DATA PREPROCESSING

BY

DR. PREETHAM KUMAR

HOD

DEPT. OF INFORMATION & COMMUNICATION TECHNOLOGY

Ref : Jiawei Han & Micheline Kamber

- **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, heterogenous sources.**
- Low-quality data will lead to low-quality mining results.

- □ *"How can the data be preprocessed in order to help improve the quality of the data and, consequently, of the mining results*
- □ *How can the data be preprocessed so as to improve the efficiency and ease of the mining process"*

- There are a number of data preprocessing techniques.
- *Data cleaning* can be applied to remove noise and correct inconsistencies in the data.
- *Data integration* merges data from multiple sources into a coherent data store, such as a data warehouse.
- *Data transformations*, such as normalization, may be applied.

- □ *Data reduction* can reduce the data size by aggregating, eliminating redundant features, or clustering, for instance.
- □ The above techniques are not mutually exclusive; they may work together.
- □ For example, data cleaning can involve transformations to correct wrong data, such as by transforming all entries for a *date* field to a common format.

- Data processing techniques, when applied before mining, can substantially improve the overall quality of the patterns mined and/or the time required for the actual mining.

Why Preprocess the Data?

- Incomplete, noisy, and inconsistent data are common place properties of large real world databases and data warehouses.
- Incomplete data can occur for a number of reasons.
  - Attributes of interest may not always be available, such as customer information for sales transaction data.
  - Other data may not be included simply because it was not considered important at the time of entry.

- Relevant data may not be recorded due to a misunderstanding, or because of equipment malfunctions.
- Furthermore, the recording of the history or modifications to the data may have been overlooked

- Missing data, particularly for tuples with missing values for some attributes, may need to be inferred.
- There are many possible reasons for noisy data (having incorrect attribute values).
  - The data collection instruments used may be faulty.
  - There may have been human or computer errors occurring at data entry.
  - Errors in data transmission can also occur.
  - There may be technology limitations, such as limited buffer size for coordinating synchronized data transfer and consumption

- Incorrect data may also result from inconsistencies in naming conventions or data codes used, or inconsistent formats for input fields, such as date.
- Duplicate tuples also require 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.

- If users believe the data are dirty, they are unlikely to trust the results of any data mining that has been applied to it.
- Furthermore, dirty data can cause confusion for the mining procedure, resulting in unreliable output.

- Getting back to task at *AllElectronics*, suppose that you would like to include data from multiple sources in your analysis.
- This would involve **integrating** multiple databases, data cubes, or files, that is, data integration.
- Yet some attributes representing a given concept may have different names in different databases, causing inconsistencies and redundancies.

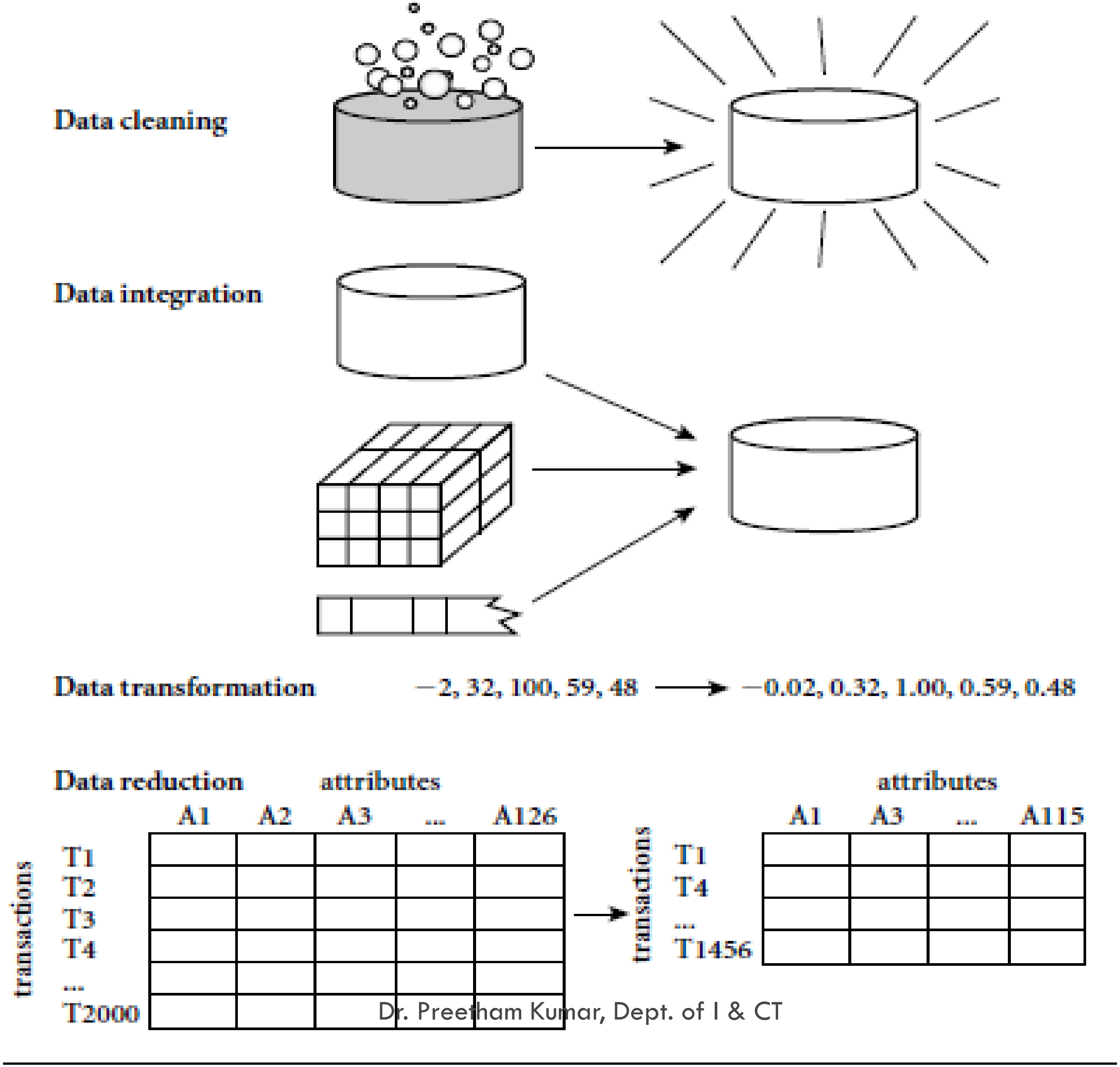
- □ For example, the attribute for customer identification may be referred to as *customer id* in one data store and *cust id* in another.
- □ Naming inconsistencies may also occur for attribute values. For example, the same first name could be registered as “Bill” in one database, but “William” in another, and “B.” in the third.
- □ Furthermore, you suspect that some attributes may be inferred from others (e.g., annual revenue).

- □ Having a large amount of redundant data may slow down or confuse the knowledge discovery process.
- □ Clearly, in addition to data cleaning, steps must be taken to help avoid **redundancies** during data integration.
- □ *data cleaning and data integration are performed as a preprocessing step when preparing the data for a data warehouse.*

- Additional data cleaning can be performed to detect and remove redundancies that may have resulted from data integration.

- *“The data set selected for analysis is HUGE, which is sure to slow down the mining process. Is there any way can reduce the size of my data set, without jeopardizing the data mining results?”*

- **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.
- There are a number of strategies for data reduction. These include
  - *data aggregation* (e.g., building a data cube),
  - *attribute subset selection* (e.g., removing irrelevant attributes through correlation analysis),
  - *dimensionality reduction* (e.g., using encoding schemes such as minimum length encoding or wavelets), and
  - *numerosity reduction* (e.g., “replacing” the data by alternative, smaller representations such as clusters or parametric models).


Data cleaning
Data integration
Data transformation
−2, 32, 100, 59, 48
−0.02, 0.32, 1.00, 0.59, 0.48
Data reduction
attributes
A1
A2
A3
...
A126
transactions
T1
T2
T3
T4
...
T2000
attributes
A1
A3
...
A115
transactions
T1
T4
...
T1456

- The above categorization is not mutually exclusive.
- For example, the removal of redundant data may be seen as a form of data cleaning, as well as data reduction.
- In summary, real-world data tend to be dirty, incomplete, and inconsistent.
- Data preprocessing techniques can improve the quality of the data, thereby helping to improve the accuracy and efficiency of the subsequent mining process.

- Data preprocessing is an important step in the knowledge discovery process, because quality decisions must be based on quality data.
- Detecting data anomalies, rectifying them early, and reducing the data to be analyzed can lead to huge payoffs for decision making.

Descriptive Data Summarization: Basic Concepts

- □ Descriptive data summarization techniques can be used to identify the typical properties of data and highlight which data values should be treated as noise or outliers.
- □ **Measuring the Central Tendency:** to measure the central tendency of data.
- □ The most common and most effective numerical measure of the “center” of a set of data is the *(arithmetic) mean.*
- □ Let *x*1;*x*2; : : : ;*xN* be a set of *N* values or observations, such as for some attribute, like *salary*.
- □ The mean of this set of values is

$$\bar{x} = \frac{\sum_{i=1}^{N} x_i}{N} = \frac{x_1 + x_2 + \cdots + x_N}{N}.$$

Distributive measure

- A distributive measure is a measure (i.e., function) that can be computed for a
  - given data set by *partitioning the data into smaller subsets*, *computing the measure for each subset*, *and then* *merging the results* *in order to arrive at the measure's value for the original (entire) data set.*
- sum() and count() are distributive measures because they can be computed in this manner. Other examples include max() and min().

Algebraic measure

- An algebraic measure is a measure that can be computed by **applying an algebraic function to one or more distributive measures**.
- Hence, *average* (or mean()) is an algebraic measure because it can be computed by sum()/count().
- When Computing data cubes, sum() and count() are typically saved in pre-computation.

- □ Sometimes, each value *xi* in a set may be associated with a weight *wi*, for *i* = 1 ,…. *N*.
- □ The weights reflect the significance, importance, or occurrence frequency attached to their respective values.
- □ In this case, we can compute

$$\bar{x} = \frac{\sum_{i=1}^{N} w_i x_i}{\sum_{i=1}^{N} w_i} = \frac{w_1 x_1 + w_2 x_2 + \cdots + w_N x_N}{w_1 + w_2 + \cdots + w_N}.$$

- □ This is called the **weighted arithmetic mean** or the **weighted average.**
- □ Note that the weighted average is another example of an algebraic measure.

- Although the mean is the single most useful quantity for describing a data set, it is not always the best way of measuring the center of the data.
- A major problem with the *mean* is its sensitivity to extreme (e.g., outlier) values. Even a small number of extreme values can corrupt the mean.
- For example, the **mean salary at a company may be substantially pushed up by that of a few highly paid managers.**
- Similarly, **the average score of a class in an exam could be pulled down quite a bit by a few very low scores**.

- □ To **offset the effect** caused by a small number of extreme values, we can instead use the **trimmed mean,** which is the **mean obtained after chopping off values at the high and low extremes**.
- □ For example, we can sort the values observed for *salary* and remove the **top and bottom 2% before computing the mean.**
- □ We should avoid trimming too large a portion (such as 20%) at both ends **as this can result in the loss of valuable information**.

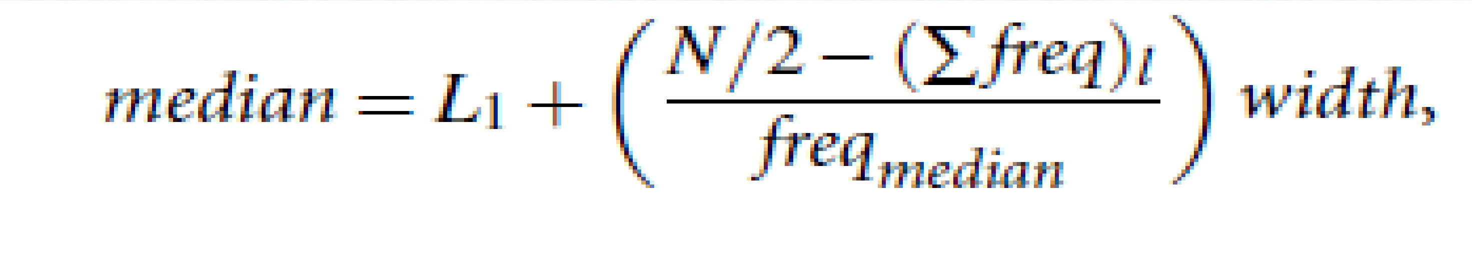
- For skewed (asymmetric) data, a better measure of the center of data is the ***median.***
- Suppose that a given data set of *N* distinct values is sorted in numerical order.
- If *N* is odd, then the median is the ***middle value*** of the ordered set; otherwise (i.e., if *N* is even), the median is the average of the **middle two values.**

Holistic measure

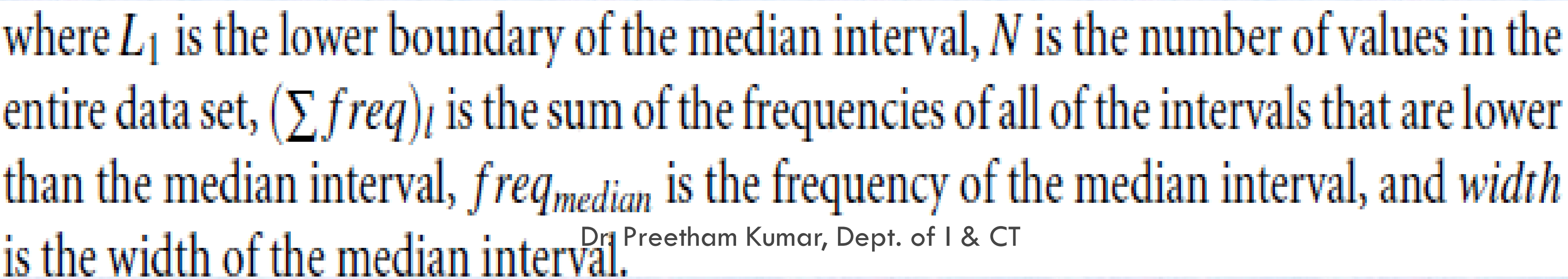
- A holistic measure is a measure that must be computed on the entire data set as a whole.
- It cannot be computed by partitioning the given data into subsets and merging the values obtained for the measure in each subset.
- The median is an example of a holistic measure.
- Holistic measures are **much more expensive** to compute than distributive measures.

- We can, however, easily *approximate* the median value of a data set.
- Assume that data are grouped in intervals according to their *xi* data values and that the frequency (i.e., number of data values) of each interval is known.
- For example, people may be grouped according to their annual salary in intervals such as 10–20K, 20–30K, and so on.
- The interval that contains the median frequency is called as the ***median interval***.

- The median of the entire data set (e.g., the median salary) by using the formula:



$$median = L_1 + \left(\frac{N/2 - (\sum freq)_l}{freq_{median}} \right) width,$$



where L_1 is the lower boundary of the median interval, N is the number of values in the entire data set, $(\sum freq)_l$ is the sum of the frequencies of all of the intervals that are lower than the median interval, $freq_{median}$ is the frequency of the median interval, and *width* is the width of the median interval.

Compute an *approximate median* value for the data.

Suppose that the values for a given set of data are grouped into intervals. The intervals and corresponding frequencies are as follows.

Age	***frequency***
1–5	200
5–15	450
15–20	300
20–50	1500
50–80	700
80–110	44

$$median = L_1 + \left(\frac{N/2 - (\sum freq)_l}{freq_{median}} \right) width,$$

Substituting in the above equation the values

*L*1 = 20, *N*= 3194, (∑freq) = 950,

freqmedian = 1500, *width* = 30,

We *get median* = 32.94 years.

Mode

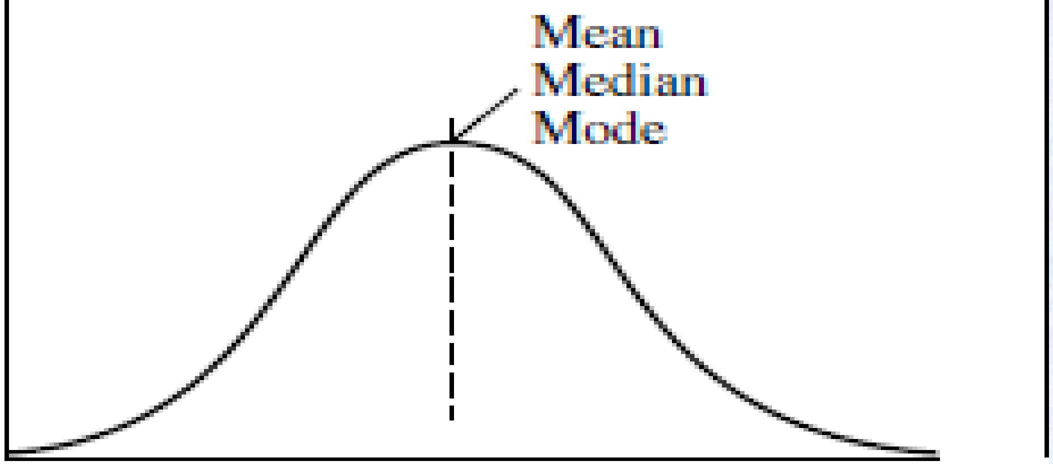
- □ Another measure of central tendency is the *mode*.
- □ The mode for a set of data is the value **that occurs most frequently in the set.**
- □ It is possible for the **greatest frequency to correspond to several different values**, which results in **more than one mode**.
- □ Data sets with one, two, or three modes are respectively called **unimodal, bimodal, and trimodal**.

Mode

- A data set with **two or more modes is multimodal.**
- If each data value occurs only once, then there is no mode.
- For unimodal frequency curves that are moderately skewed (asymmetrical), we have the following empirical relation:

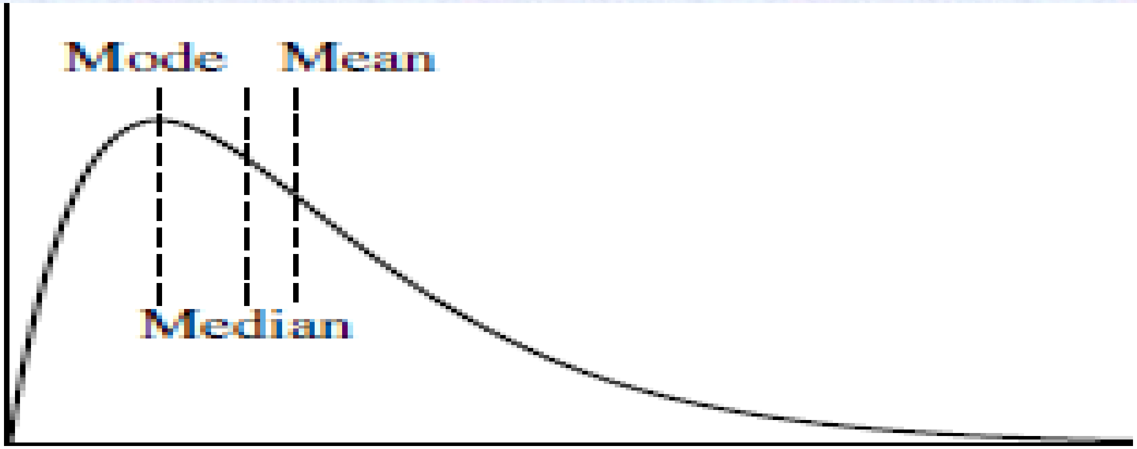
    ***Mean- mode* = 3X(*mean-median*)**

- In a unimodal frequency curve with perfect symmetric data distribution, the mean, median, and mode are all at the same center value, as shown in Figure



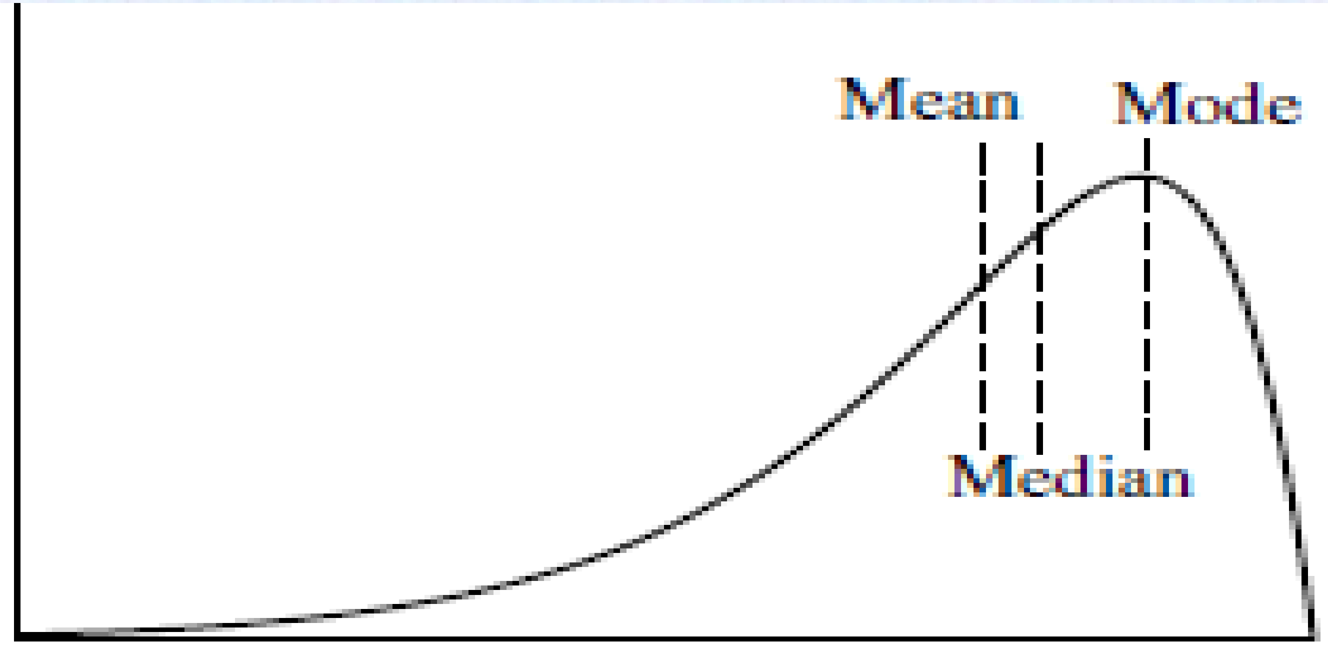
(a) symmetric data

Mean, median, and mode of sym

- However, data in most real applications are not symmetric.
- They may instead be either **positively skewed**, where the **mode occurs at a value that is smaller than the median** (Figure (b))
- 

(b) positively skewed data

□ negatively skewed, where the mode occurs at a value greater than the median(Figure (c)).



(c) negatively skewed data

Mid Range

- The midrange can also be used to assess the central tendency of a data set.
- It is the average of the **largest** and **smallest** values in the set.
- This algebraic measure is easy to compute using the SQL aggregate functions, max() and min().

Measuring the Dispersion of Data

- The degree to which numerical data tend to spread is called the **dispersion, or variance** of the data.
- The most common measures of data dispersion are
  - *range*
  - the *five-number summary* (based on *quartiles*),
  - the *interquartile range*
  - the *standard deviation.*

Range, Quartiles, Outliers, and Boxplots

- Let $x1;x2; : : : ;xN$ be a set of observations for some attribute.
- The **Range** of the set is the difference between the **largest** (max()) and **smallest** (min()) values.
- We assume that the data are sorted in increasing numerical order.

Percentile

- The ***kth*** percentile of a set of data in numerical order is the **value *xi*** having the property that ***k* percent of the data entries** lie **at or below *xi*.**
- The *median* is the 50th percentile since median is the value at the middle position of the data set and 50% of the data lie below this point.

First Quartile, Third Quartile

- The most commonly used percentiles other than the median (Second quartile)are quartiles.
  - **The first quartile, denoted by Q1, is the 25th percentile;**
  - **the third quartile, denoted by Q3, is the 75th percentile.**
- The quartiles, including the median, give some indication of the center, spread, and shape of a distribution.

Interquartile Range

- The distance between the first (Q1)and third quartiles(Q3) is a simple measure of spread that gives the range covered by the middle half of the data.
- This distance is called the interquartile range (*IQR*) and is defined as

$$IQR = Q3\text{-}Q1.$$

- A common rule for identifying **suspected outliers** is to
- single out values falling **at least 1.5*_IQR_ above the third quartile** or **below the first quartile.**

- BecauseQ1, the median(Q2), andQ3 together contain no information about the endpoints (e.g., tails) of the data, **a fuller summary of the shape of a distribution** can be obtained by providing the **lowest and highest data values as well.** This is known as the ***five-number summary***.

- **The five-number** summary of a distribution consists of the median, the quartilesQ1 andQ3, and the smallest

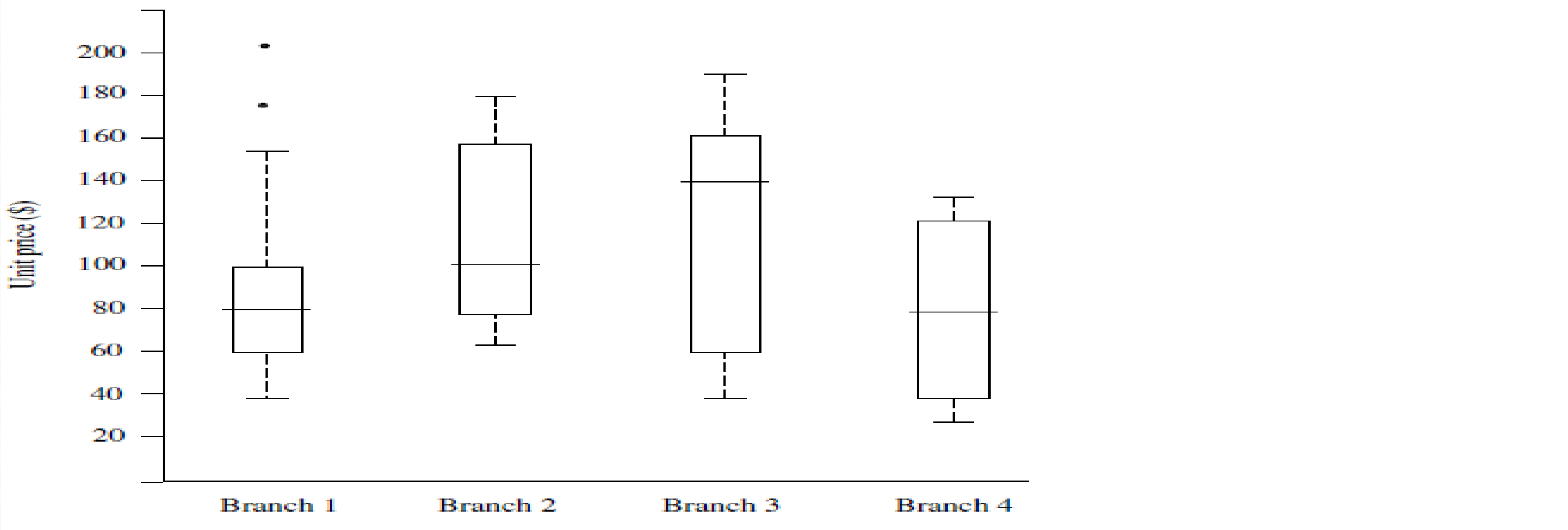
and largest individual observations, written in the order

  - Minimum;
  - Q1;
  - Median;
  - Q3;
  - Maximum:
- Boxplots are a popular way of visualizing a distribution.

Boxplots

- **A boxplot incorporates the five-number summary as follows:**
  - Typically, the ends of the box are at the quartiles, (Q1 and Q3) so that the box length is the interquartile range, *IQR*.
  - The median is marked by a line within the box.
  - Two lines (called *whiskers*) outside the box extend to the smallest (*Minimum*) and largest (*Maximum*) observations.

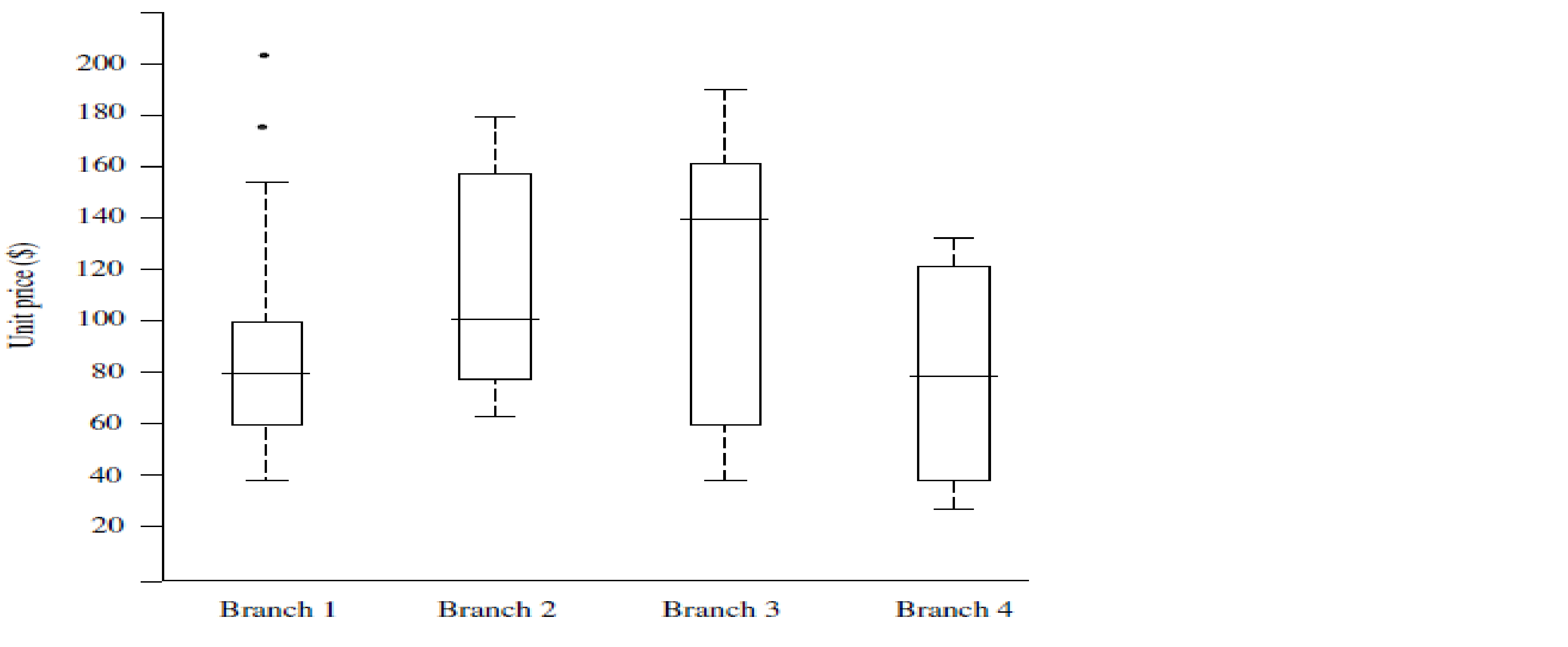
- When dealing with a moderate number of observations, it is worthwhile to plot potential outliers individually.



Boxplot for the unit price data for items sold at four branches of *AllElectronics* during a giver time period.

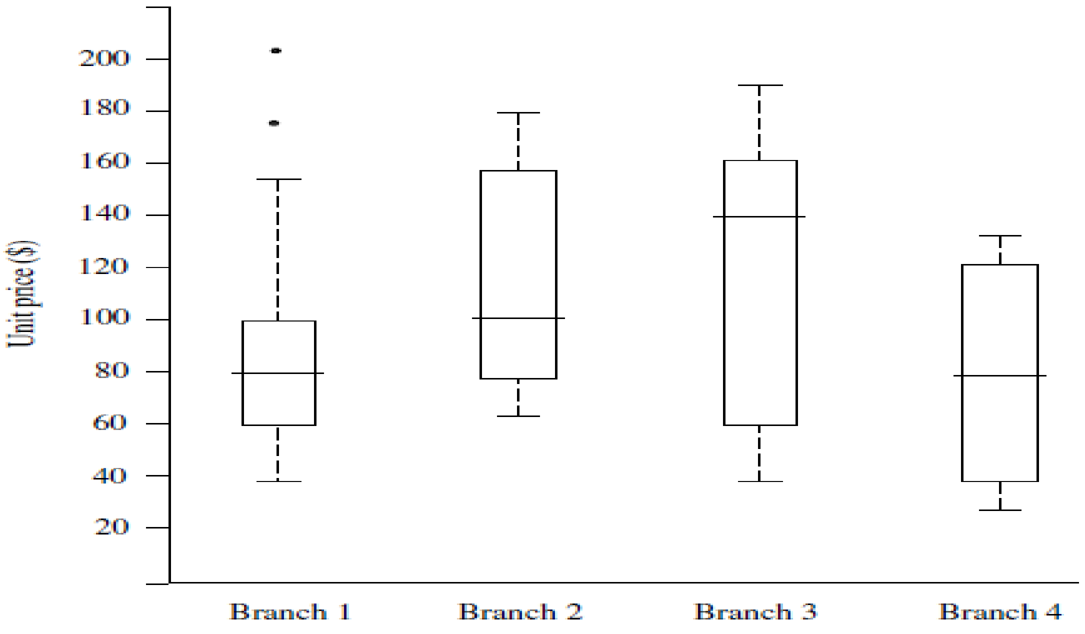
- To do this in a boxplot, the whiskers are *extended to the extreme low and high observations* *only* ***if these values are*** **less than 1.5****IQR*** **beyond** the quartiles.
- Otherwise, the whiskers terminate at the most extreme observations occurring within1.5 **IQR* of the quartiles.

- For branch 1, we see that the median price of items sold is $80, Q1 is $60, Q3 is $100.



Boxplot for the unit price data for items sold at four branches of *AllElectronics* during a given time period.

- Notice that two outlying observations for this branch were plotted individually, as their values of 175 and 202 are more than 1.5 times the IQR here of 40.



Boxplot for the unit price data for items sold at four branches of *AllElectronics* during a giver time period.

Suppose that the data for analysis includes the attribute *age*. The *age* values for the data tuples are (in increasing order)

13, 15, 16, 16, 19, 20, 20, 21, 22, 22, 25, 25, 25, 25, 30, 33, 33, 35, 35, 35, 35, 36, 40, 45, 46, 52, 70.

(a) What is the *mean* of the data? What is the *median*?

Ans: The (arithmetic) mean of the data is:

$$\bar{x} = \frac{1}{N}\sum_{i=1}^{N} x_i$$

Mean = 809/27 = 30. The median (middle value of the ordered set, as the number of values in the set is odd) of the data is: 25.

(b) What is the *mode* of the data? Comment on the data's modality (i.e., bimodal, trimodal, etc.)

Ans:

- □ This data set has two values that occur with the same highest frequency and is, therefore, bimodal.
- □ The modes (values occurring with the greatest frequency) of the data are 25 and 35.

(c) What is the *midrange* of the data?

Ans: The midrange (average of the largest and smallest values in the data set) of the data is: (70+13)/2 =41.5

(d) Can you find (roughly) the first quartile (Q1) and the third quartile (Q3) of the data?

Ans:

The First quartile(Q1) (corresponding to the 25th percentile) of the data is: **20.**

The third quartile(Q3) (corresponding to the 75th percentile) of the data is: **35.**

(e) Give the *five-number summary* of the data.

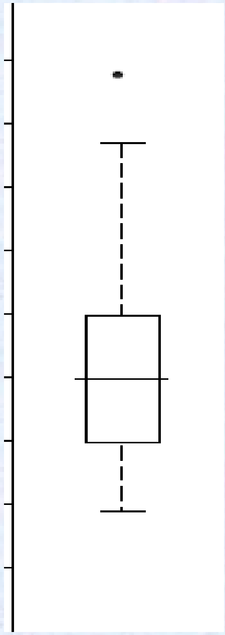
Ans: The Five number summary of a distribution consists of the minimum value, First quartile, median value, third quartile, and maximum value.

13, 15, 16, 16, 19, 20, 20, 21, 22, 22, 25, 25, 25, 25, 30, 33, 33, 35, 35, 35, 35, 36, 40, 45, 46, 52, 70.

It provides a good summary of the shape of the distribution and for this data is: 13, 20, 25, 35, 70.

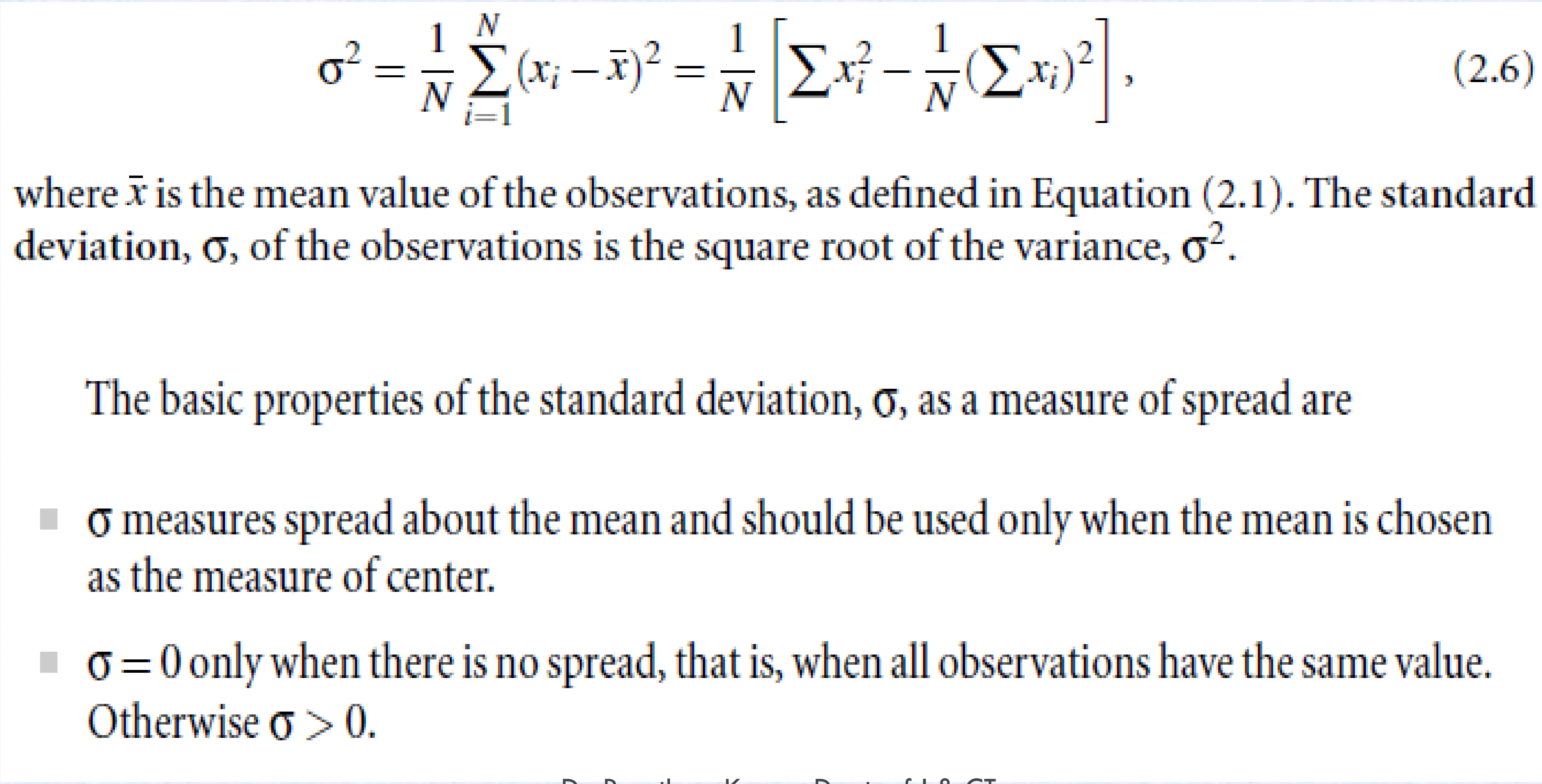
(f) Show a *boxplot* of the data (It will be similar to the below figure)

-



Variance and Standard Deviation

- The variance of *N* observations, *x*1;*x*2; : : : ;*xN*, is



$$\sigma^2 = \frac{1}{N}\sum_{i=1}^{N}(x_i - \bar{x})^2 = \frac{1}{N}\left[\sum x_i^2 - \frac{1}{N}\left(\sum x_i\right)^2\right], \qquad (2.6)$$

where $\bar{x}$ is the mean value of the observations, as defined in Equation (2.1). The standard deviation, σ, of the observations is the square root of the variance, σ^2.

The basic properties of the standard deviation, σ, as a measure of spread are

- σ measures spread about the mean and should be used only when the mean is chosen as the measure of center.
- $\sigma = 0$ only when there is no spread, that is, when all observations have the same value. Otherwise $\sigma > 0$.

Graphic Displays of Basic Descriptive Data Summaries

- Aside from the bar charts, pie charts, and line graphs used in most statistical or graphical data presentation software packages, there are other **popular types of graphs for the display of data summaries and distributions**.
- These include
  - *histograms,*
  - *quantile plots,*
  - *q-q plots,*
  - *scatter plots*, and
  - *loess curves.*
- Such graphs are very helpful for the visual inspection of data.

Histograms

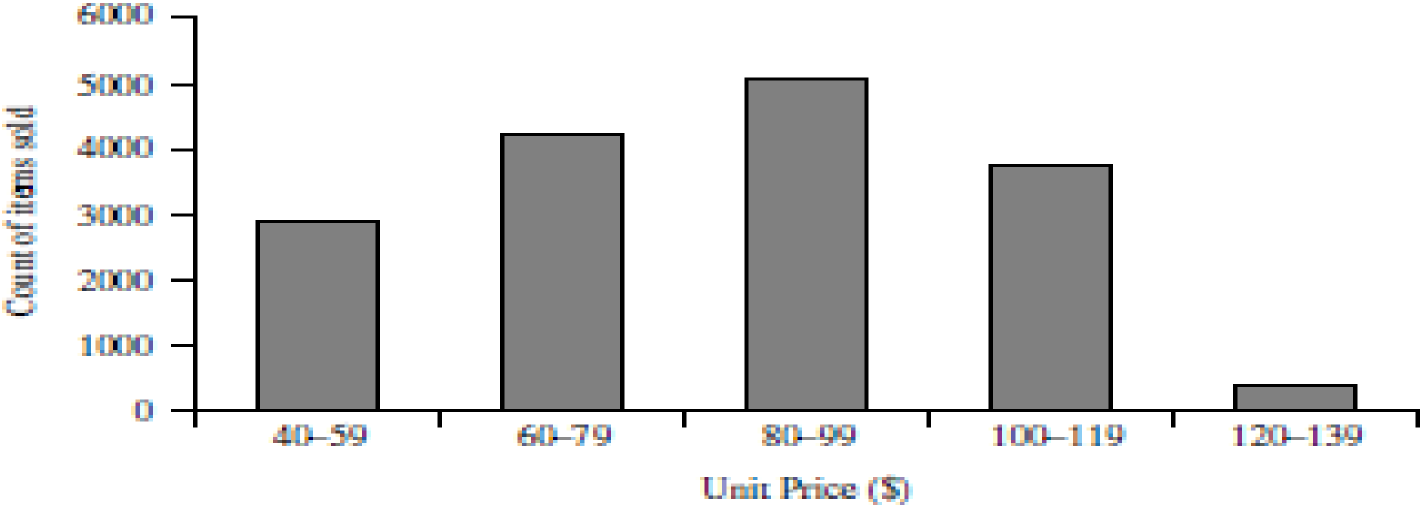
- Plotting histograms, or frequency histograms, is a graphical method for summarizing the distribution of a given attribute.
- A histogram for an attribute *A* **partitions the data distribution of *A* into disjoint subsets**, or *buckets*.
- Typically, the width of each bucket is uniform.
- Each **bucket is represented by a rectangle** whose **height is equal to the count or relative frequency** of the values at the bucket.
- Histograms are at least a century old and are a widely used univariate graphical method.

- If *A* is categoric, such as *automobile model* or *item type*, then one rectangle is drawn for each known value of *A*, and the resulting graph is more commonly referred to as a bar chart.
- If *A* is numeric, the term *histogram* is preferred.
- Partitioning rules for constructing histograms for numerical attributes are discussed in Section 2.5.4.
- In an equal-width histogram, for example, each bucket represents an equal-width range of numerical attribute *A*.

- The following Figure shows a histogram for the data set of Table 2.1, where buckets are

defined by equal-width ranges representing $20 increments and the frequency is the count of items sold.

- However, they may not be as effective as the quantile plot, q-q plot, and boxplot methods for comparing groups of univariate observations



igure 2.4 A histogram for the data set of Table 2.1.

Table 2.1 A set of unit price data for items sold at a branch of *AllElectronics*.

Unit price ($)	*Count of items sold*
40	275
43	300
47	250
..	..
74	360
75	515
78	540
..	..
115	320
117	270
120	350

quantile plot

- A **quantile plot** is a simple and effective way to have a first look at a **univariate data distribution**.
  - First, it displays all of the data for the given attribute (allowing the user to assess both the overall behavior and unusual occurrences).
  - Second, it plots quantile information. The mechanism used in this step is slightly different from the percentile computation
  - Let *xi*, for *i* = 1 to *N*, be the data sorted in increasing order so that *x*1 is the smallest observation and *xN* is the largest

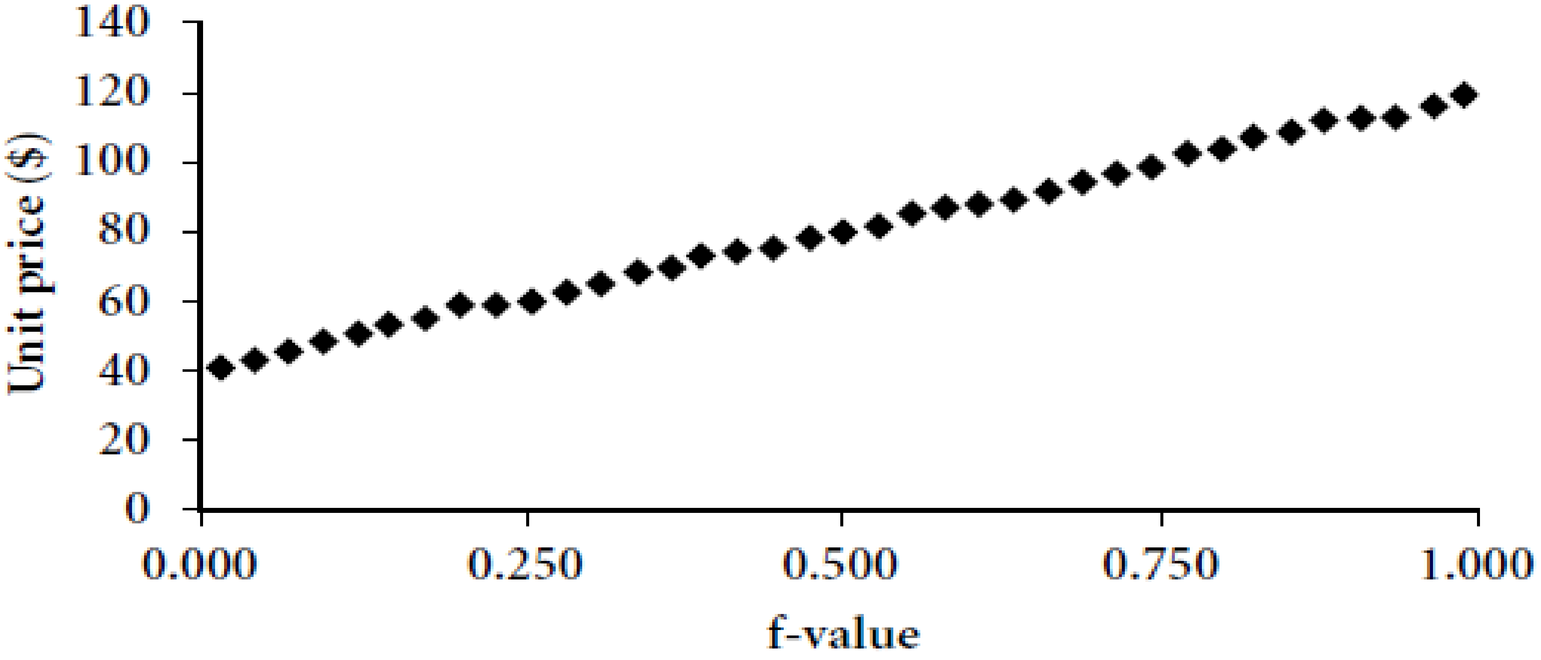
- Each observation, *xi*, is paired with a percentage, *fi*, which indicates that approximately 100 *fi* % of the data are below or equal to the value, *xi*.
- We say "approximately" because there may not be a value with exactly a fraction, *fi*, of the data below or equal to *xi*.
- Note that the 0.25 quantile corresponds to quartile Q1, the 0.50 quantile is the median, and the 0.75 quantile is Q3.

$$f_i = \frac{i - 0.5}{N}.$$

- These numbers increase in equal steps of 1/*N*, ranging from 1/2*N* (which is slightly above zero , we get it when i=1) to 1-1/2*N* (which is slightly below one, we get when i=N).
- On a quantile plot, *xi* is graphed against *fi.*
- This allows us to compare different distributions based on their quantiles.
- For example, given the quantile plots of sales data for two different time periods,

- □ we can compare theirQ1, median,Q3, and other *fi* values at a glance.

- □ Figure shows a quantile plot for the *unit price* data of Table 2.1.

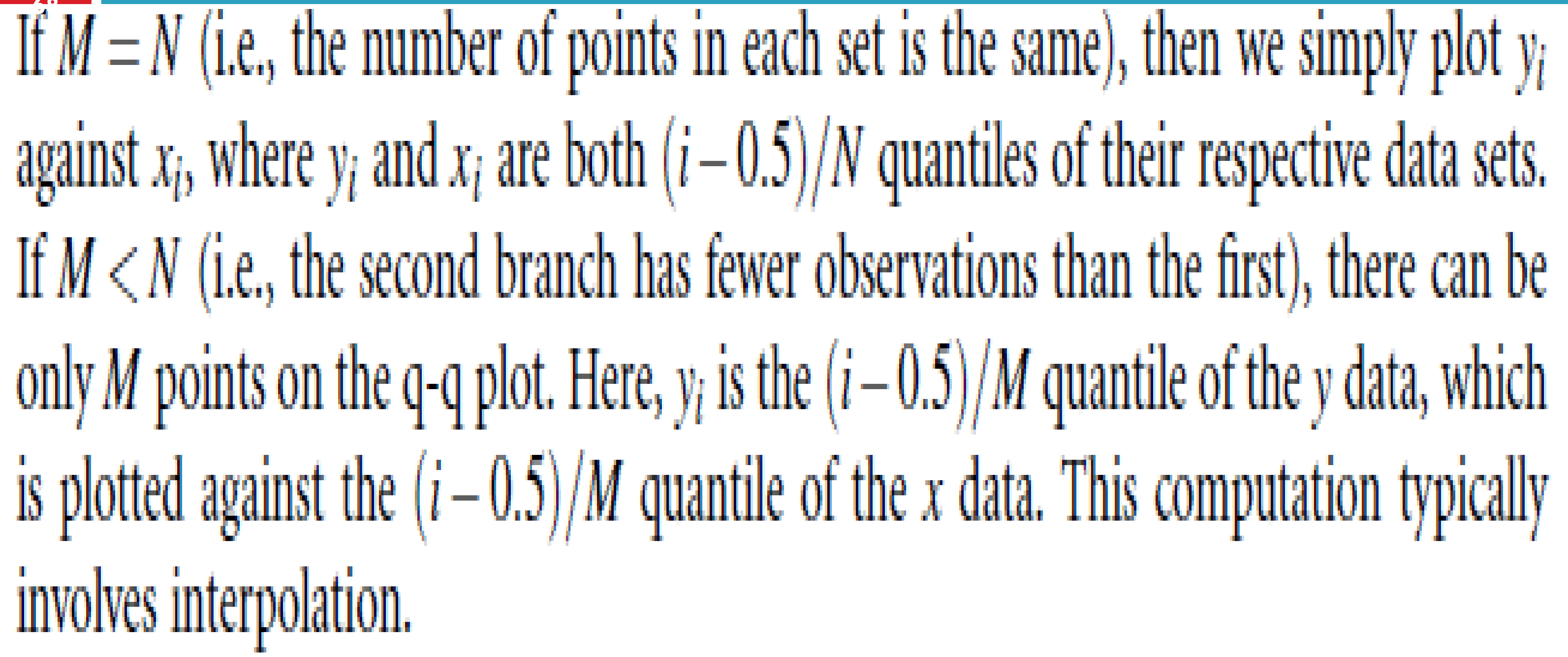


A quantile plot for the unit price data of Table 2.1.

quantile-quantile plot

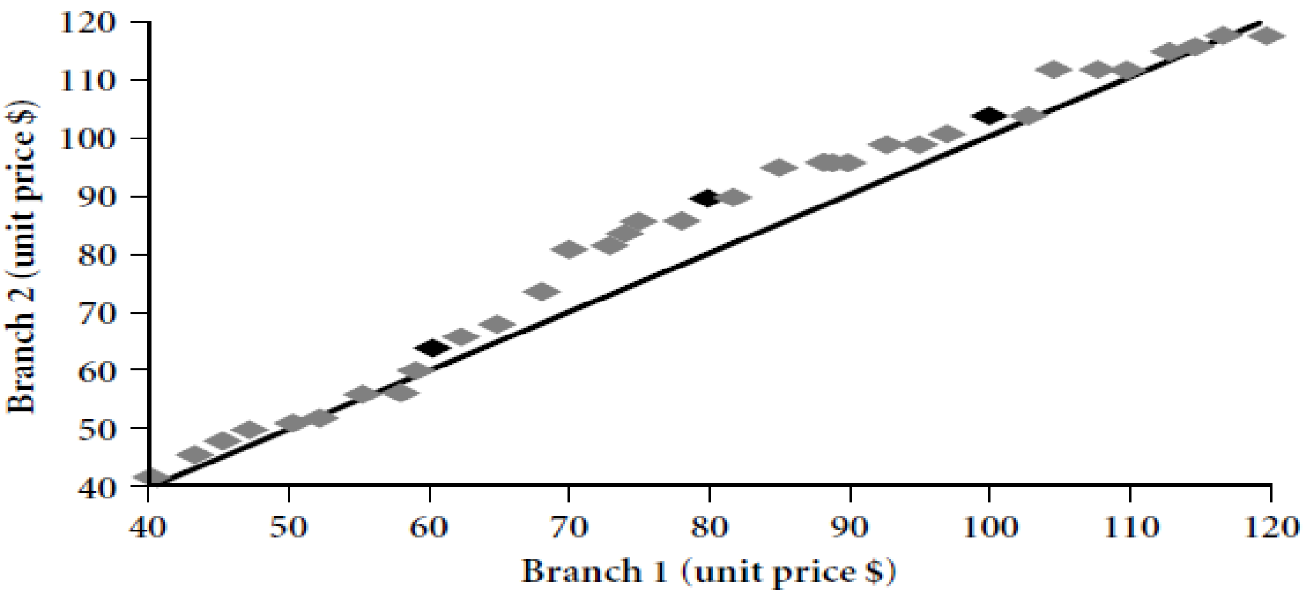
- A quantile-quantile plot, or q-q plot, graphs the **quantiles of one univariate distribution** against **the corresponding quantiles of another**.
- It is a powerful visualization tool in that it allows the user to view whether there is a shift in going from one distribution to another.

- □ Suppose that we have two sets of observations for the variable ***unit price***, taken from two different branch locations.
- □ Let $x1; : : : ;xN$ be the data from the first branch, and $y1; : : : ;yM$ be the data from the second, where each data set is sorted in increasing order.



If $M = N$ (i.e., the number of points in each set is the same), then we simply plot y_i against x_i, where y_i and x_i are both $(i-0.5)/N$ quantiles of their respective data sets. If $M < N$ (i.e., the second branch has fewer observations than the first), there can be only M points on the q-q plot. Here, y_i is the $(i-0.5)/M$ quantile of the y data, which is plotted against the $(i-0.5)/M$ quantile of the x data. This computation typically involves interpolation.

- Figure shows a quantile-quantile plot for *unit price* data of items sold at two different branches of *AllElectronics* during a given time period. Each point corresponds to the same quantile for each data set and shows the unit price of items sold at branch 1 versus branch 2 for that quantile.



A quantile-quantile plot for unit price data from two different branches.

- For example, here the lowest point in the left corner corresponds to the 0.03 quantile.

(To aid in comparison, we also show a straight line that represents the case of when, for each given quantile, the unit price at each branch is the same. In addition, the darker points correspond to the data for Q1, the median, and Q3,respectively.)

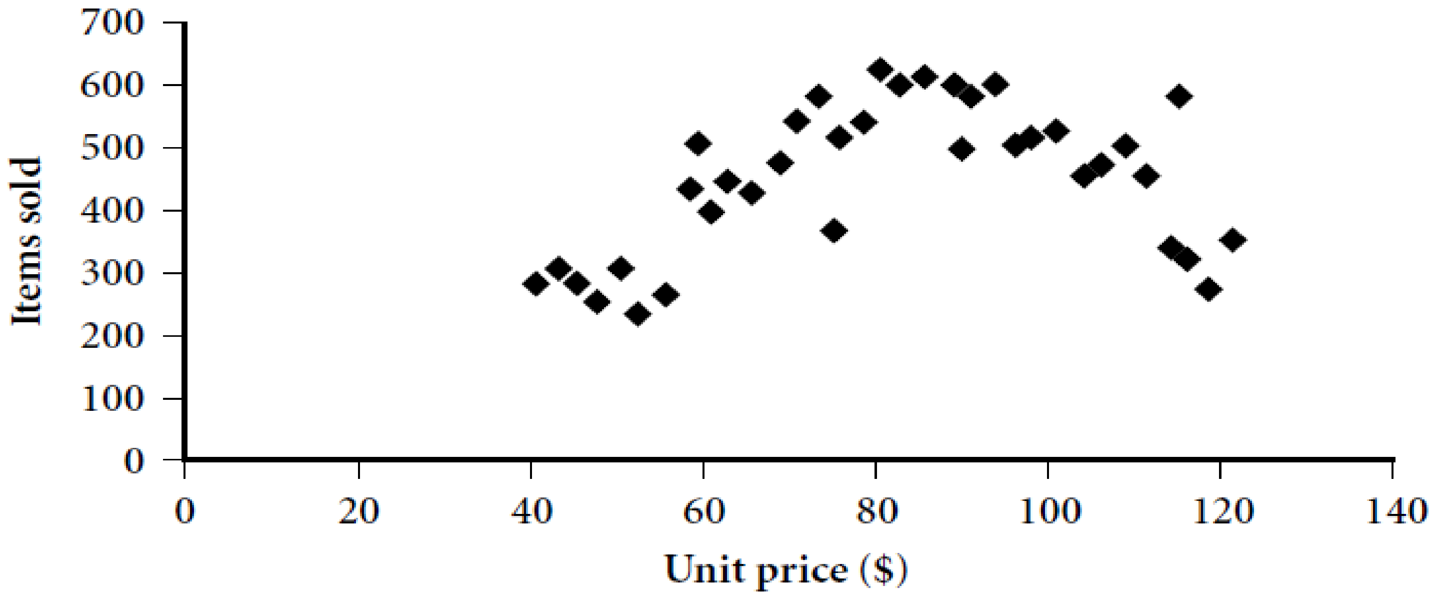
- We see that at this quantile, the unit price of items sold at branch 1 was slightly less than that at branch 2.
- In other words, 3% of items sold at branch 1 were less than or equal to $40, while 3% of items at branch 2 were less than or equal to $42.
- At the highest quantile, we see that the unit price of items at branch 2 was slightly less than that at branch 1.
- In general, we note that there is a shift in the distribution of branch 1 with respect to branch 2 in that the unit prices of items sold at branch 1 tend to be lower than those at branch 2.

Scatter Plot

- A scatter plot is one of the most effective graphical methods for determining if there appears to be a relationship, pattern, or trend between two numerical attributes.
- To construct a scatter plot, each pair of values is treated as a pair of coordinates in an algebraic sense and plotted as points in the plane.

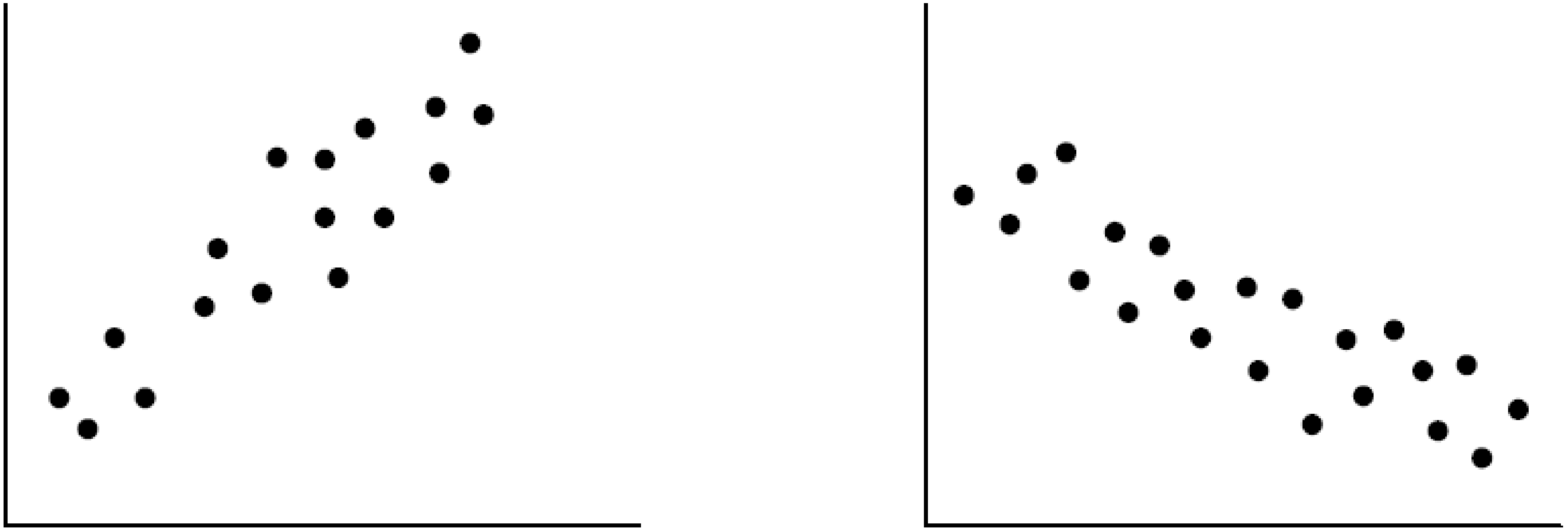
- Figure shows a scatter plot for the set of data in Table 2.1.
- The scatter plot is a useful method for providing a first

look at bivariate data to see clusters of points and outliers, or to explore the possibility of correlation relationships.



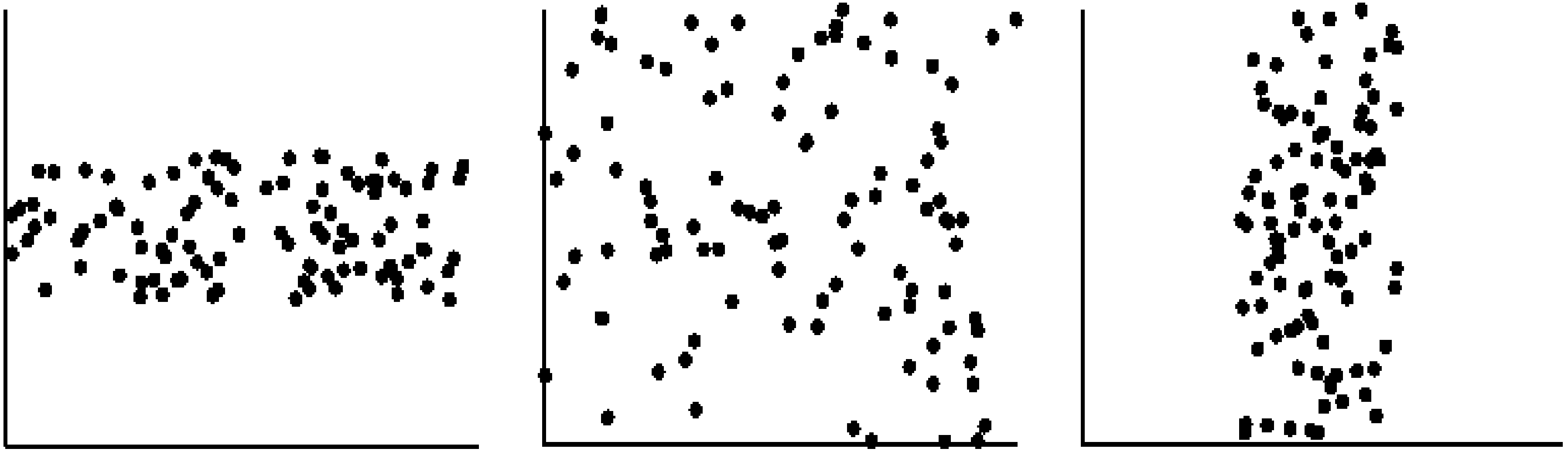
A scatter plot for the data set of Table 2.1.

- In Figure we see examples of positive and negative correlations between two attributes in two different data sets



8 Scatter plots can be used to find (a) positive or (b) negative correlations between attributes.

- Figure shows three cases for which there is no correlation relationship between the two attributes in each of the given data sets.



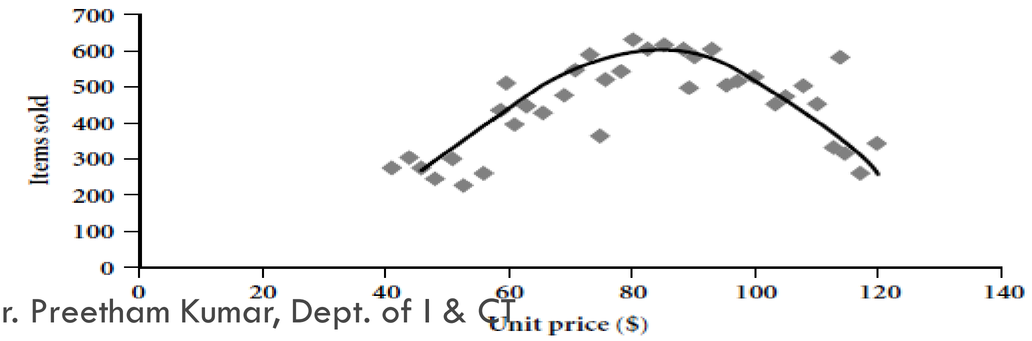
Three cases where there is no observed correlation between the two plotted attributes in each of the data sets.

scatter-plot matrix

- When dealing with several attributes, the scatter-plot matrix is a useful extension to the scatter plot.
- Given *n* attributes, a scatter-plot matrix is an *nxn* grid of scatter plots that provides a visualization of each attribute (or dimension) with every other attribute.
- The scatter-plot matrix becomes less effective as the number of attributes under study grows.

loess curve

- □ A loess curve is another important exploratory **graphic aid that adds a smooth curve** to a scatter plot in order to provide better perception of the pattern of dependence.
- □ The word *loess* is short for “local regression.” Figure shows a loess curve for the set of data in Table 2.1.



ure 2.10 A loess curve for the data set of Table 2.1.

- To fit a loess curve, values need to be set for two parameters—, α, a smoothing parameter,

and λ, the degree of the polynomials that are fitted by the regression.

While α, can be any positive number (typical values are between 1/4 and 1),

λ, can be 1 or 2.

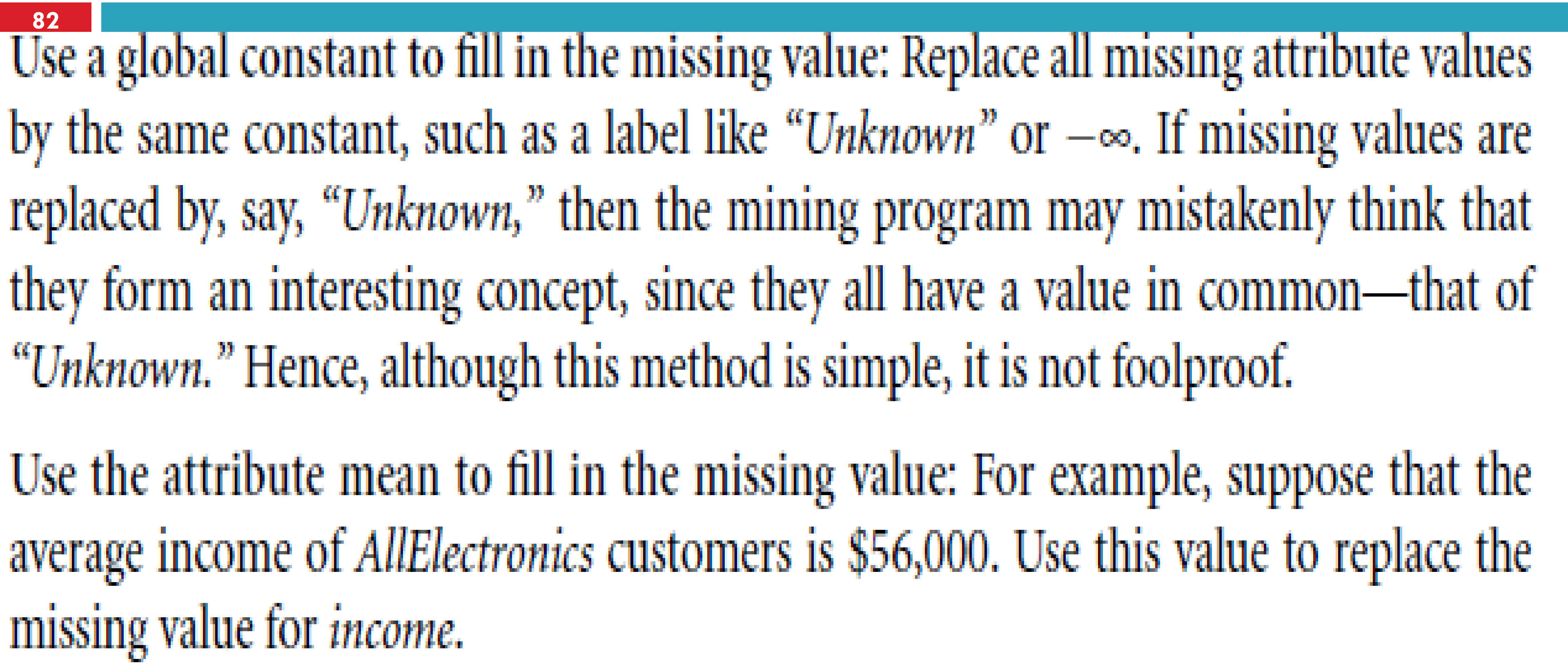
The goal in choosing α, is to produce a fit that is as smooth as possible without unduly distorting the underlying pattern in the data. The curve becomes smoother as a increases.

- □ There may be some lack of fit, however, indicating possible “missing” data patterns.
- □ If α_3 is very small, the underlying pattern is tracked, yet over fitting of the data may occur where local “wiggles” in the curve may not be supported by the data.

Data Cleaning

- **Missing Values:** How can you go about filling in the missing values for this attribute?
- **Ignore the tuple:** This is usually done when the class label is missing (assuming the mining task involves classification).
- This method is 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.

- **Fill in the missing value manually:** In general, this approach is 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*." Hence, although this method is simple, it is not foolproof.

Use the attribute mean to fill in the missing value: For example, suppose that the average income of *AllElectronics* customers is $56,000. Use this value to replace the missing value for *income*.

- **Use the attribute mean for all samples belonging to the same class as the given tuple:**
- For example, if classifying customers according to *credit risk*, replace the missing value with the average *income* value for customers in the same credit risk category as that of the given tuple.

- **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 your data set, you may construct a decision tree to predict the missing values for *income*.

Noisy Data

- ***"What is noise?"*** : Noise is a random error or variance in a measured variable.
- Given a numerical attribute such as, say, *price*, how can we "smooth" out the data to remove the noise? L
- Let's look at the following data smoothing techniques

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

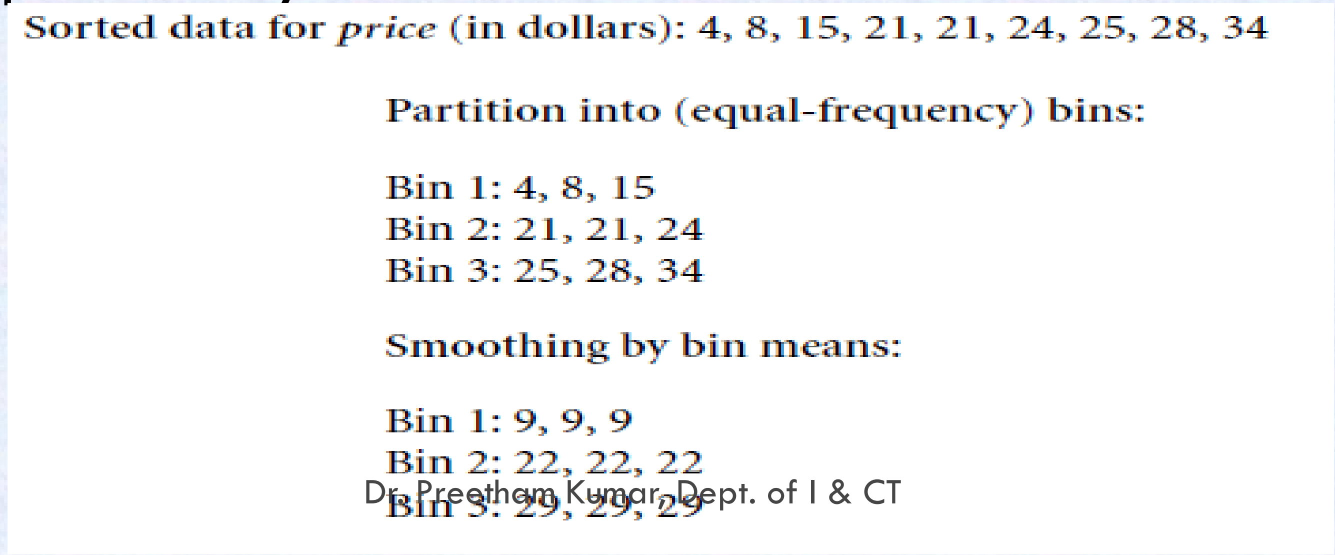
of "buckets," or *bins*.

□ Because binning methods consult the neighborhood of values, they perform *local* smoothing

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

smoothing by bin means

- □ In smoothing by bin means, each value in a bin is replaced by the mean value of the bin.
- □ For example, 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.



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:

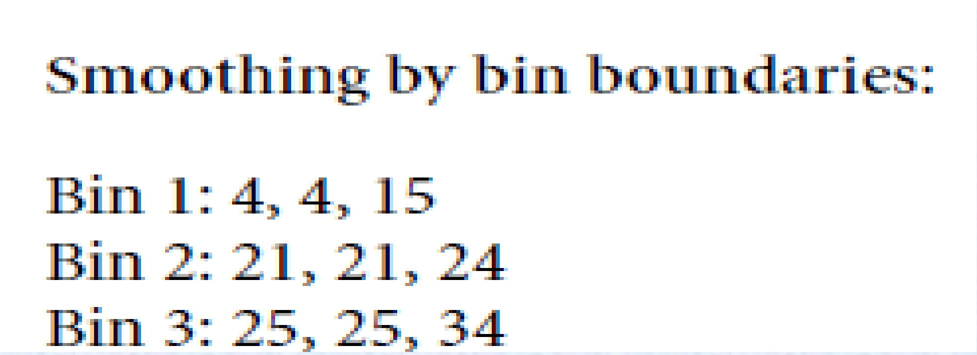
Bin 1: 9, 9, 9
Bin 2: 22, 22, 22
Bin 3: 29, 29, 29

smoothing by bin medians

- Smoothing by bin medians can be employed, in which each bin value is replaced by the bin median.
- Bin 1 : 8,8,8
- Bin2 : 21, 21, 21
- Bin3 : 28, 28,28

smoothing by bin boundaries

- □ In 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.



Smoothing by bin boundaries:

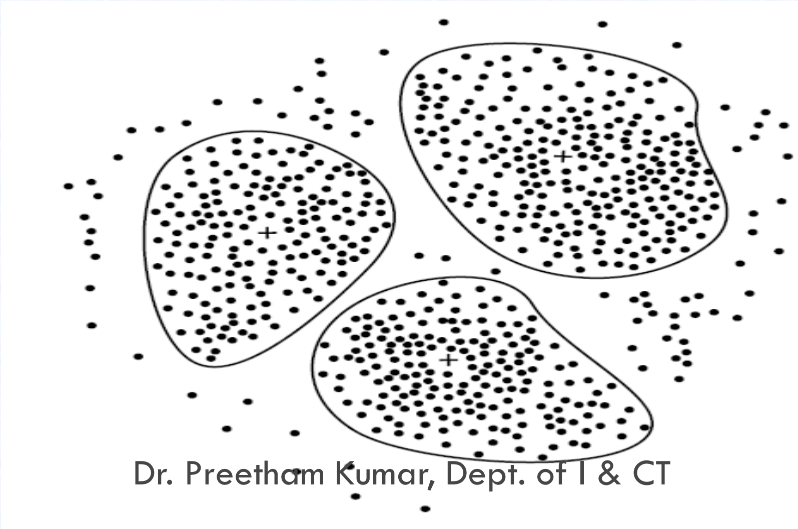
Bin 1: 4, 4, 15
Bin 2: 21, 21, 24
Bin 3: 25, 25, 34

Regression

- **Regression:** Data can be smoothed by fitting the data to a function, such as with regression.
- *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

Clustering:

- Clustering: Outliers may be detected by clustering, where similar values are organized into groups, or "clusters."
- Intuitively, values that fall outside of the set of clusters may be considered outliers.



Data Cleaning as a Process

□ **The first step in data cleaning as a process is *discrepancy detection.***

□ Discrepancies can be caused by several factors, including poorly designed data entry forms that have many optional fields, human error in data entry, deliberate errors and data decay (e.g., outdated addresses).

- □ Discrepancies may also arise from inconsistent data representations and the inconsistent use of codes.
- □ Errors in instrumentation devices that record data, and system errors, are another source of discrepancies.
- □ Errors can also occur when the data are (inadequately) used for purposes other than originally intended.
- □ There may also be inconsistencies due to data integration (e.g.,where a given attribute can have different names in different databases

“So, how can we proceed with discrepancy detection ?”

- □ As a starting point, use any knowledge you may already have regarding properties of the data.
- □ Such knowledge or “data about data” is referred to as metadata.
- □ For example , what are the domain and data type of each attribute?
- □ What are the acceptable values for each attribute ?

- □ What is the range of the length of values?
- □ Do all values fall within the expected range ?
- □ Are there any known dependencies between attributes?
- □ The descriptive data summaries presented in Section 2.2 are useful here for grasping data trends and identifying anomalies.
- □ For example, values that are more than two standard deviations away from the mean for a given attribute may be flagged as potential outliers.
- □ In this step, you may write your own scripts and/or use some of the tools that we discuss further below.
- □ From this, you may find noise, outliers, and unusual values that need investigation.

- As a data analyst, you should be on the lookout for the inconsistent use of codes and any inconsistent data representations

(such as “2004/12/25” and “25/12/2004” for *date*).

- The data should also be examined regarding unique rules, consecutive rules, and null rules.
- A unique rule says that each value of the given

attribute must be different from all other values for that attribute.

- A consecutive rule says that there can be no missing values between the lowest and highest values for the attribute, and that all values must also be unique
- A 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.

As mentioned already, reasons for missing values may include

- (1) the person originally asked to provide a value for the attribute refuses and/or finds that the information requested is not applicable (e.g., a *license-number* attribute left blank by non drivers);
- (2) the data entry person does not know the correct value; or
- (3) the value is to be provided by a later step of the process.
- The null rule should specify how to record the null condition, for example, such as to store zero for numerical attributes, a blank for character attributes, or any other conventions that may be in use (such as that entries like “don’t know” or “?” should be transformed to blank).

- There are a number of different commercial tools that can aid in the step of discrepancy detection.

- Data scrubbing tools use simple domain knowledge (e.g., knowledge of postal addresses, and spell-checking) to detect errors and make corrections in the data.

- Data auditing tools find discrepancies by analyzing the data to discover rules and relationships, and detecting data that violate such conditions.
- They are variants of data mining tools.
- For example, they may employ statistical analysis to find correlations, or clustering to identify outliers.

- □ Some data inconsistencies may be corrected manually using external references.
- □ For example, errors made at data entry may be corrected by performing a paper trace.
- □ Most errors, however, will require *data transformations.*
- □ This is the second step in data cleaning as a process.
- □ That is, once we find discrepancies, we typically need to define and apply (a series of) transformations to correct them.

- Transformations Commercial tools can assist in the data transformation step. Data migration tools allow simple transformations to be specified, such as to replace the string "*gender*" by "*sex*".
- ETL (extraction/transformation/loading) tools allow users to specify transforms through a graphical user interface (GUI).
- These tools typically support only a restricted set of transforms so that, often, we may also choose to write custom scripts for this step of the data cleaning process.

- The two-step process of discrepancy detection and data transformation (to correct discrepancies) iterates.
- This process, however, is error-prone and time-consuming.
- Some transformations may introduce more discrepancies.
- Some *nested discrepancies* may only be detected after others have been fixed.
- For example, a typo such as “20004” in a year field may only surface once all date values have been converted to a uniform format..

Data Integration and Transformation

- Data mining often requires data integration—the merging of data from multiple data stores.
- The data may also need to be transformed into forms appropriate for mining.

Data Integration

- □ Which combines data from multiple sources into a coherent data store, as in data warehousing.
- □ These sources may include multiple databases, data cubes, or flat files.

There are a number of issues to consider during data integration.

1. Schema integration and *object matching*

- □ How can equivalent real-world entities from multiple data sources be matched up?
- □ This is referred to as the entity identification problem.
- □ For example, how can the data analyst or the computer be sure that *customer id* in one database and *cust number* in another refer to the same attribute?
- □ Examples of metadata for each attribute include the name, meaning, data type, and range of values permitted for the attribute, and null rules for handling blank, zero, or null values.

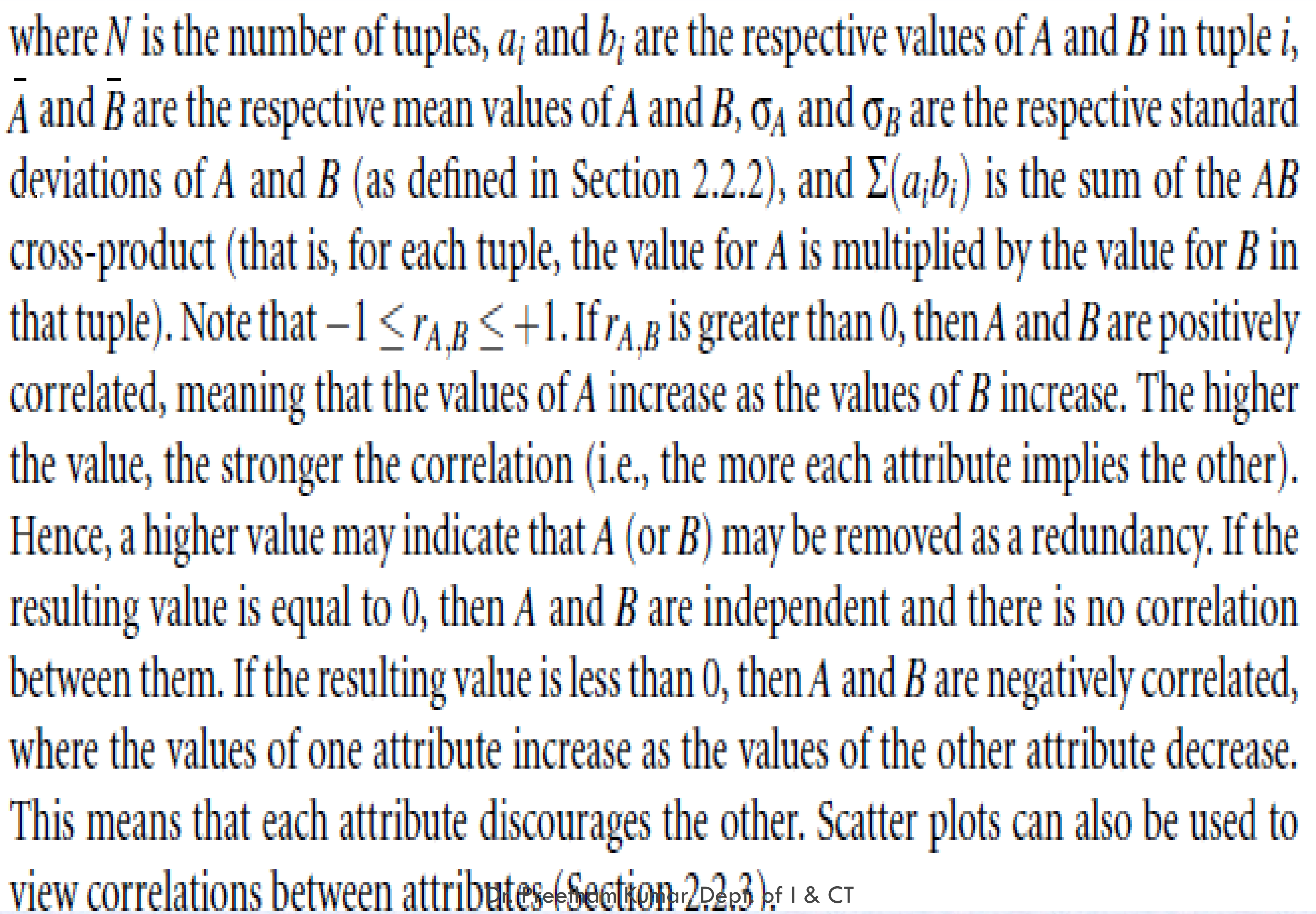
- Such metadata can be used to help avoid errors in schema integration.
- The metadata may also be used to help transform the data (e.g., where data codes for *pay type* in one database may be "*H*" and "*S*", and *1* and *2* in another).
- Hence, this step also relates to data cleaning

2. *Redundancy*

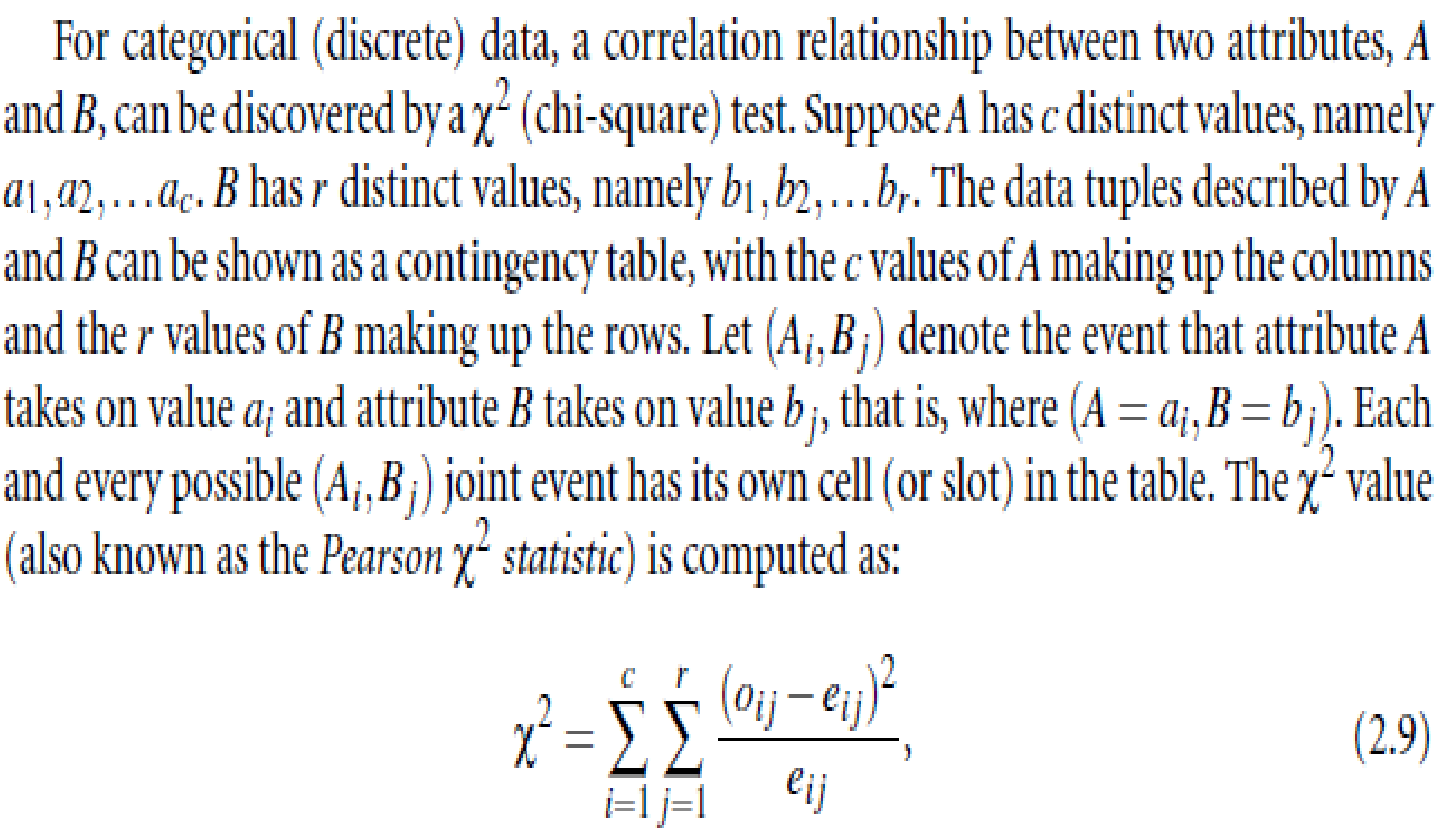
- □ An attribute (such as *annual revenue*, for instance) may be redundant if it can be "derived" from another attribute or set of attributes.
- □ Inconsistencies in attribute or dimension naming can also cause redundancies in the resulting data set.
- □ Some redundancies can be detected by correlation analysis.

- □ Given two attributes, correlation analysis can measure how strongly one attribute implies the other, based on the available data.
- □ For numerical attributes, we can evaluate the correlation between two attributes, *A*and *B*, by computing the correlation coefficient (also known as *Pearson's product moment coefficient*, named after its inventer, Karl Pearson).
- □ This is

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

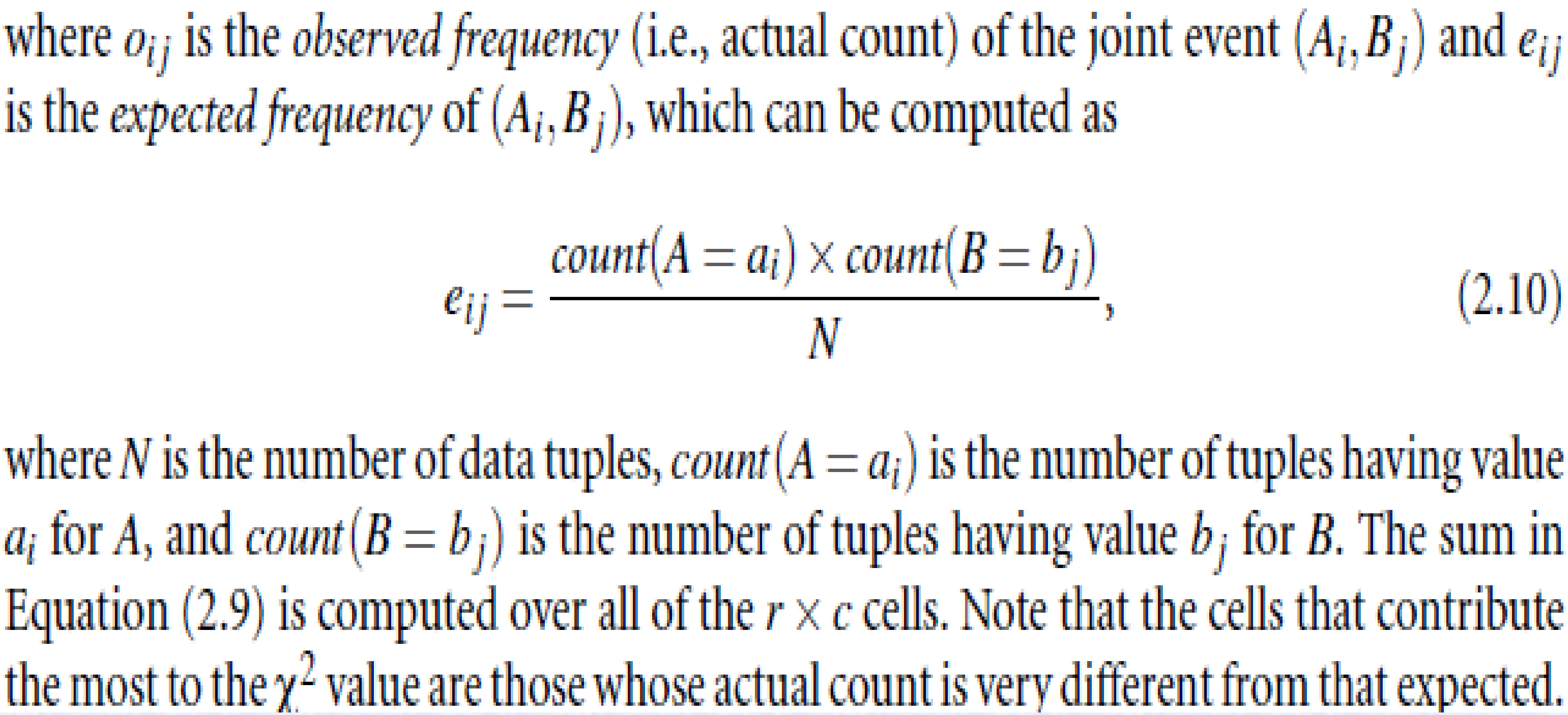


where N is the number of tuples, a_i and b_i are the respective values of A and B in tuple i, $\bar{A}$ and $\bar{B}$ are the respective mean values of A and B, σ_A and σ_B are the respective standard deviations of A and B (as defined in Section 2.2.2), and $\Sigma(a_ib_i)$ is the sum of the AB cross-product (that is, for each tuple, the value for A is multiplied by the value for B in that tuple). Note that $-1 \le r_{A,B} \le +1$. If $r_{A,B}$ is greater than 0, then A and B are positively correlated, meaning that the values of A increase as the values of B increase. The higher the value, the stronger the correlation (i.e., the more each attribute implies the other). Hence, a higher value may indicate that A (or B) may be removed as a redundancy. If the resulting value is equal to 0, then A and B are independent and there is no correlation between them. If the resulting value is less than 0, then A and B are negatively correlated, where the values of one attribute increase as the values of the other attribute decrease. This means that each attribute discourages the other. Scatter plots can also be used to view correlations between attributes (Section 2.2.3).



For categorical (discrete) 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, namely $a_1, a_2, \ldots a_c$. B has r distinct values, namely $b_1, b_2, \ldots b_r$. The data tuples described by A and B can be shown as a contingency table, with the c values of A making up the columns and the r values of B making up the rows. Let (A_i, B_j) denote the event that attribute A takes on value a_i and attribute B takes on value b_j, that is, where $(A = a_i, B = b_j)$. Each and every possible (A_i, B_j) joint event has its own cell (or slot) in the table. The χ^2 value (also known as the *Pearson* χ^2 *statistic*) is computed as:

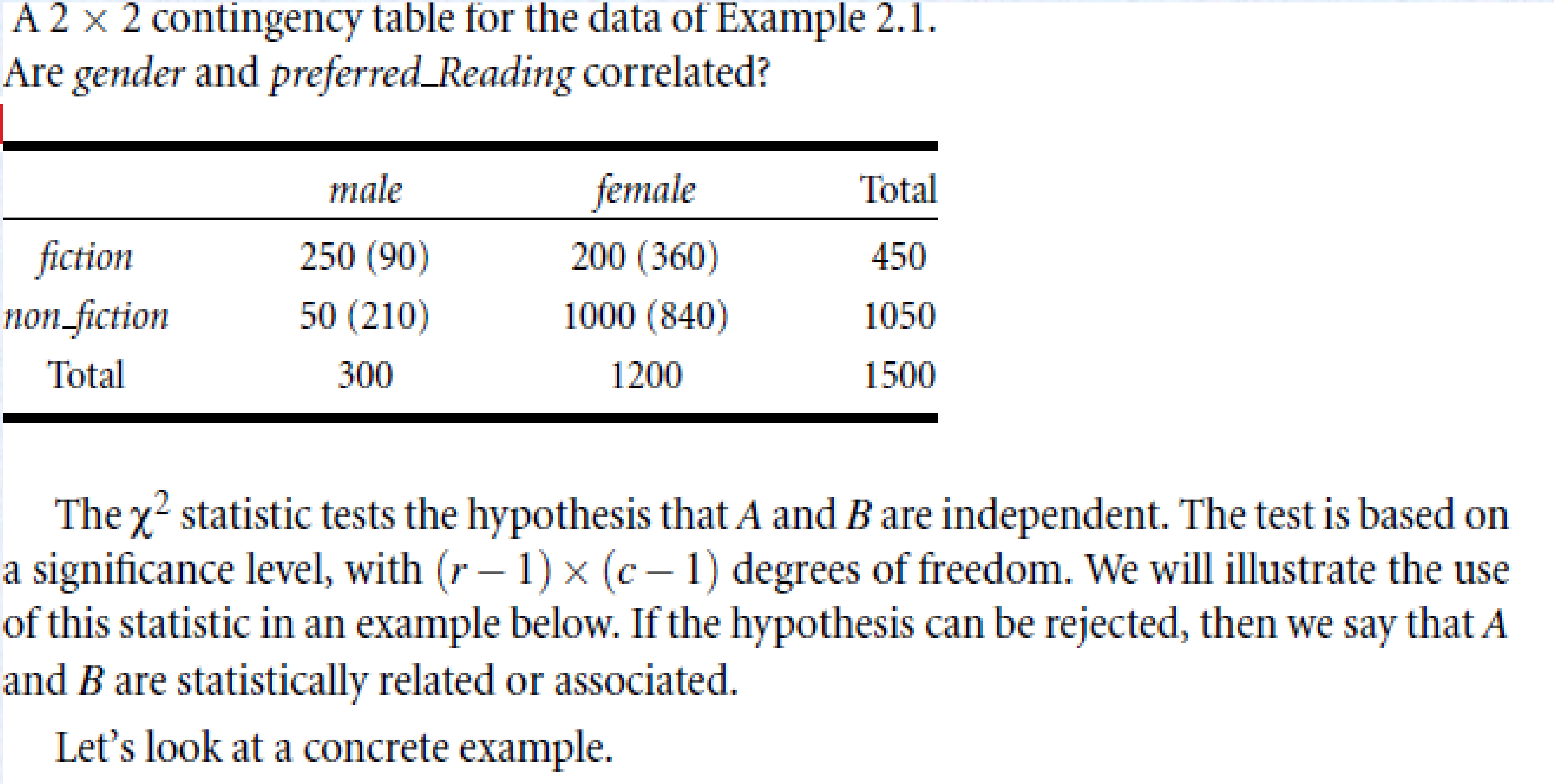
$$\chi^2 = \sum_{i=1}^{c} \sum_{j=1}^{r} \frac{(o_{ij} - e_{ij})^2}{e_{ij}}, \qquad (2.9)$$



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

$$e_{ij} = \frac{count(A = a_i) \times count(B = b_j)}{N}, \qquad (2.10)$$

where N is the number of data tuples, $count(A = a_i)$ is the number of tuples having value a_i for A, and $count(B = b_j)$ is the number of tuples having value b_j for B. The sum in Equation (2.9) is computed over all of the $r \times c$ cells. Note that the cells that contribute the most to the χ^2 value are those whose actual count is very different from that expected.



A 2 × 2 contingency table for the data of Example 2.1. Are *gender* and *preferred_Reading* correlated?

	male	*female*	Total
fiction	250 (90)	200 (360)	450
non_fiction	50 (210)	1000 (840)	1050
Total	300	1200	1500

The χ^2 statistic tests the hypothesis that A and B are independent. The test is based on a significance level, with $(r-1)\times(c-1)$ degrees of freedom. We will illustrate the use of this statistic in an example below. If the hypothesis can be rejected, then we say that A and B are statistically related or associated.

Let's look at a concrete example.

- Correlation analysis of categorical attributes using χ^2 .
- Suppose that a group of 1,500 people was surveyed.

- The gender of each person was noted.
- Each person was polled as to whether their preferred type of reading material was fiction or nonfiction.
- Thus, we have two attributes, *gender* and *preferred reading*.
- The observed frequency (or count) of each possible joint event is summarized in the contingency table shown in Table 2.2, where the numbers in parentheses are the expected frequencies (calculated based on the data distribution for both attributes, using Equation (2.10)).

Using Equation (2.10), we can verify the expected frequencies for each cell. For example, the expected frequency for the cell (male, fiction) is

$$e_{11} = \frac{count(male) \times count(fiction)}{N} = \frac{300 \times 450}{1500} = 90,$$

and so on. Notice that in any row, the sum of the expected frequencies must equal the total observed frequency for that row, and the sum of the expected frequencies in any column must also equal the total observed frequency for that column. Using Equation (2.9) for χ^2 computation, we get

$$\begin{aligned}\chi^2 &= \frac{(250-90)^2}{90} + \frac{(50-210)^2}{210} + \frac{(200-360)^2}{360} + \frac{(1000-840)^2}{840} \\ &= 284.44 + 121.90 + 71.11 + 30.48 = 507.93.\end{aligned}$$

For this 2 × 2 table, the degrees of freedom are $(2-1)(2-1) = 1$. For 1 degree of freedom, the χ^2 value needed to reject the hypothesis at the 0.001 significance level is 10.828 (taken from the table of upper percentage points of the χ^2 distribution, typically available from any textbook on statistics). Since our computed value is above this, we can reject the hypothesis that *gender* and *preferred_reading* are independent and conclude that the two attributes are (strongly) correlated for the given group of people. ■

- In addition to detecting redundancies between attributes, duplication should also be detected at the tuple level (e.g., where there are two or more identical tuples for a given unique data entry case).
- The use of de-normalized tables (often done to improve performance by avoiding joins) is another source of data redundancy.
- Inconsistencies often arise between various duplicates, due to inaccurate data entry or updating some but not all of the occurrences of the data.

3. Detection and resolution of data value conflicts.

- □ For example, for the same real-world entity, attribute values from different sources may differ.
- □ This may be due to differences in representation, scaling, or encoding.
- □ For instance, a *weight* attribute may be stored in metric units in one system and British imperial units in another.

- For a hotel chain, the *price* of rooms in different cities may involve not only different currencies but also different services (such as free breakfast) and taxes.
- An attribute in one system may be recorded at a lower level of abstraction than the “same” attribute in another.
- For example, the *total sales* in one database may refer to one branch of *All Electronics*, while an attribute of the same name in another database may refer to the total sales for *All Electronics* stores in a given region.

- When matching attributes from one database to another during integration, special attention must be paid to the *structure* of the data.
- This is to ensure that any attribute functional dependencies and referential constraints in the source system match those in the target system.
- For example, in one system, a *discount* may be applied to the order, whereas in another system it is applied to each individual line item within the order.
- If this is not caught before integration, items in the target system may be improperly discounted.

Data Transformation

- In *data transformation*, the data are transformed or consolidated into forms appropriate for mining.
- Data transformation can involve the following:

- **Smoothing,** which works to remove noise from the data. Such techniques include binning, regression, and clustering.
- Smoothing is a form of data cleaning., where users specify transformations to correct data inconsistencies
- **Aggregation,** where summary or aggregation operations are applied to the data.
- For example, the daily sales data may be aggregated so as to compute monthly and annual total amounts.
- This step is typically used in constructing a data cube for analysis of the data at multiple granularities.

- **Generalization** of the data, where low-level or “primitive” (raw) data are replaced by higher-level concepts through the use of concept hierarchies.
- For example, 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*.
- Aggregation and generalization serve as forms of data reduction

- **Normalization,** where the attribute data are scaled so as to fall within a small specified range, such as -1 to 0 or 0 to 1.
- **Attribute construction** (or *feature construction*),where new attributes are constructed and added from the given set of attributes to help the mining process

Normalization

- An attribute is normalized by scaling its values so that they fall within a small specified range, such as 0.0 to 1.0.
- Normalization is particularly useful for classification algorithms involving neural networks, or distance measurements such as nearest-neighbor classification and clustering.
- There are many methods for data normalization. We study three:
- *min-max normalization,*
- *z-score normalization*
- *normalization by decimal scaling*

Min-max normalization

□ Performs a linear transformation on the original data.

□ Suppose that *minA* and *maxA* are the minimum and maximum values of an attribute, *A*.

□ Min-max normalization maps a value, *v*, of *A* to *v'* in the range [*new minA*; *new maxA*] by computing

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

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

- □ Suppose that the minimum and maximum values for the attribute *income* are \$12,000 and \$98,000, respectively.
- □ We would like to map *income* to the range [0 to1].
- □ By min-max normalization, a value of \$73,600 for *income* is transformed to

$$\frac{73{,}600 - 12{,}000}{98{,}000 - 12{,}000}(1.0 - 0) + 0 = 0.716.$$

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, of *A* is normalized to v’ by computing

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

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.

- □ Suppose that the mean and standard deviation of the values for the attribute *income* are $54,000 and $16,000, respectively.
- □ With z-score normalization, a value of $73,600 for *income* is transformed to

$$\frac{73{,}600 - 54{,}000}{16{,}000} = 1.225.$$

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, of *A* is normalized to v' by computing

$$v' = \frac{v}{10^j},$$

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

- □ 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 1,000 (i.e., *j* = 3) so that -986 normalizes to -0.986 and 917 normalizes to 0:917

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.
- That is, mining on the reduced data set should be more efficient yet produce the same (or almost the same) analytical results.
- Strategies for data reduction include the following:

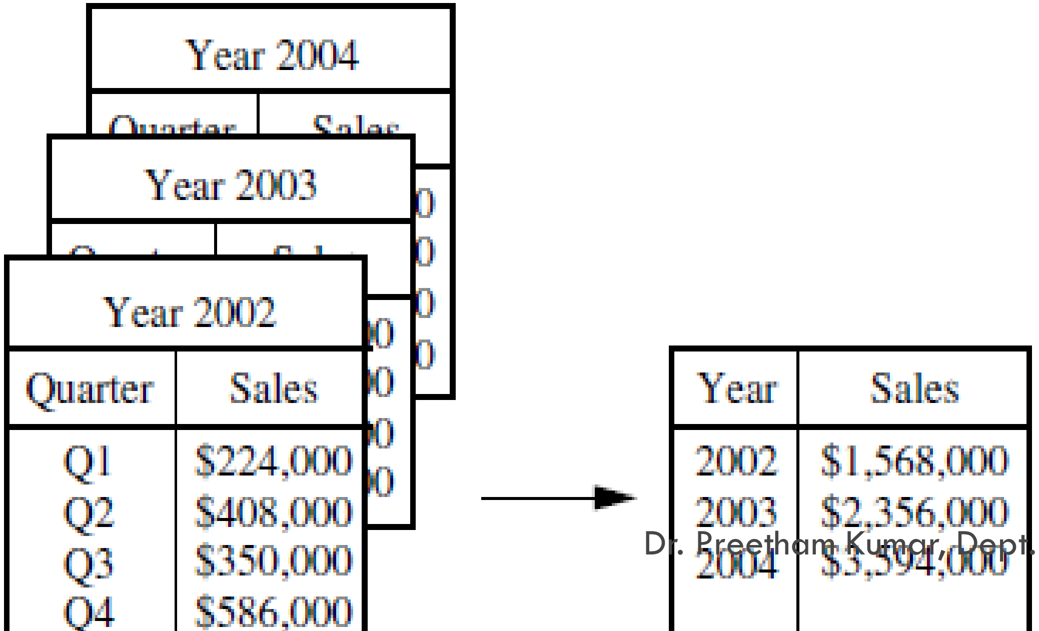
- **1.Data cube aggregation**, where aggregation operations are applied to the data in the construction of a data cube.
- **2. Attribute subset selection**, where irrelevant, weakly relevant, or redundant attributes or dimensions may be detected and removed.
- **3. Dimensionality reduction**, where encoding mechanisms are used to reduce the data set size.
- **4. Numerosity reduction** , where the data are replaced or estimated by alternative, smaller data representations such as parametric models (which need store only the model parameters instead of the actual data) or nonparametric methods such as clustering, sampling, and the use of histograms.

- **5. Discretization and concept hierarchy generation,** where raw data values for attributes are replaced by ranges or higher conceptual levels.
- 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

- Data cubes store multidimensional aggregated information.
- Imagine that you have collected the data for your analysis. These data consist of the *AllElectronics* sales per quarter, for the years 2010 to 2013.
- You are, however, 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.

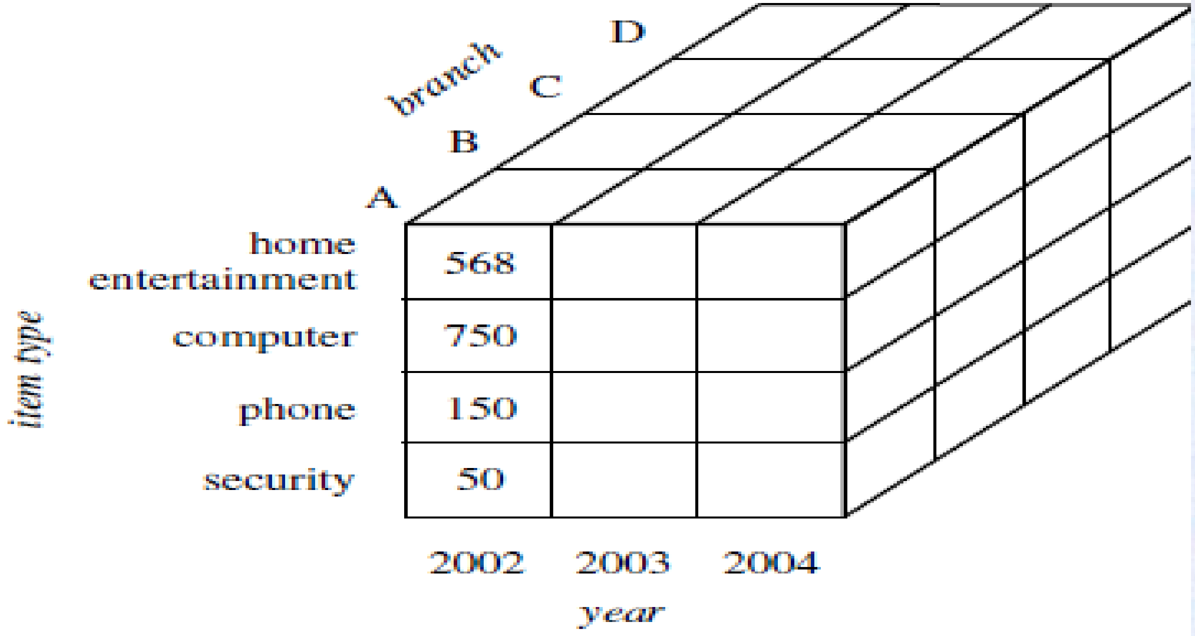
- This aggregation is illustrated in Figure.
- The resulting data set is smaller in volume, without loss of information necessary for the analysis task



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

- Data cubes store multidimensional aggregated information. For example, Figure shows a data cube for multidimensional analysis of sales data with respect to annual sales per item type for each *AllElectronics* branch.
- Each cell holds an aggregate data value, corresponding to the data point in multidimensional space.



A data cube for sales at *AllElectronics*.

- Data cubes provide fast access to precomputed, summarized data, thereby benefiting on-line analytical processing as well as data mining.
- 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*.

- In other words, the lowest level should be usable, or useful for the analysis.
- A cube at the highest level of abstraction is the *apex cuboid.*
- For the sales data of Figure, the apex cuboid would give one total—the total *sales* for all three years, for all item types, and for all branches.

- Data cubes created for varying levels of abstraction are often referred to as *cuboids*, so that a data cube may instead refer to a *lattice* of *cuboids.*
- Each higher level of abstraction further reduces the resulting data size.
- When replying to data mining requests, the *smallest* available cuboid relevant to the given task should be used.

Attribute Subset Selection

- Data sets for analysis may contain hundreds of attributes, many of which may be irrelevant to the mining task or redundant.
- For example, if the task is to classify customers as to whether or not they are likely to purchase a popular new CD at *All Electronics* when notified of a sale, attributes such as the customer’s telephone number are likely to be irrelevant, unlike attributes such as *age* or *music taste*.

- 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, especially when the behavior of the data is not well known (hence, a reason behind its analysis!).
- Leaving out relevant attributes or keeping irrelevant attributes may be detrimental, causing 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, 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, helping to make the patterns easier to understand

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

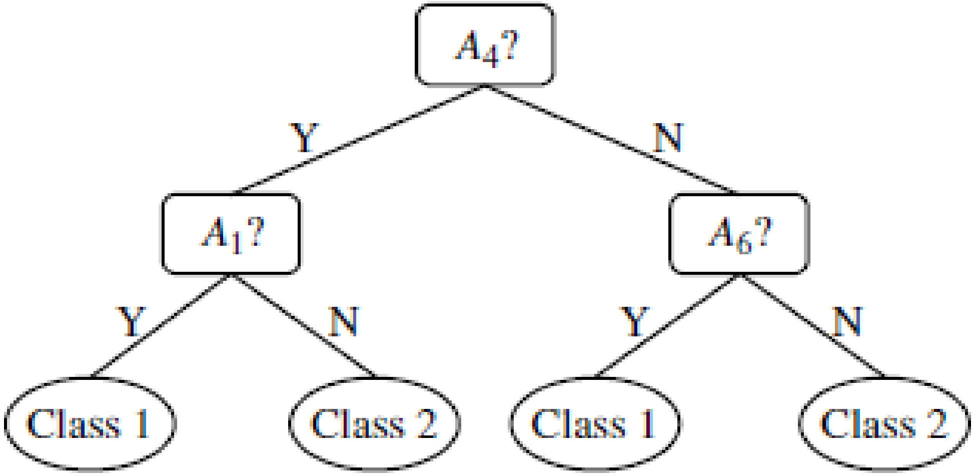
- For *n* attributes, there are 2^n possible subsets.
- An exhaustive search for the optimal subset of attributes can be prohibitively expensive, especially as *n* and the number of data classes increase.
- Therefore, heuristic methods that explore a reduced search space are commonly used for attribute 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.

- Basic heuristic methods of attribute subset selection include the following techniques,
- **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:** T
- □ 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 algorithms, such as ID3, C4.5, and CART,were originally intended for classification.
- Decision tree induction constructs a flow chart 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.
- The set of attributes appearing in the tree form the reduced subset of attributes.
- 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

Forward selection	Backward elimination	Decision tree induction
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: $\{A_1, A_4, A_6\}$	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: $\{A_1, A_4, A_6\}$	Initial attribute set: $\{A_1, A_2, A_3, A_4, A_5, A_6\}$  A_4? Y N A_1? A_6? Y N Y N Class 1 Class 2 Class 1 Class 2 => Reduced attribute set: $\{A_1, A_4, A_6\}$

Greedy (heuristic) methods for attribute subset selection.

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.

- □ There are several well-tuned algorithms for string compression.
- □ Although they are typically lossless, they allow only limited manipulation of the data
- □ Two popular and effective methods of lossy dimensionality reduction are *wavelet transforms* and *principal components analysis.*

Numerosity Reduction

- □ ***"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.
- □ For *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.)

- Log-linear models, which estimate discrete multidimensional probability distributions, are an example.
- *Nonparametric methods* for storing reduced representations of the data include histograms, clustering, and sampling.

Regression and Log-Linear Models

- Regression and log-linear models can be used to approximate the given data.
- In (simple) linear regression, the data are modeled to fit a straight line. For example, a random variable, y (called a *response variable*), can be modeled as a linear function of another random variable, x (called a *predictor variable*), with the equation

$$y = wx+b$$

- where the variance of y is assumed to be constant.

- In the context of data mining, *x* and *y* are numerical database attributes.
- The coefficients, *w* and *b* (called *regression coefficients*), specify the slope of the line and the *Y*-intercept, respectively.
- 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 is an extension of (simple) linear regression, which allows a response variable, *y*, to be modeled as a linear function of two or more predictor variables.

- 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 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).

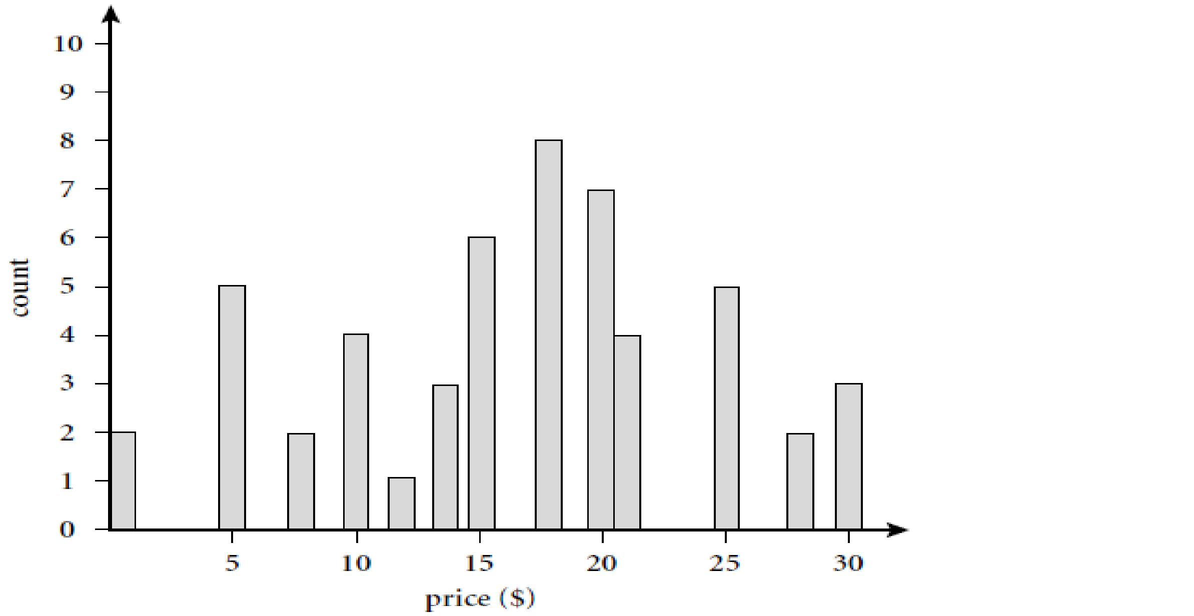
- □ Regression and log-linear models can both be used on sparse data, although their application may be limited.
- □ While both methods can handle skewed data, regression does exceptionally well.
- □ Regression can be computationally intensive when applied to high dimensional data, whereas log-linear models show good scalability for up to 10 or sodimensions

Histograms

- □ Histograms use binning to approximate data distributions and are a popular form of data reduction.
- □ A histogram for an attribute, *A*, partitions the data distribution of *A* into disjoint subsets, or *buckets*.
- □ If each bucket represents only a single attribute-value/frequency pair, the buckets are called *singleton buckets*.
- □ Often, buckets instead represent continuous ranges for thegiven attribute.

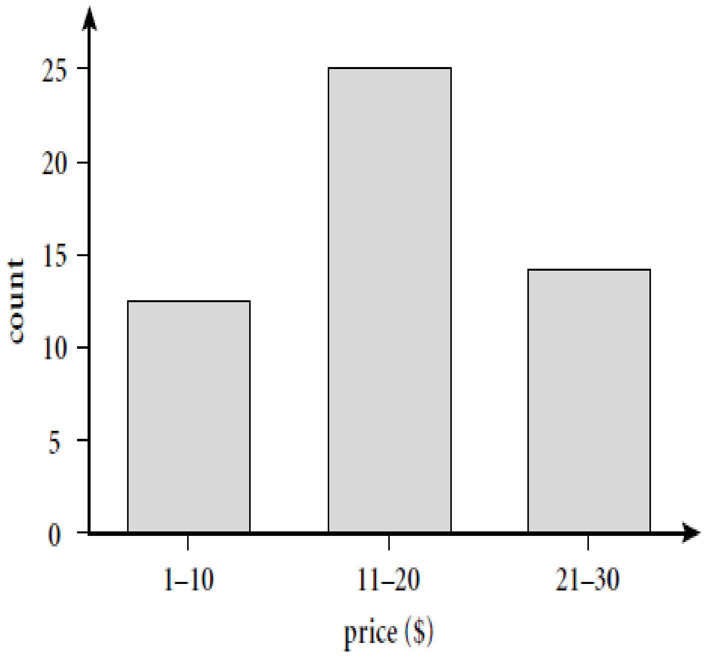
The following data are a list of prices of commonly sold items at *AllElectronics* (rounded to the nearest dollar). The numbers have been sorted:1, 1, 5, 5, 5, 5, 5, 8, 8, 10, 10, 10, 10, 12, 14, 14, 14, 15, 15, 15, 15, 15, 15, 18, 18, 18, 18, 18, 18, 18, 18, 20, 20, 20, 20, 20, 20, 20, 21, 21, 21, 21, 25, 25, 25, 25, 25, 28, 28, 30, 30, 30. shows a histogram for the data using

singleton buckets



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

- □ To further reduce the data, it is common to have each bucket denote a continuous range of values for the given attribute.
- □ In Figure, 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 (such as the width of $10 for the buckets in Figure
- **Equal-frequency (or equidepth):** In an equal-frequency histogram, the buckets are created so that, roughly, the frequency of each bucket is constant (that is, each bucket contains roughly the same number of contiguous data samples).g:

- □ **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.

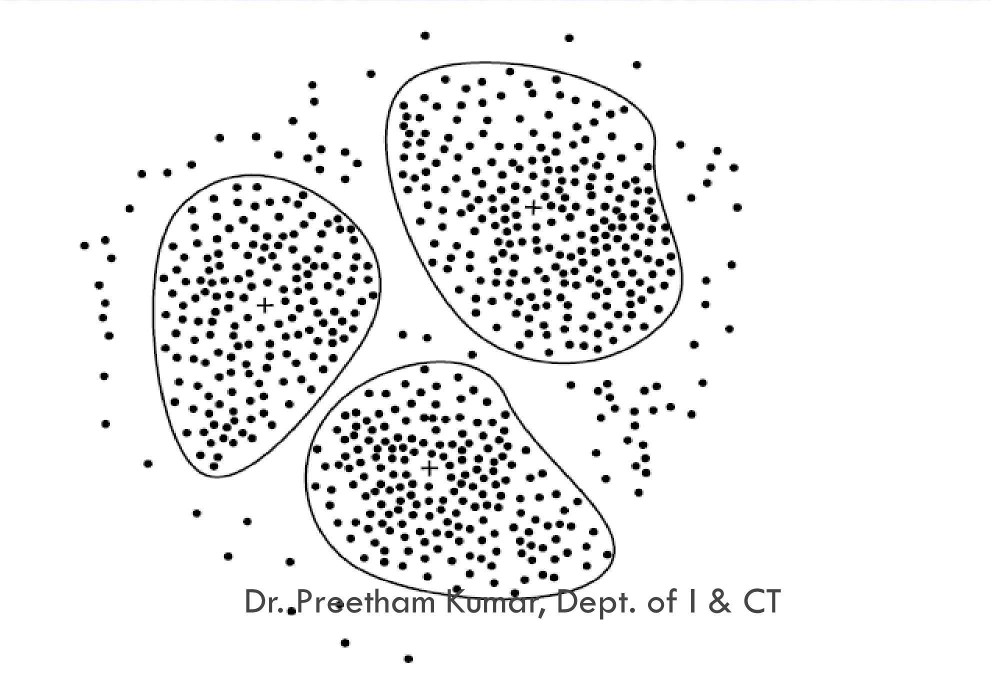
- **MaxDiff:** In a MaxDiff histogram, we consider the difference between each pair of adjacent values.
- A bucket boundary is established between each pair for pairs having the $\beta - 1$ largest differences, where β is the user-specified number of buckets.

Clustering

- Clustering techniques 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 (denoting the “average object,” or average point in space for the cluster).

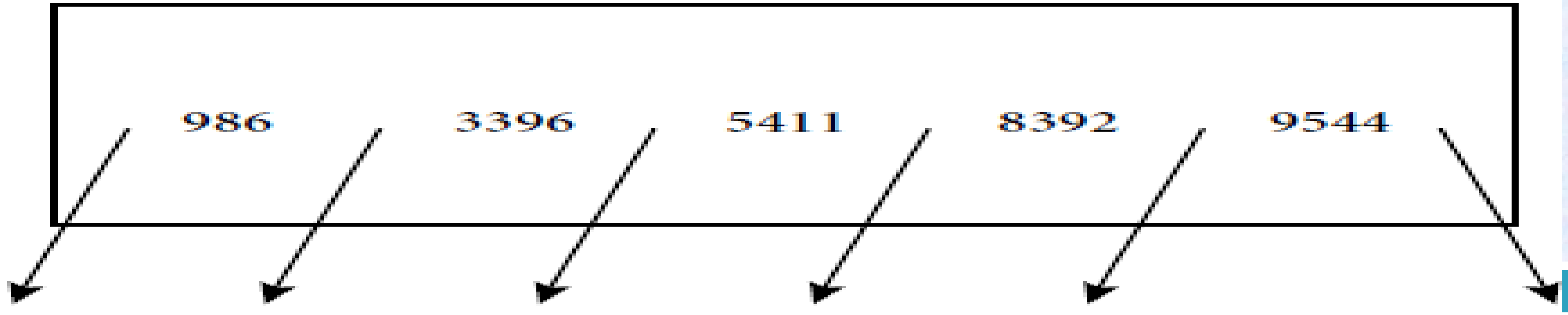
- Figure 2.12 of Section 2.3.2 shows a 2-D plot of customer data with respect to customer locations in a city, where the centroid of each cluster is shown with a “+”. Three data clusters are visible



- □ In database systems, multidimensional index trees are primarily used for providing
- □ fast data access.
- □ They can also be used for hierarchical data reduction, providing a multiresolution clustering of the data. This can be used to provide approximate answers to queries.
- □ An index tree recursively partitions the multidimensional space for a given set of data objects, with the root node representing the entire space. Such trees are typically balanced, consisting of internal and leaf nodes.

- Each parent node contains keys and pointers to child nodes that, collectively, represent the space represented by the parent node.
- Each leaf node contains pointers to the data tuples they represent (or to the actual tuples).
- An index tree can therefore store aggregate and detail data at varying levels of resolution or abstraction.

- It provides a hierarchy of clusterings of the data set, where each cluster has a label that holds for the data contained in the cluster.
- If we consider each child of a parent node as a bucket, then an index tree can be considered as a *hierarchical histogram.*
- For example, consider the root of a B+-tree as shown in following Figure, with pointers to the data keys 986, 3396, 5411, 8392, and 9544.



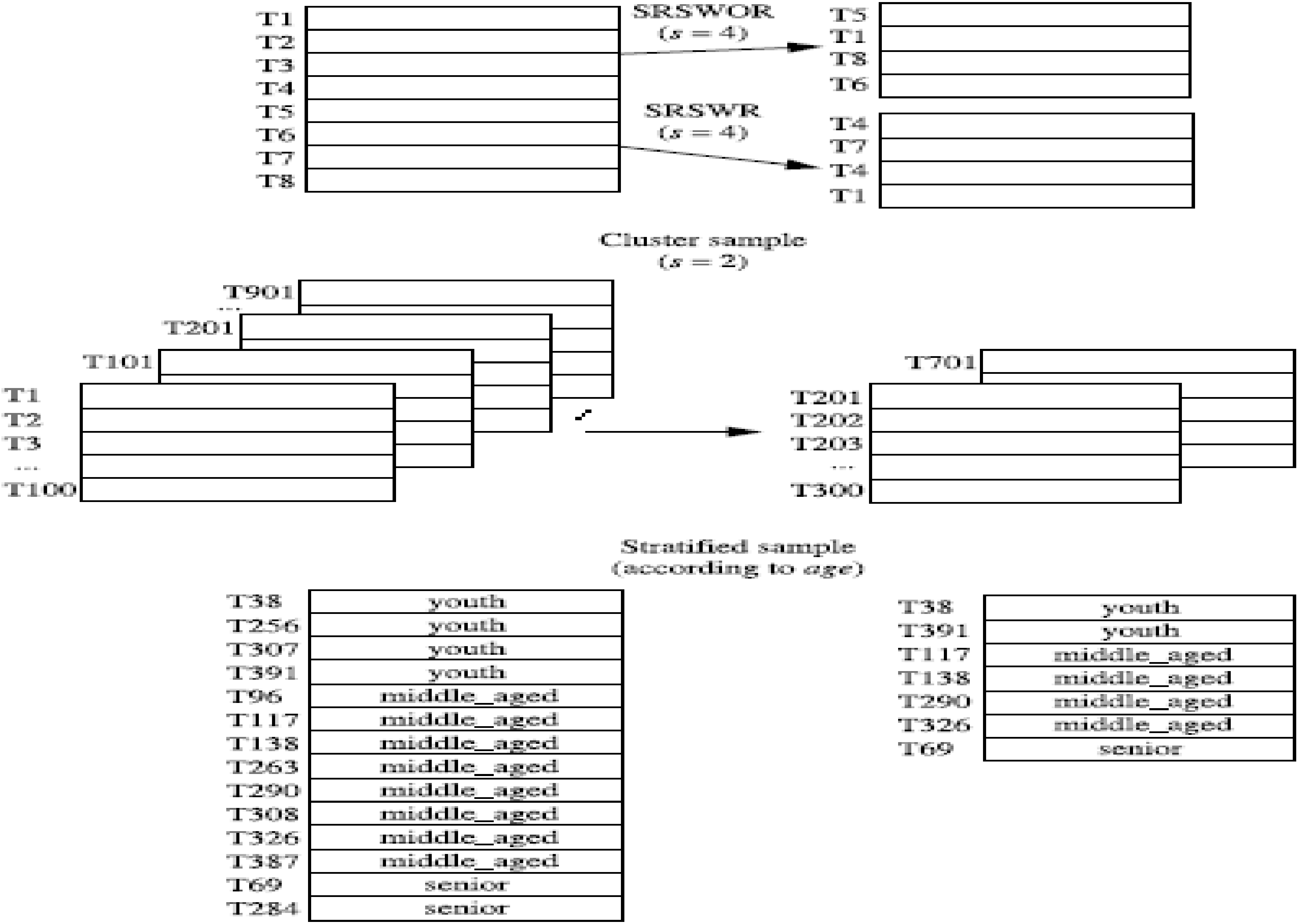
The root of a B+-tree for a given set of data.

- □ Suppose that the tree contains 10,000 tuples with keys ranging from 1 to 9999.
- □ The data in the tree can be approximated by an equal-frequency histogram of six buckets for the key ranges 1 to 985, 986 to 3395, 3396 to 5410, 5411 to 8391, 8392 to 9543, and 9544 to 9999.
- □ Each bucket contains roughly 10,000/6 items.
- □ Similarly, each bucket is subdivided into smaller buckets, allowing for aggregate data at a finer-detailed level.

- The use of multi dimensional index trees as a form of data reduction relies on an ordering of the attribute values in each dimension.
- Two-dimensional or multidimensional index trees include R-trees, quad-trees, and their
- variations. They are well suited for handling both sparse and skewed data.

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 sample (or subset) of the data.
- Suppose that a large data set, *D*, contains *N* tuples. Let's look at the most common ways that we could sample *D* for data reduction, as illustrated in Figure



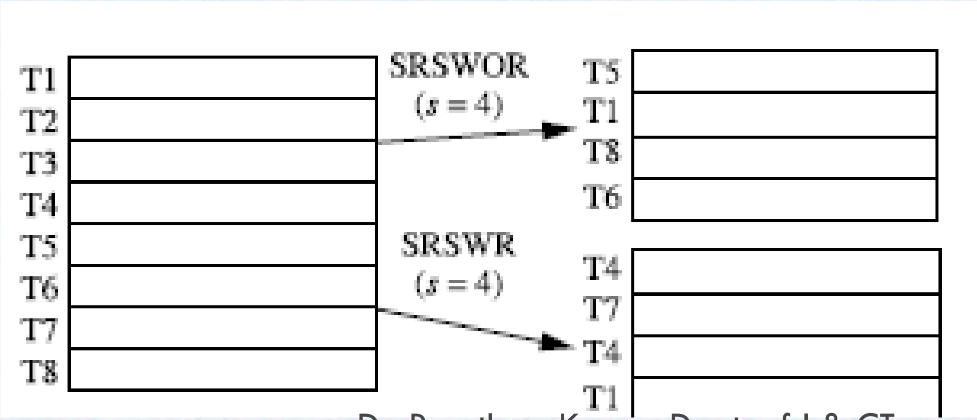
T38	youth
T256	youth
T307	youth
T391	youth
T96	middle_aged
T117	middle_aged
T138	middle_aged
T263	middle_aged
T290	middle_aged
T308	middle_aged
T326	middle_aged
T387	middle_aged
T69	senior
T284	senior

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

Sampling can be used for data reduction.

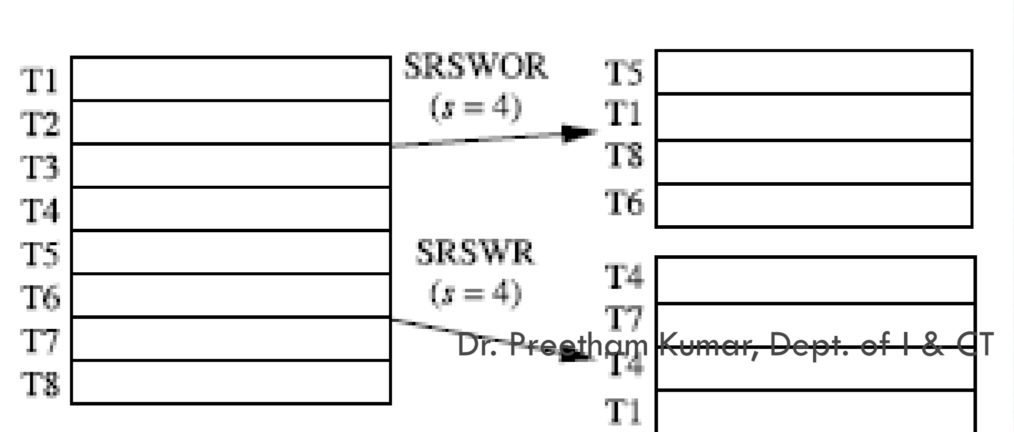
Simple random sample without replacement (SRSWOR) of size *s*:

- 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.

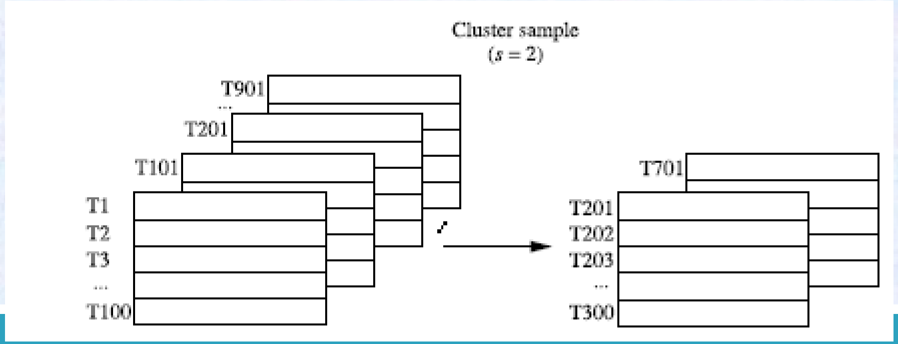


Simple random sample with replacement (SRSWR) of size *s*:

- □ 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 tuple is drawn, it is placed back in *D* so that it may be drawn again.



Cluster sample

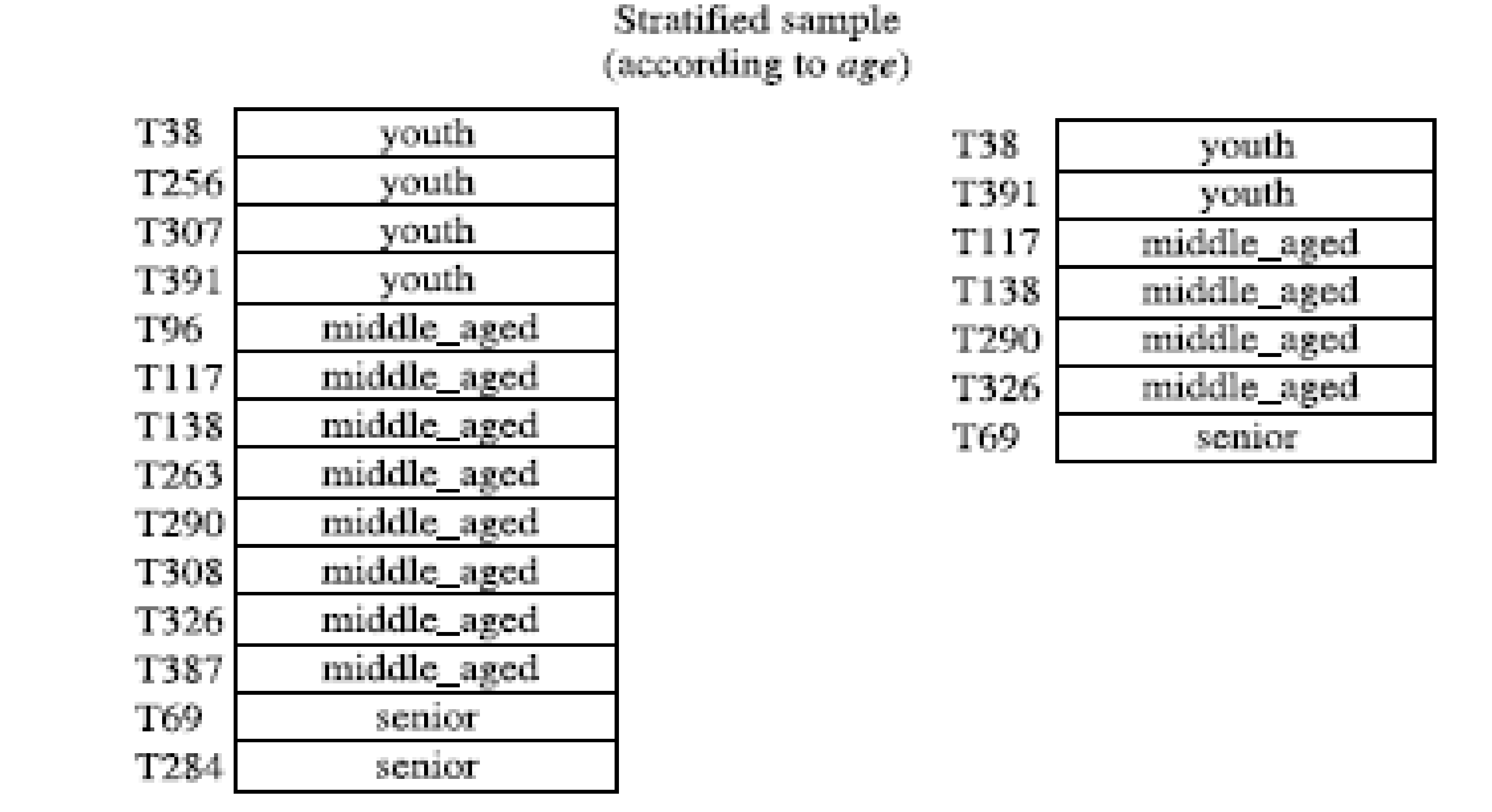


- If the tuples in *D* are grouped into *M* mutually disjoint “clusters,” then an SRS of *s* clusters can be obtained, where $s < M$.
- For example, tuples in a database are usually retrieved a page at a time, so that each page can be considered a cluster.
- A reduced data representation can be obtained 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.



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

- 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*.

Data Discretization and Concept Hierarchy Generation

- Data discretization techniques can be used to reduce the number of values for a given continuous attribute by dividing the range of the attribute into intervals.
- Interval labels can then be used to replace actual data values
- Replacing numerous values of a continuous attribute by a small number of interval labels thereby reduces and simplifies the original data.
- This leads to a concise, easy-to-use, knowledge-level representation of mining results.

- Discretization techniques can be categorized based on how the discretization is performed, such as whether it uses class information or which direction it proceeds (i.e., top-down vs. bottom-up).
- If the discretization process uses class information, then we say it is *supervised discretization.* Otherwise, it is *unsupervised.*

- 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, it is called *top-down discretization* or *splitting*.

- This contrasts with *bottom-up discretization* or *merging*, which 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.

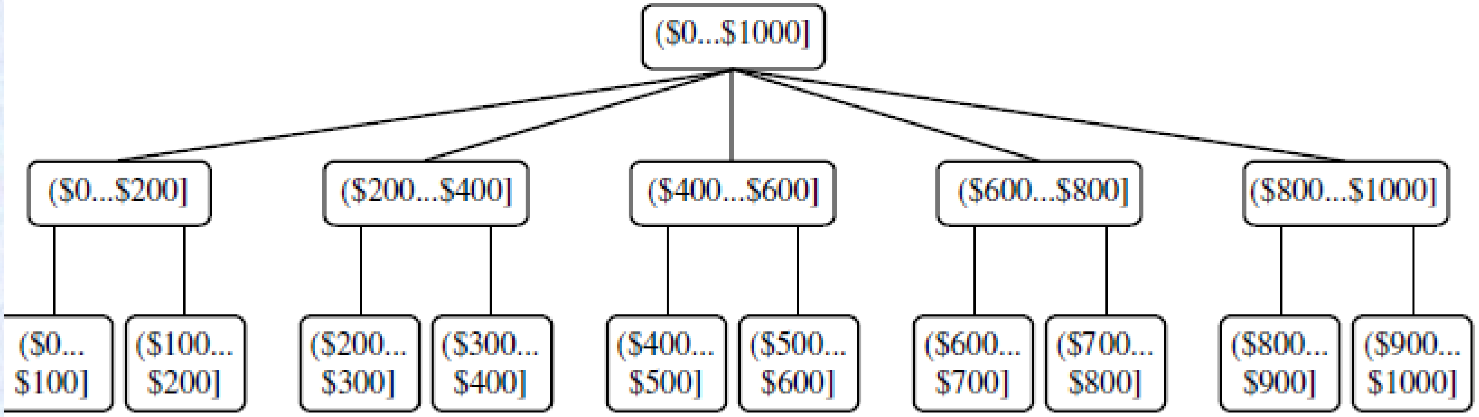
- Discretization can be performed recursively on an attribute to provide a hierarchical or multire solution partitioning of the attribute values, known as a concept hierarchy.
- Concept hierarchies are useful for mining at multiple levels of abstraction.

- A concept hierarchy for a given numerical attribute defines a discretization of the attribute.
- Concept hierarchies can be used to reduce the data by collecting and replacing low-level concepts (such as numerical values for the attribute *age*) with higher-level concepts (such as *youth, middle-aged*, or *senior*).
- Although detail is lost by such data generalization,

the generalized data may be more meaningful and easier to interpret

- This contributes to a consistent representation of data mining results among multiple mining tasks, which is a common requirement.
- In addition, mining on a reduced data set requires
- fewer input/output operations and is more efficient than mining on a larger, un-generalized data set.
- Because of these benefits, discretization techniques and concept hierarchies are typically applied before data mining as a preprocessing step, rather than during mining

- □ An example of a concept hierarchy for the attribute *price* is given in Figure.
- □ More than one concept hierarchy can be defined for the same attribute in order to accommodate the needs of various users.



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

- Manual definition of concept hierarchies can be a tedious and time-consuming task for a user or a domain expert.
- Fortunately, several discretization methods can be used to automatically generate or dynamically refine concept hierarchies for numerical attributes.
- Furthermore, many hierarchies for categorical attributes are implicit within the database schema and can be automatically defined at the schema definition level.

Discretization and Concept Hierarchy Generation for Numerical Data

- Concept hierarchies for numerical attributes can be constructed automatically based on data discretization.
- The following methods exist:
  - *binning*,
  - *histogram analysis*,
  - *entropy-based discretization*,
  - Chi-square *merging*,
  - *cluster analysis*, and
  - *discretization by intuitive partitioning*.

Binning

- Binning is a top-down splitting technique based on a specified number of bins.
- We have already discussed binning methods for data smoothing.
- These methods are also used as discretization methods for numerosity 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, as in *smoothing by bin means* or *smoothing by bin medians*, respectively..

- These techniques can be applied recursively to the resulting partitions in order to generate concept hierarchies.
- Binning does not use class information and is therefore an unsupervised discretization technique.
- It is sensitive to the user-specified number of bins, as well as the presence of outliers

Histogram Analysis

- Like binning, histogram analysis is an unsupervised discretization technique because it does not use class information.
- Histograms partition the values for an attribute, *A*, into disjoint ranges called *buckets.*
- In an *equal-width* histogram, for example, the values are partitioned into equal-sized partitions or ranges
- With an *equal frequency* histogram, the values are partitioned so that, ideally, 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 multilevel concept hierarchy, with the procedure terminating once a pre specified number of concept levels has been reached.
- A *minimum interval size* can also be used per level to control the recursive procedure..

- This specifies the minimum width of a partition, or the minimum number of values for each partition at each level.
- Histograms can also be partitioned based on cluster analysis of the data distribution.

Entropy-Based Discretization

- *Entropy* is one of the most commonly used discretization measures.
- It was first introduced by Claude Shannon in pioneering work on information theory and the concept of information gain.
- Entropy-based discretization is a supervised, top-down splitting technique.
- It explores class distribution information in its calculation and determination of split-points (data values for partitioning an attribute range).

- To discretize a numerical 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*.
- Let *D* consist of data tuples defined by a set of attributes and a class-label attribute.
- The class-label attribute provides the class information per tuple.
- The basic method for entropy-based discretization of an attribute *A* within the set is as follows:

1. Each value of *A* can be considered as a potential interval boundary or split-point (denoted *split point*) to partition the range of *A*.

- □ That is, a split-point for *A* can partition the tuples in *D* into two subsets satisfying the conditions *A* ≤ *split point* and *A* > *split point*, respectively, thereby creating a binary discretization

- Entropy-based discretization, as mentioned above, uses information regarding the class label of tuples.
- To explain the intuition behind entropy-based discretization, we must take a glimpse at classification.
- Suppose we want to classify the tuples in *D* by partitioning on attribute *A* and some split-point.
- Ideally, we would like this partitioning to result in an exact classification of the tuples.

- For example, if we had two classes, we would hope that all of the tuples of, say, class C1 will fall into one partition, and all of the tuples of class C2 will fall into the other partition.
- However, this is unlikely. For example, the first partition may contain many tuples of C1, but also some of C2.

$$Info_A(D) = \frac{|D_1|}{|D|}Entropy(D_1) + \frac{|D_2|}{|D|}Entropy(D_2),$$

- How much more information would we still need for a perfect classification, after this partitioning?
- This amount is called the *expected information requirement* for classifying a tuple in *D* based on partitioning by *A*. It is given by above formula.
- where *D*1 and *D*2 correspond to the tuples in *D* satisfying the conditions *A* $\leq$ *split point* and *A* $>$ *split point*, respectively; j*D*j is the number of tuples in *D*, and so on.
- The entropy function for a given set is calculated based on the class distribution of the tuples in the set.

$$Entropy(D_1) = -\sum_{i=1}^{m} p_i \log_2(p_i),$$

- □ For example, given *m* classes, C1,C2,…C*m*, the entropy of *D*1 is given in the above formula.
- □ where *pi* is the probability of class C*i* in *D*1, determined by dividing the number of tuples of class C*i* in *D*1 by j*D*1j, the total number of tuples in *D*1.
- □ Therefore, when selecting a split-point for attribute *A*, we want to pick the attribute value that gives the Minimum expected information requirement (i.e., min(*InfoA*(*D*))).

- This would result in the minimum amount of expected information (still) required to perfectly classify the tuples after partitioning by *A* ≤*split point* and *A*>*split point.*
- This is equivalent to the attribute-value pair with the maximum information gain

- 3. The process of determining a split-point is recursively applied to each partition obtained, until some stopping criterion is met, such as when the minimum information requirement on all candidate split-points is less than a small threshold, e, or when the number of intervals is greater than a threshold, *max interval*

Interval Merging by Chi-square Analysis

- *ChiMerge* is a **Chi-square** -based discretization method.
- which employs a bottom-up approach by finding the best neighboring intervals and then merging these to form larger intervals, recursively.
- The method is supervised in that it uses class information. The basic notion is that for accurate discretization, the relative class frequencies should be fairly consistent within an interval.
- Therefore, if two adjacent intervals have a very similar distribution of classes, then the intervals can be merged. Otherwise, they should remain separate.

- ChiMerge proceeds as follows.
- Initially, each distinct value of a numerical attribute *A* is considered to be one interval. c2 tests are performed for every pair of adjacent intervals.
- Adjacent intervals with the least c2 values are merged together, because low c2 values for a pair indicate similar class distributions.
- This merging process proceeds recursively until a predefined stopping criterion is met.

- The stopping criterion is typically determined by three conditions.
- First, merging stops when c2 values of all pairs of adjacent intervals exceed some threshold, which is determined by a specified significance level.
- A too (or very) high value of significance level for the c2 test may cause over discretization, whereas a too (or very) low value may lead to under discretization.
- Typically, the significance level is set between 0.10 and 0.01.

- Second, the number of intervals cannot be over a prespecified *max-interval*, such as 1.0 to 15.
- Finally, recall that the premise behind ChiMerge is that the relative class frequencies should be fairly consistent within an interval.

Cluster Analysis

- Cluster analysis is a popular data discretization method.
- A clustering algorithm can be applied to discretize a numerical 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 topdown splitting strategy or a bottom-up merging strategy, where each cluster forms a node of the concept hierarchy.
- □ In the former, each initial cluster or partition may be further decomposed into several sub clusters, forming a lower level of the hierarchy.
- □ In the latter, clusters are formed by repeatedly grouping neighboring clusters in order to form higher-level concepts

Discretization by Intuitive Partitioning

- □ The 3-4-5 rule can be used to segment numerical data into relatively uniform, natural seeming
- □ intervals. In general, the rule partitions a given range of data into 3, 4, or 5
- □ relatively equal-width intervals, recursively and level by level, based on the value range
- □ at the most significant digit.We will illustrate the use of the rule with an example further
- □ below. The rule is as follows:

- If an interval covers 3, 6, 7, or 9 distinct values at the most significant digit, then
- partition the range into 3 intervals (3 equal-width intervals for 3, 6, and 9; and 3
- intervals in the grouping of 2-3-2 for 7).

- If it covers 2, 4, or 8 distinct values at the most significant digit, then partition the
- range into 4 equal-width intervals.
- If it covers 1, 5, or 10 distinct values at the most significant digit, then partition the
- range into 5 equal-width intervals.

- Numeric concept hierarchy generation by intuitive partitioning.
- Suppose that profits at different branches of *AllElectronics* for the year 2004 cover a wide range, from -$351,976.00 to $4,700,896.50.
- A user desires the automatic generation of a concept hierarchy for *profit*.
- For improved readability, we use the notation (*l*...*r*] to represent the interval (*l*; *r*]. For example, (-$1,000,000...$0] denotes the range from -$1,000,000 (exclusive) to $0 (inclusive).

- Suppose that the data within the 5th percentile and 95th percentile are between -$159,876 and $1,838,761. The results of applying the 3-4-5 rule are shown in Figure

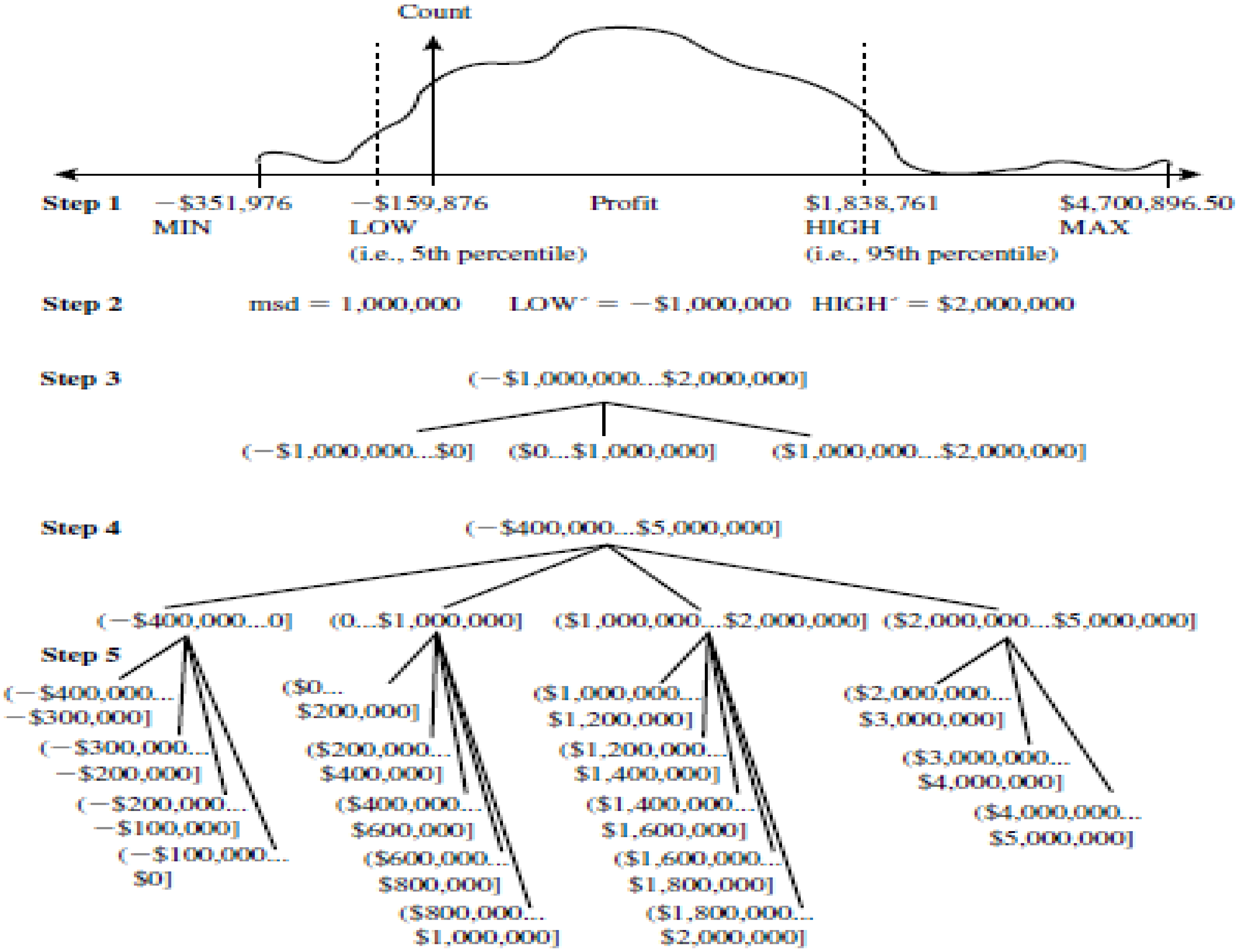
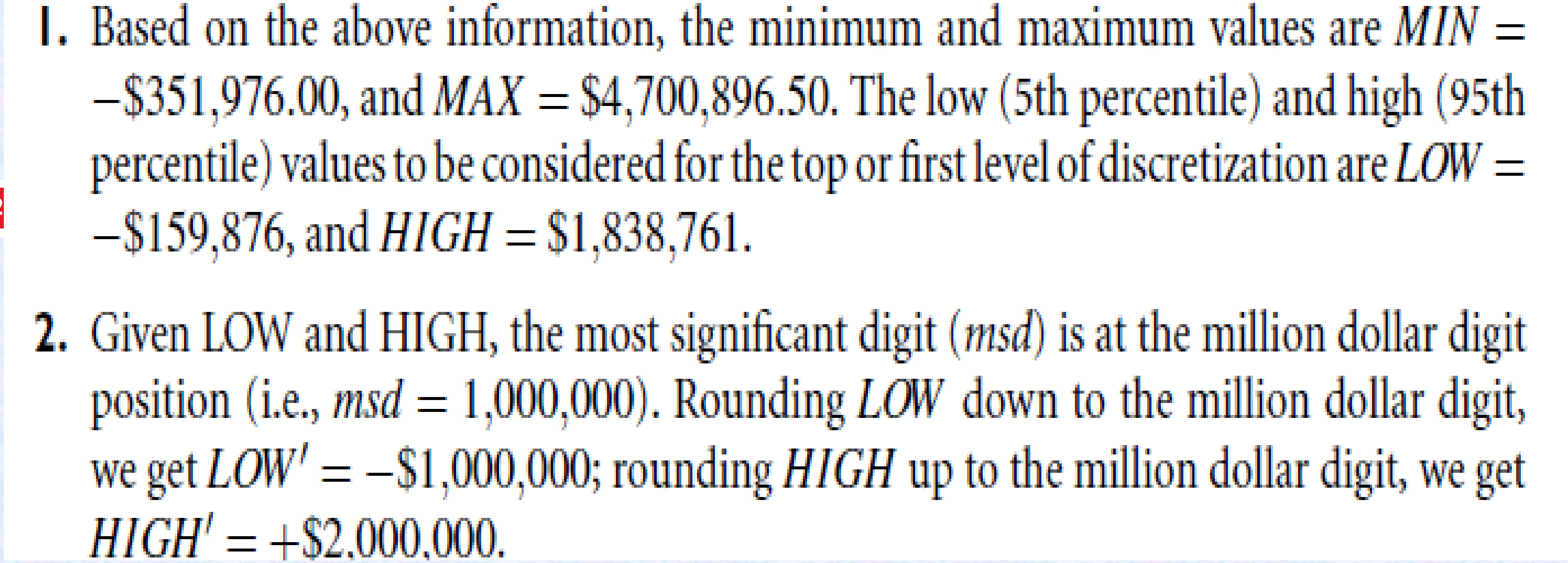
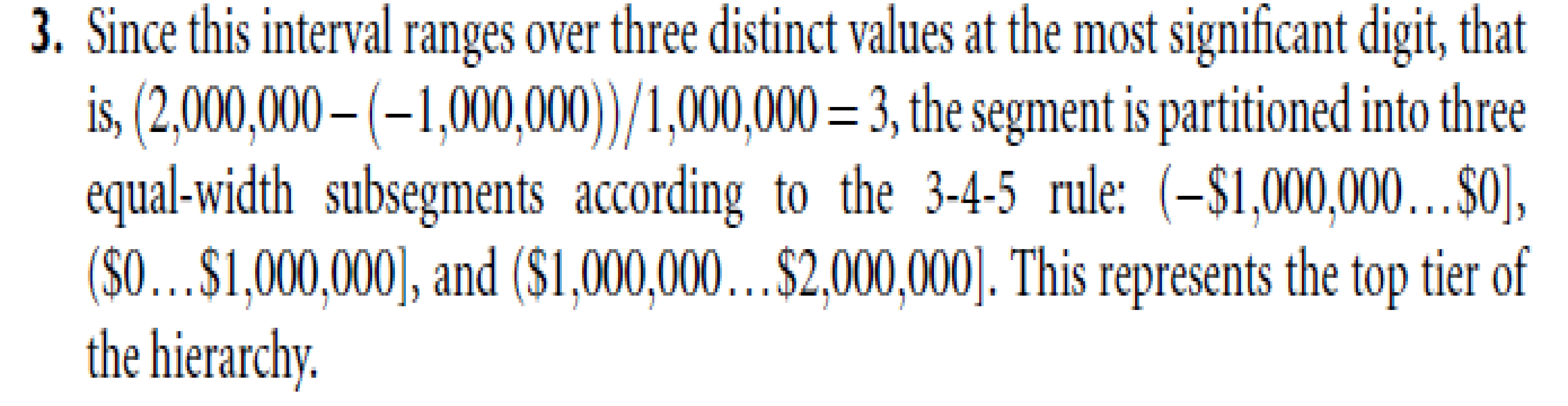


Figure 2.23 Automatic generation of a concept hierarchy for *profit* based on the 3-4-5 rule.

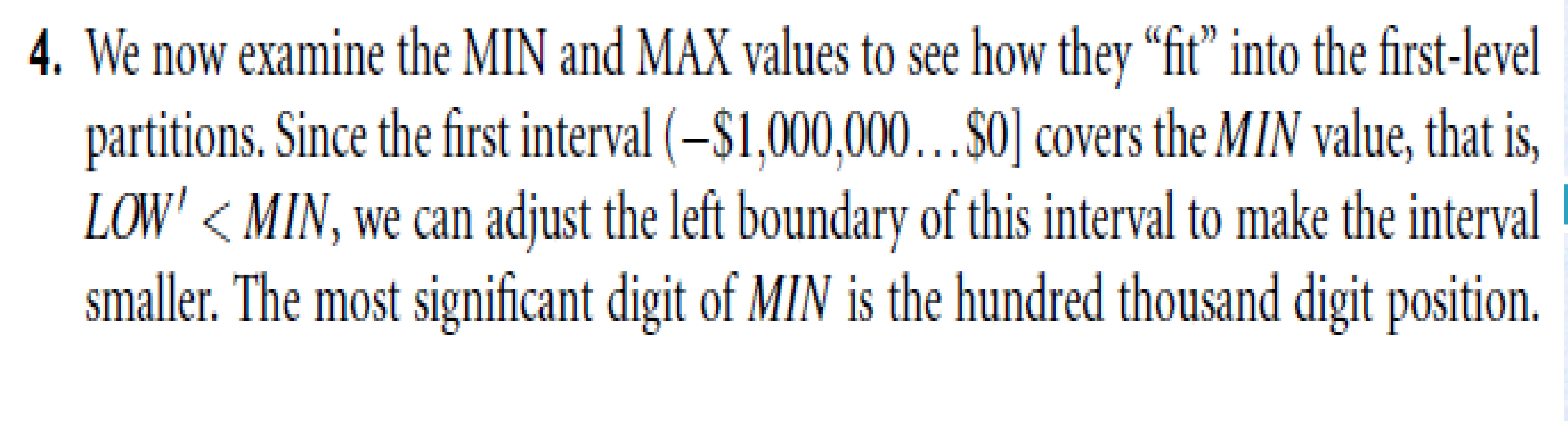


1. Based on the above information, the minimum and maximum values are $MIN = -\$351{,}976.00$, and $MAX = \$4{,}700{,}896.50$. The low (5th percentile) and high (95th percentile) values to be considered for the top or first level of discretization are $LOW = -\$159{,}876$, and $HIGH = \$1{,}838{,}761$.

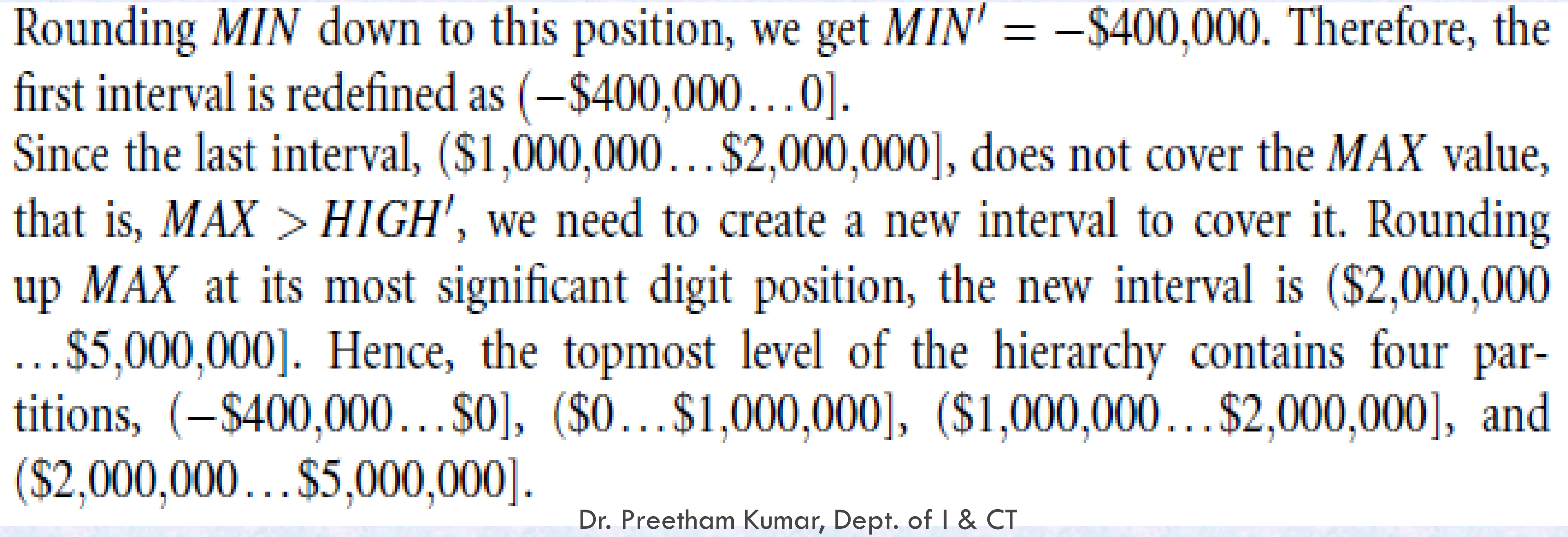
2. Given LOW and HIGH, the most significant digit (*msd*) is at the million dollar digit position (i.e., $msd = 1{,}000{,}000$). Rounding LOW down to the million dollar digit, we get $LOW' = -\$1{,}000{,}000$; rounding $HIGH$ up to the million dollar digit, we get $HIGH' = +\$2{,}000{,}000$.



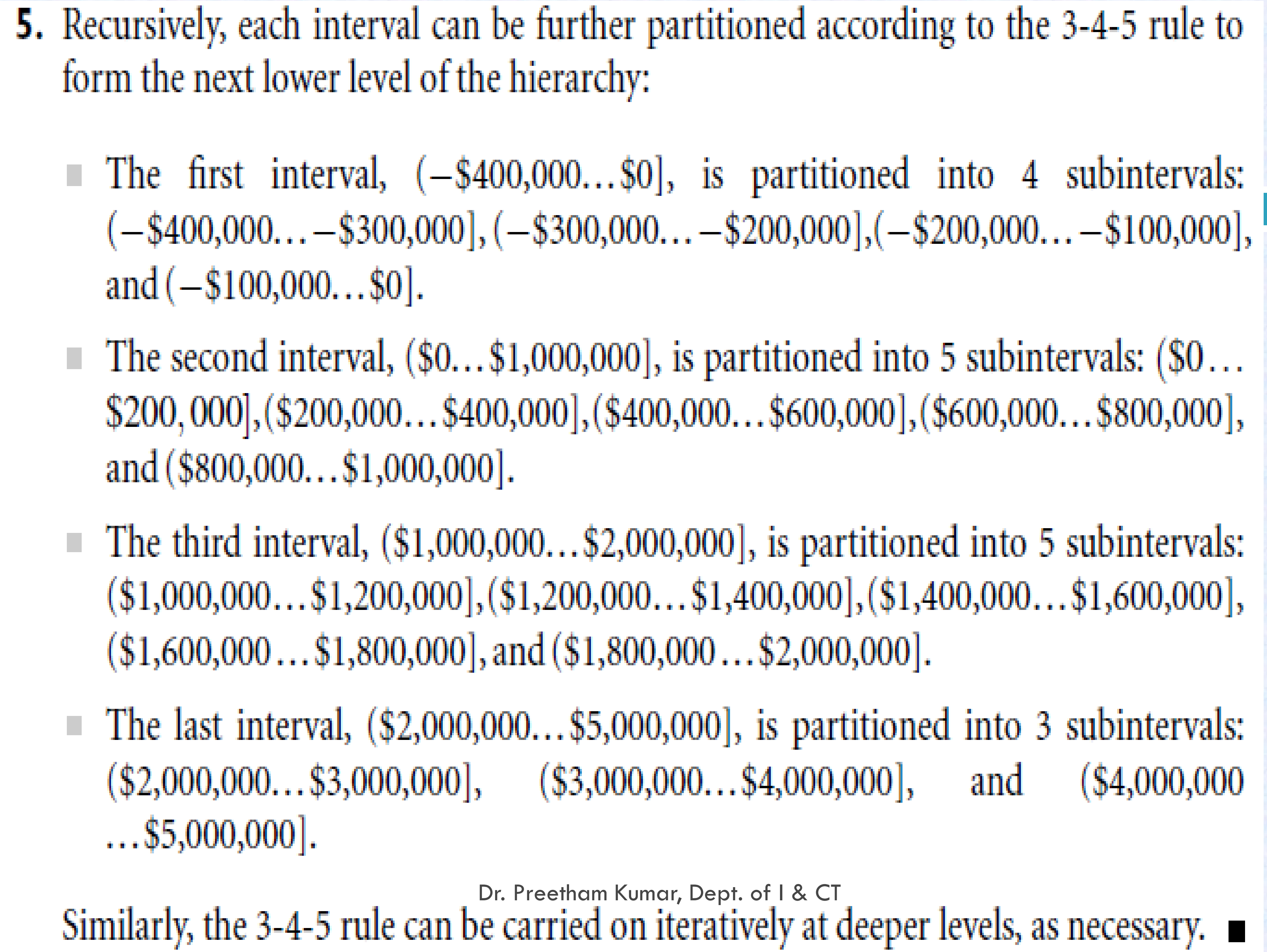
3. Since this interval ranges over three distinct values at the most significant digit, that is, $(2{,}000{,}000 - (-1{,}000{,}000))/1{,}000{,}000 = 3$, the segment is partitioned into three equal-width subsegments according to the 3-4-5 rule: $(-\$1{,}000{,}000 \ldots \$0]$, $(\$0 \ldots \$1{,}000{,}000]$, and $(\$1{,}000{,}000 \ldots \$2{,}000{,}000]$. This represents the top tier of the hierarchy.



4. We now examine the MIN and MAX values to see how they "fit" into the first-level partitions. Since the first interval (–\$1,000,000…\$0] covers the *MIN* value, that is, $LOW' < MIN$, we can adjust the left boundary of this interval to make the interval smaller. The most significant digit of *MIN* is the hundred thousand digit position.



Rounding *MIN* down to this position, we get $MIN' = -\$400{,}000$. Therefore, the first interval is redefined as (–\$400,000…0].
Since the last interval, (\$1,000,000…\$2,000,000], does not cover the *MAX* value, that is, $MAX > HIGH'$, we need to create a new interval to cover it. Rounding up *MAX* at its most significant digit position, the new interval is (\$2,000,000 …\$5,000,000]. Hence, the topmost level of the hierarchy contains four partitions, (–\$400,000…\$0], (\$0…\$1,000,000], (\$1,000,000…\$2,000,000], and (\$2,000,000…\$5,000,000].



5. Recursively, each interval can be further partitioned according to the 3-4-5 rule to form the next lower level of the hierarchy:

- The first interval, (−$400,000…$0], is partitioned into 4 subintervals: (−$400,000…−$300,000], (−$300,000…−$200,000],(−$200,000…−$100,000], and (−$100,000…$0].
- The second interval, ($0…$1,000,000], is partitioned into 5 subintervals: ($0…$200, 000],($200,000…$400,000],($400,000…$600,000],($600,000…$800,000], and ($800,000…$1,000,000].
- The third interval, ($1,000,000…$2,000,000], is partitioned into 5 subintervals: ($1,000,000…$1,200,000],($1,200,000…$1,400,000],($1,400,000…$1,600,000], ($1,600,000…$1,800,000], and ($1,800,000…$2,000,000].
- The last interval, ($2,000,000…$5,000,000], is partitioned into 3 subintervals: ($2,000,000…$3,000,000], ($3,000,000…$4,000,000], and ($4,000,000 …$5,000,000].

Similarly, the 3-4-5 rule can be carried on iteratively at deeper levels, as necessary. ■

Manipal
MANIPAL INSTITUTE OF TECHNOLOGY
Manipal Institute of Technology

All the Best