

Data Pre-processing

Contents

- Data Cleaning
- Data Integration and Transformation
- Data Reduction
- Data discretization
- Concept hierarchy generation

Data Pre-processing

- **Real-world databases** are highly susceptible to *noisy, missing, and inconsistent data due to their typically huge size* (often several gigabytes or more) and their likely *origin from multiple, heterogenous sources*.
- Low-quality data will lead to low-quality mining results.
- *How can the data be pre-processed in order to help improve the quality of the data and, consequently, of the mining results?*
- *How can the data be pre-processed so as to improve the efficiency and ease of the mining process?*

Data Quality

- What kinds of data quality problems?
- How can we detect problems with the data?
- What can we do about these problems?
- Examples of data quality problems:
 - Noise and outliers
 - Noise: random error or variance in a measured variable
 - Outliers are data objects with characteristics that are considerably different than most of the other data objects in the data set
 - Missing values
 - Duplicate data

Data Quality- Missing Values and Duplicate Data

- Reasons for missing values
 - Information is not collected
(e.g., people decline to give their age and weight)
 - Attributes may not be applicable to all cases
(e.g., annual income is not applicable to children)
- Handling missing values
 - Eliminate Data Objects
 - Estimate Missing Values
 - Ignore the Missing Value During Analysis
 - Replace with all possible values (weighted by their probabilities)
- Data set may include data objects that are **duplicates**, or almost duplicates of one another
- Major issue when merging data from heterogenous sources

Data Quality: Why Preprocess the Data?

Data have quality if they satisfy the requirements of the intended use.

Accuracy: correct or wrong, accurate or not

- There are many possible reasons for inaccurate data.
 - Human or computer **errors occurring at data entry**.
 - Users may **purposely submit incorrect data values for mandatory fields when they do not wish to submit personal information** such as choosing the default value “January 1” displayed for birthday.
 - Incorrect data may also result **from inconsistencies in naming conventions or data codes**, or inconsistent formats for input fields (e.g., date).

Completeness: not recorded, unavailable, ...

- Attributes of interest may not always be available
- Data may not be included simply because they were **not considered important at the time of entry**.
- Relevant data **may not be recorded due to a misunderstanding**.
- Missing data, tuples with missing values for some attributes, may need to be inferred.

Data Quality: Why Preprocess the Data?

Consistency: some modified but some not, dangling, ...

- **Inconsistencies in data codes, or inconsistent formats** for input fields (e.g., *date*).
- Eg: discrepancies in the department codes used to categorize items.

Timeliness: timely update?

- Is the data is **timely updated**?

Believability: how trustable is the data? Is it correct?

- How much the data are trusted by users?
- **The past errors can effect the trustability** of the data.

Interpretability: **how easily** the data can be **understood**?

Major Tasks in Data Preprocessing

- **Data cleaning** can be applied to **remove noise and correct inconsistencies** in the data.
 - ❖ Fill in missing values, smooth noisy data, identify or remove outliers, and resolve inconsistencies
- **Data integration** merges **data from multiple sources into a coherent data store**, such as a data warehouse.
 - ❖ Integration of multiple databases, data cubes, or files.
- **Data transformations - Data are scaled** to fall within a smaller range like 0.0 to 1.0.
 - **Normalization , Data Discretization and Concept hierarchy generation** are forms of data transformations.
 - For example, normalization may **improve the accuracy and efficiency** of mining algorithms involving distance measurements.
- **Data reduction** obtains a **reduced representation of the data set that is much smaller in volume**, yet produces the same (or almost the same) analytical results.
 - ❖ Can reduce the data size by **aggregating, eliminating redundant features, or clustering**.
 - ❖ Data aggregation, Attribute subset selection, Dimensionality reduction, Numerosity reduction, Data compression, by generalization with the use of Concept hierarchy
- **Data Discretization-** a form of data reduction that is very useful for **the automatic generation of concept hierarchies from numerical data**.

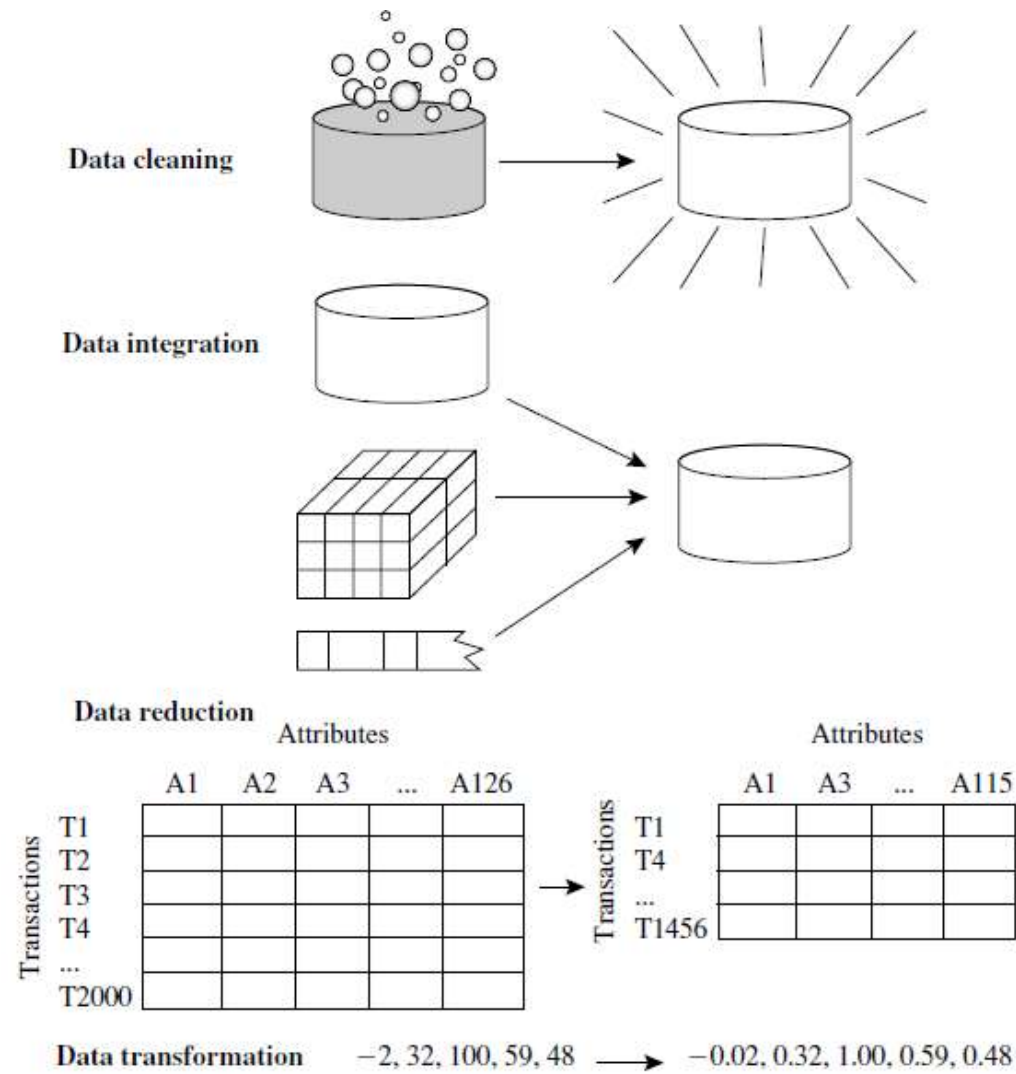
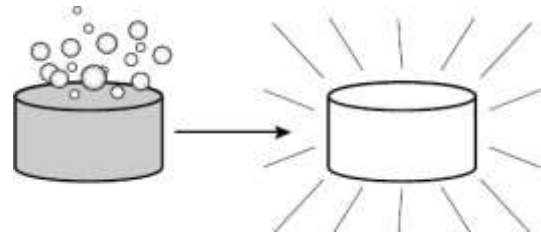


Figure 3.1 Forms of data preprocessing.

Data Cleaning

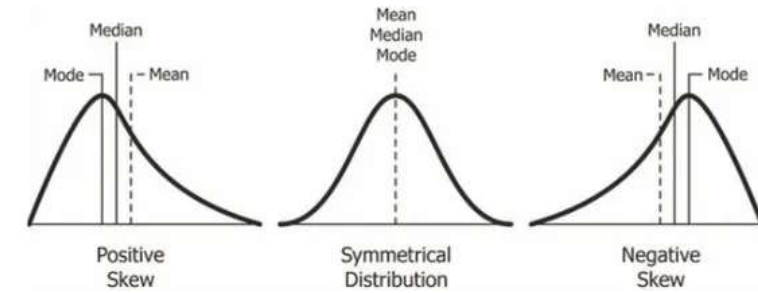
- **Data cleaning** routines work to “clean” the data by filling in missing values, smoothing noisy data, identifying or removing outliers, and resolving inconsistencies.



How to Handle Missing Data?

- **Ignore the tuple:** Usually done when the class label is missing (assuming the mining task involves **classification**).
- Not very effective, unless the tuple contains several attributes with missing values.
- Poor when the percentage of missing values per attribute varies considerably.
- By ignoring the tuple, **we do not make use of the remaining attributes' values** in the tuple which could have been useful to the task at hand.
- **Fill in the missing value manually:** **Time consuming** and may not be feasible given a large data set with many missing values.
- **Use a global constant to fill in the missing value:** **Replace** all missing attribute values **by the same constant** such as a label like “*Unknown*” or $-\infty$.
- If missing values are replaced by, say, “*Unknown*,” then the mining program may mistakenly think that they form an interesting concept, since they all have a value in common—that of “*Unknown*.”
- This method is simple, it is not foolproof.

How to Handle Missing Data?



- **Use a measure of central tendency for the attribute to fill in the missing value:**
 - For e.g., the **mean or median-** normal (symmetric) data distributions, the mean can be used, while skewed data distribution should employ the median.

For example, suppose that the data distribution regarding the income of *AllElectronics* customers is symmetric and that the mean income is \$56,000. Use this value to replace the missing value for *income*.
- **Use the attribute mean or median for all samples belonging to the same class as the given tuple:**
 - For example, if classifying customers according to *credit risk*, we may replace the missing value with the mean *income* value for customers in the same credit risk category as that of the given tuple. If the data distribution for a given class is skewed, the median value is a better choice.
- **Use the most probable value to fill in the missing value:**
- Determined with regression, inference-based tools using a Bayesian formalism, or decision tree induction.
- For example, using the other customer attributes in your data set, you may construct a decision tree to predict the missing values for *income*.

Noisy Data and How to Handle Noisy Data?

- **Noise:** random error or variance in a measured variable
- Outliers may represent noise.
- Given a numeric attribute such as, say, *price*, how can we “**smooth**” out the data to remove the noise?

Data Smoothing Techniques:

- **Binning**
- **Regression**
- **Clustering**
- **Combined computer and human inspection**

Noisy Data and How to Handle Noisy Data?

Data Smoothing Techniques:

- **Binning**
 - smooth a sorted data value by consulting its “neighborhood,” that is, the values around it.
 - sorted values are distributed into a number of “buckets,” or bins.
 - Because binning methods consult the neighborhood of values, they perform *local smoothing*.
 - First sort data and partition into (equal-frequency) bins
 - Then one can smooth by bin mean, smooth by bin median, smooth by bin boundaries, etc.

Binning Methods for Data Smoothing

- **Binning methods** smooth a sorted data by distributing them into bins (buckets).

Smoothing by bin means:

- Each value in a bin is replaced by the mean value of the bin.

Smoothing by bin medians:

- Each bin value is replaced by the bin median.

Smoothing by bin boundaries:

- The minimum and maximum values in a given bin are identified as the bin boundaries.
- Each bin value is then replaced by the closest boundary value.

Binning Methods for Data Smoothing: Example

- Sorted data for *price* (in dollars): 4, 8, 15, 21, 21, 24, 25, 28, 34
- **Partition into (equal-frequency) bins:**
 - Bin 1: 4, 8, 15
 - Bin 2: 21, 21, 24
 - Bin 3: 25, 28, 34
- **Smoothing by bin means:**
 - Bin 1: 9, 9, 9
 - Bin 2: 22, 22, 22
 - Bin 3: 29, 29, 29
- **Smoothing by bin medians:**
 - Bin 1: 8, 8, 8
 - Bin 2: 21, 21, 21
 - Bin 3: 28, 28, 28
- **Smoothing by bin boundaries:**
 - Bin 1: 4, 4, 15
 - Bin 2: 21, 21, 24
 - Bin 3: 25, 25, 34

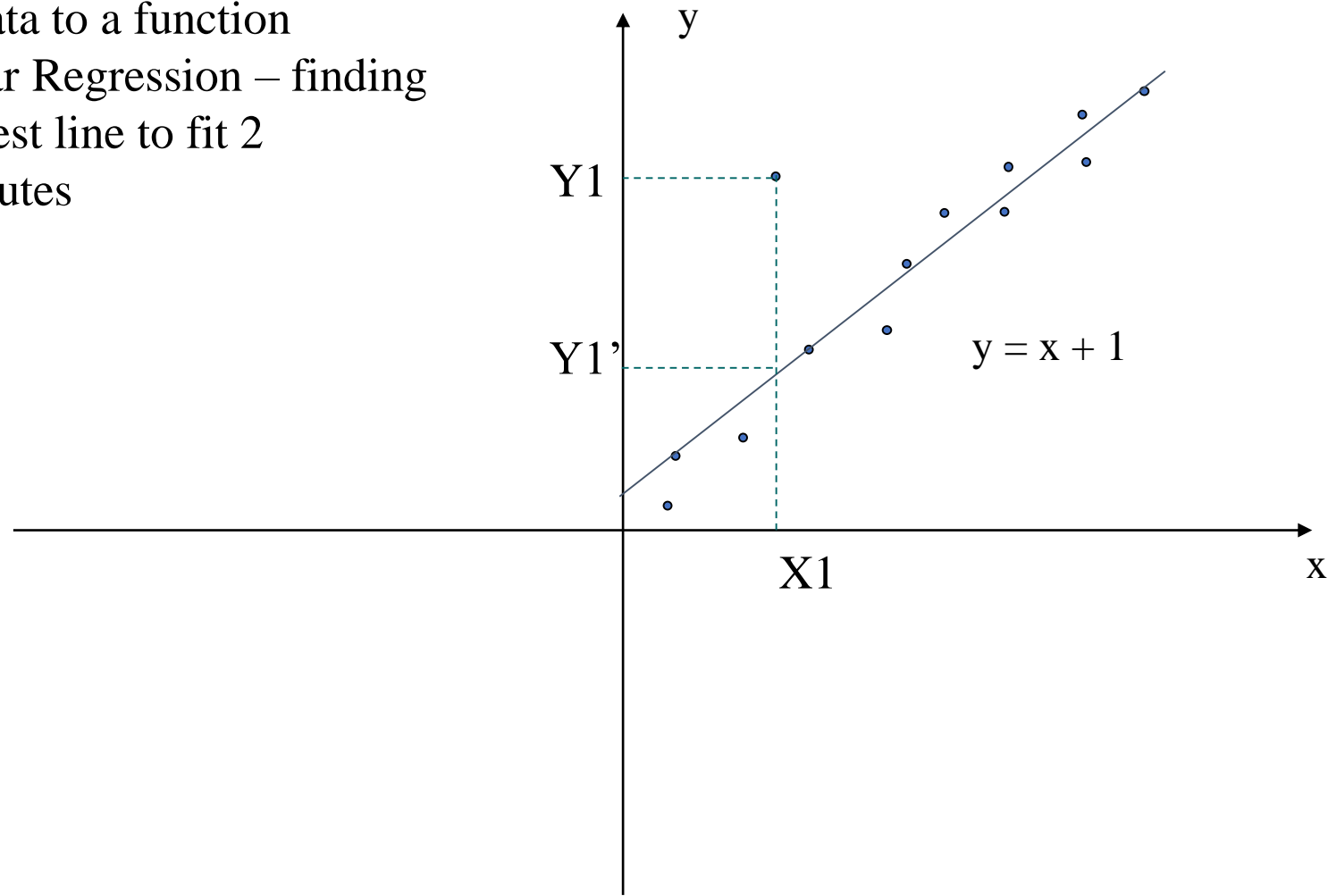
Noisy Data and How to Handle Noisy Data?

Data Smoothing Techniques:.

- **Regression**
 - Smooth by fitting the data into regression functions
 - A technique that conforms data values to a function.
 - **Linear regression** involves finding the “best” line to fit two attributes (or variables) so that one attribute can be used to predict the other.
 - **Multiple linear regression**
 - ❖ An extension of linear regression.
 - ❖ More than two attributes are involved and the data are fit to a multidimensional surface.
- **Clustering**
 - Similar values are organized into groups, or “clusters”.
 - detect and remove outliers
 - values that fall outside of the set of clusters may be considered outliers
- **Combined computer and human inspection**
 - detect suspicious values and check by human (e.g., deal with possible outliers)

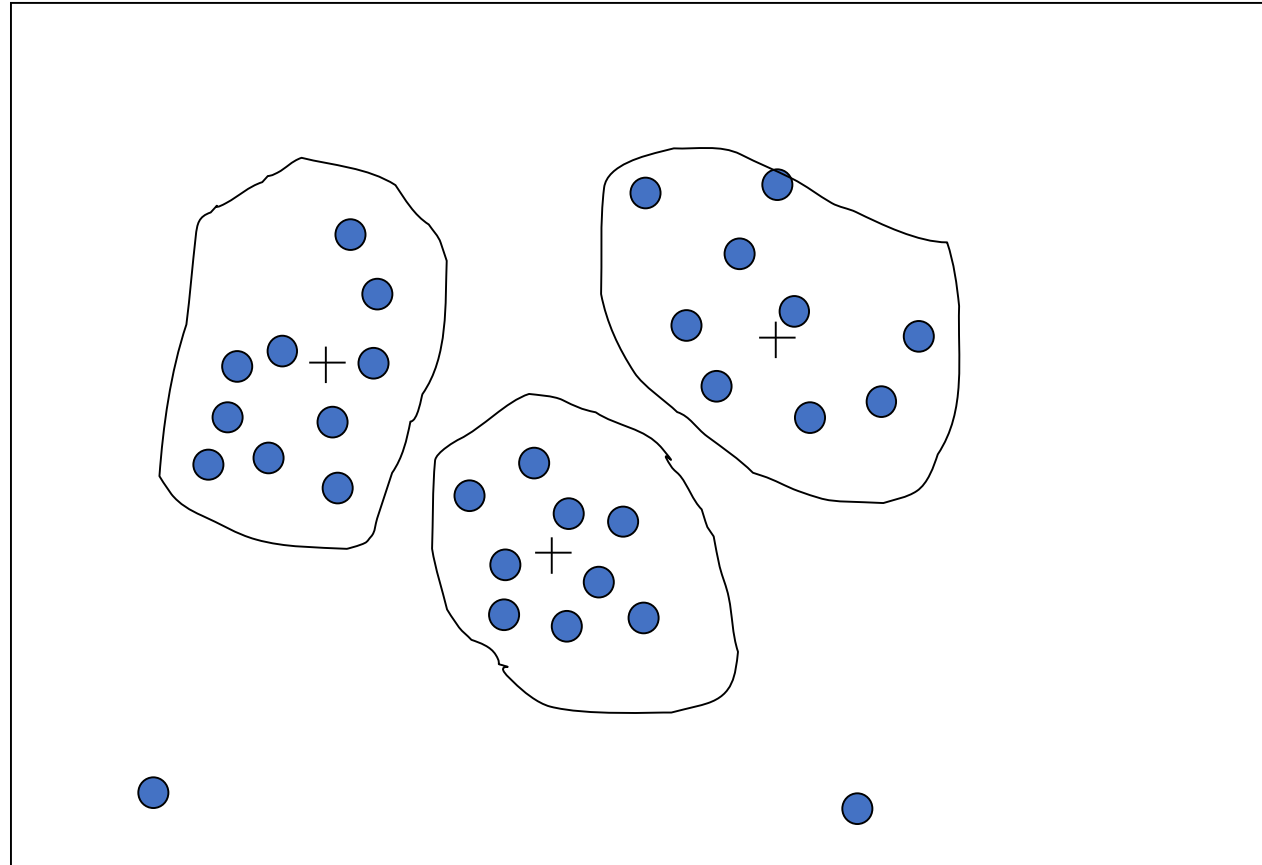
Regression

- Fit data to a function
- Linear Regression – finding the best line to fit 2 attributes



Cluster Analysis

- Outliers may be detected
- Similar values are organized into groups



Data Smoothing

- *Many methods for data smoothing are also methods for data reduction involving discretization.*
- **Eg1:** The **binning** techniques reduce the number of distinct values per attribute.
 - This acts as a form of data reduction for logic-based data mining methods, such as **decision tree induction**, which repeatedly make value comparisons on sorted data.
- **Eg2: Concept hierarchies** are a form of data discretization that can also be used for data smoothing.
 - A concept hierarchy for price, for example, may map real price values into **inexpensive**, **moderately priced**, and **expensive**, thereby reducing the number of data values to be handled by the mining process.

Data Cleaning as a Process

- **Data discrepancy detection**
 - Discrepancies can be caused by
 - Poorly designed data entry forms that have many optional fields
 - Human error in data entry
 - Deliberate errors (e.g., respondents not wanting to divulge information about themselves)
 - Data decay (e.g., outdated addresses).
 - Inconsistent data representations and inconsistent use of codes. (e.g., “2010/12/25” and “25/12/2010” for *date*).
 - **Field overloading** -Developers squeeze new attribute definitions into unused (bit) portions of already defined attributes.
 - Errors in instrumentation devices that record data and system errors.
 - When the data are (inadequately) used for purposes other than originally intended.
 - Inconsistencies due to data integration (e.g., where a given attribute can have different names in different databases)

Data Cleaning as a Process

Step 1: Data discrepancy detection

- Use metadata-Knowledge regarding properties of the data
(e.g., domain and datatype, range, dependency b/w attr., distribution)

For example, values that are more than two standard deviations away from the mean for a given attribute may be flagged as potential outliers.

- Check data regarding **uniqueness rule, consecutive rule and null rule**
 - **Unique rule** - Each value of the given attribute must be different from all other values for that attribute.
 - **Consecutive rule** - There can be no missing values between the lowest and highest values for the attribute, and that all values must also be unique (e.g., as in check numbers).
 - **Null rule** - specifies the use of blanks, question marks, special characters, or other strings that may indicate the null condition (e.g., where a value for a given attribute is not available), and how such values should be handled.

Data Cleaning as a Process

- **Data discrepancy Correction**

- 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 relationship** to detect data that violate such conditions (e.g., **correlation and clustering** to find outliers)

- Data inconsistencies may **be corrected manually** using external references.

- For example, errors made at data entry may be corrected by performing a paper traces.

Data Cleaning as a Process

Step 2: Data Transformation

- Data migration and integration
 - **Data migration tools**: allow simple transformations to be specified Eg. Replacing 'gender' by 'sex'.
 - **ETL** (Extraction/Transformation/Loading) tools: allow users to specify transformations through a graphical user interface.
 - Support only a **restricted set of transforms**.
- Integration of the two processes
 - 2 step process is **error prone and time consuming**.
 - Transformations **may introduce more errors**. (Eg. A typo '20004' in a year field may only surface when all the date values are converted to a uniform formats)
 - Interactivity increased using
 - **New data cleaning tools** (e.g., Potter's Wheels)-publicly available data cleaning tool- integrates **discrepancy detection and transformation**.
 - **Using declarative language** to specify data transformation operators.

- Data Quality and Major Tasks in Data Preprocessing
- Data Cleaning
- **Data Integration**
- Data Transformation
- Data Reduction
- Data Discretization

Data Integration

- Data integration:
 - Combines data from multiple sources into a coherent store
- Issues in integration
 - Schema integration and object matching can be tricky
 - Integrate data from different sources
 - Result- Entity identification problem
 - Eg: *customer_id* in one database and *cust _number* in another.
 - Detection and resolution of data value conflicts
 - For the same real world entity, attribute values from different sources are different Eg. Pay_type in one database be “H” and “S” and “1” and “2” in another
 - Possible reasons for conflicts: different representations, different scales, e.g., Weight stored in different units
 - Redundancy
 - Duplicate tuples
 - Denormalized tables

Handling Redundancy in Data Integration

- **Redundancy** is another important issue in data integration.
- An attribute may be **redundant** if it can be “derived” from another attribute or set of attributes.
- Inconsistencies in attribute or dimension naming can also cause redundancies in the resulting data set after data integration.
- Redundant data occur often after integration of multiple databases
 - *Object 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.
- Careful integration of the data from multiple sources may help reduce/avoid redundancies and inconsistencies and improve mining speed and quality.
- Redundant attributes may be able to be detected by **correlation analysis**.
 - χ^2 (*chi-square*) test for nominal attributes.
 - **correlation coefficient** and **covariance** for numeric attributes

Correlation Analysis (Numerical Data)

- For numeric attributes, we can evaluate the *correlation between two attributes*, A and B, by computing the **correlation coefficient**(also called **Pearson's product moment coefficient**).

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

where n is the number of tuples, \bar{A} and \bar{B} are the respective means of A and B, σ_A and σ_B are the respective standard deviation of A and B, and $\sum(AB)$ is the sum of the AB cross-product.

- Note that $-1 \leq r_{A,B} \leq 1$
- If $r_{A,B} > 0$: A and B are **positively correlated** (A's values increase as B's).

The higher, the stronger correlation.

- $r_{A,B} = 0$: **independent**;
- $r_{A,B} < 0$: **negatively correlated** (A's values increase as B's decreases-one discourages the other).

Correlation Analysis (for Numeric Data)

Covariance

- The **mean values** of A and B, are also known as the **expected values** on A and B.

$$E(A) = \bar{A} = \frac{\sum_{i=1}^n a_i}{n} \qquad E(B) = \bar{B} = \frac{\sum_{i=1}^n b_i}{n}$$

- The **covariance** between A and B is defined as:

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

- Covariance** is similar to **correlation coefficient**:

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

- It can also be shown that:

$$Cov(A, B) = E(A \cdot B) - \bar{A}\bar{B}$$

- This equation simplifies the calculation of $Cov(A, B)$.

Covariance: Example

- Suppose two stocks A and B have the following values in one week:
(2, 5), (3, 8), (5, 10), (4, 11), (7, 14).
- Question: If the stocks are affected by the same industry trends, will their prices rise or fall together?
 - $E(A) = (2 + 3 + 5 + 4 + 7) / 5 = 21/5 = 4.2$
 - $E(B) = (5 + 8 + 10 + 11 + 14) / 5 = 48/5 = 9.6$
 - $\text{Cov}(A,B) = (2 \times 5 + 3 \times 8 + 5 \times 10 + 4 \times 11 + 7 \times 14) / 5 - 4.2 \times 9.6 = 4.88$
- Thus, A and B rise together since $\text{Cov}(A, B) > 0$.

χ^2 (Chi-Square) Test

Correlation Test (for Nominal Data)

- For **nominal data**, a *correlation relationship* between two attributes, A and B, can be discovered by a χ^2 (**chi-square**) test.
- Suppose A has c distinct values, $a_1 \dots a_c$. B has r distinct values, $b_1 \dots b_r$.

χ^2 (chi-square) Test:

$$\chi^2 = \sum_{i=1}^c \sum_{j=1}^r \frac{(o_{ij} - e_{ij})^2}{e_{ij}}$$

where o_{ij} is the *observed frequency* (i.e., actual count) of the **joint event** (A_i, B_j) and e_{ij} is the *expected frequency* of (A_i, B_j) , which can be computed as

$$e_{ij} = \frac{\text{count}(A = a_i) \times \text{count}(B = b_j)}{n}$$

where n is the number of data tuples, $\text{count}(A=a_i)$ is the number of tuples having value a_i for A, and $\text{count}(B = b_j)$ is the number of tuples having value b_j for B.

- 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.

Chi-Square Calculation: An Example

- **Contingency Table** for two attributes **LikeScienceFiction** and **PlayChess**

	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(col.)	300	1200	1500

- Numbers in cells are *observed frequencies* (numbers in parenthesis are *expected counts* calculated based on the data distribution in the two categories).

$$e_{\text{LSF,PC}} = \text{count}(\text{LSF}) * \text{count}(\text{PC}) / n = 300 * 450 / 1500 = 90$$

- χ^2 (chi-square) calculation

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

Chi-Square Calculation: An Example

- For this 2x2 table, the degrees of freedom are $(2-1)(2-1) = 1$.
 - There are two possible values for *LikeScienceFiction* attribute and two possible values for *PlayChess* attribute.
- For 1 degree of freedom, the χ^2 value needed to reject the hypothesis at the 0.001 significance level is 10.83 (from table of upper percentage points of χ^2 distribution).

Degrees of Freedom (df)	Probability (p)										
	0.95	0.90	0.80	0.70	0.50	0.30	0.20	0.10	0.05	0.01	0.001
1	0.004	0.02	0.06	0.15	0.46	1.07	1.64	2.71	3.84	6.64	10.83
2	0.10	0.21	0.45	0.71	1.39	2.41	3.22	4.60	5.99	9.21	13.82
3	0.35	0.58	1.01	1.42	2.37	3.66	4.64	6.25	7.82	11.34	16.27
4	0.71	1.06	1.65	2.20	3.36	4.88	5.99	7.78	9.49	13.28	18.47
5	1.14	1.61	2.34	3.00	4.35	6.06	7.29	9.24	11.07	15.09	20.52
6	1.63	2.20	3.07	3.83	5.35	7.23	8.56	10.64	12.59	16.81	22.46
7	2.17	2.83	3.82	4.67	6.35	8.38	9.80	12.02	14.07	18.48	24.32
8	2.73	3.49	4.59	5.53	7.34	9.52	11.03	13.36	15.51	20.09	26.12
9	3.32	4.17	5.38	6.39	8.34	10.66	12.24	14.68	16.92	21.67	27.88
10	3.94	4.86	6.18	7.27	9.34	11.78	13.44	15.99	18.31	23.21	29.59
	Nonsignificant								Significant		

- Since our computed value 507.93 is above 10.83, we can reject the hypothesis that *LikeScienceFiction* and *PlayChess* are independent and conclude that the two attributes are (strongly) correlated for the given group of people.

- Data Quality and Major Tasks in Data Preprocessing
- Data Cleaning
- Data Integration
- **Data Transformation**
- Data Reduction
- Data Discretization

Data Transformation

- In **data transformation**, the data **are transformed or consolidated** into **forms appropriate for mining**.
- In data transformation, a function that **maps the entire set of values of a given attribute to a new set of replacement values** such that each old value can be identified with one of the new values.

Data transformation strategies

- **Smoothing:** Remove noise from the data.

Techniques -binning, regression, and clustering.

- **Attribute construction (or feature construction)**

New attributes are constructed and added from the given set of attributes. Eg: *area* from *height* and *width*

- **Aggregation** :Summary or aggregation operations are applied to the data.
- Typically used in constructing a *data cube* for data analysis at multiple abstraction levels.

E.g: the daily sales data may be aggregated so as to compute monthly and annual total amounts.

- **Normalization:** Attribute *data are scaled* so as to fall within a smaller range, such as -1.0 to 1.0, or 0.0 to 1.0.
- **Discretization** :Raw values of a *numeric attribute* (e.g., *age*) are *replaced by interval labels* (e.g., 0–10, 11–20, etc.) *or conceptual labels* (e.g., *youth*, *adult*, *senior*).
- The labels *can be* recursively *organized into higher-level concepts*, resulting in a **concept hierarchy** for the numeric attribute.
- **Concept hierarchy generation for nominal data-** where attributes such as *street* can be generalized to higher-level concepts, like *city* or *country*.
- Many hierarchies for nominal attributes are implicit within the database schema and can be automatically defined at the schema definition level.

Normalization

- An attribute is **normalized** by **scaling** its values so that they **fall within a small specified range**.
- **A larger range of an attribute gives a greater effect (weight) to that attribute.**
 - This means that an attribute with a larger range can have **greater weight at data mining tasks** than an attribute with a smaller range.
- Normalizing the data attempts to give all attributes an equal weight.
 - Normalization is particularly **useful for classification algorithms** involving neural networks or distance measurements such as nearest-neighbor classification and clustering.

Some Normalization Methods:

- **Min-max normalization**
- **Z-score normalization**
- **Normalization by decimal scaling**

Min-Max Normalization

- **Min-max normalization** performs a **linear transformation** on the original data.
- Suppose that \min_A and \max_A are minimum and maximum values of an attribute A.
- **Min-max normalization** maps a value, v_i of an attribute A to v'_i in the range $[\text{new_min}_A, \text{new_max}_A]$ by computing:

$$v'_i = \frac{v_i - \min_A}{\max_A - \min_A} (\text{new_max}_A - \text{new_min}_A) + \text{new_min}_A$$

- Min-max normalization preserves the relationships among the original data values.
- It will encounter an “out-of-bounds” error if a future input case for normalization falls outside of the original data range for A.
- We can standardize the range of all the numerical attributes to $[0,1]$ by applying **min-max normalization** with $\text{newmin}=0$ and $\text{newmax}=1$ to all the numeric attributes.

Min-Max Normalization: Example

- Suppose that the range of the attribute *income* is \$12,000 to \$98,000. We want to normalize *income* to range [0.0, 1.0].

- Then \$73,000 is mapped to
$$v'_i = \frac{v_i - \min_A}{\max_A - \min_A} (\text{new_max}_A - \text{new_min}_A) + \text{new_min}_A$$

$$\text{newvalue}(73000) = \frac{73000 - 12000}{98000 - 12000} (1.0 - 0.0) + 0 = 0.716$$

- Suppose that the range of the attribute *income* is \$12,000 to \$98,000. We want to normalize *income* to range [1.0, 5.0].
- Then \$73,000 is mapped to

$$\text{newvalue}(73000) = \frac{73000 - 12000}{98000 - 12000} (5.0 - 1.0) + 1.0 = 3.864$$

Z-score Normalization

- In **z-score normalization** (or *zero-mean normalization*), the values for an attribute A are normalized based on the *mean and standard deviation of A* .
- A value v_i of attribute A is normalized to v'_i by computing

$$v'_i = \frac{v_i - \bar{A}}{\sigma_A}$$

where \bar{A} and σ_A are the mean and standard deviation of attribute A .

- **z-score normalization** is *useful when the actual minimum and maximum of an attribute are unknown*.
- Suppose that the mean and standard deviation of the values for the attribute *income* are \$54,000 and \$16,000. With z-score normalization a value of \$73,600 for *income*:

$$\text{newvalue}(73600) = \frac{73600 - 54000}{16000} = 1.225$$

Normalization by Decimal Scaling

- **Normalization by decimal scaling** normalizes by moving the decimal point of values of attribute A.
- The number of decimal points moved depends on the maximum absolute value of A.
- A value v_i of attribute A is normalized to v'_i by computing

$$v'_i = \frac{v_i}{10^j}$$

where j is the smallest integer such that $\max(|v'_i|) < 1$.

Example:

- Suppose that the recorded values of A range from -986 to 917.
- The maximum absolute value of A is 986.
- To normalize by decimal scaling, we therefore divide each value by 1000 so that -986 normalizes to -0.986 and 917 normalizes to 0.917.

Data Reduction

- **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) analytical 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:** e.g., remove unimportant attributes
 - Wavelet transforms
 - Principal Components Analysis (PCA)
 - Feature subset selection, feature creation
 - **Numerosity reduction:**
 - Regression and log-linear Models
 - Histogram
 - Clustering
 - Sampling
 -

Data Reduction

- **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) analytical 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:**
 - Data cube aggregation
 - Attribute subset selection
 - Dimensionality reduction e.g., remove unimportant attributes
 - Wavelet transforms
 - Principal Components Analysis (PCA)
 - Numerosity reduction
 - Regression and log-linear Models
 - Histogram
 - Clustering
 - Sampling
 - Discretization and concept hierarchy generation

Data Reduction

Data reduction strategies:

Data cube aggregation

- Aggregation operations are applied to the data in the construction of a data cube.

Attribute subset selection

- Irrelevant, weakly relevant, or redundant attributes or dimensions may be detected and removed.

Dimensionality reduction

- Encoding mechanisms are used to reduce the data set size. e.g., remove unimportant attributes

Numerosity reduction

- The data are replaced or estimated by alternative, smaller data representations
- Parametric models (which need store only the model parameters instead of the actual data)
- Nonparametric methods such as clustering, sampling, and the use of histograms

Discretization and concept hierarchy generation

- Raw data values for attributes are replaced by ranges or higher conceptual levels.
- A form of numerosity reduction that is very useful for the automatic generation of concept hierarchies.
- Discretization and concept hierarchy generation are powerful tools for data mining, in that they allow the mining of data at multiple levels of abstraction

Data Cube Aggregation

- If the data has sales per quarters, and we are interested in annual sales
- The **data can be aggregated** so that **the resulting data summarize** the total sales per year instead of per quarter.
- The resulting data set is **smaller in volume**, without loss of information necessary for the analysis task.

Year 2004	
Quarter	Sales
Q1	\$224,000
Q2	\$408,000
Q3	\$350,000
Q4	\$586,000

Year 2003	
Quarter	Sales
Q1	\$224,000
Q2	\$408,000
Q3	\$350,000
Q4	\$586,000

Year 2002	
Quarter	Sales
Q1	\$224,000
Q2	\$408,000
Q3	\$350,000
Q4	\$586,000

Year	Sales
2002	\$1,568,000
2003	\$2,356,000
2004	\$3,594,000

sales per quarter are aggregated to provide the annual sales.

Attribute Subset Selection

- Data sets for analysis may contain hundreds of attributes, many of which may be **irrelevant** to the mining task or **redundant**.
- **Redundant Attributes** **duplicate** much or all of the **information** contained in one or more other attributes.
 - **Eg: Customer Age and Date of Birth:** Both attributes provide information about the age of the customer.
- **Irrelevant Attributes** contain almost **no useful information** for the data mining task.
 - Students' IDs are irrelevant to predict students' grade.
- **Attribute Subset Selection** reduces the data set size by **removing irrelevant or redundant attribute**.
 - **Goal** - to find a minimum set of attributes such that the resulting probability distribution of the data classes is as close as possible to the original distribution obtained using all attributes.
 - Attribute subset selection **reduces the number of attributes appearing in the discovered patterns**, helping to make the patterns easier to understand.

Attribute Subset Selection

- How can we find a ‘good’ subset of the original attributes?
- There are 2^n possible attribute combinations of n attributes
- Typical heuristic attribute selection methods used(reduce search space, typically *greedy, i.e; they always make what looks to be the best choice at the time*)
- The “best” (and “worst”) attributes are typically determined using tests of statistical significance, which assume that the attributes are independent of one another.

Heuristic Search in Attribute Subset Selection

1. Stepwise forward selection:

- Starts with an empty set of attributes as the reduced set.
- The best of the original attributes is determined and added to the reduced set.
- At each subsequent iteration or step, the best of the remaining original attributes is added to the set.

2. Stepwise backward elimination:

- Starts with the full set of attributes.
- At each step, it removes the worst attribute remaining in the set.

3. Combination of forward selection and backward elimination:

- Combine 1 and 2
- Each iteration selects the best attribute and removes the worst from among the remaining attributes.

Dimensionality Reduction

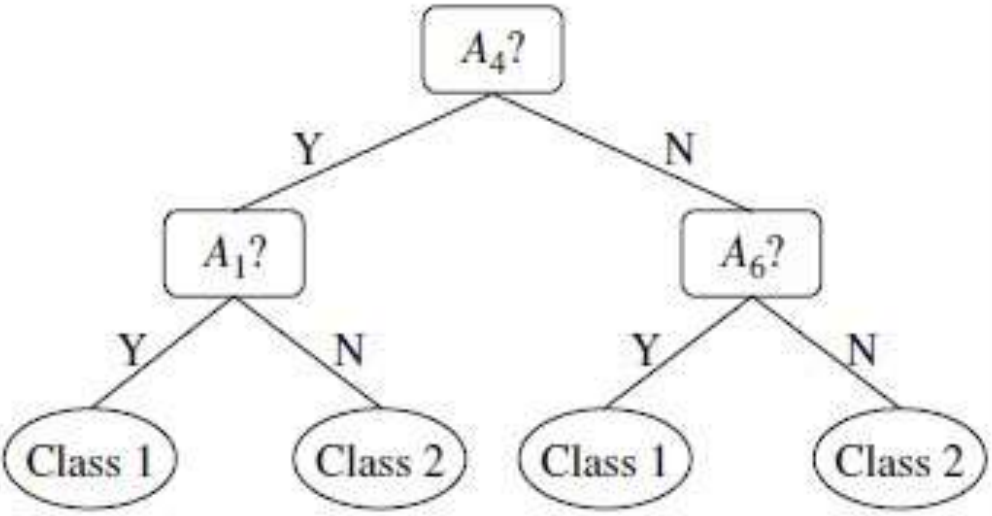
Heuristic Search in Attribute Subset Selection

4. Decision tree induction:

- Decision tree algorithms (e.g., ID3, C4.5, and CART) were originally intended for classification.
- Decision tree induction constructs a flowchart like structure (node - attribute, branch - outcome of the test, and leaf node - a class prediction)
- At each node, the algorithm chooses the “best” attribute to partition the data into individual classes.
- When decision tree induction is used for attribute subset selection, a tree is constructed from the given data.
- All attributes that do not appear in the tree are assumed to be irrelevant.
- The set of attributes appearing in the tree form the reduced subset of attributes.

Dimensionality Reduction

Heuristic Search in Attribute Subset Selection

Forward selection	Backward elimination	Decision tree induction
<p>Initial attribute set: $\{A_1, A_2, A_3, A_4, A_5, A_6\}$</p> <p>Initial reduced set: $\{\}$ $\Rightarrow \{A_1\}$ $\Rightarrow \{A_1, A_4\}$ \Rightarrow Reduced attribute set: $\{A_1, A_4, A_6\}$</p>	<p>Initial attribute set: $\{A_1, A_2, A_3, A_4, A_5, A_6\}$ $\Rightarrow \{A_1, A_3, A_4, A_5, A_6\}$ $\Rightarrow \{A_1, A_4, A_5, A_6\}$ \Rightarrow Reduced attribute set: $\{A_1, A_4, A_6\}$</p>	<p>Initial attribute set: $\{A_1, A_2, A_3, A_4, A_5, A_6\}$</p>  <pre> graph TD A4["A4?"] -- Y --> A1["A1?"] A4 -- N --> A6["A6?"] A1 -- Y --> C1_1((Class 1)) A1 -- N --> C2_1((Class 2)) A6 -- Y --> C1_2((Class 1)) A6 -- N --> C2_2((Class 2)) </pre> <p>\Rightarrow Reduced attribute set: $\{A_1, A_4, A_6\}$</p>

Dimensionality Reduction

- In *dimensionality reduction*, **data encoding or transformations are applied** so as to obtain a **reduced or “compressed”** representation of the original **data**.
- If the **original data can be *reconstructed* from the compressed data** without any loss of information, the data reduction is called ***lossless***.
- If, instead, we can **reconstruct only an approximation** of the original data, then the data reduction is called ***lossy***.
- Two popular and effective methods of lossy dimensionality reduction:
 - *wavelet transforms*
 - *principal components analysis(PCA)*.

Dimensionality Reduction

Wavelet Transformation

- **Discrete wavelet transform** is a **dimension reduction technique**.
- The **discrete wavelet transform (DWT)** is a **linear signal processing technique** that, when applied to a n -dimensional data vector X , transforms it to a numerically different n -dimensional vector, X' , of **wavelet coefficients**.
- **Compressed approximation**: store only a small fraction of the strongest of the **wavelet coefficients**.
 - All wavelet coefficients larger than some **user-specified threshold** can be retained.
 - **All other** coefficients are set to **0**.
 - The resulting data representation is therefore **very sparse**, so that **operations** that can take advantage of **data sparsity** are **computationally very fast** if performed in wavelet space.

Dimensionality Reduction

Discrete Wavelet Transformation

The general procedure for applying a discrete wavelet transform uses a hierarchical *pyramid algorithm* that halves the data at each iteration, resulting in fast computational speed. The method is as follows:

1. The length, L , of the input data vector must be an integer power of 2. This condition can be met by padding the data vector with zeros as necessary ($L \geq n$).
2. Each transform involves applying two functions.

The first applies some data smoothing, such as a sum or weighted average.

The second performs a weighted difference, which acts to bring out the detailed features of the data.

3. The two functions are applied to pairs of data points in X , that is, to all pairs of measurements $(x_{2i}; x_{2i+1})$. This results in two sets of data of length $L=2$.

In general, these represent a smoothed or low-frequency version of the input data and the high frequency content of it, respectively.

4. The two functions are recursively applied to the sets of data obtained in the previous loop, until the resulting data sets obtained are of length 2.
5. Selected values from the data sets obtained in the above iterations are designated the wavelet coefficients of the transformed data.

Dimensionality Reduction

Discrete Wavelet Transformation

- Wavelets: A math tool for space-efficient hierarchical decomposition of functions
- 8-dimensional data vector $S = [2, 2, 0, 2, 3, 5, 4, 4]$ can be transformed to 8-dimensional wavelet coefficient vector $S_\wedge = [2^{3/4}, -1^{1/4}, 1/2, 0, 0, -1, -1, 0]$
- Compression: many small detail coefficients can be replaced by 0's, and only the significant coefficients are retained

Resolution	Averages	Detail Coefficients
8	$[2, 2, 0, 2, 3, 5, 4, 4]$	
4	$[2, 1, 4, 4]$	$[0, -1, -1, 0]$
2	$[1\frac{1}{2}, 4]$	$[\frac{1}{2}, 0]$
1	$[2\frac{3}{4}]$	$[-1\frac{1}{4}]$

Dimensionality Reduction

Principal Component Analysis (PCA)

- Suppose that the data to be reduced consist of **tuples described by n dimensions**.
- **Principal components analysis (PCA)** searches for *k n -dimensional orthogonal vectors* that can best be used to represent data, $k \leq n$.
- The **original data are thus projected onto a much smaller space**, resulting in dimensionality reduction.
- Unlike attribute subset selection, which reduces the attribute set size by retaining a subset of the initial set of attributes, PCA “**combines**” the essence of attributes by **creating an alternative, smaller set of variables**.
- The initial data can then be projected onto this smaller set.
- PCA often **reveals relationships that were not previously suspected** and thereby allows **interpretations** that would not ordinarily result.

Dimensionality Reduction

Principal Component Analysis (PCA)

Principal Component Analysis Steps: Given N data vectors from n -dimensions, **find $k \leq n$ orthogonal vectors** (principal components) that can be best used to represent data.

- **Normalize input data:** Each attribute falls within the same range
- **Compute k orthonormal (unit) vectors, i.e., principal components**
- **Each input data (vector) is a linear combination of the k principal component vectors**
- The **principal components are sorted in order of decreasing “significance” or strength**
- Since the components are sorted, **the size of the data can be reduced by eliminating the weak components**, i.e., those with low variance (i.e., using the strongest principal components, it is possible to reconstruct a good approximation of the original data)

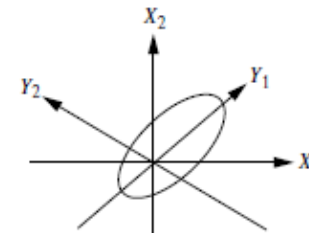


Figure 2.17 Principal components analysis. Y_1 and Y_2 are the first two principal components for the given data.

Numerosity Reduction Sampling

- “*Can we reduce the data volume by choosing alternative, ‘smaller’ forms of data representation?*”
- Techniques of *numerosity reduction* can indeed be applied for this purpose.
- These techniques may be parametric or nonparametric.
- *Parametric methods*
- *A model is used* to estimate the data, so that typically *only the data parameters need* to be stored, instead of the actual data. (Outliers may also be stored.)
- *Eg: Log-linear models* - estimate discrete multidimensional probability distributions
- *Nonparametric methods* for storing reduced representations of the data include histograms, clustering, and sampling.

Numerosity Reduction

Regression and Log-Linear Models

- **Linear regression**: Data are modeled to fit a straight line
 - Eg. A random variable y (response variable) can be modeled as **a linear function of another random variable x** (predictor variable) with eq $y=wx+b$, w and b are regression coefficients
 - 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 **two or more predictor variables** $Y = b_0 + b_1 X_1 + b_2 X_2$
- **Log-linear model**: approximates **discrete multidimensional probability distributions**
 - Each tuple of n dimensions can be **considered as a point in a multidimensional space**
 - Can estimate the probability of each point for a set of discretized **attributes** based on a smaller subset of dimensional combinations.

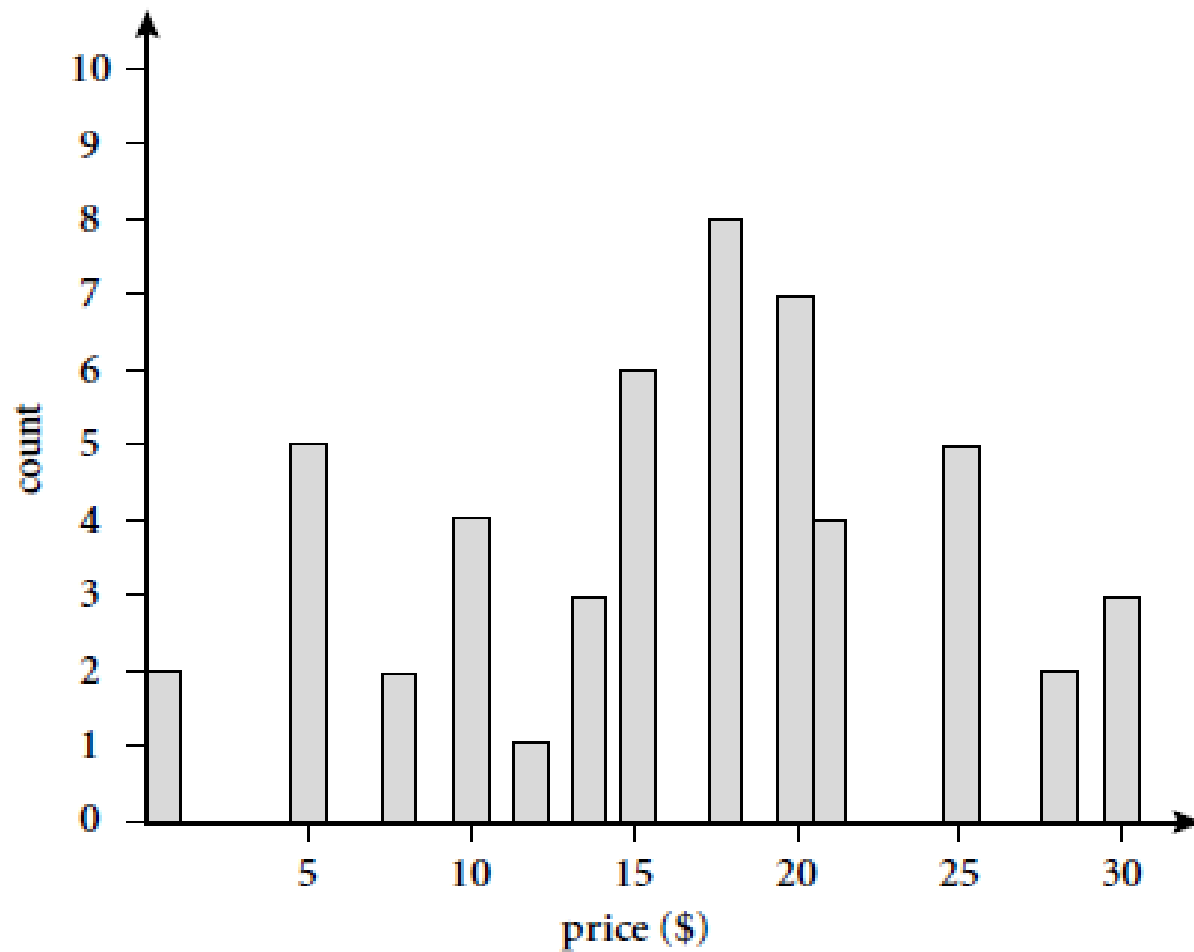
Numerosity Reduction

Histograms

- Divide data into **buckets**
- If each bucket represents only a single attribute-value/frequency pair, the buckets are called *singleton buckets*.
- Eg. The following data are a list of prices of commonly sold items at *AllElectronics*.

The numbers have been sorted:

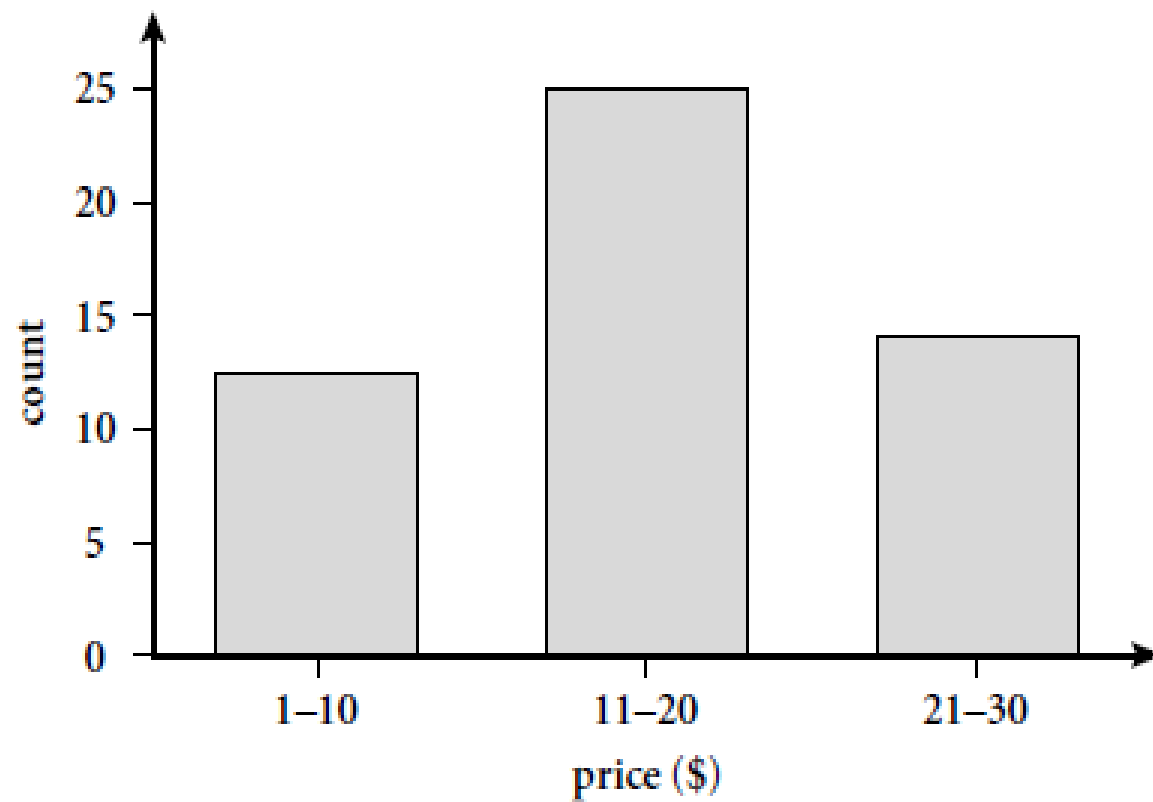
1, 1, 5, 5, 5, 5, 5, 8, 8, 10, 10, 10, 10, 12, 14, 14, 14, 15, 15, 15, 15,
15, 15, 18, 18, 18, 18, 18, 18, 18, 18, 18, 20, 20, 20, 20, 20, 20, 20, 21,
21, 21, 21, 25, 25, 25, 25, 25, 28, 28, 30, 30, 30.



A histogram for *price* using singleton buckets—each bucket represents one price-value/frequency pair.

Histograms- Partitioning rules

- **Equal-width**: equal bucket range
- **Equal-frequency** (or equal-depth)
- **V-optimal**: with the least *histogram variance* (weighted sum of the original values that each bucket represents, bucket weight is equal to the number of values in that bucket)
- **MaxDiff**: Difference between each pair of adjacent values.
- Set bucket boundary between each pair for pairs having the $\beta-1$ largest differences(β -user-specified number of buckets)



An equal-width histogram for *price*, where values are aggregated so that each bucket has a uniform width of \$10.

Example:

- **Data :** 0, 4, 12, 16, 16, 18, 24, 26, 28

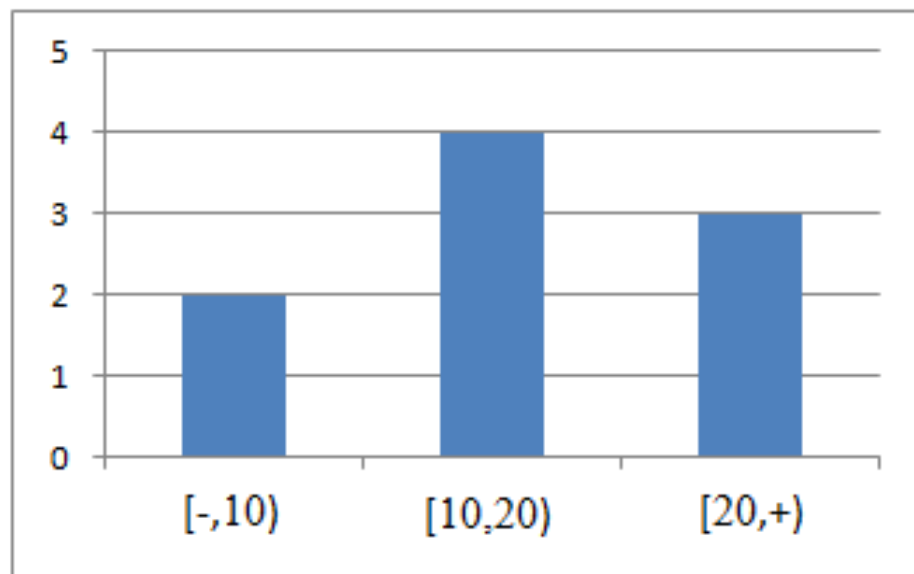
- **Equal width**

- Bin 1: 0, 4 [-,10)
- Bin 2: 12, 16, 16, 18 [10,20)
- Bin 3: 24, 26, 28 [20,+)

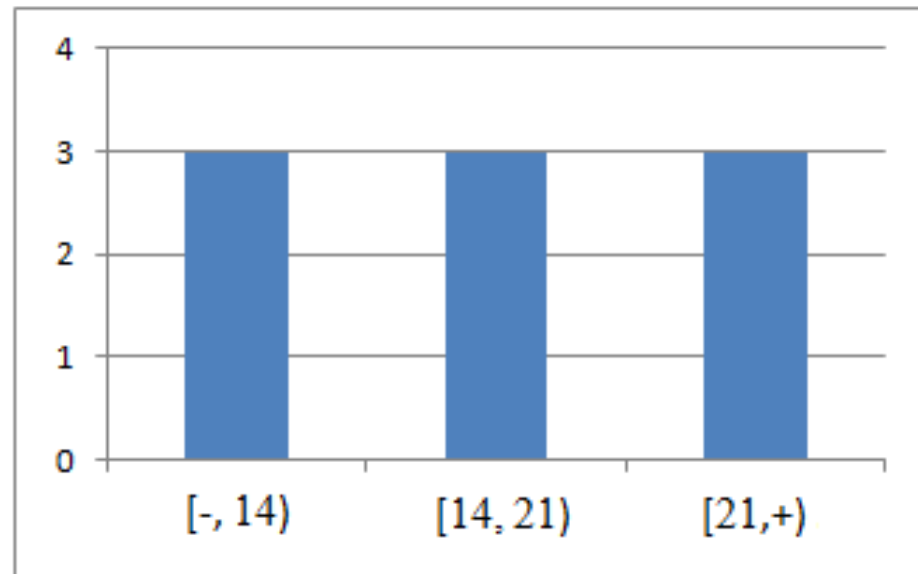
- **Equal frequency**

- Bin 1: 0, 4, 12 [-, 14)
- Bin 2: 16, 16, 18 [14, 21)
- Bin 3: 24, 26, 28 [21,+)

Equal width



Equal frequency



Numerosity Reduction

Clustering

- Consider data tuples as objects.
- They partition the objects **into groups or *clusters***, so that **objects within a cluster are “similar”** to one another and “dissimilar” to objects in other clusters.
- **Similarity** is commonly defined in terms of how “close” the objects are in space, **based on a distance function**.
- The “**quality**” of a cluster may be represented by its *diameter*, the maximum distance between any two objects in the cluster.
- *Centroid distance* is an alternative measure of cluster quality and is defined as the **average distance of each cluster object from the cluster centroid**.

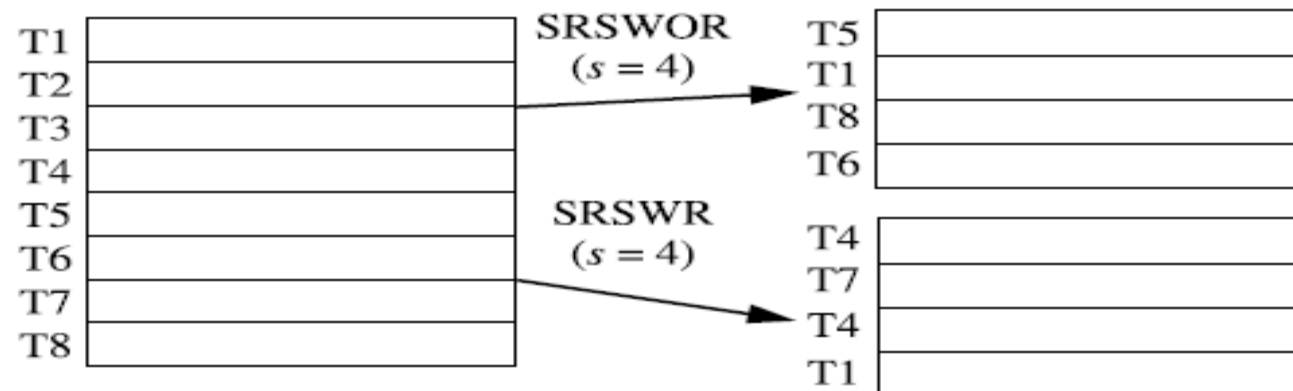
Numerosity Reduction

Sampling

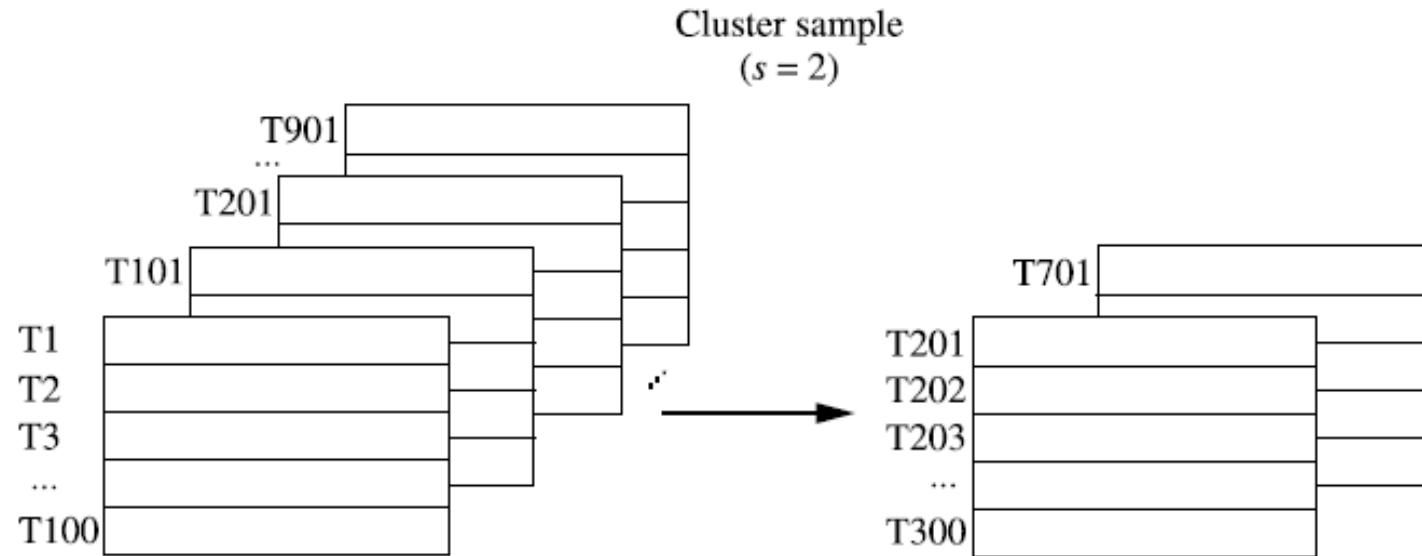
- **Sampling** is the main technique employed for data selection.
 - It is often used for both the preliminary investigation of the data and the final data analysis.
- Statisticians sample because obtaining the entire set of data of interest is too expensive or time consuming.
- **Sampling is used in data mining** because processing the entire set of data of interest is too expensive or time consuming.
- The **key principle for effective sampling** is the following:
 - using a sample will work almost as well as using the entire data sets, if the sample is representative
 - A sample is representative if it has approximately the same property (of interest) as the original set of data

Sampling

- **Sampling**: obtaining a small sample s to represent the whole data set N
- Allow a mining algorithm to run in complexity that is potentially sub-linear to the size of the data.
- **Simple random sample without replacement (SRSWOR)** of size s : drawing s of the N tuples from D ($s < N$), where the probability of drawing any tuple in D is $1/N$
- As each item is selected, it is removed from the population
- **Simple random sample with replacement (SRSWR)** of size s : each time a tuple is drawn from D , it is recorded and then replaced
- Objects are not removed from the population as they are selected for the sample.



- **Cluster sample:** *Tuples in D are grouped into M mutually disjoint “clusters,” then an SRS of s clusters can be obtained, where $s < M$*



- **Stratified sample:** If D is divided into mutually disjoint parts called *strata*, a stratified sample of D is generated by obtaining an SRS at each stratum.
 - Split the data into several partitions; then draw random samples from each partition.
 - In the simplest version, equal numbers of objects are drawn from each group even though the groups are of different sizes.
 - In an other variation, the number of objects drawn from each group is proportional to the size of that group.

Stratified sample
(according to *age*)

T38	youth
T256	youth
T307	youth
T391	youth
T96	middle_aged
T117	middle_aged
T138	middle_aged
T263	middle_aged
T290	middle_aged
T308	middle_aged
T326	middle_aged
T387	middle_aged
T69	senior
T284	senior

T38	youth
T391	youth
T117	middle_aged
T138	middle_aged
T290	middle_aged
T326	middle_aged
T69	senior

Discretization

Discretization: To transform a **numeric (continuous) attribute into a categorical attribute.**

- Some data mining algorithms require that data be in the form of categorical attributes.
- In discretization:
 - The **range** of a continuous attribute is **divided into intervals.**
 - Then, interval labels can be used to **replace actual data values to obtain a categorical attribute.**

Simple Discretization Example: *income* attribute is discretized into a categorical attribute.

- Target categories (low, medium, high).
- Calculate average *income*: AVG.
 - If $\text{income} > 2 * \text{AVG}$, $\text{new_income_value} = \text{"high"}$.
 - If $\text{income} < 0.5 * \text{AVG}$, $\text{new_income_value} = \text{"low"}$.
 - Otherwise, $\text{new_income_value} = \text{"medium"}$.

Discretization Methods

- A basic distinction between discretization methods for classification is **whether class information is used (supervised) or not (unsupervised)**.
- Some of discretization methods are as follows:

Unsupervised Discretization: If class information is not used, then relatively simple approaches are common.

- Binning
- Clustering analysis

Supervised Discretization:

- Classification (e.g., decision tree analysis)
- Correlation (e.g., χ^2) analysis

Discretization by Binning

- Attribute values can be discretized by **applying equal-width or equal-frequency binning**.
- Binning approaches **sorts the attribute values first**, then partition them into the bins.
 - **equal width approach** divides the range of the attribute into a user-specified number of intervals each having the same width.
 - **equal frequency (equal depth) approach** tries to put the same number of objects into each interval.
- After bins are determined, all values are replaced by **bin labels** to discretize that attribute.
 - Instead of bin labels, **values may be replaced by bin means (or medians)**.
- Binning **does not use class information** and is therefore an **unsupervised** discretization technique.

Discretization by Binning: Example equal-width approach

- Suppose a group of 12 values of *price* attribute has been sorted as follows:

<i>price</i>	5	10	11	13	15	35	50	55	72	89	204	215
--------------	---	----	----	----	----	----	----	----	----	----	-----	-----

equal-width partitioning: The width of each interval is $(215-5)/3 = 70$.

- Partition them into *three bins*

bin1	5, 10, 11, 13, 15, 35, 50, 55, 72
bin2	89
bin3	204, 215

- Replace each value with its bin label to discretize.

[illegible]

Discretization by Binning: Example

equal-frequency approach

- Suppose a group of 12 values of *price* attribute has been sorted as follows:

<i>price</i>	5	10	11	13	15	35	50	55	72	89	204	215
--------------	---	----	----	----	----	----	----	----	----	----	-----	-----

equal-frequency partitioning:

- Partition them into *three bins*: each interval contains 4 values

bin1	5, 10, 11, 13
bin2	15, 35, 50, 55
bin3	72, 89, 204, 215

- Replace each value with its bin label to discretize.

<i>price</i>	5	10	11	13	15	35	50	55	72	89	204	215
<i>categorical attr.</i>	1	1	1	1	2	2	2	2	3	3	3	3

Discretization by Clustering

- A **clustering** algorithm can be applied to discretize a numeric attribute.
 - The values of the attribute are partitioned into clusters by a clustering algorithm.
 - Each value in a cluster is replaced by the label of that cluster to discretize.
- Clustering **takes the distribution and closeness of attribute values into consideration**, and therefore is **able to produce high-quality discretization results**.

Simple clustering: *partition data from biggest gaps.*

- Example: **partition data along 2 biggest gaps into three bins.**

bin1	5, 10, 11, 13, 15
bin2	35, 50, 55, 72, 89
bin3	204, 215

- Replace each value with its bin label to discretize.

<i>price</i>	5	10	11	13	15	35	50	55	72	89	204	215
<i>categorical attr.</i>	1	1	1	1	1	2	2	2	2	2	3	3

Discretization by Classification

- Techniques used for a **classification** algorithm such as **decision tree** can be applied to discretization.
- *Decision tree approaches to discretization* are **supervised**, that is, *they make use of class label information*.
- These techniques *employ a top-down splitting approach* for attribute values:
 - *Class distribution information is used* in the calculation and determination of *split-points*.
 - The main idea is to **select split-points** so that a given resulting **partition contains as many tuples of the same class** as possible.
 - **Entropy** is the most commonly used measure for this purpose.

Discretization by Correlation Analysis

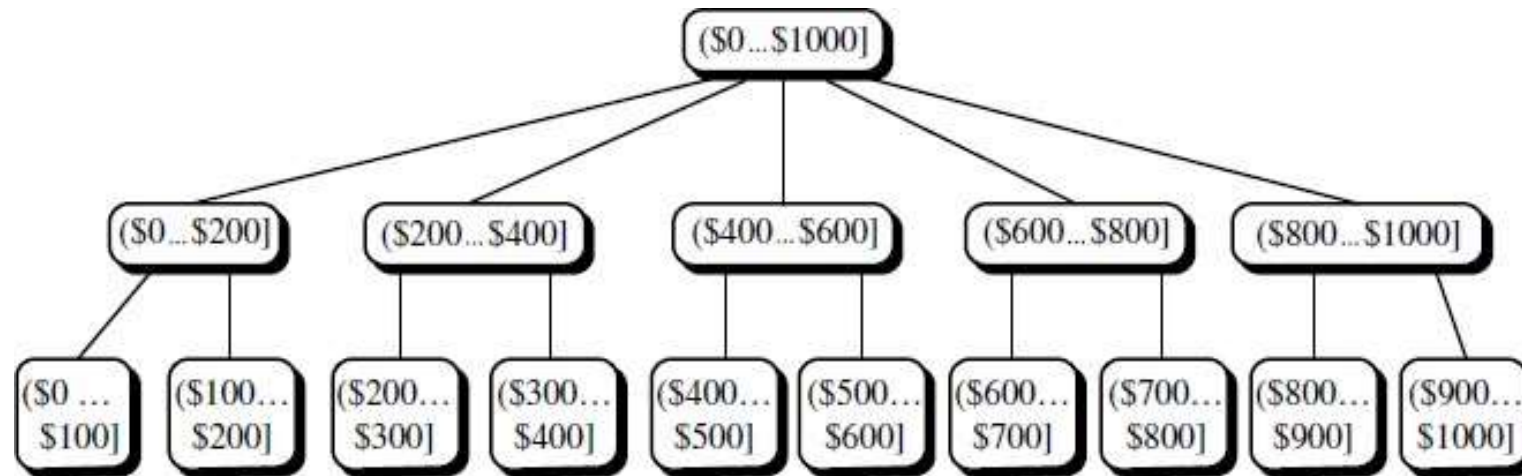
- Measures of correlation can be used for discretization.
- *ChiMerge* is a χ^2 – based discretization method.
 - ChiMerge employs a bottom-up approach.
 - ChiMerge finds the best neighboring intervals and then merge them to form larger intervals, recursively.
 - ChiMerge is supervised since it uses class information.
- ChiMerge proceeds as follows:
 - Initially, each distinct value of a numeric attribute is considered to be one interval,
 - χ^2 tests are performed for every pair of adjacent intervals.
 - Adjacent intervals with the least χ^2 values merged together, because low χ^2 values for a pair indicate similar class distributions.
 - This merging process proceeds recursively until a predefined stopping criterion is met.

Concept Hierarchy

- A **concept hierarchy** defines a sequence of mappings from a set of low-level concepts to higher-level, more general concepts.
- Many concept hierarchies are implicit within the database schema.
- Concept hierarchies may be provided manually by system users or may be automatically generated based on statistical analysis of the data distribution.
- A concept hierarchy that is **a total or partial order among attributes**.
- Concept hierarchies may also be defined by discretizing or grouping values for a given dimension.

Concept Hierarchy

- A concept hierarchy for the attribute *price*

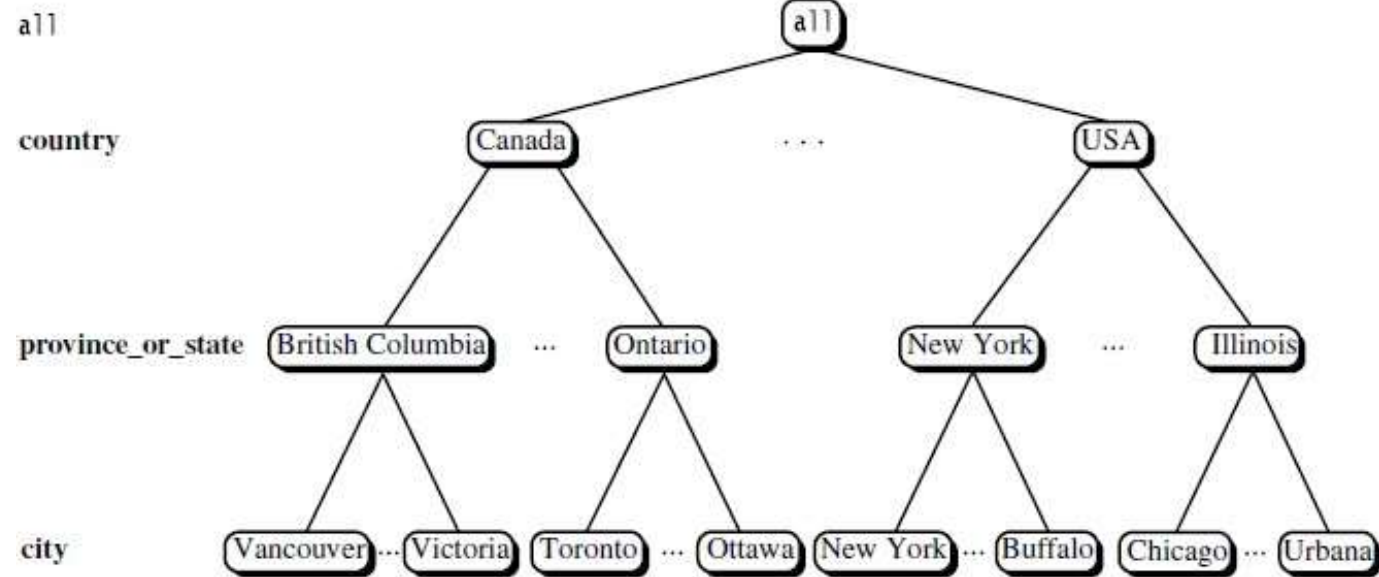


- A concept hierarchy for a numeric attribute can be constructed automatically or it can be created by the user.
- A concept hierarchy can be used for discretization.

Concept Hierarchy

- Concept hierarchies for nominal attributes can also be created.

location attribute



time attribute

