

- Why data preprocessing?
- Data cleaning
- Data integration and transformation
- Data reduction
- Discretization and concept hierarchy generation
- Summary

Why Data Preprocessing?

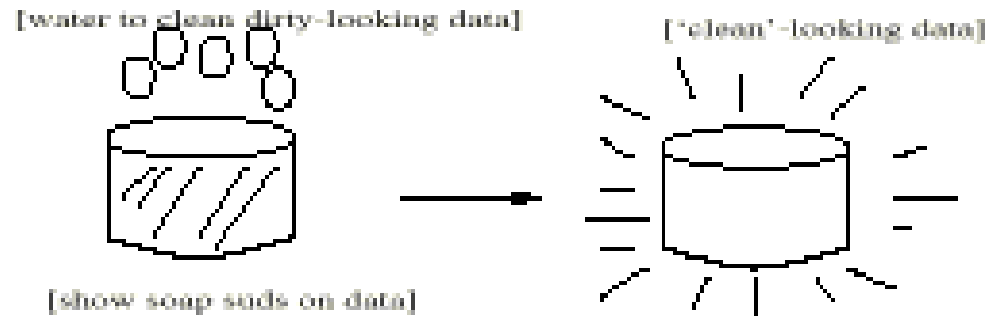
- Data in the real world is dirty
 - **incomplete**: lacking attribute values, lacking certain attributes of interest, or containing only aggregate data
 - **noisy**: containing errors or outliers
 - **inconsistent**: containing discrepancies in codes or names
- No quality data, no quality mining results!
 - Quality decisions must be based on quality data
 - Data warehouse needs consistent integration of quality data
- A multi-dimensional measure of data quality:
 - A well-accepted multi-dimensional view:
 - accuracy, completeness, consistency, timeliness, believability, value added, interpretability, accessibility
 - Broad categories:
 - intrinsic, contextual, representational, and accessibility.

Major Tasks in Data Preprocessing

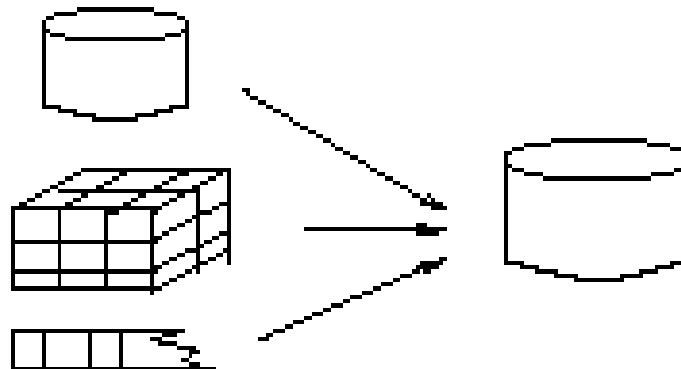
- Data cleaning
 - Fill in missing values, smooth noisy data, identify or remove outliers, and resolve inconsistencies
- Data integration
 - Integration of multiple databases, data cubes, files, or notes
- Data transformation
 - Normalization (scaling to a specific range)
 - Aggregation
- Data reduction
 - Obtains reduced representation in volume but produces the same or similar analytical results
 - Data discretization: with particular importance, especially for numerical data
 - Data aggregation, dimensionality reduction, data compression, generalization

Forms of data preprocessing

Data Cleaning



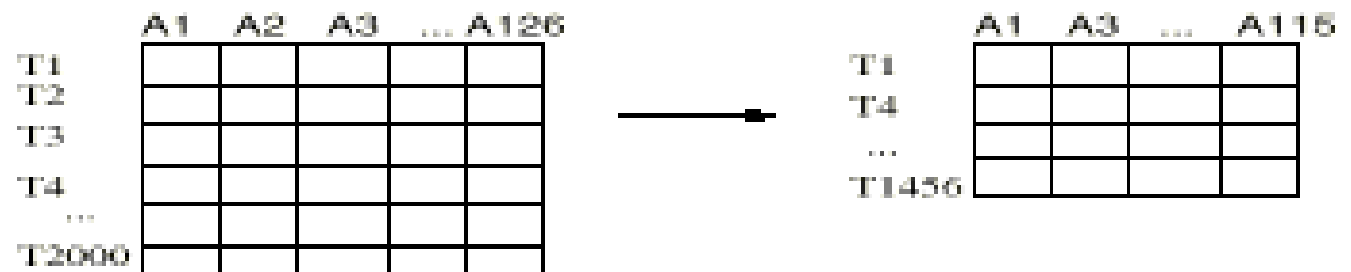
Data Integration



Data Transformation

-2, 32, 100, 59, 48 → -0.02, 0.32, 1.00, 0.59, 0.48

Data Reduction



Agenda

- Why preprocess the data?
- Data cleaning
- Data integration and transformation
- Data reduction
- Discretization and concept hierarchy generation
- Summary

Data Cleaning

- Data cleaning tasks
 - Fill in missing values
 - Identify outliers and smooth out noisy data
 - Correct inconsistent 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

How to Handle Missing Data?

- Ignore the tuple: usually done when class label is missing (assuming the task is classification—not effective in certain cases)
- Fill in the missing value **manually**: tedious + infeasible?
- Use a **global constant** to fill in the missing value: e.g., “unknown”, a new class?!
- Use the **attribute mean** to fill in the missing value
- Use the **attribute mean for all samples of the same class** to fill in the missing value: smarter
- Use the **most probable value** to fill in the missing value: inference-based such as regression, Bayesian formula, decision tree

Noisy Data

- Q: What is noise?
- A: Random error in a measured variable.
- Incorrect attribute values may be due to
 - faulty data collection instruments
 - data entry problems
 - data transmission problems
 - technology limitation
 - inconsistency in naming convention
- Other data problems which requires data cleaning
 - duplicate records
 - incomplete data
 - inconsistent data

How to Handle Noisy Data?

- Binning method:
 - first sort data and partition into (equi-depth) bins
 - then one can smooth by bin means, smooth by bin median, smooth by bin boundaries, etc.
 - used also for discretization (discussed later)
- Clustering
 - detect and remove outliers
- Semi-automated method: combined computer and human inspection
 - detect suspicious values and check manually
- Regression
 - smooth by fitting the data into regression functions

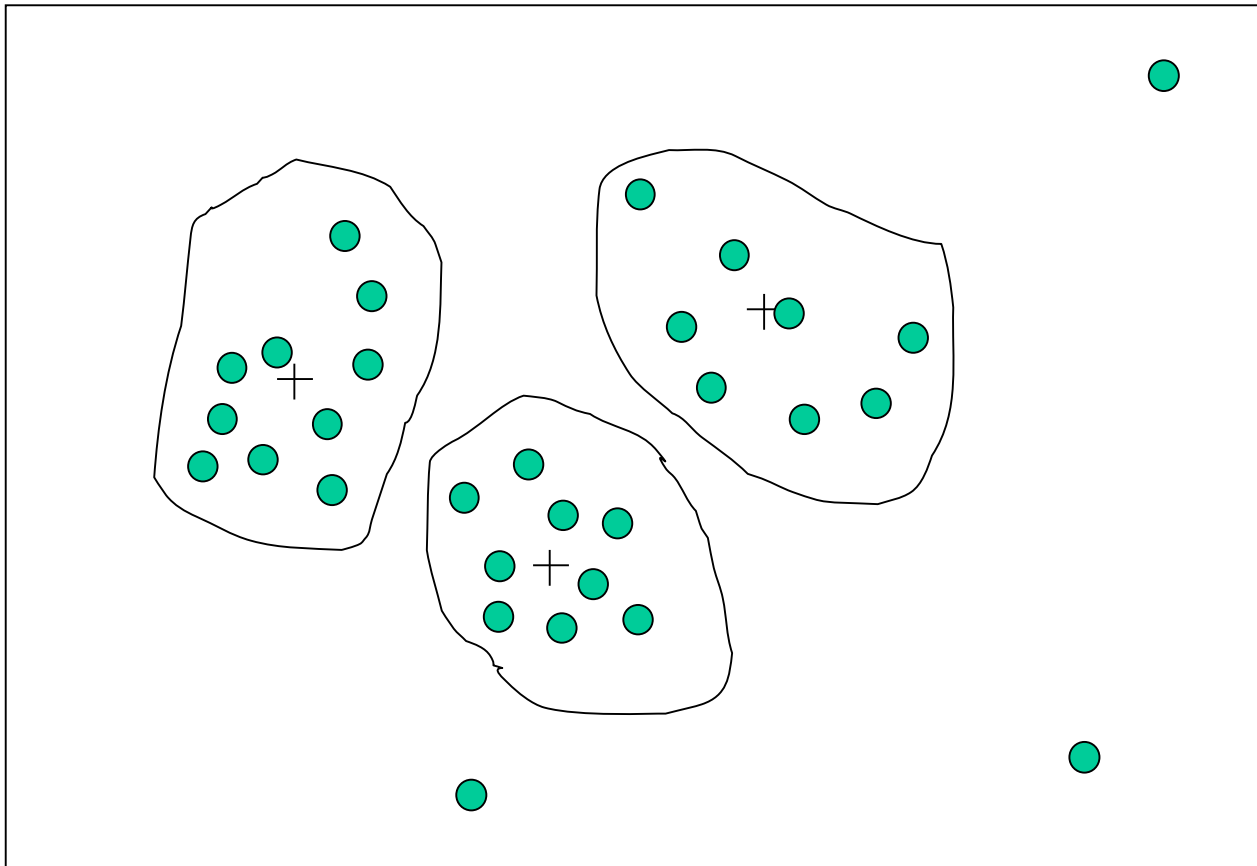
Simple Discretization Methods: Binning

- **Equal-width** (distance) partitioning:
 - It 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: $W = (B-A)/N$.
 - The most straightforward
 - But outliers may dominate presentation
 - Skewed data is not handled well.
- **Equal-depth** (frequency) partitioning:
 - It divides the range into N intervals, each containing approximately same number of samples
 - Good data scaling
 - Managing categorical attributes can be tricky.

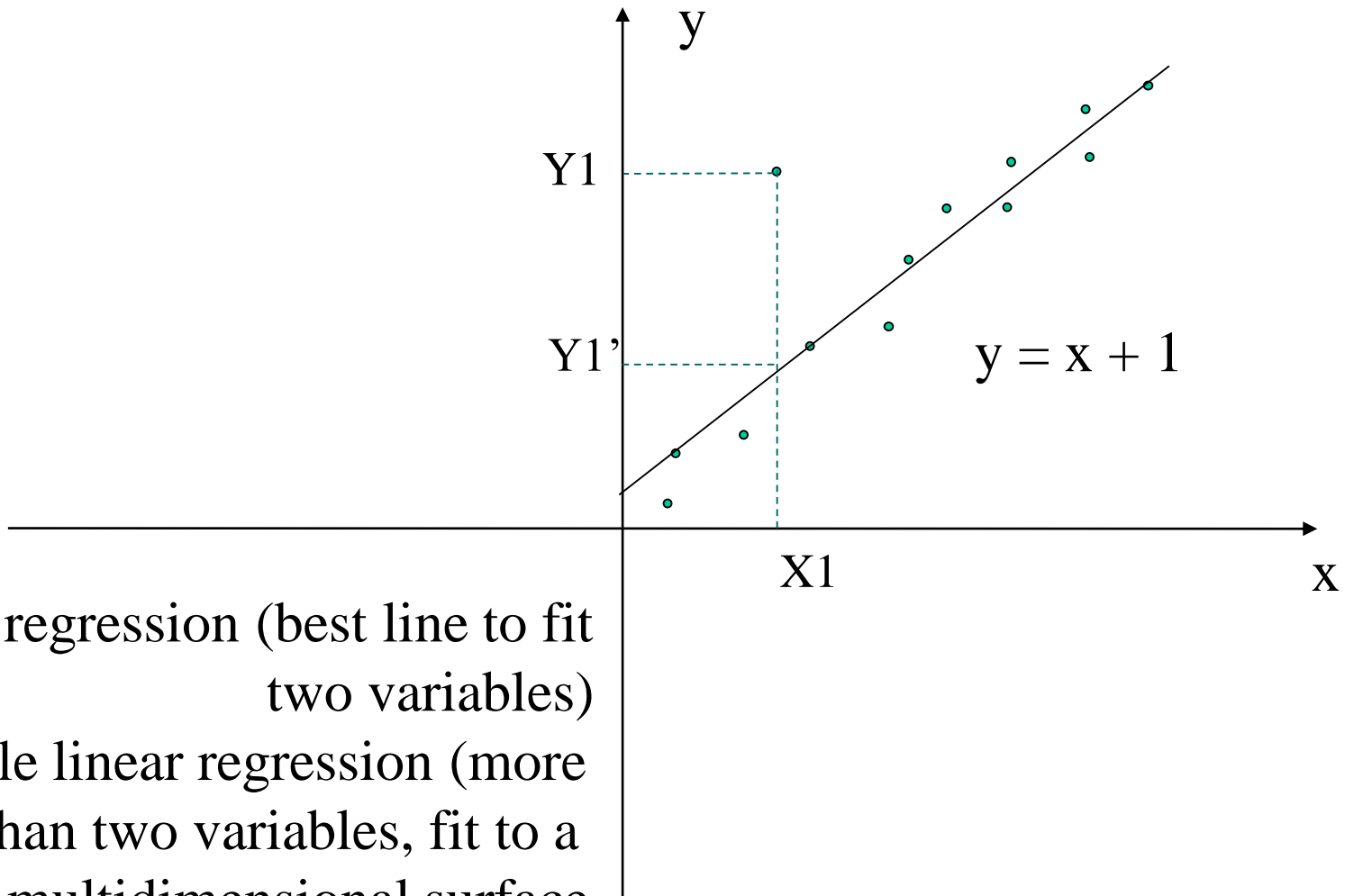
Binning Methods for Data Smoothing

- * Sorted data for price (in dollars): 4, 8, 9, 15, 21, 21, 24, 25, 26, 28, 29, 34
- * Partition into (equi-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

Cluster Analysis



Regression



- Linear regression (best line to fit two variables)
- Multiple linear regression (more than two variables, fit to a multidimensional surface)

Data Integration

- Data integration:
 - combines data from multiple sources into a coherent store
- Schema integration
 - integrate metadata from different sources
 - Entity identification problem: identify real world entities from multiple data sources, e.g., A.cust-id \equiv B.cust-#
- Detecting and resolving data value conflicts
 - for the same real world entity, attribute values from different sources are different
 - possible reasons: different representations, different scales, e.g., metric vs. British units, different currency

Handling Redundant Data in Data Integration

- Redundant data occur often when integrating multiple DBs
 - The same attribute may have different names in different databases
 - One attribute may be a “derived” attribute in another table, e.g., annual revenue
- Redundant data may be able to be detected by correlational analysis

$$r_{A,B} = \frac{\sum (A - \bar{A})(B - \bar{B})}{(n-1)\sigma_A\sigma_B}$$

- Careful integration can help reduce/avoid redundancies and inconsistencies and improve mining speed and quality

Data Transformation

- Smoothing: remove noise from data (binning, clustering, regression)
- Aggregation: summarization, data cube construction
- Generalization: concept hierarchy climbing
- Normalization: scaled to fall within a small, specified range
 - min-max normalization
 - z-score normalization
 - normalization by decimal scaling
- Attribute/feature construction
 - New attributes constructed from the given ones

Data Transformation: Normalization

Particularly useful for classification (NNs, distance measurements, nn classification, etc)

- min-max normalization

$$v' = \frac{v - \min A}{\max A - \min A} (\text{new}_{\max} A - \text{new}_{\min} A) + \text{new}_{\min} A$$

- z-score normalization

$$v' = \frac{v - \text{mean} A}{\text{stand}_{\text{dev}} A}$$

- normalization by decimal scaling

$$v' = \frac{v}{10^j} \quad \text{Where } j \text{ is the smallest integer such that } \text{Max}(|v'|) < 1$$

Data Reduction

- Obtains a reduced representation of the data set that is much smaller in volume but yet produces the same (or almost the same) analytical results
- Data reduction strategies
 - Data cube aggregation-combined data to construct data cube
 - Dimensionality reduction-reducing number of variables
 - Attribute set selection-highly relevant attributes
 - Discretization and concept hierarchy generation

Data Cube Aggregation

- Multiple levels of aggregation in data cubes
 - Further reduce the size of data to deal with
- Reference appropriate levels
 - Use the smallest representation capable to solve the task
- Queries regarding aggregated information should be answered using data cube, when possible

Dimensionality Reduction

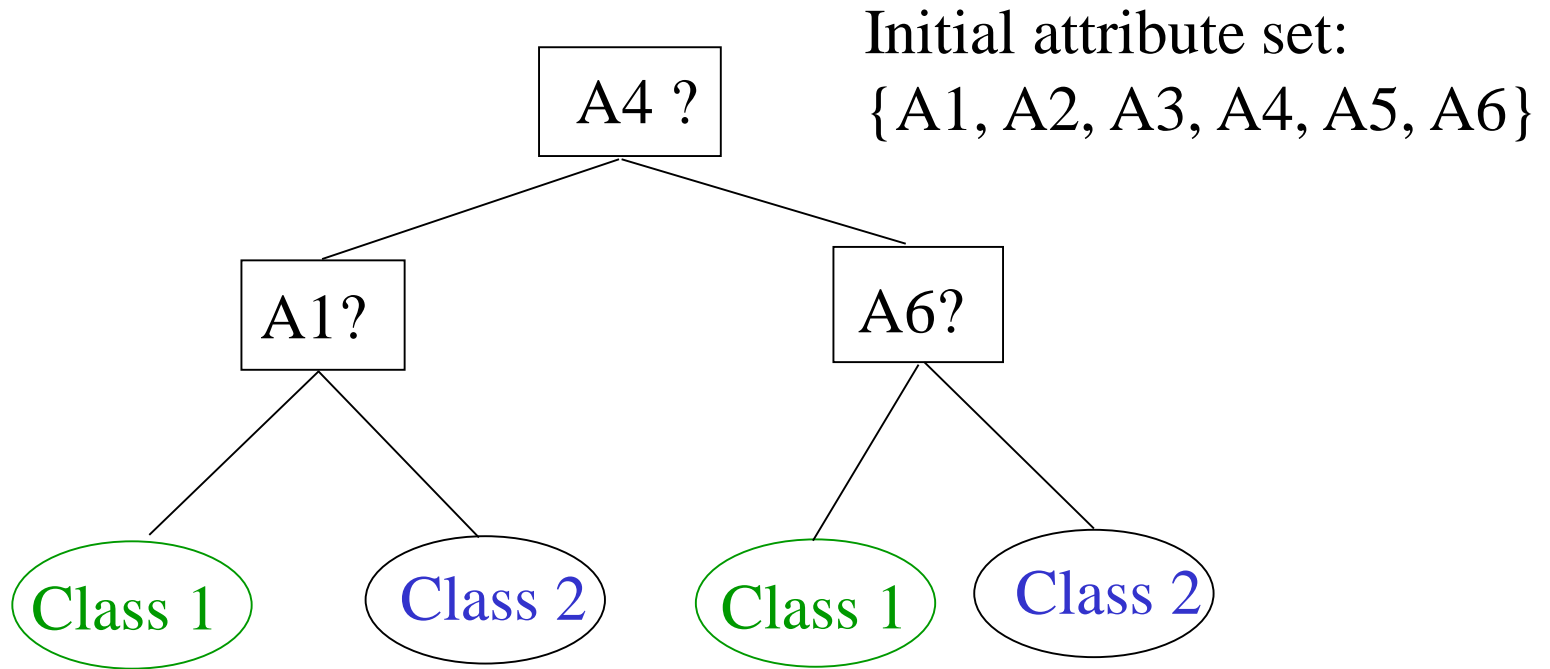
- **Problem:** Feature selection (i.e., **attribute subset selection**):
 - Select a **minimum set of features** such that the probability distribution of different classes given the values for those features is as close as possible to the original distribution given the values of all features
 - **Nice side-effect:** reduces # of attributes in the discovered patterns (which are now easier to understand)
- **Solution:** Heuristic methods (due to exponential # of choices) usually greedy:
 - step-wise forward selection
 - step-wise backward elimination
 - combining forward selection and backward elimination
 - decision-tree induction

Example of Decision Tree Induction

nonleaf nodes: tests

branches: outcomes of tests

leaf nodes: class prediction

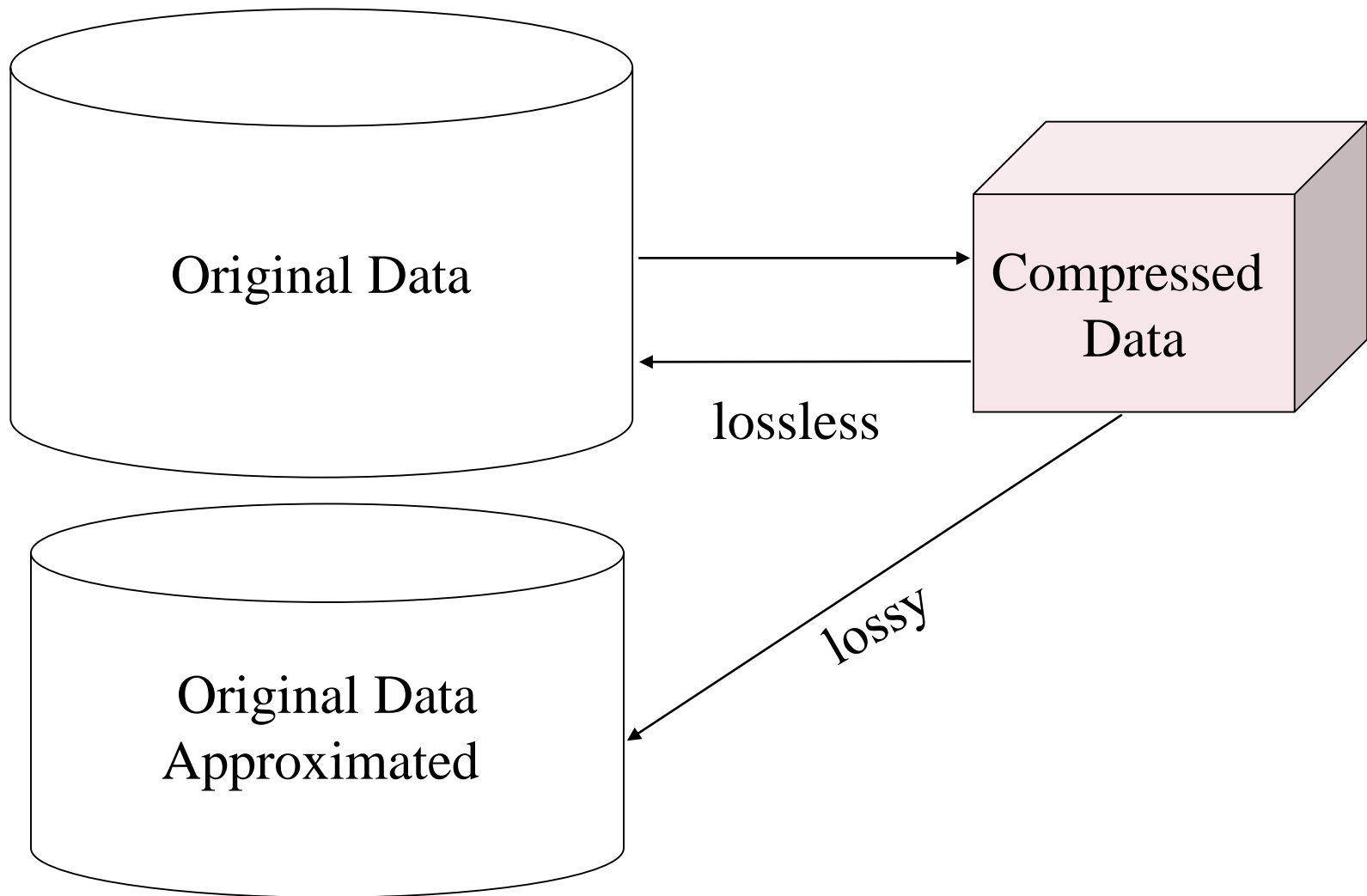


-----> Reduced attribute set: $\{A1, A4, A6\}$

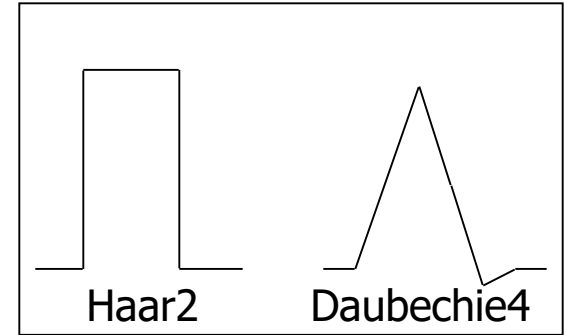
Data Compression

- String compression
 - There are extensive theories and well-tuned algorithms
 - Typically lossless
 - But only limited manipulation is possible without expansion
- Audio/video, image 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 vary slowly with time

Data Compression



Wavelet Transforms



- Discrete wavelet transform (DWT):
linear signal processing
- Compressed approximation: store only a small fraction of the strongest of the wavelet coefficients
- Similar to discrete Fourier transform (DFT), but better lossy compression, localized in space (conserves local details)
- Method (hierarchical pyramid algorithm):
 - Length, L , must be an integer power of 2 (padding with 0s, when necessary)
 - Each transform has 2 functions:
 - smoothing (e.g., sum, weighted avg.), weighted difference
 - Applies to pairs of data, resulting in two sets of data of length $L/2$
 - Applies the two functions recursively, until reaches the desired length

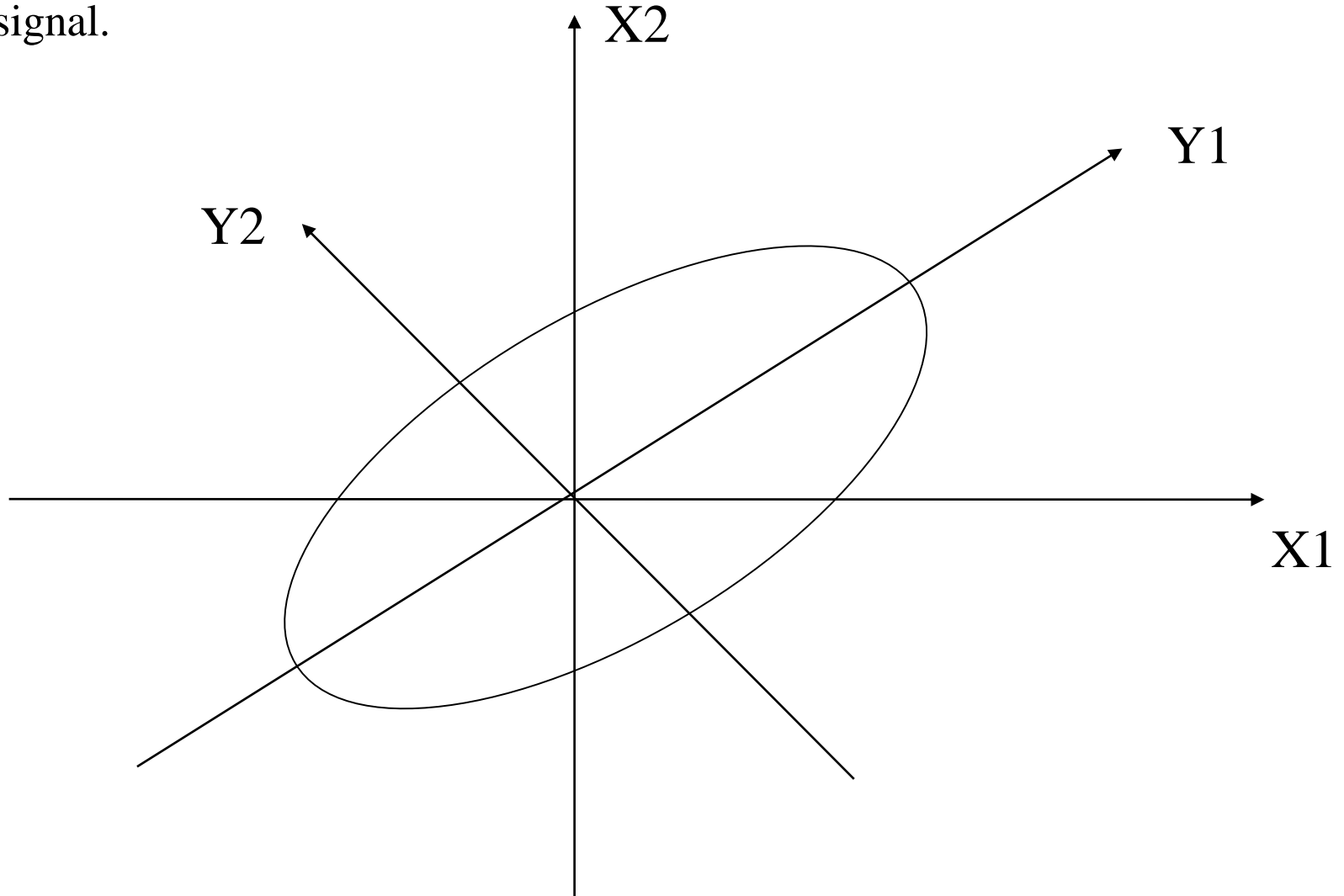
Principal Component Analysis (PCA)

Karhunen-Loeve (K-L) method

- Given N data vectors from k -dimensions, find $c \leq k$ orthogonal vectors that can be best used to represent data
 - The original data set is reduced (projected) to one consisting of N data vectors on c principal components (reduced dimensions)
- Each data vector is a linear combination of the c principal component vectors
- Works for ordered and unordered attributes
- Used when the number of dimensions is large

Principal Component Analysis

- The principal components (new set of axes) give important information about variance.
- Using the strongest components one can reconstruct a good approximation of the original signal.



Numerosity Reduction

- Parametric methods

- Assume the data fits some model, estimate model parameters, store only the parameters, and discard the data (except possible outliers)
- E.g.: Log-linear models: obtain value at a point in m -D space as the product on appropriate marginal subspaces

- Non-parametric methods

- Do not assume models
- Major families: histograms, clustering, sampling

Regression and Log-Linear Models

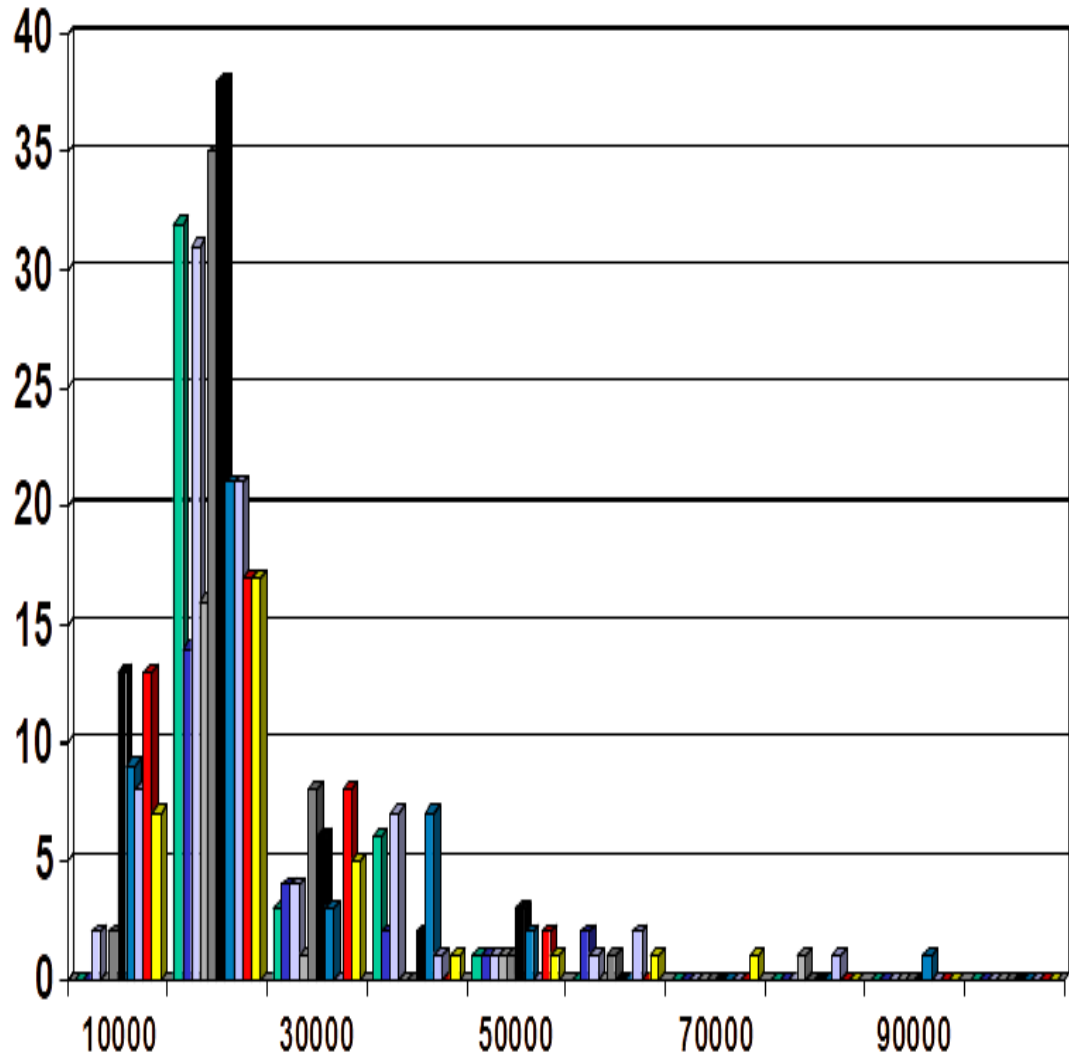
- **Linear regression:** Data are modeled to fit a straight line:
 - Often uses the least-square method to fit the line
- **Multiple regression:** allows a response variable y to be modeled as a linear function of multidimensional feature vector (predictor variables)
- **Log-linear model:** approximates discrete multidimensional joint probability distributions

Regression Analysis and Log-Linear Models

- **Linear regression:** $Y = \alpha + \beta X$
 - Two parameters , α and β specify the line and are to be estimated by using the data at hand.
 - using the least squares criterion to the known values of $Y_1, Y_2, \dots, X_1, X_2, \dots$
- **Multiple regression:** $Y = b_0 + b_1 X_1 + b_2 X_2$.
 - Many nonlinear functions can be transformed into the above.
- **Log-linear models:**
 - The multi-way table of joint probabilities is approximated by a product of lower-order tables.
 - Probability: $p(a, b, c, d) = \alpha_{ab} \beta_{ac} \chi_{ad} \delta_{bcd}$

Histograms

- Approximate data distributions
- Divide data into buckets and store average (sum) for each bucket
- A bucket represents an attribute-value/frequency pair
- Can be constructed optimally in one dimension using dynamic programming
- Related to quantization problems.



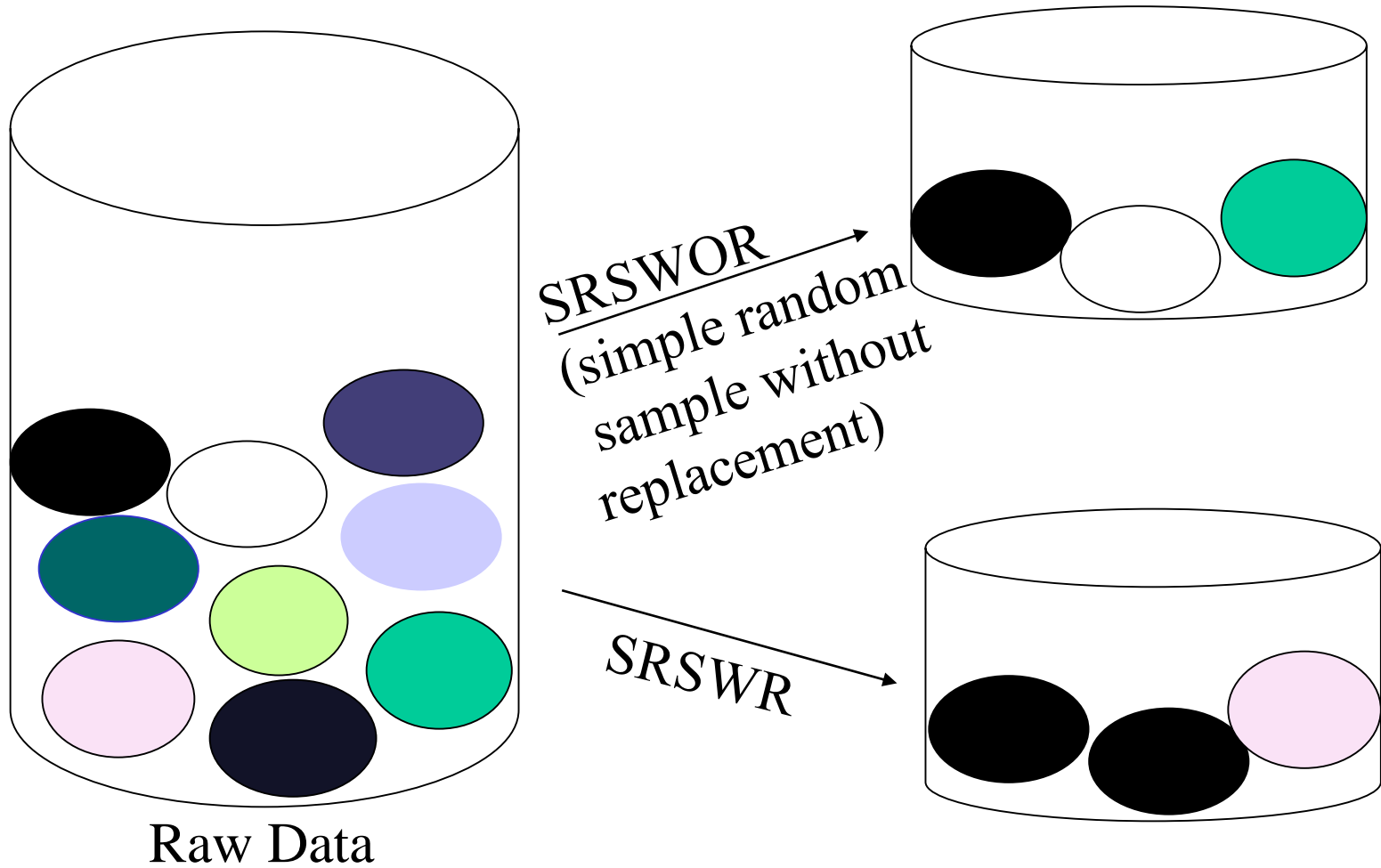
Clustering

- Partition data set into clusters, and store cluster representation only
- **Quality of clusters** measured by their **diameter** (max distance between any two objects in the cluster) or **centroid distance** (avg. distance of each cluster object from its centroid)
- Can be very effective if data is clustered but not if data is “smeared”
- Can have hierarchical clustering (possibly stored in multi-dimensional index tree structures (B+-tree, R-tree, quad-tree, etc))
- There are many choices of clustering definitions and clustering algorithms (further details later)

Sampling

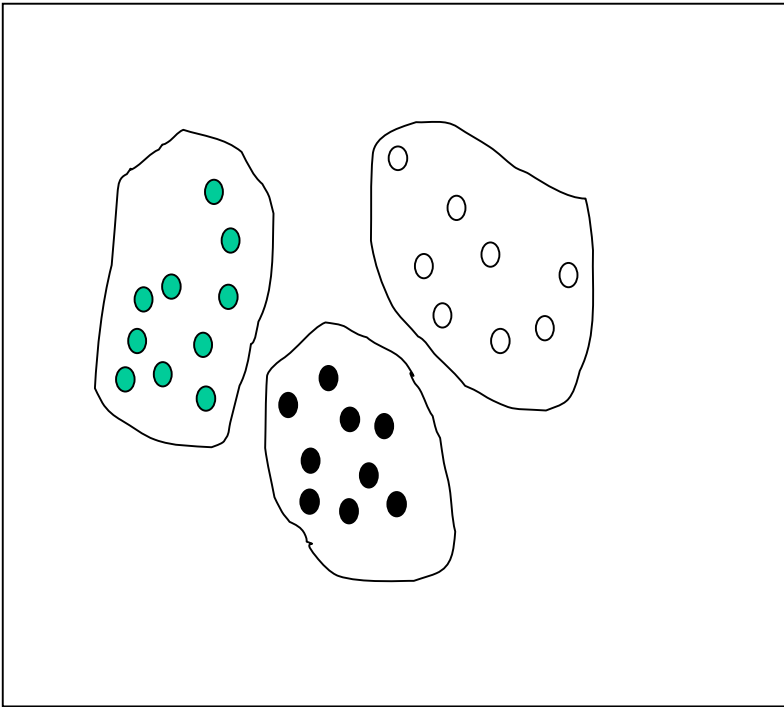
- Allow a mining algorithm to run in complexity that is potentially sub-linear to the size of the data
- Cost of sampling: proportional to the size of the sample, increases linearly with the number of dimensions
- Choose a **representative** subset of the data
 - Simple random sampling may have very poor performance in the presence of skew
- Develop adaptive sampling methods
 - Stratified sampling:
 - Approximate the percentage of each class (or subpopulation of interest) in the overall database
 - Used in conjunction with skewed data
- Sampling may not reduce database I/Os (page at a time).
- Sampling: natural choice for progressive refinement of a reduced data set.

Sampling

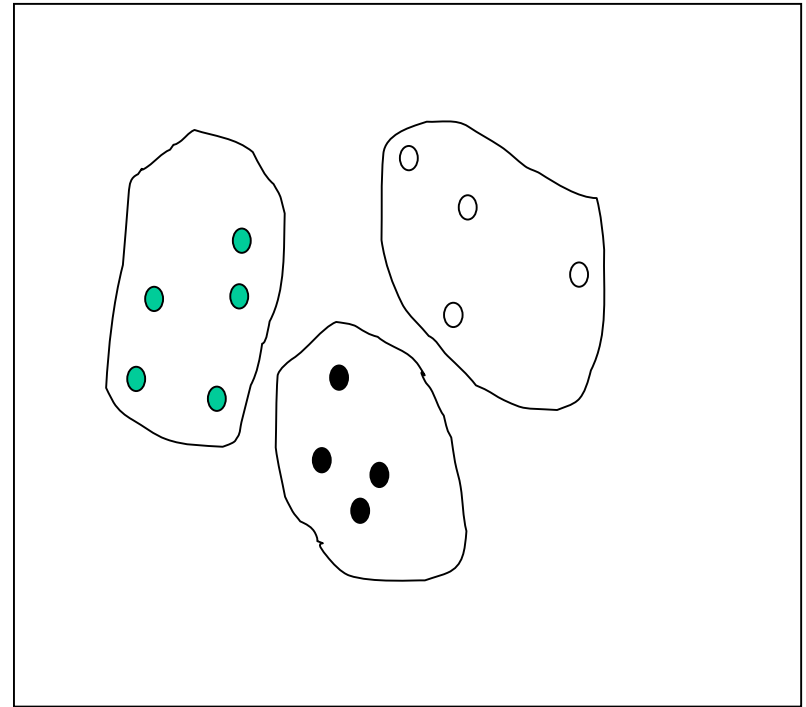


Sampling

Raw Data



Cluster/Stratified Sample



Hierarchical Reduction

- Use multi-resolution structure with different degrees of reduction
- Hierarchical clustering is often performed but tends to define partitions of data sets rather than “clusters”
- Parametric methods are usually not amenable to hierarchical representation
- Hierarchical aggregation
 - An index tree hierarchically divides a data set into partitions by value range of some attributes
 - Each partition can be considered as a bucket
 - Thus an index tree with aggregates stored at each node is a hierarchical histogram

Agenda

- Why preprocess the data?
- Data cleaning
- Data integration and transformation
- Data reduction
- Discretization and concept hierarchy generation
- Summary

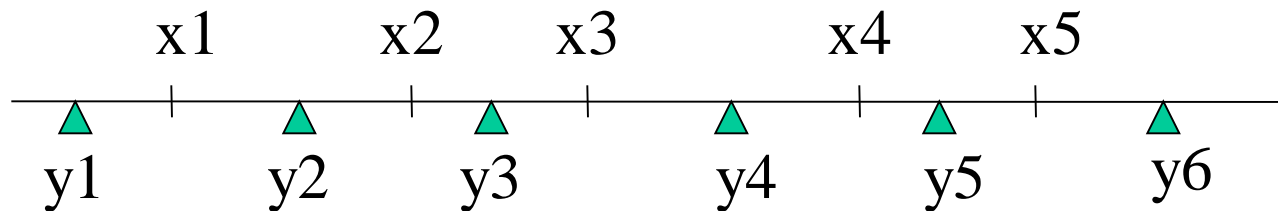
Discretization/Quantization

- Three types of attributes:

- Nominal — values from an unordered set
- Ordinal — values from an ordered set
- Continuous — real numbers

- Discretization/Quantization:

✉ divide the range of a continuous attribute into intervals



- Some classification algorithms only accept categorical attributes.
- Reduce data size by discretization
- Prepare for further analysis

Discretization and Concept Hierarchy

- Discretization
 - reduce the number of values for a given continuous attribute by dividing the range of the attribute into intervals. Interval labels can then be used to replace actual data values.
- Concept Hierarchies
 - reduce the data by collecting and replacing low level concepts (such as numeric values for the attribute age) by higher level concepts (such as young, middle-aged, or senior).

Discretization and concept hierarchy generation for numeric data

- Hierarchical and recursive decomposition using:
 - Binning (data smoothing)
 - Histogram analysis (numerosity reduction)
 - Clustering analysis (numerosity reduction)
- Entropy-based discretization
- Segmentation by natural partitioning

Entropy-Based Discretization

- Given a set of samples S , if S is partitioned into two intervals S_1 and S_2 using threshold T on the value of attribute A , the information gain resulting from the partitioning is:

$$I(S, T) = \frac{|S_1|}{|S|} E(S_1) + \frac{|S_2|}{|S|} E(S_2)$$

where the entropy function E for a given set is calculated based on the class distribution of the samples in the set. Given m classes the entropy of S_1 is:

$$E(S_1) = - \sum_{i=1}^m p_i \log_2(p_i)$$

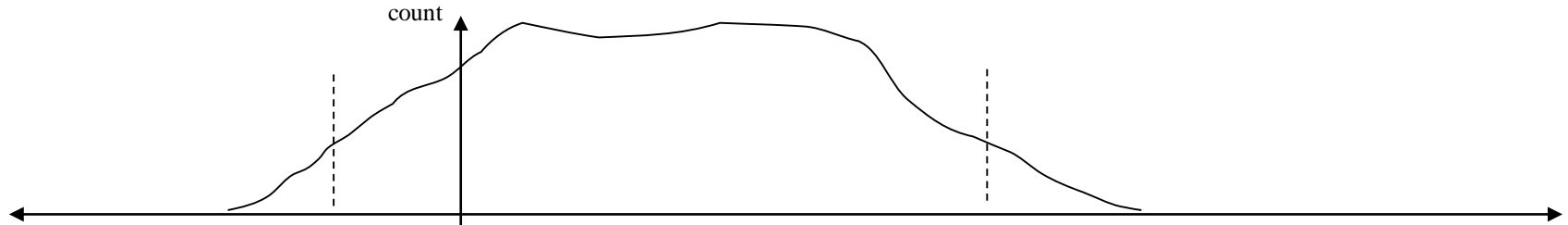
where p_i is the probability of class i in S_1 .

- The threshold that maximizes the information gain over all possible thresholds is selected as a binary discretization.
- The process is recursively applied to partitions obtained until some stopping criterion is met, e.g., $E(S) - I(S, T) < \delta$
- Experiments show that it may reduce data size and improve classification accuracy

Segmentation by natural partitioning

- 3-4-5 rule can be used to segment numeric data into relatively uniform, “natural” intervals.
- It partitions a given range into 3,4, or 5 equiwidth intervals recursively level-by-level based on the value range of the most significant digit.
 - * If an interval covers 3, 6, 7 or 9 distinct values at the most significant digit, partition the range into 3 equi-width intervals
 - * If it covers 2, 4, or 8 distinct values at the most significant digit, partition the range into 4 intervals
 - * If it covers 1, 5, or 10 distinct values at the most significant digit, partition the range into 5 intervals


Example of 3-4-5 rule



Step 1:	-\$351	-\$159	profit	\$1,838	\$4,700
	Min	Low (i.e, 5%-tile)		High(i.e, 95%-0 tile)	Max

Step 2:	msd=1,000	Low=-\$1,000	High=\$2,000
			(-\$1,000 - \$2,000)

Step 3:



```
graph TD; A["(-$1,000 - $2,000)"] --- B["(-$1,000 - 0)"]; A --- C["(0 - $1,000)"]; A --- D["($1,000 - $2,000)"]
```

Step 4:

Diagram illustrating a decision tree for Step 4, showing the progression of values from the root node to terminal nodes.

Root Node: $(-\$4000 - \$5,000)$

Branches from Root:

- Node 1: $(-\$400 - 0)$
 - Node 1.1: $(-\$400 - -\$300)$
 - Node 1.2: $(-\$300 - -\$200)$
 - Node 1.3: $(-\$200 - -\$100)$
 - Node 1.4: $(-\$100 - 0)$
- Node 2: $(0 - \$1,000)$
 - Node 2.1: $(0 - \$200)$
 - Node 2.2: $(\$200 - \$400)$
 - Node 2.3: $(\$400 - \$600)$
 - Node 2.4: $(\$600 - \$800)$
 - Node 2.5: $(\$800 - \$1,000)$
- Node 3: $(\$1,000 - \$2,000)$
 - Node 3.1: $(\$1,000 - \$1,200)$
 - Node 3.2: $(\$1,200 - \$1,400)$
 - Node 3.3: $(\$1,400 - \$1,600)$
 - Node 3.4: $(\$1,600 - \$1,800)$
 - Node 3.5: $(\$1,800 - \$2,000)$
- Node 4: $(\$2,000 - \$5,000)$
 - Node 4.1: $(\$2,000 - \$3,000)$
 - Node 4.2: $(\$3,000 - \$4,000)$
 - Node 4.3: $(\$4,000 - \$5,000)$

Concept hierarchy generation for categorical data

- Categorical data: no ordering among values
- Specification of a partial ordering of attributes explicitly at the schema level by users or experts
- Specification of a portion of a hierarchy by explicit data grouping
- Specification of a set of attributes, but not of their partial ordering
- Specification of only a partial set of attributes

Concept hierarchy generation w/o data semantics - Specification of a set of attributes

Concept hierarchy can be automatically generated based on the number of distinct values per attribute in the given attribute set. The attribute with the most distinct values is placed at the lowest level of the hierarchy (**limitations?**)



Agenda

- Why preprocess the data?
- Data cleaning
- Data integration and transformation
- Data reduction
- Discretization and concept hierarchy generation
- Summary

Summary

- Data preparation is a big issue for both warehousing and mining
- Data preparation includes
 - Data cleaning and data integration
 - Data reduction and feature selection
 - Discretization
- A lot a methods have been developed but still an active area of research