

CST466

DATA MINING

MODULE-2

Module - 2 (Data Preprocessing)

Data Preprocessing-Need of data preprocessing, Data Cleaning- Missing values, Noisy data, Data Integration and Transformation, Data Reduction-Data cube aggregation, Attribute subset selection, Dimensionality reduction, Numerosity reduction, Discretization and concept hierarchy generation.

DATA PREPROCESSING:

- Today's 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, heterogeneous sources.
- Low-quality data will lead to low-quality mining results.
- Following are the major steps in data preprocessing;
 - **Data cleaning** can be applied to remove noise and correct inconsistencies in data.
 - **Data integration** merges data from multiple sources into a coherent data store such as a data warehouse.
 - **Data reduction** can reduce data size by, for instance, aggregating, eliminating redundant features, or clustering.
 - **Data transformations** (e.g., normalization) may be applied, where data are scaled to fall within a smaller range like 0.0 to 1.0.
- These techniques are not mutually exclusive; they may work together.
 - **Eg:** Data cleaning can involve transformations to correct wrong data, such as by transforming all entries for a date field to a common format.

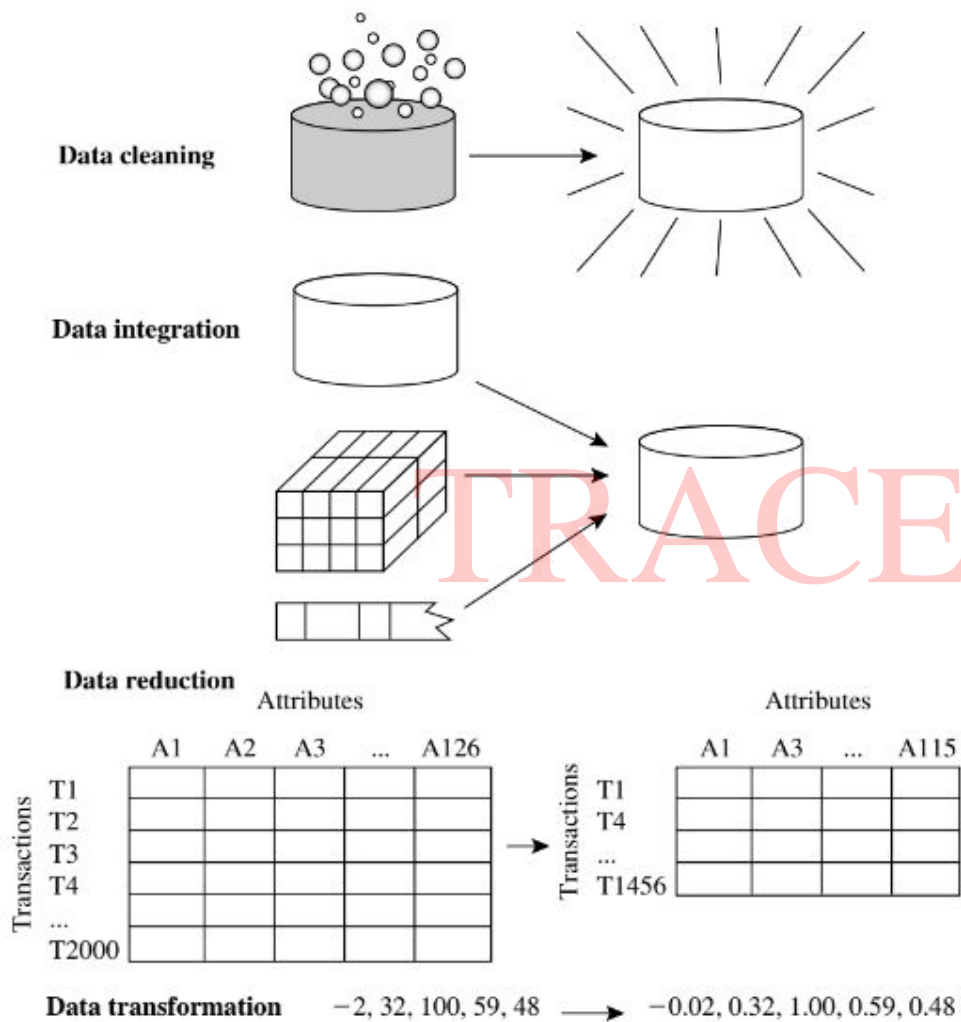


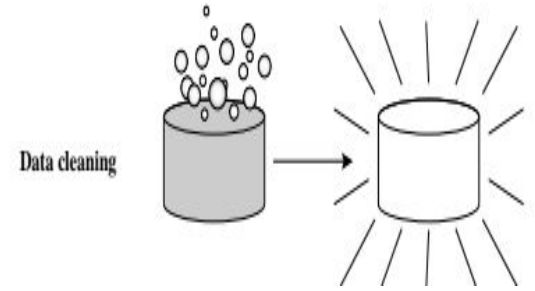
Fig: Forms of data preprocessing

NEED OF DATA PREPROCESSING:

- Today's 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, heterogeneous sources.
- **Low-quality data will lead to low-quality mining results.**
 - Eg: Duplicate or missing data may cause incorrect or even misleading results.
- Data warehouse require consistent integration of quality data.
- So, it is important to make sure that data is preprocessed and it turn have good quality.

DATA CLEANING:

- Real-world data tend to be **incomplete, noisy, and inconsistent**.
- Data cleaning (or data cleansing) routines attempt to;
 - **Fill in missing values**
 - **Smooth out noise while identifying outliers**
 - **Correct inconsistencies in the data.**



Missing Values:

- Many tuples may not have recorded value for several attributes.

Eg: Consider an electronic store. It may have *sales* and *customer* data. Many tuples have no recorded value for several attributes such as customer *income*.

- To fill the missing values for this attribute, we can use the following methods;
 1. Ignore the tuple
 2. Fill in the missing value manually
 3. Use a global constant to fill in the missing value
 4. Use a measure of central tendency for the attribute (e.g., the mean or median) to fill in the missing value.
 5. Use the attribute mean or median for all samples belonging to the same class as the given tuple
 6. Use the most probable value to fill in the missing value

1.Ignore the tuple:

- Usually done when the class label is missing.
- Not very effective, unless the tuple contains several attributes with missing values.
- It is especially 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.
- Such data could have been useful to the task at hand.

2.Fill in the missing value manually:

- 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 $-\infty$.
- 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.” Hence, although this method is simple, it is not foolproof.

4.Use a measure of central tendency for the attribute (e.g., the mean or median) to fill in the missing value:

- The measure of central tendency will be used to fill the missing value.

5. 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.

6. Use the most probable value to fill in the missing value:

- This may be determined with regression, inference-based tools using a Bayesian formalism, or decision tree induction.
 - For example, using the other customer attributes in the data set, we may construct a decision tree to predict the missing values for *income*.

Note:

- *Methods 3 through 6 bias the data — the filled-in value may not be correct.*
- *Method 6 is a popular strategy.*
- *In comparison to the other methods, it uses the most information from the present data to predict missing values.*

Noisy Data:

- Noise is a random error or variance in a measured variable.

Eg: Consider a numeric attribute named *price*. We have to “smooth” out the data to remove the noise.

- Following are the common **data smoothing techniques** that can be used to remove noise;
 1. **Binning**
 2. **Regression**
 3. **Outlier analysis**

TRACE KTU

Binning:

- Binning methods smooth a sorted data value by consulting its “neighborhood,” that is, the values around it.
- The sorted values are distributed into a number of “buckets,” or bins.
- Because binning methods consult the neighborhood of values, they perform local smoothing.

- The following figure illustrates some binning techniques.

Partition into bins:

- The data for price are first sorted and then partitioned into equal-frequency bins of size 3 (i.e., each bin contains three values).

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 boundaries:

Bin 1: 4, 4, 15

Bin 2: 21, 21, 24

Bin 3: 25, 25, 34

Smoothing by bin means:

- Each value in a bin is replaced by the mean value of the bin.
- Eg: The mean of the values 4, 8, and 15 in Bin-1 is 9.
 - Therefore, each original value in this bin is replaced by the value 9.
- Similarly, smoothing by bin medians can be employed, in which 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.

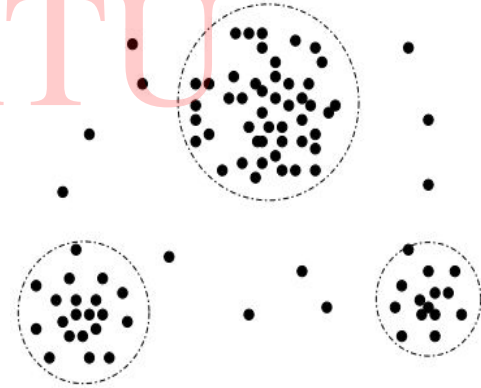
Regression:

- Regression is 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* is an extension of linear regression, where more than two attributes are involved and the data are fit to a multidimensional surface.

TRACE KTU

Outlier analysis:

- Outliers may be detected by clustering, for example, where similar values are organized into groups, or “clusters.”
- Intuitively, values that fall outside of the set of clusters may be considered outliers.



A 2-D customer data plot with respect to customer locations in a city, showing three data clusters. Outliers may be detected as values that fall outside of the cluster sets.

DATA REDUCTION:

- Data reduction techniques can be applied **to obtain a reduced representation of the data set** that is much smaller in volume, yet closely maintains the integrity of the original data.
 - ie, mining on the reduced data set should be more efficient yet produce the same (or almost the same) analytical results.
- Following are the common 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/dimensions are detected and removed.
 - **Dimensionality reduction**
 - Encoding mechanisms are used to reduce the data set size.
 - **Numerosity reduction**
 - The data are replaced or estimated by alternative, smaller data representations such as;
 - Parametric models (Store only the model parameters instead of the actual data)

Or

 - Non-parametric models (such as clustering, sampling, and the use of histograms)
 - **Discretization & concept hierarchy generation**
 - Raw data values for attributes are replaced by ranges or higher conceptual levels.

DATA CUBE AGGREGATION:

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

Eg:

- Imagine that we have collected the data for some analysis.
- The data consist of the *AllElectronics* sales per quarter, for the years 2008 to 2010.
- Suppose, we are interested in the annual sales (total per year), rather than the total per quarter.
- Thus, the data can be aggregated so that the resulting data summarize the total sales per year instead of per quarter.
- This aggregation is illustrated in the figure.
- The resulting data set is smaller in volume, without loss of information necessary for the analysis task.

Year 2010	
Quarter	Sales
Year 2009	
Year 2008	
Quarter	Sales
Q1	\$224,000
Q2	\$408,000
Q3	\$350,000
Q4	\$586,000

Year	Sales
2008	\$1,568,000
2009	\$2,356,000
2010	\$3,594,000

Fig:

- Sales data for a given branch of an Electronics Shop for the years 2008 through 2010.
- On the left, the sales are shown per quarter.
- On the right, the data are aggregated to provide the annual sales.

Note:

Introduction to Attribute Subset Selection:

- Datasets for analysis may contain hundreds of attributes, many of which may be irrelevant to the mining task or redundant.

Eg: Suppose,

Given a data mining task:

Classify customers based on whether or not they are likely to purchase a popular new CD at *AllElectronics* when notified a sale.

- Attributes such as the *customer's telephone number* are likely to be irrelevant.
- Attributes such as *age* or *music_taste* are likely to be relevant.
- Although it may be possible for a domain expert to pick out some of the useful attributes, this can be a difficult and time-consuming task.
- Leaving out relevant attributes or keeping irrelevant attributes may be detrimental and may cause confusion for the mining algorithm employed.
- This can result in discovered patterns of poor quality.
- In addition, the added volume of irrelevant or redundant attributes can slow down the mining process.

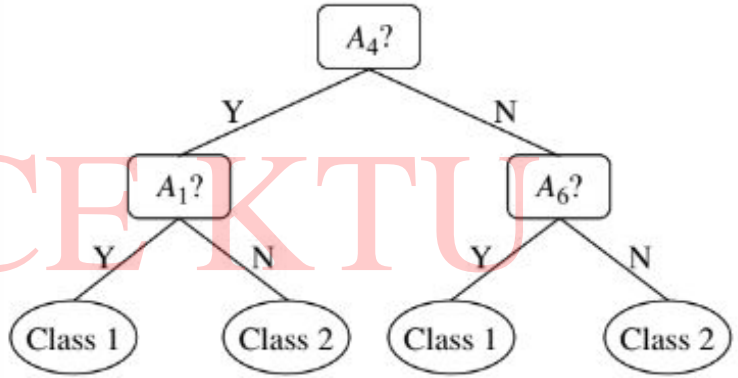
ATTRIBUTE SUBSET SELECTION:

- Attribute subset selection **reduces the data set size by removing irrelevant or redundant attributes** (or dimensions).
- The goal of attribute subset selection is 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.**
- Mining on a reduced set of attributes has an additional benefit:
 - It reduces the number of attributes appearing in the discovered patterns.
 - This helps to make the patterns easier to understand.

“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 and the number of data classes increase.
- Therefore, **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.
- Their strategy is to make a locally optimal choice in the hope that this will lead to a globally optimal solution.
- Such greedy methods are effective in practice and may come close to estimating an optimal solution.

The following table shows some basic heuristic methods of 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: $\{A_1, A_2, A_3, A_4, A_5, A_6\}$</p> <p>$\Rightarrow \{A_1, A_3, A_4, A_5, A_6\}$ $\Rightarrow \{A_1, A_4, A_5, A_6\}$ \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>

Greedy (heuristic) methods for attribute subset selection.

Stepwise forward selection:

- The procedure 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.

Stepwise backward elimination:

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

Combination of forward selection and backward elimination:

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

Decision tree induction:

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

- When decision tree induction is used for attribute subset selection, a tree is constructed from the given data.
- At each node, the algorithm chooses the “best” attribute to partition the data into individual classes.
- **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.**

Note:

- *The stopping criteria for the methods may vary.*
- *The procedure may employ a threshold on the measure used to determine when to stop the attribute selection process.*

NUMEROSITY REDUCTION:

- Numerosity reduction techniques **replace the original data volume by alternative, 'smaller' forms of data representation.**
- These techniques are of two types;
 - **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.
 - Eg: **Regression and log-linear models**
 - **Non-parametric methods**
 - Used for storing reduced representations of the data.
 - Eg: **Histograms, Clustering, Sampling**

Regression and Log-Linear Models:

- Regression and log-linear models can be used to approximate the given data.
- **Linear Regression:**
 - **The data are modeled to fit a straight line.**
 - Eg: With the equation $y = wx + b$, a random variable, y (called a response variable), can be modeled as a linear function of another random variable, x (called a predictor variable).

- In the context of data mining, x and y are numeric database attributes.
- The coefficients, w and b are called regression coefficients.
 - w specify the slope of the line.
 - b specify the y-intercept.
- These coefficients can be solved for by the method of least squares, which minimizes the error between the actual line separating the data and the estimate of the line.
- **Multiple linear regression:**
 - It is an extension of linear regression, which **allows a response variable, y , to be modeled as a linear function of two or more predictor variables.**
- **Log-linear models:**
 - 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 therefore also useful for dimensionality reduction (since the lower-dimensional points together typically occupy less space than the original data points) and data smoothing (since aggregate estimates in the lower-dimensional space are less subject to sampling variations than the estimates in the higher-dimensional space).

Histograms:

- Histograms **use binning to approximate data distributions.**
- **A histogram for an attribute, A, partitions the data distribution of A into disjoint subsets, referred to as buckets or bins.**
- If each bucket represents only a single attribute–value/frequency pair, the buckets are called **singleton buckets.**
- Other than singleton buckets, there are **buckets which denotes a continuous value range**, for the given attribute.

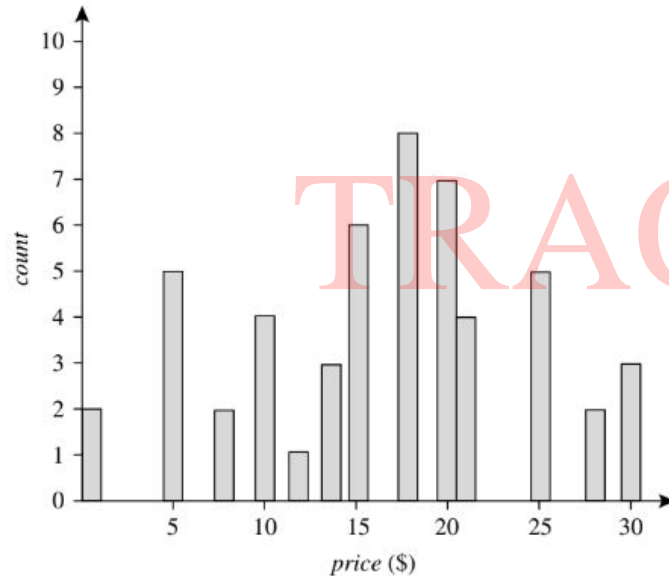
TRACE KTU

Eg: The following data are a list of AllElectronics *prices* for commonly sold items. 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, 20, 20, 20, 20, 20, 20, 21, 21, 21, 21, 25, 25, 25, 25, 25, 28, 28, 30, 30, 30.

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, 20, 20, 20, 20, 20, 20, 21, 21, 21, 21, 25, 25, 25, 25, 25, 28, 28, 30, 30, 30.

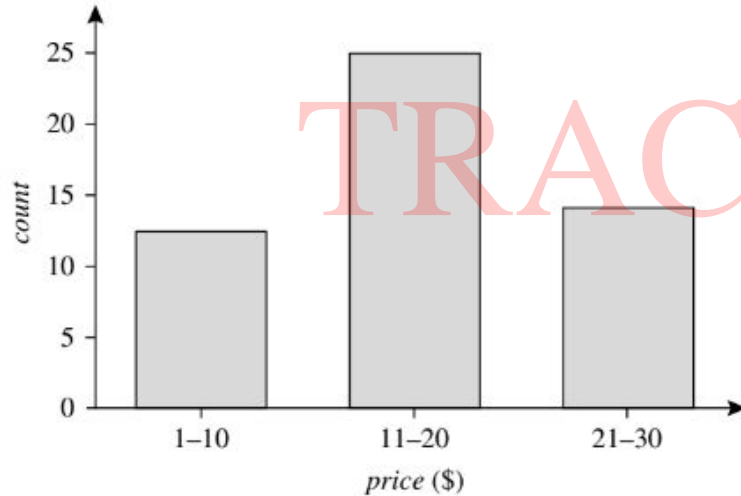
- Following shows a histogram for the data using **singleton buckets** - each bucket represents only a single attribute-value/frequency pair.



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

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.

- To further reduce the data, it is common to have **each bucket denote a continuous value range**, for the given attribute.
- Here, each bucket represents a different \$10 range for price.



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

“How are the buckets determined and the attribute values partitioned?”

- There are several partitioning rules, including the following:
 - **Equal-width:** ✓
 - In an equal-width histogram, **the width of each bucket range is uniform.**
 - **Bin width = (max_value-min_value)/No.of bins.**
 - **Equal-frequency (or equal-depth):** ✓
 - In an equal-frequency histogram, **the buckets are created so that, roughly, the frequency of each bucket is constant** (i.e., each bucket contains roughly the same number of contiguous data samples).
 - **V-Optimal:**
 - If we consider all of the possible histograms for a given number of buckets, the V-Optimal histogram is the one with the least variance.
 - Histogram variance is a weighted sum of the original values that each bucket represents, where bucket weight is equal to the number of values in the bucket.
 - **MaxDiff:**
 - In a MaxDiff histogram, we consider the difference between each pair of adjacent values.
 - A bucket boundary is established between each pair for pairs having the $\beta-1$ largest differences, where β is the user-specified number of buckets.

- Histograms are highly effective at approximating both sparse and dense data, as well as highly skewed and uniform data.
- Multidimensional histograms can capture dependencies between attributes.
- These histograms have been found effective in approximating data with up to five attributes.

Question 1:

Q: Suppose a group of 12 sales *price* records has been sorted as follows:

5, 10, 11, 13, 15, 35, 50, 55, 72, 92, 204, 215.

Partition them into three bins by each of the following methods:

- (a) equal-frequency (equi-depth) partitioning.
- (b) equal-width partitioning.

Given sorted values for *price*: 5, 10, 11, 13, 15, 35, 50, 55, 72, 92, 204, 215

Solution:

(a) equal-frequency (equi-depth) partitioning.

In an equal-depth histogram, the buckets are created so that, roughly, the **frequency of each bucket is constant**.

Partition the data into equi-depth bins of depth 4:

Bin 1: 1: 5, 10, 11, 13

Bin 2: 15, 35, 50, 55

Bin 3: 72, 92, 204, 215

TRACE KTU

(b) equal-width partitioning.

In an equal-width histogram, **the width of each bucket range is uniform**.

Partitioning the data into 3 equi-width bins will require the width to be $(215-5)/3 = 70$. [Bin width = (max_value-min_value)/No.of bins.]

Bin 1: 5, 10, 11, 13, 15, 35, 50, 55, 72

Bin 2: 92

Bin 3: 204, 215

Explanation to equi-width partitioning:

- We have bin-width (w)=70
- According to this, we divide the data into three categories:

○ 5 to 75	$[\min, \min+w) = [5, 75)$	$[\min, \min+w-1] = [5, 74]$
○ 75 to 145	$[\min+w, \min+2w) = [75, 145)$	$[\min+w, \min+2w-1] = [75, 144]$
○ 145 to 215	$[\min+2w, \max] = [145, 215]$	$[\min+2w, \max] = [145, 215]$
- Now we keep the numbers in the above ranges.
- Thus,
 - Bin1: 5,10,11,13,15,35,50,55,72
 - Bin2: 92
 - Bin3: 204,215

Question 2:

- Suppose a group of *price* records has been sorted as follows:

Given data : 10,15,16,18,20,30,35,42,48,50,52,55

Partition them into four bins by equi-width binning.

Solution:

$x = 4$	
$w = (55-10)/4 = 12$	
$[\min, \min+w-1]$	[10, 21]
$[\min+w, \min+2*w-1]$	[22, 33]
$[\min+2*w, \min+3*w-1]$	[34, 45]
$[\min+3*w, \max]$	[46, 55]

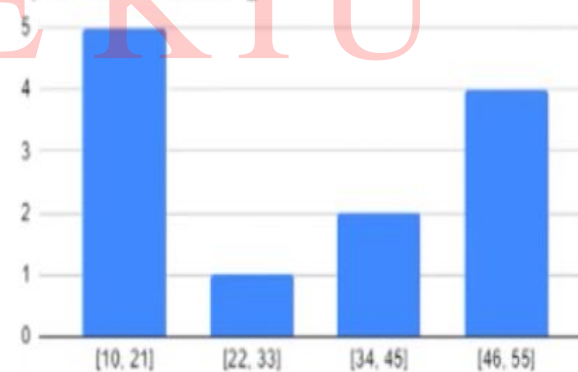
Sorted values: 10,15,16,18,20,30,35,42,48,50,52,55

HISTOGRAM

Equi-width binning:

- **Bin 1:** 10,15,16,18,20
- **Bin 2:** 20
- **Bin 3:** 35,42
- **Bin 4:** 48,50,52,55

Equal Width Binning



Clustering:

- Clustering techniques consider data tuples as **objects**.
- They **partition the objects into** groups, or **clusters** in such a way 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;
 - Diameter of the cluster- the maximum distance between any two objects in the cluster.
 - or
 - Centroid distance - the average distance of each cluster object from the cluster centroid.
- **In data reduction, the cluster representations of the data are used to replace the actual data.**
- The effectiveness of this technique depends on the data’s nature.
- It is very effective for data that can be organized into distinct clusters.

Sampling:

- Sampling can be used as a data reduction technique because it **allows a large data set to be represented by a much smaller random data sample** (or subset).
- Following are the most common ways that we could sample data set D for data reduction;
 - **Simple random sample without replacement (SRSWOR) of size s**
 - **Simple random sample with replacement (SRSWR) of size s**
 - **Cluster sample**
 - **Stratified sample**

Note:

- *An advantage of sampling for data reduction is that the cost of obtaining a sample is proportional to the size of the sample, s , as opposed to N , the data set size.*
 - *Hence, sampling complexity is potentially sublinear to the size of the data.*
- *Other data reduction techniques can require at least one complete pass through D .*

- Suppose that a large data set, D , contains N tuples.

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

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

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

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

Cluster sample:

- If the tuples in D are grouped into M mutually disjoint “clusters,” then an SRS of s clusters can be obtained, where $s < M$.
- For example, tuples in a database are usually retrieved a page at a time, so that each page can be considered a cluster.
- A reduced data representation can be obtained by applying, say, SRSWOR to the pages, resulting in a cluster sample of the tuples.
- Other clustering criteria conveying rich semantics can also be explored.
- For example, in a spatial database, we may choose to define clusters geographically based on how closely different areas are located.

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.
- This helps ensure a representative sample, especially when the data are skewed.
- For example, a stratified sample may be obtained from customer data, where a stratum is created for each customer age group.
- In this way, the age group having the smallest number of customers will be sure to be represented.

TRACE KTU

DIMENSIONALITY REDUCTION:

- In dimensionality reduction, **data encoding or transformations are applied** so as **to obtain a reduced or compressed representation of the original data.**
- Dimensionality reduction **reduces the number of random variables or attributes under consideration.**
- 2 types;
 - **Lossless dimensionality reduction:**
 - If the original data can be reconstructed from the compressed data without any loss of information, the data reduction is called lossless.
 - **Lossy dimensionality reduction:**
 - Only an approximation of the original data can be reconstructed from the compressed data.
- Following are the two popular and effective methods of lossy dimensionality reduction;
 - **Wavelet transforms**
 - **Principal Component Analysis (PCA)**

Wavelet Transforms:

- The discrete wavelet transform (DWT) is a linear signal processing technique that, when **applied to a data vector X , transforms it to a numerically different vector, X'** , of wavelet coefficients.
- The **two vectors are of the same length**.
- When applying this technique to data reduction, we consider each tuple as an n -dimensional data vector, that is, $X = (x_1, x_2, \dots, x_n)$, depicting n measurements made on the tuple from n database attributes.

“How can this technique be useful for data reduction if the wavelet transformed data are of the same length as the original data?”

- The wavelet transformed data can be truncated.
- A compressed approximation of the data can be retained by storing only a small fraction of the strongest of the wavelet coefficients.

Eg: 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.

- The technique also works to remove noise without smoothing out the main features of the data, making it effective for data cleaning as well.
- Given a set of coefficients, an approximation of the original data can be constructed by applying the inverse of the DWT used.

DWT v/s DFT:

- The DWT is closely related to the discrete Fourier transform (DFT), a signal processing technique involving sines and cosines.
- However DWT achieves better lossy compression.
 - ie, if the same number of coefficients is retained for a DWT and a DFT of a given data vector, the DWT version will provide a more accurate approximation of the original data.
 - Hence, for an equivalent approximation, the DWT requires less space than the DFT.
 - Unlike the DFT, wavelets are quite localized in space, contributing to the conservation of local detail.
- There is only one DFT, yet there are several families of DWTs.
- Popular wavelet transforms include the Haar-2, Daubechies-4, and Daubechies-6.

Principal Components Analysis: (PCA)

- Also called Karhunen-Loeve, or K-L, method.
- PCA aims to **find the directions of maximum variance in high-dimensional data and projects it onto a new subspace with equal or fewer dimensions than the original one.**
- Suppose that the data to be reduced consist 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 **$k \leq n$.**
- **The original data are thus projected onto a much smaller space,** resulting in dimensionality reduction.
- PCA often reveals relationships that were not previously suspected and thereby allows interpretations that would not ordinarily result.

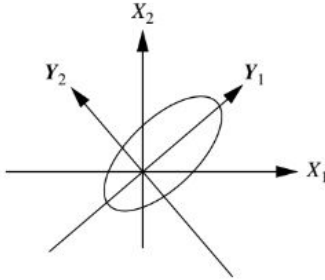
PCA v/s Attribute subset selection:

- *In attribute subset selection, we reduced the attribute set size by retaining a subset of the initial set of attributes whereas PCA “combines” the essence of attributes by creating an alternative, smaller set of variables.*
- *PCA - lossy reduction.*
- *Attribute subset selection - lossless*

PCA - Procedure:

1. 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 smaller domains.
2. PCA computes k orthonormal vectors that provide a basis for the normalized input data.
 - These are unit vectors that each point in a direction perpendicular to the others.
 - These vectors are referred to as the principal components.
 - The input data are a linear combination of the principal components.
3. 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.
 - ie, 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.

Eg: Figure shows the first two principal components, Y_1 and Y_2 , for the given set of data originally mapped to the axes X_1 and X_2 . This information helps identify groups or patterns within the data.



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

4. 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 TRANSFORMATION:

- In data transformation, **data are transformed or consolidated into forms appropriate for mining.**
- Strategies for data transformation include the following:
 - **1. Smoothing**
 - To remove noise from the data.
 - Techniques include binning, regression, and clustering.
 - **2. Attribute construction** (or feature construction)
 - New attributes are constructed and added from the given set of attributes to help the mining process.
 - **3. Aggregation**
 - Summary or aggregation operations are applied to the data.
 - Eg: The daily sales data may be aggregated so as to compute monthly and annual sales.
 - This step is typically used in constructing a data cube for data analysis at multiple abstraction levels.
 - **4. Normalization**
 - The 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 .
 - **5. Generalization** of the data:
 - Low-level data are replaced by higher-level concepts through the use of concept hierarchies.
 - Eg: Categorical attributes, like street, can be generalized to higher-level concepts like city or country.
 - Similarly, values for numerical attributes, like age, may be mapped to higher-level concepts like youth, middle-aged, and senior.

Note:

Introduction to Normalization:

- The measurement unit used can affect the data analysis.
- Eg: Changing measurement units from meters to inches for height, or from kilograms to pounds for weight, may lead to very different results.
- In general, **expressing an attribute in smaller units will lead to a larger range for that attribute**, and thus tend to give such an attribute greater effect or “weight.”
- To help avoid dependence on the choice of measurement units, the data should be normalized or standardized.
- This involves **transforming the data to fall within a smaller or common range** such as $[-1,1]$ or $[0.0, 1.0]$.
- **Normalizing the data attempts to give all attributes an equal weight.**

Normalization:

- Involves **transforming the data to fall within a smaller or common range** such as $[-1,1]$ or $[0.0, 1.0]$.
- Following are the common methods for data normalization.
 - **Min-max normalization**
 - **z-score normalization**
 - **Normalization by decimal scaling**
- Let **A** be a numeric attribute with **n** observed values, **v1, v2,..., vn**.

Min-max normalization:

- Min-max normalization performs a **linear transformation** on the original data.
- Min-max normalization preserves the relationships among the original data values.
- Suppose that **min_A** and **max_A** are the minimum and maximum values of an attribute, **A**.
- Min-max normalization maps a value, **v_i** of A to **v'_i** in the range **[new_min_A, new_max_A]** by computing;

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

Sample question:

Q: Suppose the min and max values for the attribute *income* are \$12000 and \$98000 respectively. Apply min-max normalization to map the attribute *income* with value \$73600 to the range [0.0, 1.0].

Ans:

- Min-max normalization maps a value, v_i of A to v_i' in the range $[\text{new_min}_A, \text{new_max}_A]$ by computing;

$$v_i' = \frac{v_i - \text{min}_A}{\text{max}_A - \text{min}_A} (\text{new_max}_A - \text{new_min}_A) + \text{new_min}_A.$$

$$v_i' = \frac{73600 - 12000}{98000 - 12000} (1.0 - 0.0) + 0.0$$

$$v_i' = 0.716$$

z - score normalization: (or zero-mean normalization)

- The values for an attribute, A, are **normalized based on the mean** (i.e., average) **and standard deviation of A**.
- A value, v_i of 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, respectively, of attribute A.

- This method of normalization is **useful when the actual minimum and maximum of attribute A are unknown**, or when there are outliers that dominate the min-max normalization.

- A variation of this z-score normalization replaces the standard deviation σ_A by the mean absolute deviation of A denoted by s_A
- The mean absolute deviation of A, denoted s_A , is;

$$s_A = \frac{1}{n}(|v_1 - \bar{A}| + |v_2 - \bar{A}| + \dots + |v_n - \bar{A}|).$$

- Thus, z-score normalization using the mean absolute deviation is;

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

- **The mean absolute deviation, s_A , is more robust to outliers than the standard deviation, σ_A .**
- When computing the mean absolute deviation, the deviations from the mean (i.e., $|x_i - \bar{x}|$) are not squared; hence, the effect of outliers is somewhat reduced.

Note:

- The formulas for the variance and the standard deviation for both population and sample data set are given below;

	Population	Sample
Variance	$\sigma^2 = \frac{\sum_{i=1}^N (x_i - \mu)^2}{N}$	$S^2 = \frac{\sum_{i=1}^n (x_i - \bar{x})^2}{n - 1}$
Standard deviation	$\sigma = \sqrt{\frac{\sum_{i=1}^N (x_i - \mu)^2}{N}}$	$S = \sqrt{\frac{\sum_{i=1}^n (x_i - \bar{x})^2}{n - 1}}$

Sample question:

Q: Suppose that the mean and standard deviation of the values for the attribute *income* are \$54,000 and \$16,000, respectively. Apply z-score normalization to map the attribute *income* with value \$73,600.

Ans:

- A value, v_i of A is normalized to v'_i by computing;

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

$$v'_i = \frac{73600 - 54000}{16000}$$

$$v'_i = 1.225$$

Normalization by decimal scaling:

- Normalization is done 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 $\max(|v'_i|) < 1$.

Sample question:

Q: Suppose that the recorded values of A range from -986 to 917. Normalize by decimal scaling.

Ans:

Here, $j=3$ = no. of digits in values -986 and 917

We therefore divide each value by 1000 (i.e., $j = 3$) so that -986 normalizes to -0.986 and 917 normalizes to 0.917.

Normalization - University Questions:

Q1. A set of data is given: $A=\{8,10,15,20\}$. Normalize the data by z-score normalization.

Q2. A set of data is given: $A=\{115,233,484,543\}$. Normalize the data by min-max normalization (range: $[0.0,1.0]$).

Q3: Use the two methods below to normalize the following group of data:

100,200,300,500,900

a) min-max normalization by setting min=0, max=1

b) z-score normalization

Q4: A set of data is given : $\{-10, 201, 301, -401, 501, 601, 701\}$. Normalize the given data by decimal scaling.

Q5: Normalize the following group of data : $\{1000,2000,3000,9000\}$ using min-max normalization by setting min:0 and max:1

DISCRETIZATION & CONCEPT HIERARCHY GENERATION:

- Data discretization refers to a method of converting a huge number of data values into smaller ones so that the evaluation and management of data become easy.
- Alternatively, data discretization can be defined as a **method of converting attribute values of continuous data into a finite set of intervals with minimum data loss.**
 - ie, **the range of the attribute is divided into intervals.**
 - **Now, these interval labels can be used to replace actual values.**
- Replacing numerous values of a continuous attribute by a small number of interval labels thereby **reduces and simplifies the original data.**
- Eg: Suppose we have an attribute Age with the given values;

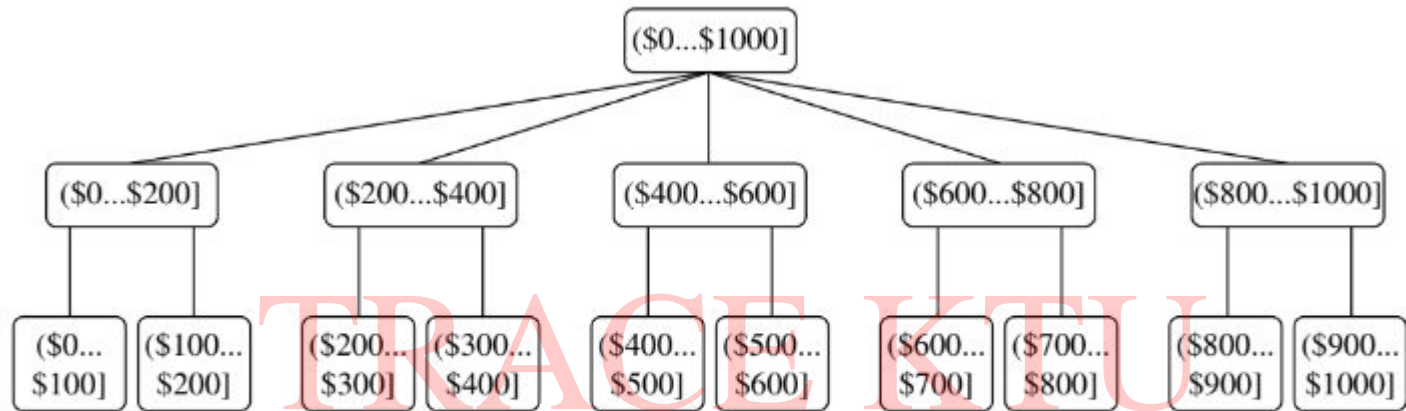
TRACE KTU

Table before Discretization

Age	1,5,9,4,7,11,14,17,13,18, 19,31,33,36,42,44,46,70,74,78,77
-----	--

Attribute	Age	Age	Age	Age
	1,5,4,9,7	11,14,17,13,18,19	31,33,36,42,44,46	70,74,77,78
After Discretization	Child	Young	Mature	Old

- Discretization techniques can be categorized;
 - **Based on how the discretization is performed.**
 - If the discretization process uses class information - **Supervised discretization.**
 - If the discretization process does not use class information - **Unsupervised discretization.**
 - **Based on the direction in which discretization proceeds.**
 - If the process starts by first finding one or a few points (called split points or cut points) to split the entire attribute range, and then repeats this recursively on the resulting intervals - **Top-down discretization (also called Splitting)**
 - If the process starts by considering all of the continuous values as potential split-points, removes some by merging neighborhood values to form intervals, and then recursively applies this process to the resulting intervals - **Bottom-up discretization (also called Merging)**



A concept hierarchy for the attribute *price*, where an interval $(\$X \dots \$Y]$ denotes the range from $\$X$ (exclusive) to $\$Y$ (inclusive).

- Following are the strategies for discretization & concept hierarchy generation for numerical data;
 - 1. Discretization by Binning
 - 2. Discretization by Histogram Analysis
 - 3. Discretization by Cluster, Decision Tree, and Correlation Analyses

Discretization by Binning:

- Binning is a top-down splitting technique based on a specified number of bins.

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 boundaries:

Bin 1: 4, 4, 15
Bin 2: 21, 21, 24
Bin 3: 25, 25, 34

- Binning method can be used as a method for data smoothing as well as for data reduction and concept hierarchy generation.
- Eg: 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, as in smoothing by bin means or smoothing by bin medians, respectively.
- These techniques can be applied recursively to the resulting partitions to generate concept hierarchies.

Discretization by Histogram Analysis:

- Histogram analysis is an unsupervised discretization technique.
- A histogram partitions the values of an attribute, A , into disjoint ranges called buckets or bins.
- Various partitioning rules can be used to define histograms such as;
 - Equi-width histogram - the values are partitioned into equal-size partitions or ranges.
 - Equal-frequency histogram - the values are partitioned so that each partition contains the same number of data tuples.
- The histogram analysis algorithm can be applied recursively to each partition in order to automatically generate a multi-level concept hierarchy, with the procedure terminating once a prespecified number of concept levels has been reached.

Discretization by Cluster, Decision Tree and Correlation Analyses:

- Clustering, decision tree analysis, and correlation analysis can be used for data discretization.

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 takes the distribution of A into consideration, as well as the closeness of data points, and therefore is able to produce high-quality discretization results.

- 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.

Discretization by Decision tree:

- Techniques to generate decision trees for classification can be applied to discretization also.
- Such techniques employ a top-down splitting approach.
- Decision tree approaches to discretization are supervised.
 - ie, they make use of class label information.
- Eg: Suppose, we have a data set of patient symptoms (the attributes).
 - Each patient has an associated diagnosis class label.
 - Class distribution information is used in the calculation and determination of split-points (data values for partitioning an attribute range).
 - Intuitively, 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.
 - To discretize a numeric attribute, A, the method selects the value of A that has the minimum entropy as a split-point, and recursively partitions the resulting intervals to arrive at a hierarchical discretization.
 - Such discretization forms a concept hierarchy for A.

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 by finding the best neighboring intervals and then merging them to form larger intervals, recursively.
- ChiMerge is supervised as it uses class information.
- ChiMerge proceeds as follows.
 - Initially, each distinct value of a numeric attribute A 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 are 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 GENERATION FOR NOMINAL DATA:

- Nominal attributes have a finite (but possibly large) number of distinct values, with no ordering among the values.
 - Eg: Geographic location, job category, and item type.
- Following are the four methods for the generation of concept hierarchies for nominal data:
 1. Specification of a partial ordering of attributes explicitly at the schema level by users or experts.
 2. Specification of a portion of a hierarchy by explicit data grouping
 3. Specification of a set of attributes, but not of their partial ordering.
 4. Specification of only a partial set of attributes.

Specification of a partial ordering of attributes explicitly at the schema level by users or experts:

- Concept hierarchies for nominal attributes or dimensions typically involve a group of attributes.
- A user or expert can easily define a concept hierarchy by specifying a partial or total ordering of the attributes at the schema level.

- Eg:
 - Suppose that a relational database contains the following group of attributes:
 - street, city, province or state, and country.
 - Similarly, a data warehouse location dimension may contain the same attributes.
 - A hierarchy can be defined by specifying the total ordering among these attributes at the schema level such as $\text{street} < \text{city} < \text{province or state} < \text{country}$.

Specification of a portion of a hierarchy by explicit data grouping:

- In a large database, it is unrealistic to define an entire concept hierarchy by explicit value enumeration.
- On the contrary, we can easily specify explicit groupings for a small portion of intermediate-level data.
- Eg:
 - After specifying that province and country form a hierarchy at the schema level, a user could define some intermediate levels manually, such as " $\{\text{Alberta, Saskatchewan, Manitoba}\} \subset \text{prairies Canada}$ " and " $\{\text{British Columbia, prairies Canada}\} \subset \text{Western Canada}$."

Specification of a set of attributes, but not of their partial ordering:

- A user may specify a set of attributes forming a concept hierarchy, but omit to explicitly state their partial ordering.
- The system can then try to automatically generate the attribute ordering so as to construct a meaningful concept hierarchy.

Specification of only a partial set of attributes:

- Sometimes a user can be careless when defining a hierarchy, or have only a vague idea about what should be included in a hierarchy.
- Consequently, the user may have included only a small subset of the relevant attributes in the hierarchy specification.
- For example, instead of including all of the hierarchically relevant attributes for location, the user may have specified only street and city.
- To handle such partially specified hierarchies, it is important to embed data semantics in the database schema so that attributes with tight semantic connections can be pinned together.
- In this way, the specification of one attribute may trigger a whole group of semantically tightly linked attributes to be “dragged in” to form a complete hierarchy. Users, however, should have the option to override this feature, as necessary.