Data Mining

UNIT-II

B.Tech(CSE)-VI SEM

UNIT : II -Data Preprocessing

- Overview of Data Preprocessing
- Data Cleaning
- Data Integration
- Data Reduction
- Data Transformation
- Data Discretization

Data Preprocessing

- It is a data mining technique that involves transforming raw data in to an understandable format.
- To make data more suitable for data mining.
- To improve the data mining analysis with respect to time, cost and quality.

Why preprocess the data?

- Data in the real world is:

  - incomplete: lacking attribute values, lacking certain attributes of interest, or containing only aggregate data

  - e.g., *Occupation*=“ ” (missing data)

  - noisy: containing noise, errors, or outliers

  - e.g., *Salary*=“–10” (an error)

  - inconsistent: lack of compatibility or similarity between two or more facts. e.g.,

  - *Age*=“42”, *Birthday*=“03/07/2010”
  - Was rating “1, 2, 3”, now rating “A, B, C”
  - discrepancy between duplicate records

- No quality data , no quality mining results:

  - Quality decisions must be based on quality data
  - Data warehouse needs consistent integration of quality data

Why preprocess the data?

- No quality data, no quality mining results!
  - Quality decisions must be based on quality data
    - e.g., duplicate or missing data may cause incorrect or even misleading statistics.

Data preparation, cleaning, and transformation comprises the majority of the work in a data mining application (90%).

Measures of Data Quality

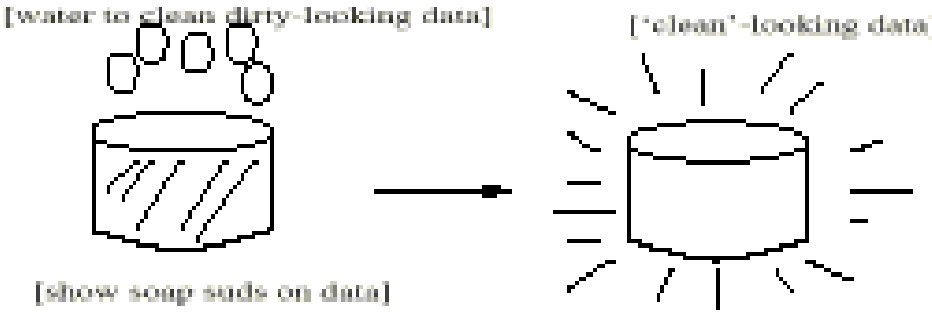
- Accuracy
- Completeness
- Consistency
- Timeliness
- Believability
- Value added
- Interpretability
- Accessibility

Major Tasks in Data Preprocessing

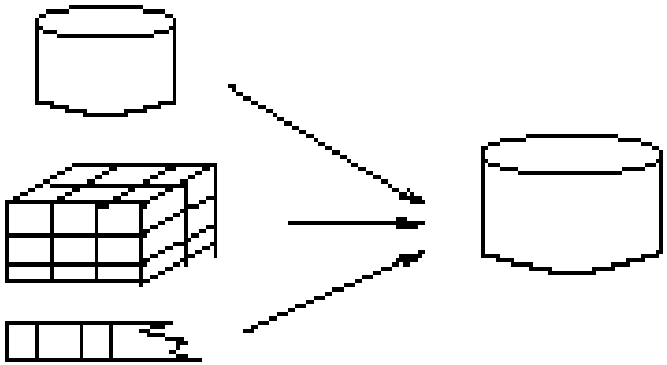
- **Data Cleaning:** Fill in Missing Values, Smooth noisy data, identify or remove outliers, and resolve inconsistencies.
- **Data Integration:** Integration of Multiple databases, data cubes, or files.
- **Data Reduction:** Obtains reduced representation in volume but produces the same or similar analytical results
  - Dimensionality reduction
  - Numerosity reduction
  - Data compression
- **Data Transformation and Discretization (for numerical data):** Normalization and Concept hierarchy generation

Forms of Data preprocessing

Data Cleaning



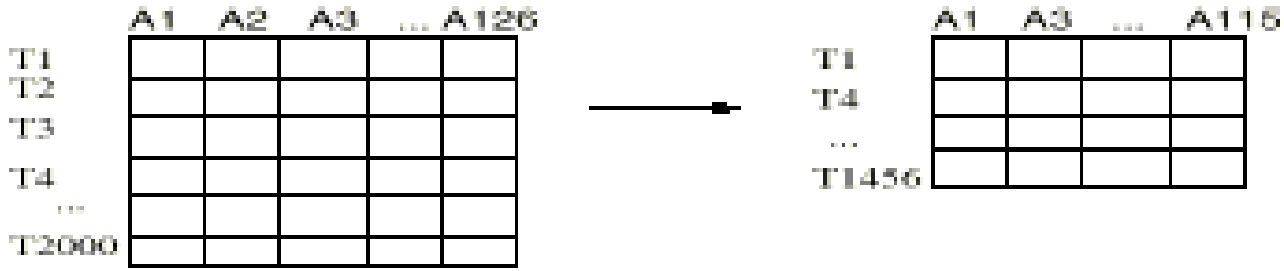
Data Integration



Data Transformation

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

Data Reduction



Data cleaning

- Data cleaning attempts to fill in missing values, smooth out noise while identifying outliers and correct inconsistencies in the data.

- **Data cleaning Tasks:**

1. Missing Values
2. Noisy Data
3. Inconsistent Data

Data Cleaning – Missing Values

1. **Ignore the tuple:** This is usually done when the class label is missing. This method is not very effective, unless the tuple contains several attributes with missing values.
2. **Fill in the missing values manually:** In general, this approach is time- consuming and may not be feasible given a large data set with many missing values.
3. **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 "-∞".
4. **Use the attribute mean to fill in the missing values**
5. **Use the attribute mean for all samples belonging to the same class as the given tuple.**
6. **Use the most probable value to fill in the missing value:** This may be determined with inference-based such as Bayesian formula or decision tree

Data Cleaning – Noisy Data

- **Noise:** random error or variance 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
    - inconsistent data

How to Handle Noisy Data?

1. Binnig
2. Clustering
3. Combined Computer and human inspection
4. Regression

Data Cleaning – Noisy Data

1. **Binning:**

- 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.

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:

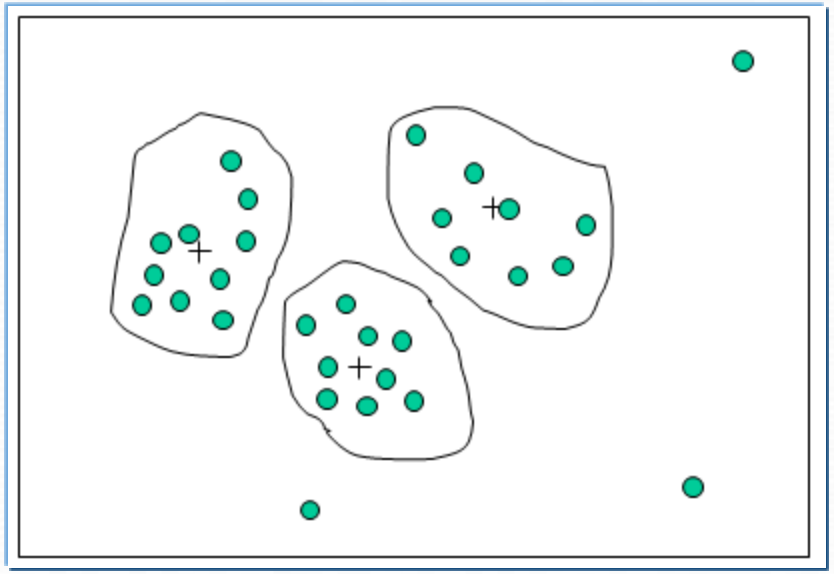
- **Bin 1:** 9, 9, 9, 9-
- **Bin 2:** 23, 23, 23, 23
- **Bin 3:** 29, 29, 29, 29

*Smoothing by bin boundaries:

- **Bin 1:** 4, 4, 4, 15
- **Bin 2:** 21, 21, 25, 25
- **Bin 3:** 26, 26, 26, 34

Data Cleaning – Noisy Data

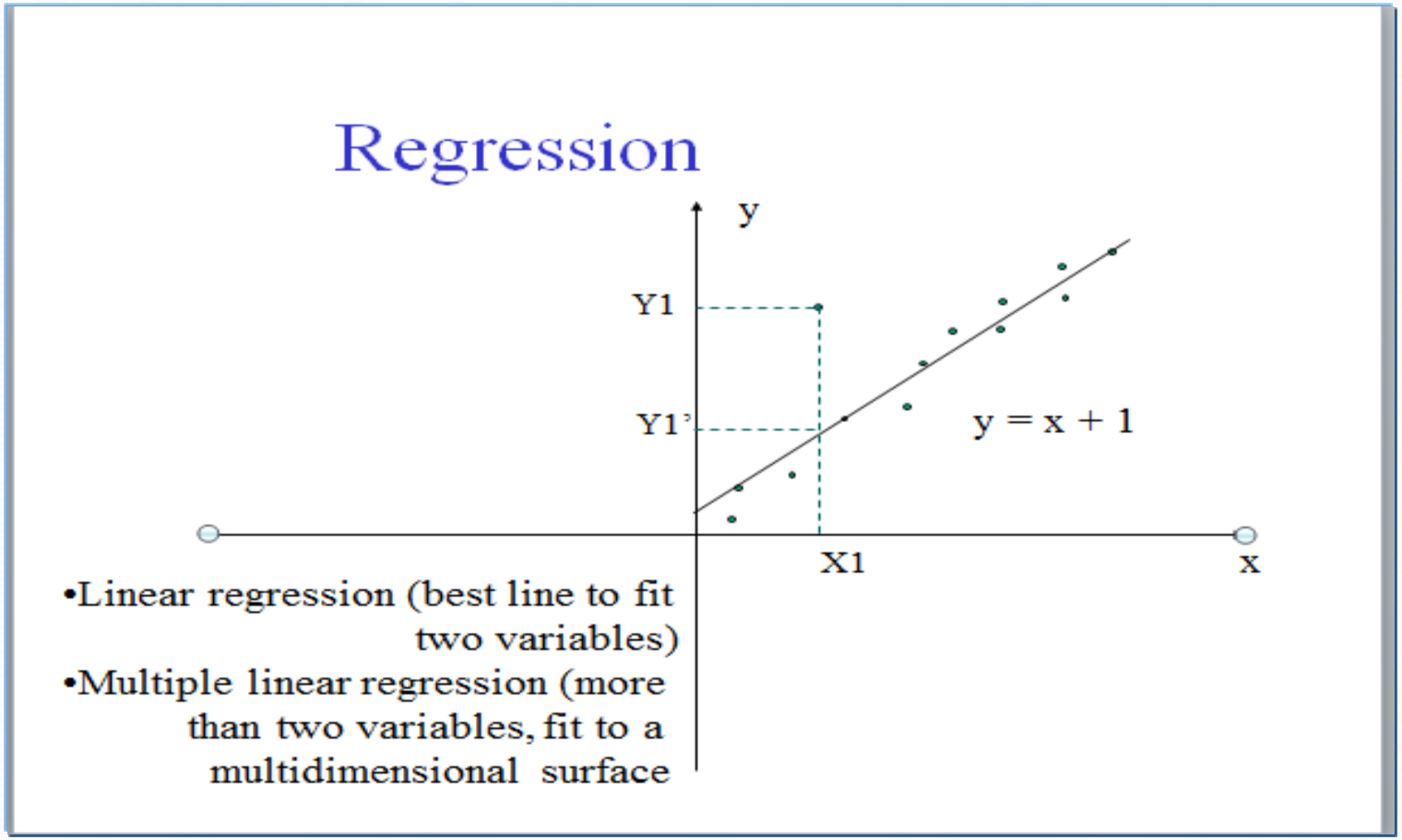
2. **Clustering:**
   - Similar values are organized into groups (clusters).
   - Values that fall outside of clusters considered outliers



3. **Combined computer and human inspection:** Outliers may be identified through a combination of computer and human inspection. Outlier patterns may be informative or garbage.

Data Cleaning – Noisy Data

4. **Regression:** Data can be smoothed by fitting the data to a function such as with regression. (linear regression/multiple linear regression)



Data Integration

- The merging of data from multiple data sources. The data may also need to be transformed into forms appropriate for mining. The data sources may include multiple databases, datacubes (or) flat files.

- **Data Integration – Issues**

  1. Entity Identification Problem
  2. Redundancy and Correlation Analysis
  3. Tuple Duplication
  4. Data Conflict Detection and Resolution

Data Integration

1. Entity Identification Problem :

- Integrate metadata (about the data) from different sources.
- The real world entities from multiple source be matched referred to as the **entity identification problem.**

  For example, How can the data analyst and computer be sure that customer id in one data base and customer number in another reference to the same attribute. A.cust-id=B.cust-# (same entity?)

Schema integration: Integrate metadata from different sources. **e.g.,** A.cust-id ≡ B.cust-#

Data Integration

2. Redundancy and Correlation Analysis:

- An attribute may be redundant if it can be "**derived**" or obtaining from another attribute or set of attribute.
- Inconsistencies in attribute can also cause redundancies in the resulting data set.
- Some redundancies can be detected by **correlation analysis**.
- For nominal data , we use the X^2 (chi-square) test.

Data Integration

2. Redundancy and Correlation Analysis:

- An attribute may be redundant if it can be "**derived**" or obtaining from another attribute or set of attribute.
- Inconsistencies in attribute can also cause redundancies in the resulting data set.
- Some redundancies can be detected by **correlation analysis**.
- For nominal data , we use the X^2 (chi-square) test.
- For numeric attributes, we can use the correlation coefficient and covariance.

Correlation Analysis (Nominal Data):

- X^2 (chi-square) test

$$\chi^2 = \sum \frac{(Observed - Expected)^2}{Expected}$$

- The larger the X^2 value, the more likely the variables are related
- The cells that contribute the most to the X^2 value are those whose actual count is very different from the expected count
- Correlation does not imply causality
  - # of hospitals and # of car-theft in a city are correlated
  - Both are causally linked to the third variable: population

Chi-Square 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(col.)	300	1200	1500

- X^2 (chi-square) calculation (numbers in parenthesis are expected counts calculated based on the data distribution in the two categories)

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

- It shows that like_science_fiction and play_chess are correlated in the group

Correlation Analysis (Numeric Data)

- Correlation coefficient (also called Pearson's product moment coefficient)

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

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 $\Sigma(a_i b_i)$ is the sum of the AB cross-product.

- 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; $r_{AB} < 0$: negatively correlated

Covariance (Numeric Data)

- Covariance is similar to correlation

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

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

where n is the number of tuples, $\bar{A}$ and $\bar{B}$ are the respective mean or **expected values** of A and B, σ_A and σ_B are the respective standard deviation of A and B.

- **Positive covariance**: If $Cov_{A,B} > 0$, then A and B both tend to be larger than their expected values.
- **Negative covariance**: If $Cov_{A,B} < 0$ then if A is larger than its expected value, B is likely to be smaller than its expected value.
- **Independence**: $Cov_{A,B} = 0$ 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

Co-Variance: An Example

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

- It can be simplified in computation as

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

- Suppose two stocks A and B have the following values in one week: (2, 5), (3, 8), (5, 10), (4, 11), (6, 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 + 6)/ 5 = 20/5 = 4
  - E(B) = (5 + 8 + 10 + 11 + 14) /5 = 48/5 = 9.6
  - Cov(A,B) = (2×5+3×8+5×10+4×11+6×14)/5 − 4 × 9.6 = 4
- Thus, A and B rise together since Cov(A, B) > 0.

Data Integration

3. Tuple Duplication:

Along with redundancies data integration has also deal with the duplicate tuples. Duplicate tuples may come in the resultant data if the denormalized table has been used as a source for data integration.

4. Data value conflict Detection and Resolution:

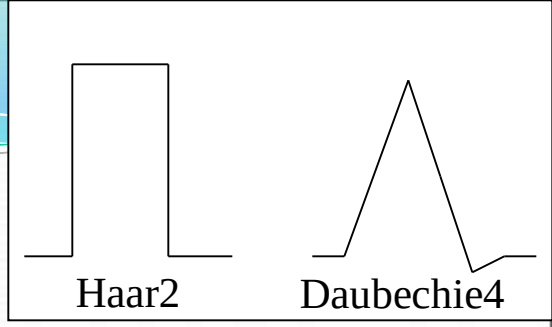
- Attribute values from another different sources may differ for the same real world entity.
- An attribute in one system may be recorded at a lower level abstraction then the "same" attribute in another.

For Example, the total sales in one database may refer to one branch of All Electronics, an attribute of the same name in another database may refer to the total sales for All Electronics store in a given region.

Data Reduction

- **Data Reduction** techniques are applied to obtain a reduced representation of the data set that is much smaller in volume, yet closely maintain the integrity of the original data.
- **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 is the process of reducing the number of random variables or attributes under consideration.
    - Wavelet transforms
    - Principal Components Analysis (PCA)
    - Feature subset selection, feature creation
  - **Numerosity reduction** replaces original data by alternative,small forms of data representation
    - Regression and Log-Linear Models(parametric methods)
    - Histograms, clustering, sampling, Data cube aggregation(non parametric methods)
  - **Data compression**

Data Reduction: Wavelet Transforms



- Discrete wavelet transform (DWT) for linear signal processing, multi-resolution analysis
- 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
- Method: (Hierarchical pyramid algorithm)
  - Length, L, must be an integer power of 2 (padding with 0's, when necessary)
  - Each transform has 2 functions: smoothing, difference
  - Applies to pairs of data, resulting in two set of data of length L/2
  - Applies two functions recursively, until reaches the desired length

Data Reduction - Principal Component Analysis (PCA) (also called Karhunen-Loeve (K-L) method):

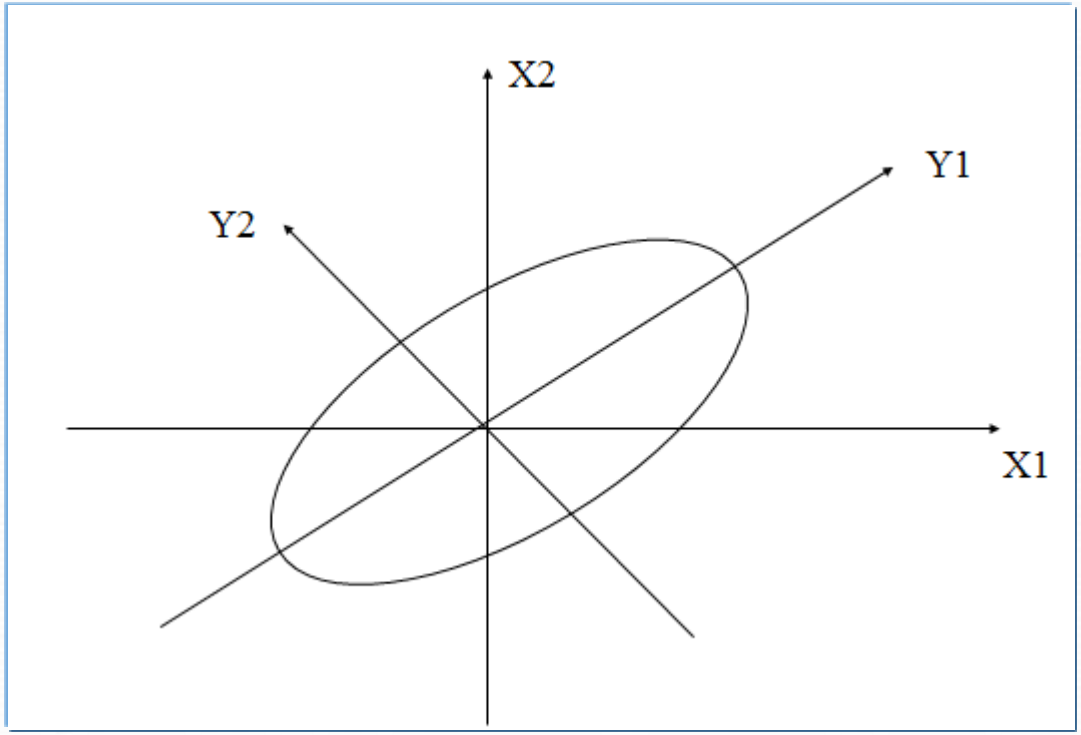
- The data consists of tuples or data vectors described by *n attributes or dimensions*.
- ***PCA*** searches for *k n-dimensional orthogonal vectors that* can best be used to represent the data, where *k* <=*n*.
- The original data are thus projected onto a much smaller space, resulting in dimensionality reduction.
- PCA is computationally inexpensive, and can be applied to ordered or unordered data(attributes).
- PCA can handle sparse and skewed data.
- Multidimensional data can be handled by reducing the data into two dimensional.
- 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.

Data Reduction: Principal Component Analysis (Steps)

- The basic procedure is as follows:
- The input data are normalized, so that each attribute falls within the same range. This step helps ensure that attributes with large domains will not dominate attributes with small domains. Compute k orthonormal vectors, i.e., principal Components.
- PCA computes k orthonormal vector that provide a basis for the normalized input data. These are unit vectors that each point in a direction perpendicular to the other. These vectors are referred to as the principal components. The input data are a linear combination of the principal components.
- The principal components are sorted in order of decreasing "significance" or strength. The principal components essentially serve as a new set of axes for the data, providing important information about variance. That is, the sorted axes are such that the first axis shows the most variance among the data, the second axis shows the next highest variance, and so on.
- Because the components are sorted in decreasing order of "significance," the data size can be reduced by eliminating the weaker components, that is, those with low variance. Using the strongest principal components, it should be possible to reconstruct a good approximation of the original data.

Data Reduction - Principal Component Analysis (PCA) (also called Karhunen-Loeve (K-L) method):

- Find a projection that captures the largest amount of variation in data
- The original data are projected onto a much smaller space, resulting in dimensionality reduction. We find the eigenvectors of the covariance matrix, and these eigenvectors define the new space



Principal components analysis. Y1 and Y2 are the first two Principal components for the given data.

Data Reduction

- Works for numeric data only
- Principal Components are used as inputs to multiple regression and clustering analysis

Conclusion:

- PCA handles better the sparse Data.
- DWT more suitable for data of high Dimensionality.

Data Reduction: Attribute Subset Selection

- **Attribute subset selection (feature selection):** Reduce the data set size by removing irrelevant or redundant attributes.

- **Goal**: select a minimum set of features (attributes) 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.

- It reduces the number of attributes appearing in the discovered patterns, helping to make the patterns easier to understand. redundant attributes

Data Reduction: Attribute Subset Selection

- **How can we find a 'good' subset of the original attributes?**

- For n attributes, there are 2^n possible subsets.
- An exhaustive search for the optimal subset of attributes can be prohibitively expensive, especially as n increase.
- Heuristic methods that explore a reduced search space are commonly used for attribute subset selection.
- These methods are typically greedy in that, while searching through attribute space, they always make what looks to be the best choice at the time.
- Such greedy methods are effective in practice and may come close to estimating an optimal solution.

Data Reduction: Attribute Subset Selection

Stepwise forward selection:

- The procedure starts with an empty set of attributes as the reduced set.
- **First:** The best single-feature is picked.
- **Next:** At each subsequent iteration or step, the best of the remaining original attributes is added to the set

Initial attribute set:
$\{A_1, A_2, A_3, A_4, A_5, A_6\}$

Initial reduced set:
$\{\}$
=> $\{A_1\}$
=> $\{A_1, A_4\}$
=> Reduced attribute set:
$\{A_1, A_4, A_6\}$

Data Reduction: Attribute Subset Selection

Stepwise backward elimination:

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

Initial attribute set:
$\{A_1, A_2, A_3, A_4, A_5, A_6\}$

=> $\{A_1, A_3, A_4, A_5, A_6\}$
=> $\{A_1, A_4, A_5, A_6\}$
=> Reduced attribute set:
$\{A_1, A_4, A_6\}$

Combining forward selection and backward elimination:

- The stepwise forward selection and backward elimination methods can be combined.
- At each step, the procedure selects the best attribute and removes the worst from among the remaining attributes.

Data Reduction:Attribute Subset Selection

Decision tree induction:

- Decision tree algorithms, such as ID3, C4.5, and CART, were originally intended for classification.

- Decision tree induction constructs a flowchart-like structure where each internal (nonleaf) node denotes a test on an attribute, each branch corresponds to an outcome of the test, and each external (leaf) node denotes a class prediction.

- At each node, the algorithm chooses the "best" attribute to partition the data into individual classes.

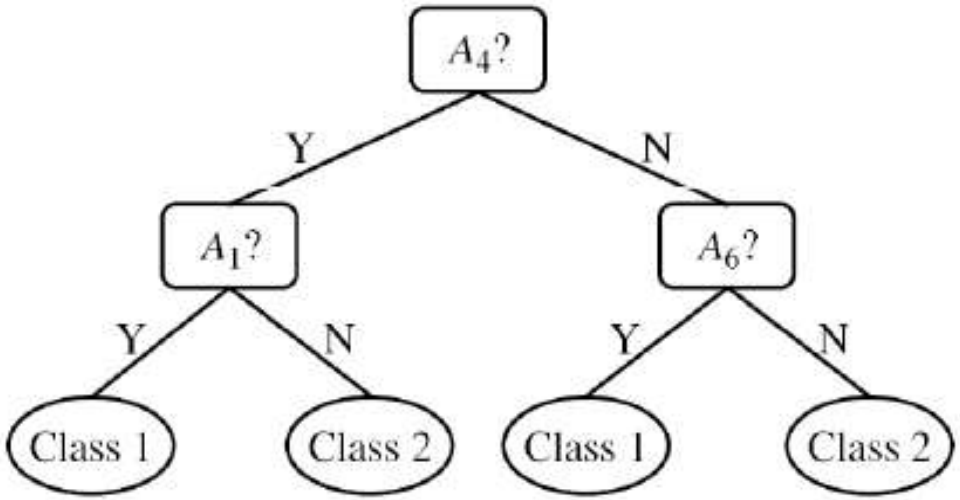
- When decision tree induction is used for attribute subset selection, a tree is constructed from the given data.

- All attributes that do not appear in the tree are assumed to be irrelevant.

Data Reduction : Attribute Subset Selection

Decision tree induction

Initial attribute set:
$\{A_1, A_2, A_3, A_4, A_5, A_6\}$



=> Reduced attribute set:
$\{A_1, A_4, A_6\}$

Data Reduction : Attribute Subset Selection

- In some cases, we may want to create new attributes based on others. Such attribute construction can help improve accuracy and understanding of structure in high dimensional data. For example, we may wish to add the attribute area based on the attributes height and width.
- By combining attributes, attribute construction can discover missing information about the relationships between data attributes that can be useful for knowledge discovery.

Data Reduction: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.

$$Y = \alpha + \beta X$$

  - 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 $Y_1, Y_2, \ldots, X_1, X_2, \ldots$.
- **Multiple regression:**
  - Allows a response variable y to be modeled as a linear function of multidimensional feature vector (predictor variables).

$$Y = b_0 + b_1 X_1 + b_2 X_2.$$

Many nonlinear functions can be transformed into the above

- **Log-linear model:**
  - approximates discrete multidimensional joint probability distributions.
  - The multi-way table of joint probabilities is approximated by a product of lower-order tables.
  - Probability: $p(a, b, c, d) = \alpha ab\ \beta ac\ \chi ad\ \delta bcd$

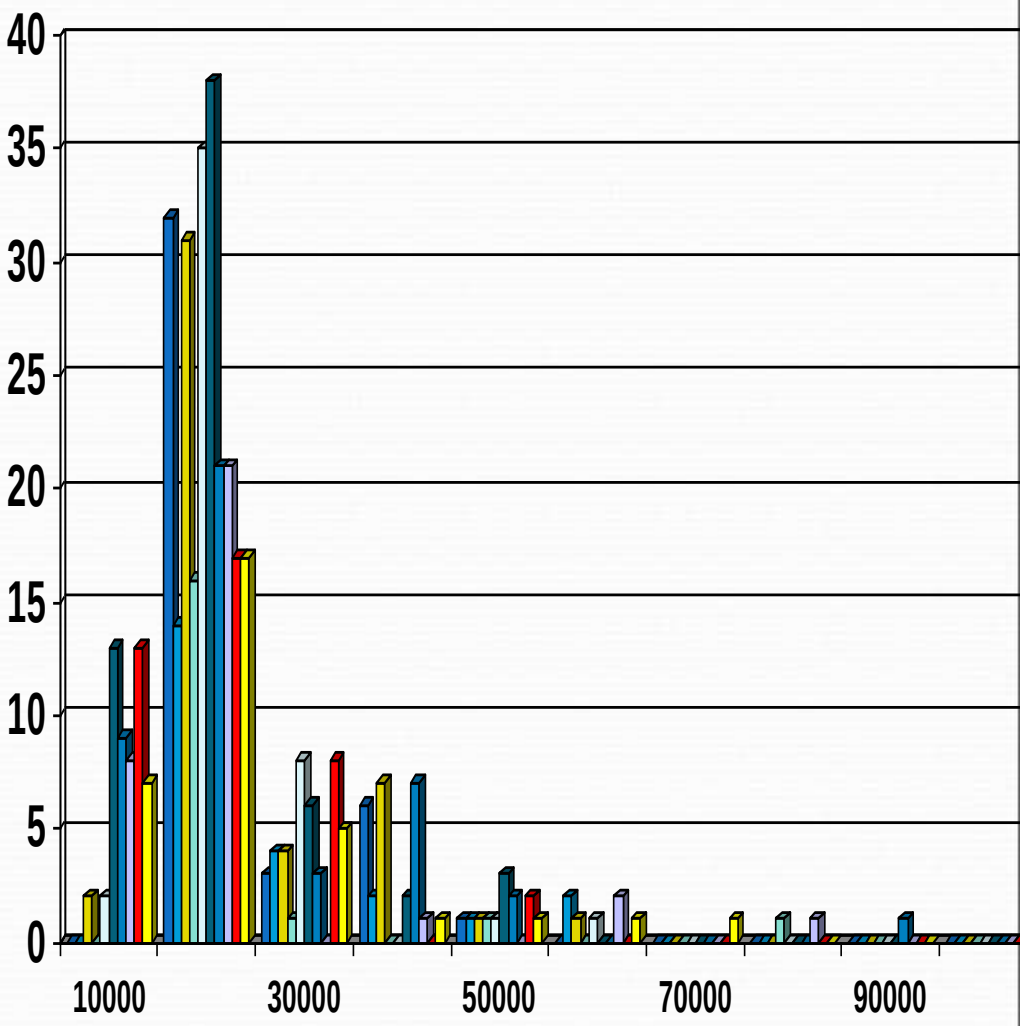
Log Linear Models

- **Log-linear models** approximate discrete multidimensional probability distributions.
- Given a set of tuples in *n* dimensions (e.g., described by *n* attributes), we can consider each tuple as a point in an *n*-dimensional space. Log-linear models can be used to estimate the probability of each point in a multidimensional space for a set of discretized attributes, based on a smaller subset of dimensional combinations. This allows a higher-dimensional data space to be constructed from lower-dimensional spaces.
- Log-linear models are useful for Dimensionality Reduction and Data Smoothing.

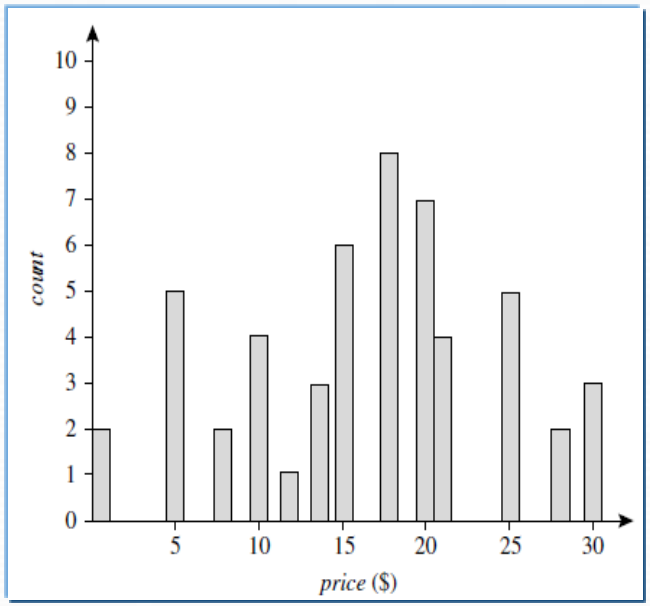
- Regression and log-linear models can both be used on sparse data, although their application may be limited.
- Among both, regression handles skewed data in better way
- Regression can be computationally intensive when applied to high-dimensional data.
- og-linear models show good scalability for up to 10 or so dimensions.

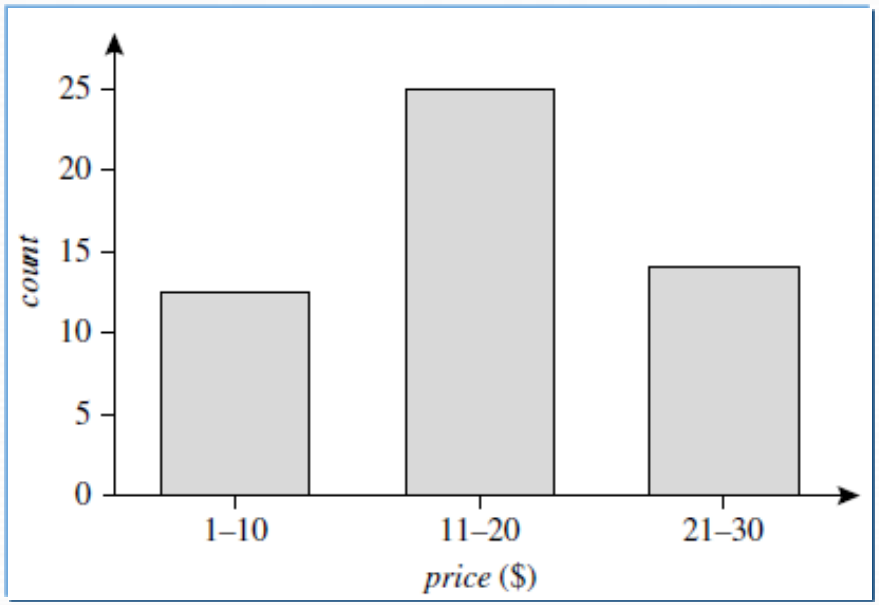
Data Reduction: Histogram

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



Histograms





a)Histogram for price – using singleton buckets
b) An equal depth Histogram for price.

Histogram

- Histograms are highly effective at approximating both sparse and dense data, as well as highly skewed and uniform data.
- The histograms described for single attributes can be extended for multiple attributes.
- *Multidimensional histograms can capture* dependencies between attributes.
- Singleton buckets are useful for storing high-frequency outliers.

Data Reduction: 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))

Data Reduction: Sampling

- Sampling: obtaining a small sample s to represent the whole data set N
- Suppose that a large data set, D, contains N instances.
- The most common ways that we could sample D for data reduction:

  -Simple random sample without replacement (SRSWOR)

  -Simple random sample with replacement (SRSWR)

  -Cluster sample

  -Stratified sample

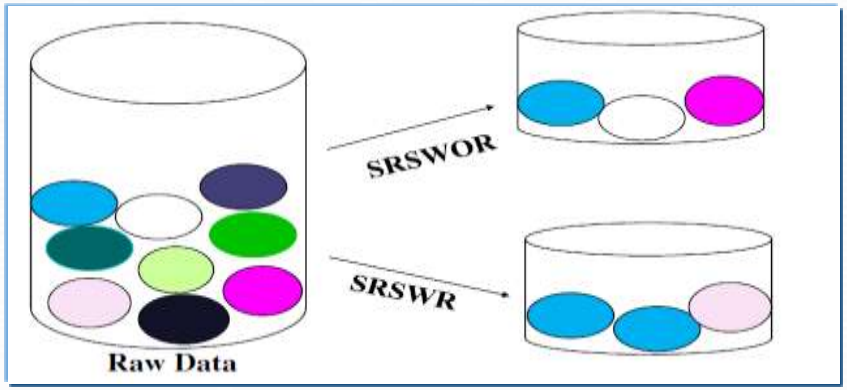
Data Reduction: Sampling

- **Simple random sample without replacement (SRSWOR) of size s:**

  - This is created by drawing s of the N instances from D (s <N), where the probability of drawing any tuple in D is 1=N, that is, all instances are equally likely to be sampled.

- **Simple random sample with replacement (SRSWR) of size s:**

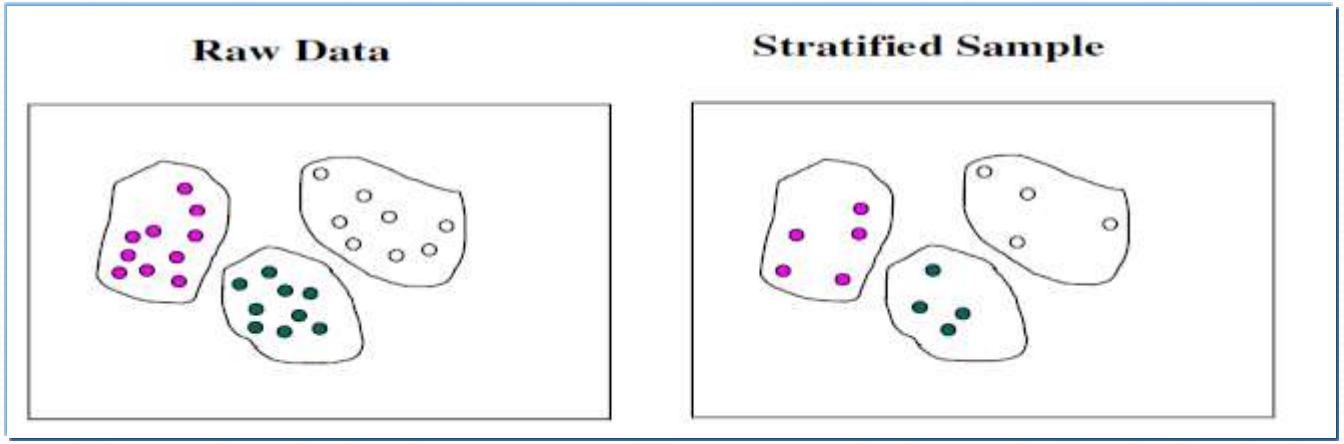
  - This is similar to SRSWOR, except that each time a tuple is drawn from D, it is recorded and then replaced.
  - That is, after a instance is drawn, it is placed back in D so that it may be drawn again.



Data Reduction: Sampling

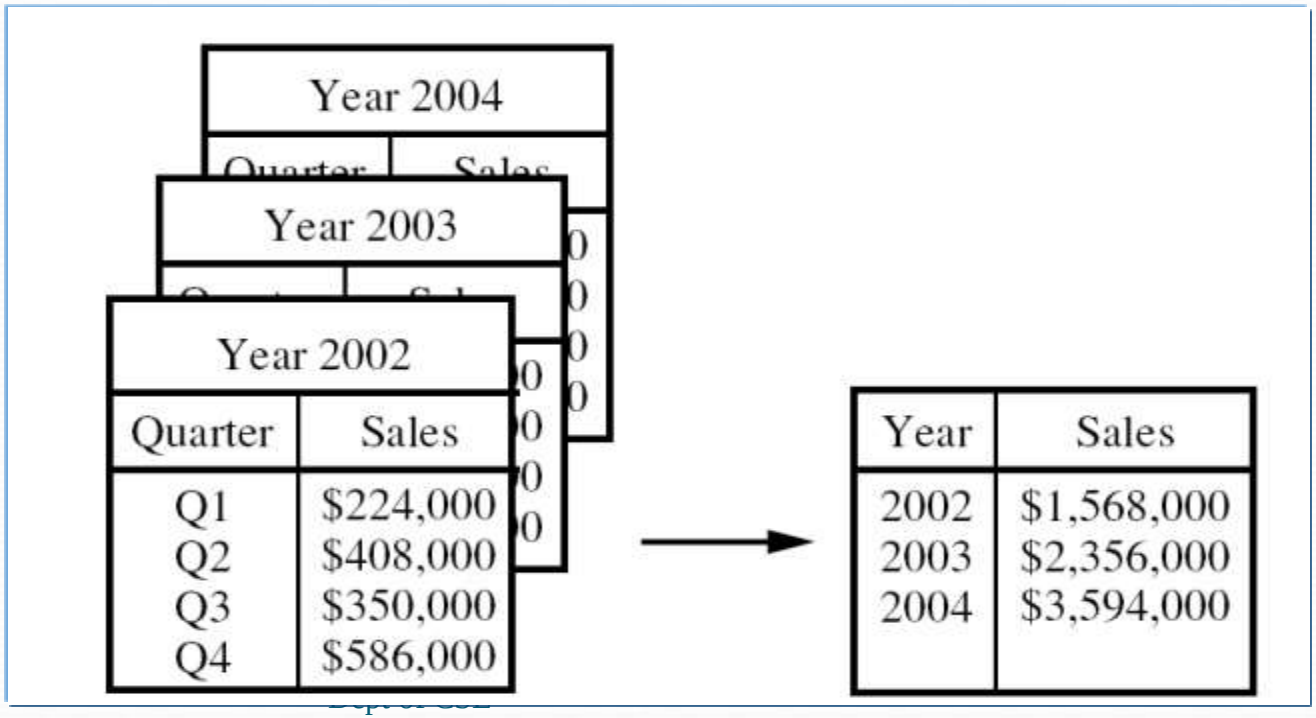
Stratified Sample:

- Simple random sampling may have very poor performance in the presence of skew
- Approximate the percentage of each class (or subpopulation of interest) in the overall database
- Used in conjunction with skewed data



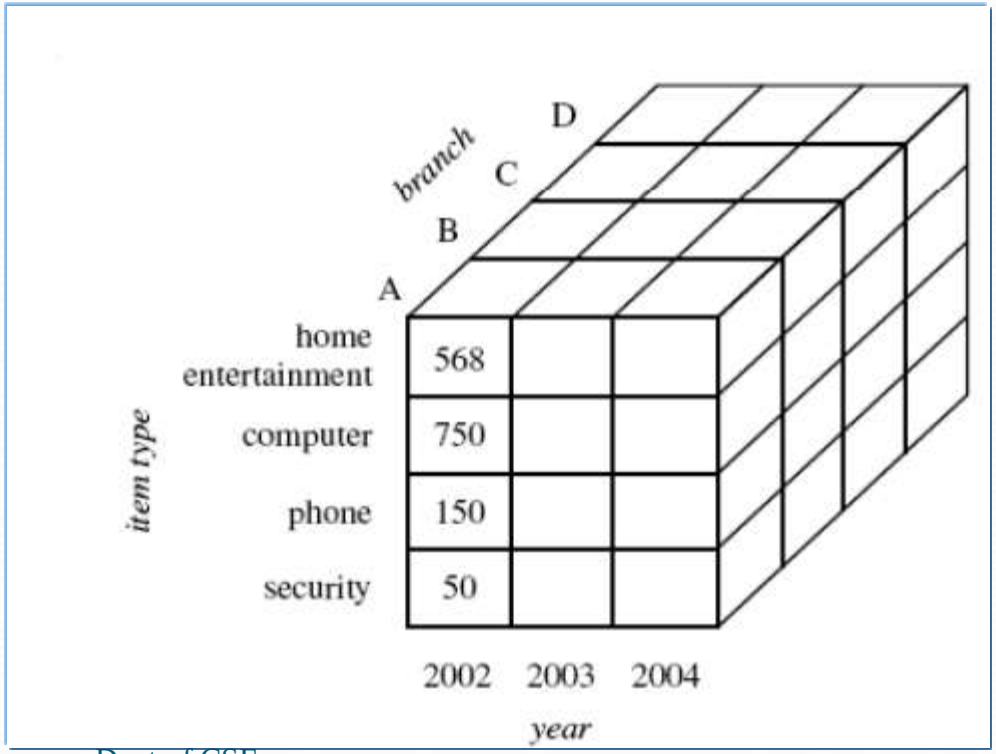
Data Reduction: Data Cube Aggregation

- On the left, the sales are shown per quarter. On the right, the data are aggregated to provide the annual sales
- Sales data for a given branch of AllElectronics for the years 2002 to 2004.



Data Reduction:Data Cube Aggregation

- Data cubes store multidimensional aggregated information.
- Data cubes provide fast access to precomputed, summarized data, thereby benefiting on-line analytical processing as well as data mining.
- A data cube for sales at AllElectronics.



Data Reduction: Data Cube Aggregation

Base cuboid:

- The cube created at the lowest level of abstraction is referred to as the base cuboid.
- The base cuboid should correspond to an individual entity of interest, such as sales or customer.

Apex cuboid:

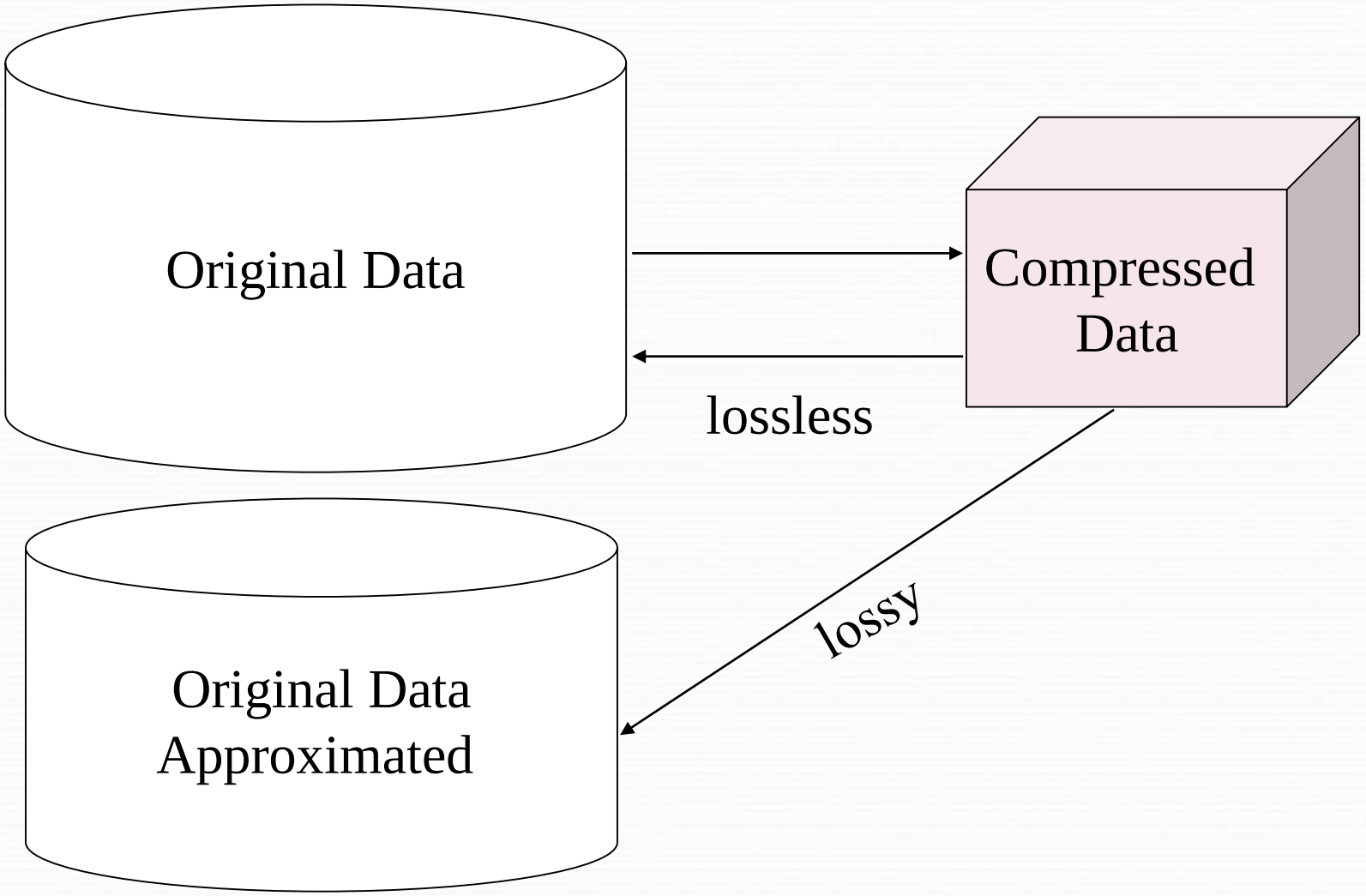
- A cube at the highest level of abstraction is the apex cuboid.

- For the sales data, the apex cuboid would give one total the total sales.

Data Reduction 3: 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 vary slowly with time
- Dimensionality and numerosity reduction may also be considered as forms of data compression

Data Compression



Data Transformation and Data Discretization:

- Transforming or Consolidating data into mining suitable form is known as Data Transformation.

- Data Transformation strategies:
  1. Smoothing
  2. Attribute/feature construction
  3. Aggregation
  4. Normalization
  5. Discretization
  6. Concept hierarchy generation for nominal data

Data Transformation

1. **Smoothing:**

   which works to remove noise from data. Techniques include binning, clustering, regression.

2. **Attribute construction (or feature construction):**

   where new attributes are constructed and added from the given of attributes to help and simplify the mining process.

3. **Aggregation:**

   where summary or aggregation operations are applied to the data. **For example**, daily Sales data may be aggregated to compute monthly & annual total amounts.

4. **Normalization:** Where the attribute data are scaled so as to fall within a specified range such as, -1.0 to 1.0 or 0.0 to 1.0.

Data Transformation

5. Discretization:

Where the 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 raw values of a The data where low-level or "primitive" data are placed by higher-level concepts through the use of concept hierarchies.

6. Concept hierarchy generation for nominal data:

where the attributes of lower level concepts like street can be generalized to higher-level concept city or country.

Data Transformation by Normalization:

- Data normalization involves converting all data variable into a given range.
- Techniques that are used for normalization are:
  - min-max normalization
  - z-score normalization
  - normalization by decimal scaling
- **Min-Max Normalization:**
  - This transforms the original data linearly.
  - Suppose that: min_A is the minima and max_A is the maxima of an attribute, P

$$v' = \frac{v - min_A}{max_A - min_A}(new\_max_A - new\_min_A) + new\_min_A$$

Where, v is the value you want to plot in the new range. v' is the new value you get after normalizing the old value.

Ex. Let income range \$12,000 to \$98,000 normalized to [0.0, 1.0]. Then \$73,000 is mapped to $\frac{73{,}600 - 12{,}000}{98{,}000 - 12{,}000}(1.0 - 0) + 0 = 0.716$

Data Transformation

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

$$v' = \frac{v - mean_A}{stand\_dev_A}$$

Ex. Let μ = 54,000, σ = 16,000. Then $\frac{73{,}600 - 54{,}000}{16{,}000} = 1.225$

- **Decimal Scaling:**
  - It normalizes the values of an attribute by changing the position of their decimal points
  - The number of points by which the decimal point is moved can be determined by the absolute maximum value of attribute A.

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

ILO

- Illustrate Discritization methods

Discretization

- **Data discretization** is defined as a process of converting continuous **data** attribute values into a finite set of intervals and associating with each interval some specific **data** value
-
- **Discretization** is the process of putting values into buckets so that there are a limited number of possible states. ... If your **data mining** solution uses relational **data**, you can control the number of buckets to use for grouping **data** by setting the value of the DiscretizationBucketCount property

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
  - Numeric—real numbers, **e.g.**, integer or real numbers
- **Discretization**: Divide the 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
    - Top-down split, unsupervised
  - Histogram analysis
    - Top-down split, unsupervised
  - 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)

Discritization by binning

- Binning is a top-down splitting technique based on a specified number of bins.
- Binning methods methods are also used as discretization methods for data reduction and concept hierarchy generation.
- For example, attribute values can be discretized by applying equal-width or equal-frequency binning, and then replacing each bin value by the bin mean or median. These techniques can be applied recursively to the resulting partitions to generate concept hierarchies.
- Binning does not use class information and is therefore an unsupervised discretization technique.

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 (**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

Discretization by Histogram Analysis

- histogram analysis is an unsupervised discretization technique because it does not use class information.
- A histogram partitions the values of an attribute, *A*, into disjoint ranges called *buckets* or *bins*.
- In an *equal-width* histogram, for example, the values are partitioned into equal-size partitions or ranges
- With an *equal-frequency* histogram, the values are partitioned so that, ideally, each partition contains the same number of data tuples.

Discretization by Histogram Analysis

- The histogram analysis algorithm can be applied recursively to each partition in order to automatically generate a multilevel concept hierarchy, with the procedure terminating once a prespecified number of concept levels has been reached.

- A *minimum interval size* can also be used per level to control the recursive procedure.

Discretization by clustering

- Cluster analysis is a popular data discretization method.
- A clustering algorithm can be applied to discretize a numeric attribute, *A*, by partitioning the values of *A* into clusters or groups..
- Clustering can be used to generate a concept hierarchy for *A* by following either a top-down splitting strategy or a bottom-up merging strategy, where each cluster forms a node of the concept hierarchy.
- Each initial cluster or partition may be further decomposed into several subclusters, forming a lower level of the hierarchy, then clusters are formed by repeatedly grouping neighboring clusters in order to form higher-level concepts.

Discretization by Decision Tree

- Techniques to generate decision trees for classification can be applied to discretization.

- decision tree approaches to discretization are supervised, that is, they make use of class label information.

- Because decision tree–based discretization uses class information, it is more likely that the interval boundaries (split-points) are defined to occur in places that may help improve classification accuracy.

Discretization by Classification & Correlation Analysis

- Classification (e.g., decision tree analysis)
  - Supervised: Given class labels, e.g., cancerous vs. benign
  - Using *entropy* to determine split point (discretization point)
  - Top-down, recursive split
- Correlation analysis (e.g., Chi-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

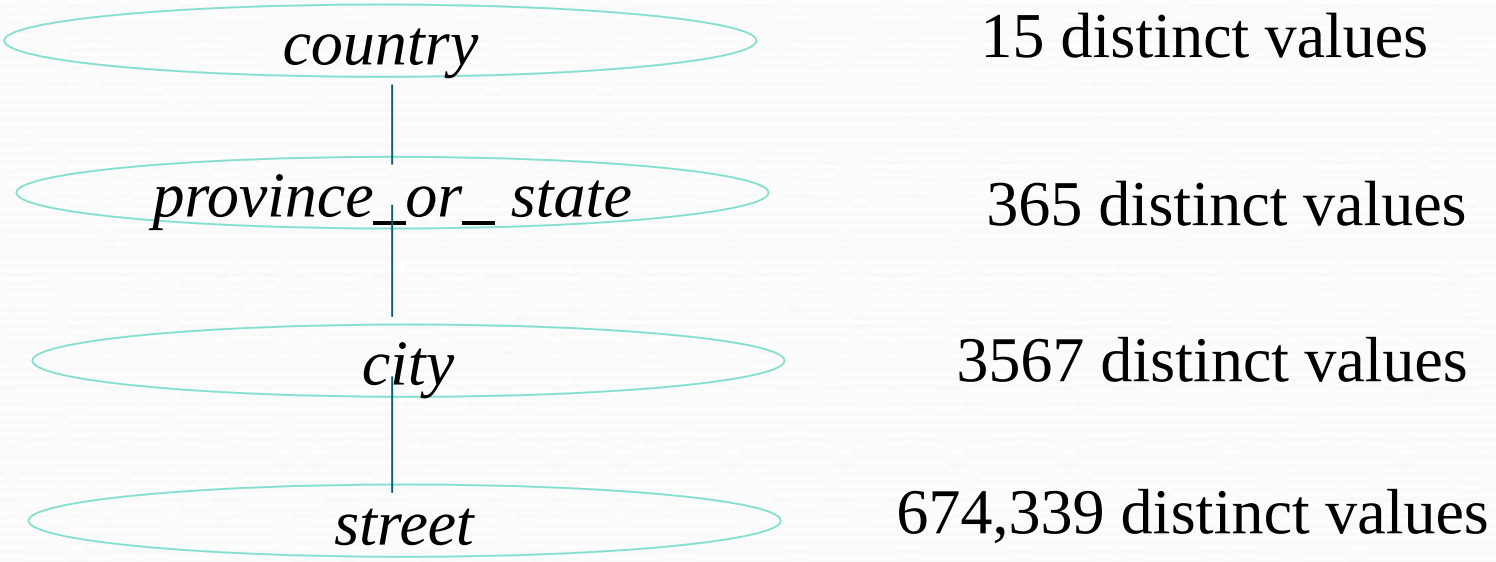
- **Concept hierarchy** organizes concepts (i.e., attribute values) hierarchically and is usually associated with each dimension in a data warehouse
- Concept hierarchies facilitate drilling and rolling in data warehouses to view data in multiple granularity
- Concept hierarchy formation: Recursively reduce the data by collecting and replacing low level concepts (such as numeric values for *age*) by higher level concepts (such as *youth*, *adult*, or *senior*)
- Concept hierarchies can be explicitly specified by domain experts and/or data warehouse designers
- Concept hierarchy can be automatically formed for both numeric and nominal data. For numeric 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
  - *street* < *city* < *state* < *country*
- Specification of a hierarchy for a set of values by explicit data grouping
  - {Urbana, Champaign, Chicago} < Illinois
- Specification of only a partial set of attributes
  - E.g., only *street* < *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: {*street, city, state, country*}

Automatic Concept Hierarchy Generation

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



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