathbf{b}_1 \\ \mathbf{b}_2 \\ \vdots \\ \mathbf{b}_n \end{bmatrix} = \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{bmatrix} \tag{16}$$

Each row is an orthonormal basis vector $\mathbf{b}_i$ with n-components. Thus we have a $n \times n$-matrix.

Principal Component Analysis: The Problem (III)

Is there another basis, which is a linear combination of the initially chosen basis, that best re-expresses our data set?

Let $\mathbf{X}$ we the data matrix with $n \times m$ records.

Let $\mathbf{Y}$ be another $n \times m$-matrix, related by some linear transformation $\mathbf{U}$, such that

$$\mathbf{U}\mathbf{X} = \mathbf{Y}, \tag{17}$$

$$\begin{bmatrix} \mathbf{u}_1 \\ \mathbf{u}_2 \\ \vdots \\ \mathbf{u}_n \end{bmatrix} \begin{bmatrix} \mathbf{x}_1 & \mathbf{x}_2 & \cdots & \mathbf{x}_m \end{bmatrix} = \begin{bmatrix} \mathbf{u}_1\mathbf{x}_1 & \mathbf{u}_1\mathbf{x}_2 & \cdots & \mathbf{u}_1\mathbf{x}_m \\ \mathbf{u}_2\mathbf{x}_1 & \mathbf{u}_2\mathbf{x}_2 & \cdots & \mathbf{u}_2\mathbf{x}_m \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{u}_n\mathbf{x}_1 & \mathbf{u}_n\mathbf{x}_2 & \cdots & \mathbf{u}_n\mathbf{x}_m \end{bmatrix}. \tag{18}$$

Eq. 17 and 18 represent the **change of a basis** and can be interpreted as:

1. $\mathbf{U}$ is a matrix that transforms $\mathbf{X}$ into $\mathbf{Y}$.
2. Geometrically, $\mathbf{U}$ is a rotation and stretch which again transforms $\mathbf{X}$ into $\mathbf{Y}$.
3. The rows of $\mathbf{U}$, $\{\mathbf{u}_1, \ldots, \mathbf{u}_n\}$, are a set of new basis vectors for expressing the columns of $\mathbf{X}$.

Principal Component Analysis: The Problem (IV)

Questions we have to answer:

What is the best way to "re-express" **X**?
What is a good choice of basis **U**?

Main assumptions:

Continuity of data space.
Linearity of the basis vectors.
Mean and variance are sufficient statistics.
Large variance have important dynamics in data.
Principal components are orthogonal.

PCA assumes that the basis vectors are orthonormal, i.e., $\mathbf{u}_i \cdot \mathbf{u}_j = \delta_{ij}$, such that **U** is an orthonormal matrix.

PCA assumes that the directions with the largest variances are most *principal*.

Principal Component Analysis: Solving the Problem

Find some orthonormal matrix $\mathbf{U}$ where $\mathbf{Y} = \mathbf{UX}$ such that $\mathbf{Cov}(\mathbf{Y}, \mathbf{Y}) \equiv \frac{1}{n-1}\mathbf{YY}^T$ is diagonalized.
The rows of $\mathbf{U}$ are the *principal components* of $\mathbf{X}$.
Recall, high diagonal values of $\mathbf{Cov}(\mathbf{Y}, \mathbf{Y})$ mean a covariance matrix with large variance, thus interesting dynamics. Large off-diagonal values mean high covariance, thus high redundancy.
We rewrite $\mathbf{Cov}(\mathbf{Y}, \mathbf{Y})$ in terms of $\mathbf{U}$:

$$\mathbf{Cov}(\mathbf{Y}, \mathbf{Y}) = \frac{1}{n-1}\mathbf{YY}^T \tag{19}$$

$$= \frac{1}{n-1}(\mathbf{UX})(\mathbf{UX})^T \tag{20}$$

$$= \frac{1}{n-1}\mathbf{UXX}^T\mathbf{U}^T \tag{21}$$

$$= \frac{1}{n-1}\mathbf{UAU}^T. \tag{22}$$

We have a new matrix $\mathbf{A} \equiv \mathbf{XX}^T$, which is **symmetric**.

Principal Component Analysis: Diagonalization of Symmetric Matrices (I)

The symmetric matrix $\mathbf{A}$ can be diagonalized by an orthogonal matrix of its eigenvectors.

Orthogonally diagonalizable means that there exists some $\mathbf{E}$, such that $\mathbf{A} = \mathbf{E}\mathbf{D}\mathbf{E}^T$, where $\mathbf{D}$ is a diagonal matrix and $\mathbf{E}$ is a matrix diagonalizing $\mathbf{A}$. Let's compute $\mathbf{A}^T$:

$$\mathbf{A}^T = (\mathbf{E}\mathbf{D}\mathbf{E}^T)^T = \mathbf{E}^{TT}\mathbf{D}^T\mathbf{E}^T = \mathbf{E}\mathbf{D}\mathbf{E}^T = \mathbf{A}. \tag{23}$$

Thus, if $\mathbf{A}$ is orthogonally diagonalizable, it must also be symmetric.

Principal Component Analysis: Diagonalization of Symmetric Matrices (II)

Let $\mathbf{E} = [\mathbf{e}_1 \; \mathbf{e}_2 \; \cdots \; \mathbf{e}_m]$ be a matrix of normalized eigenvectors $\mathbf{e}_i$. We briefly cover, that a matrix can only be orthogonally diagonalized iff that matrix's eigenvectors are linearly independent. Further, we show that such a matrix has not only linearly independent, but orthogonal eigenvectors.

Let $\mathbf{A}$ be a matrix, not necessarily symmetric, with linearly independent eigenvectors. Let further $\mathbf{D}$ be a diagonal matrix where the *i*th eigenvalue is placed in $\mathbf{D}_{ii}$.

We show that $\mathbf{AE} = \mathbf{ED}$.

$$\text{Left hand side:} \quad \mathbf{AE} = [\mathbf{Ae}_1 \; \mathbf{Ae}_2 \; \cdots \; \mathbf{Ae}_m], \tag{24}$$

$$\text{Right hand side:} \quad \mathbf{ED} = [\lambda \mathbf{e}_1 \; \lambda \mathbf{e}_2 \; \cdots \; \lambda \mathbf{e}_m]. \tag{25}$$

Note, that if $\mathbf{AE} = \mathbf{ED}$ then $\mathbf{Ae}_i = \lambda_i \mathbf{e}_i$ for all i.

This is the definition of the eigenvalue equation, thus $\mathbf{AE} = \mathbf{ED}$ follows.

$\implies \mathbf{A} = \mathbf{EDE}^{-1} = \mathbf{EDE}^T$.

Intermezzo: Inverse of Orthogonal Matrix is its Transposed

Let $\mathbf{E}$ be an orthogonal matrix, then $\mathbf{E}^{-1} = \mathbf{E}^T$.

Write $\mathbf{E}$ as $\mathbf{E} = \begin{bmatrix}\mathbf{e}_1 & \mathbf{e}_2 & \cdots & \mathbf{e}_n\end{bmatrix}$, where $\mathbf{e}_i$ is the ith column vector. We now have to show that $\mathbf{E}^T\mathbf{E} = \mathbf{I}$, where $\mathbf{I}$ is the identity matrix.

Note that $\left(\mathbf{E}^T\mathbf{E}\right)_{ij} = \mathbf{e}_i^T\mathbf{e}_j$. As $\mathbf{E}$ is orthogonal, the dot product is 0 for any two column vectors.

The only exception is the dot product of one particular column with itself,
which equals one:

$$\left(\mathbf{E}^T\mathbf{E}\right)_{ij} = \mathbf{e}_i^T\mathbf{e}_j = \begin{cases} 1 & i = j, \\ 0 & i \neq j. \end{cases} \tag{26}$$

$\mathbf{E}^T\mathbf{E} = \mathbf{I}$ and by definition we have that $\mathbf{E}^{-1}\mathbf{E} = \mathbf{I}$.

Therefore, we conclude that $\mathbf{E}^{-1} = \mathbf{E}^T$.

Principal Component Analysis: Diagonalization of Symmetric Matrices (III)

Now we show that a symmetric matrix has orthogonal eigenvectors.

Let λ_1 and λ_2 be distinct eigenvalues for eigenvectors $\mathbf{e}_1$ and $\mathbf{e}_2$, such that

$$\begin{aligned}
\lambda_1 \mathbf{e}_1 \cdot \mathbf{e}_2 &= (\lambda_1 \mathbf{e}_1)^T \mathbf{e}_2 && (27)\\
&= (\mathbf{A}\mathbf{e}_1)^T \mathbf{e}_2 && (28)\\
&= \mathbf{e}_1^T \mathbf{A}^T \mathbf{e}_2 && (29)\\
&= \mathbf{e}_1^T \mathbf{A} \mathbf{e}_2 && (30)\\
&= \mathbf{e}_1^T (\lambda_2 \mathbf{e}_2) && (31)\\
\lambda_1 \mathbf{e}_1 \cdot \mathbf{e}_2 &= \lambda_2 \mathbf{e}_1 \cdot \mathbf{e}_2 . && (32)
\end{aligned}$$

By the last relation we can equate that $(\lambda_1 - \lambda_2)\mathbf{e}_1 \cdot \mathbf{e}_2 = 0$.

By conjecture, the eigenvalues are unique, thus it holds that $\mathbf{e}_1 \cdot \mathbf{e}_2 = 0$.

We conclude that the eigenvectors of a symmetric matrix are orthogonal.

Principal Component Analysis: Diagonalization of Symmetric Matrices (IV)

We defined $\mathbf{A} \equiv \mathbf{X}\mathbf{X}^T$, where $\mathbf{A}$ is symmetric.

Further, we computed $\mathbf{Cov}(\mathbf{Y}, \mathbf{Y}) = \frac{1}{n-1}\mathbf{U}\mathbf{A}\mathbf{U}^T$.

By the intermezzo we know that $\mathbf{A} = \mathbf{E}\mathbf{D}\mathbf{E}^T$,
for some eigenvector matrix $\mathbf{E}$ of $\mathbf{A}$ and diagonal matrix $\mathbf{D}$.

What if A is degenerated?

- $\mathbf{A}$ has $r \leq n$ orthonormal eigenvectors, where $r = \text{rank } \mathbf{A}$.
- If $r < n$, then $\mathbf{A}$ is degenerated.
    - All data occupy a subspace of dimension $r \leq n$.
- We select $(n - r)$ additional orthonormal vectors to *fill up* the matrix $\mathbf{E}$.
    - These additional vectors do not effect the final solution because the variances associated with these directions are zero.

Principal Component Analysis: The Trick

Select $\mathbf{U}$ to be a matrix where each row $\mathbf{u}_i$ is ein eigenvector of $\mathbf{X}\mathbf{X}^T$.
Thus, $\mathbf{U} \equiv \mathbf{E}^T$.
Substituting into the result from our intermezzo we get: $\mathbf{A} = \mathbf{U}^T\mathbf{D}\mathbf{U}$. We rewrite the equation for $\mathbf{Cov}(\mathbf{Y},\mathbf{Y})$ as

$$\mathbf{Cov}(\mathbf{Y},\mathbf{Y}) = \frac{1}{n-1}\mathbf{U}\mathbf{A}\mathbf{U}^T \tag{33}$$

$$= \frac{1}{n-1}\mathbf{U}(\mathbf{U}^T\mathbf{D}\mathbf{U})\mathbf{U}^T \tag{34}$$

$$= \frac{1}{n-1}(\mathbf{U}\mathbf{U}^T)\mathbf{D}(\mathbf{U}\mathbf{U}^T) \tag{35}$$

$$= \frac{1}{n-1}(\mathbf{U}\mathbf{U}^{-1})\mathbf{D}(\mathbf{U}\mathbf{U}^{-1}) \tag{36}$$

$$\mathbf{Cov}(\mathbf{Y},\mathbf{Y}) = \frac{1}{n-1}\mathbf{D}. \tag{37}$$

Principal Component Analysis: Summary

We conclude, that our choice of $\mathbf{U}$ diagonalizes $\mathbf{Cov}(\mathbf{Y}, \mathbf{Y})$.

The results are summarized as follows:

- The principal components of $\mathbf{X}$ are the eigenvectors of $\mathbf{X}\mathbf{X}^T$; or the rows of $\mathbf{U}$.
- The *i*th diagonal value of $\mathbf{Cov}(\mathbf{Y}, \mathbf{Y})$ is the variance of $\mathbf{X}$ along $\mathbf{u}_i$.
- The computation of PCA involves subtracting the mean of each measurement type.
- The computation of PCA involves computing the eigenvectors of $\mathbf{X}\mathbf{X}^T$.

Principal Component Analysis: Algorithm

```
PCA(data) {
    [N,M] = size(data); # (N dimensions, M trials).
    means = mean(data,2);
    data = data - repmat(means,1,M);
    covariance = 1 / (N-1) $*$ data $*$ data$^T$;
    [PC,V] = eig(covariance); # PC - each column is a PC. V - Nx1 matrix of variances.
    V = diag(V); # Extract diagonal of matrix as vector.
    [junk, rindices] = sort(-1 $*$ V); # Sort the variances in decreasing order.
    V = V(rindices);
    PC = PC(:,rindices);
    signals = PC$^T$ $*$ data;
    return signals; # Signals - NxM matrix of projected data.
};
```

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 6).
- 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, . . .

Parametric Data Reduction: Regression and Log-Linear Models

Linear regression:

Data modeled to fit a **straight line**.
Often uses the **least-square method** to fit the line.

Multiple regression:

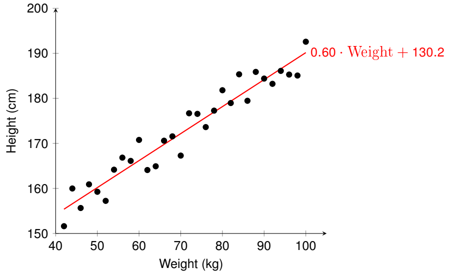
Allows a response random variable Y to be modeled as a linear function of a multidimensional feature vector $(x_1, x_2, \ldots, x_n)$.

Log-linear model:

Approximates discrete multidimensional probability distributions.

Regression Analysis

- A collective name for techniques for the modeling and analysis of numerical data consisting of values of a **dependent variable** (also called response variable or measurement) and of one or more **independent variables** (aka. explanatory variables or predictors).
- The parameters are estimated so as to give a best fit of the data.
- The **best fit** is evaluated by using the **least-squares method**, but other criteria have also been used.
- Used for prediction (including forecasting of time-series data), inference, hypothesis testing, and modeling of causal relationships.



Regression Analysis and Log-Linear Models

Linear regression: $y = wx + b$.

- Two regression coefficients, w and b,
- specify the line and are to be estimated by using the data at hand.
- Using the least-squares criterion to the known values of $y_1, y_2, \ldots, x_1, x_2, \ldots$

Multiple regression: $y = w_1 x_1 + w_2 x_2 + \ldots + w_n x_n + b$.

- Many nonlinear functions can be transformed into the above.

Log-linear models:

- Approximate discrete multidimensional probability distributions.
- Estimate the probability of each point (tuple) in a multi-dimensional space for a set of discretized attributes, based on a smaller subset of dimensional combinations.
- Useful also for dimensionality reduction and data smoothing.

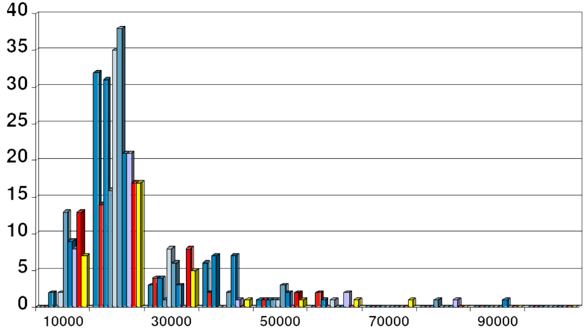
Histogram Analysis

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 repetition.

Once an object is selected, it is removed from the population.

Sampling with repetition.

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 which 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.

Chapter III: Preprocessing

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 $[\min, \max]$):

$$a_{\text{new}} = \frac{a - \min_A}{\max_A - \min_A}(\max - \min) + \min\,. \tag{38}$$

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:

$$a_{\text{new}} := z(a) = \frac{a - \mu_A}{\sigma_A}, \text{ with } \mu \text{ being the mean and } \sigma \text{ the standard deviation.} \tag{39}$$

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

Normalization by decimal scaling:

$$a_{\text{new}} = \frac{a}{10^k}, \text{ where } k \text{ is the smallest integer such that } \max(|a_{\text{new}}|) < 1. \tag{40}$$

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)

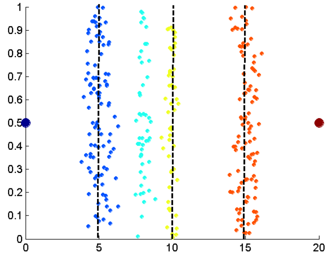


Figure: a) Equal interval width (binning).

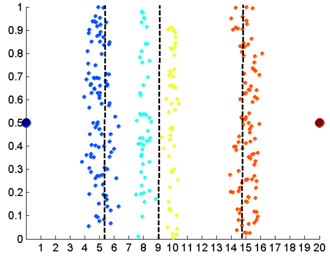


Figure: b) Equal frequency (binning).

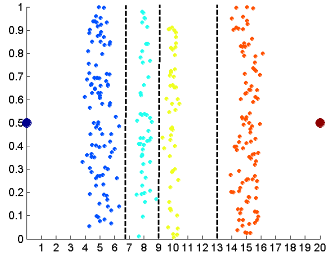


Figure: c) K-means clustering.

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.

$\#(\{''\textit{Urbana}'', ''\textit{Champaign}'', ''\textit{Chicago}''\}) \prec \#(\text{Illinois})$.

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:

$$\#(\text{streets}) = 674.339 > \#(\text{city}) = 3567, \tag{41}$$

$$\#(\text{city}) = 3567 > \#(\text{province or state}) = 356, \tag{42}$$

$$\#(\text{province or state}) = 356 > \#(\text{country}) = 15. \tag{43}$$

Chapter III: Preprocessing

- Data preprocessing: an overview.
  - Data quality.
  - Major tasks in data preprocessing.
- Data cleaning.
- Data integration.
- Data reduction.
- Data transformation and data discretization.
- **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.

References (I)

D. P. Ballou and G. K. Tayi: Enhancing data quality in data warehouse environments. Comm. of ACM, 42:73-78, 1999.

A. Bruce, D. Donoho and H.-Y. Gao: Wavelet analysis. IEEE Spectrum, Oct. 1996.

T. Dasu and T. Johnson: Exploratory Data Mining and Data Cleaning. John Wiley, 2003.

J. Devore and R. Peck: Statistics: The Exploration and Analysis of Data. Duxbury Press, 1997.

H. Galhardas, D. Florescu, D. Shasha, E. Simon and C.-A. Saita: Declarative data cleaning: Language, model, and algorithms. VLDB'01.

M. Hua and J. Pei: Cleaning disguised missing data: A heuristic approach. KDD'07.

H. V. Jagadish et al.: Special Issue on Data Reduction Techniques. Bulletin of the Technical Committee on Data Engineering, 20(4), Dec. 1997.

References (2)

H. Liu and H. Motoda (eds.): Feature Extraction, Construction, and Selection: A Data Mining Perspective. Kluwer Academic, 1998.

J. E. Olson. Data Quality: The Accuracy Dimension. Morgan Kaufmann, 2003.

D. Pyle: Data Preparation for Data Mining. Morgan Kaufmann, 1999.

V. Raman and J. Hellerstein: Potter's Wheel: An Interactive Framework for Data Cleaning and Transformation, VLDB'01.

T. Redman: Data Quality: The Field Guide. Digital Press (Elsevier), 2001.

R. Wang, V. Storey and C. Firth: A framework for analysis of data quality research. IEEE Trans. Knowledge and Data Engineering, 7:623-640, 1995.

J. Shlens: A Tutorial on Principal Component Analysis. 2005, URL: `https://www.cs.cmu.edu/~elaw/papers/pca.pdf`.

Thank you for your attention.
Any questions about the third chapter?

Ask them now, or again, drop me a line:
`luciano.melodia@fau.de`.