Chapter 2: Getting to Know Your Data

- Real-world data are typically noisy, enormous in volume (often several gigabytes or more), and may originate form a odgepodge of heterogenous sources
- Getting familiar with your data. Knowledge about your data is useful for data preprocessing the first major task of the data mining process.
- First we want to know the following things:
  1. What are the types of *attributes* or fields that make up your data?
  2. What kind of values does each attribute have?
  3. Which attributes are discrete, and which are continuous-valued?
  4. What do the data *look like*? How are the values distributed?
  5. Are there ways we can visualize the data to get a better sense of it all?
  6. Can we spot any outliers?
  7. Can we measure the similarity of some data objects with respect to others?

Data Objects and Attribute Types

- Data sets are made up of data objects.
- A **data object** represents an entity—
  1. In a sales database: The objects may be customers, store items, and sales
  2. In a medical database: The objects may be patients;
  3. In a university database: the objects may be students, professors, and courses.
- Data objects are typically described by attributes.
- Data objects can also be referred to as *samples, examples, instances, data points*, or *objects*.
- If the data objects are stored in a database, they are *data tuples*. i.e. the rows of a database correspond to the data objects, and the columns correspond to the attributes

What Is an Attribute?

- An **attribute** is a data field, representing a characteristic or feature of a data object.
- The nouns *attribute*, *dimension*, *feature*, and *variable* are often used interchangeably in the literature.
- The term *dimension* is commonly used in data warehousing.
- Machine learning literature tends to use the term *feature*
- Statisticians prefer the term *variable*.
- Data mining and database professionals commonly use the term *attribute*, and we do here as well.
- Attributes describing a customer object can include, for example, *customer ID*, *name*, and *address*.
- Observed values for a given attribute are known as *observations*.
- A set of attributes used to describe a given object is called an *attribute vector* (or *feature vector*).
- The distribution of data involving one attribute (or variable) is called *univariate*.
- A *bivariate* distribution involves two attributes, and so on.
- The **type** of an attribute is determined by the set of possible values—nominal, binary, ordinal, or numeric—the attribute can have

Nominal Attributes (distinctness)

- Nominal means: "relating to names."
- The values of a **nominal attribute** are symbols or *names of things*.
- Each value represents some kind of category, code, or state, and so
- Nominal attributes are also referred to as **categorical**.
- The values do not have any meaningful order.
- In computer science, the values are also known as *enumerations*.

Example: *Nominal Attributes*

1. *Hair color : Black*, *brown*, *blond*, *red*, *auburn*, *gray*, and *white*
2. *Marital status : Single, married, divorced*, and *widowed*
3. *Occupation: Teacher, dentist, programmer, farmer* etc.

Note:

- Although we said that the values of a nominal attribute are symbols or "names of things,"
- It is possible to represent such symbols or "names" with numbers.
    1. *Hair_color : Black* (0) *Brown(1)* , and so on.
    2. *Customor_ID*: Possible values that are all numeric.
- Mathematical operations (+, - ,*, / etc.) on values of nominal attributes are not meaningful. It makes no sense to subtract one customer ID number from another, unlike, say, subtracting an age value from another (where *age* is a numeric attribute).
- Even though a nominal attribute may have integers as values, it is not considered a numeric attribute because the integers are not meant to be used quantitatively(i. e. they are considerd as qualitative).

- Nominal attribute values do not have any meaningful order about them and are not quantitative, it makes no sense to find the mean (average value) or median(middle value) and mode (most commonly occurring value)

Binary Attributes

- A **binary attribute** is a nominal attribute with only two categories or states: 0 or 1, where 0 and 1 represents as 0 : Absent 1: present.
- Binary attributes are referred to as **Boolean** if the two states correspond to *true* and *false*.

  Example:

  Object: Patient
  Attribute: Smoker
  Values: 1 (Smokes) 0 (Does not smokes)

  Object: Patient
  Attribute: Medical_Test
  Result: 1: Positive 0: Negative

- A binary attribute is **symmetric** if both of its states are equally valuable and carry the same weight; that is, there is no preference on which outcome should be coded as 0 or 1.

  Example:

  Attribute: *Gender*
  Values: Male and *Female*

- A binary attribute is **asymmetric** if the outcomes of the states are not equally important, such as the *positive* and *negative* outcomes of a medical test for HIV

  Example: 1: HIV +VE 0: HIV –VE (In rare case we use)

Ordinal Attributes

- An **ordinal attribute** is an attribute with possible values that have a meaningful order or *ranking* among them, but the magnitude between successive values is not known.

  Example 1: Attribute: Drink_Size
  Values: Small, Medium, Large
- The values have a meaningful sequence (which corresponds to increasing drink size) however, we cannot tell from the values *how much* bigger, say, a medium is than a large

  Example 2: Attribute: Grade
  Values: *A+, A, A-, B+, B- etc.*

  Example 3: Attribute: *Professional rank*
  Values: *Assistant*, *Associate*, and Professors

  Example 4: Attribute: Army ranks.
  Values: P*rivate, Private first class, Specialist, Corporal, and Sergeant*

- Ordinal attributes are useful for registering subjective assessments of qualities that cannot be measured objectively

  Uses of ordinal attributes: Surveys for ratings

  Example: Customer satisfaction
  0: very dissatisfied, 1: somewhat dissatisfied, 2: neutral, 3: satisfied, and *4: very satisfied.*

- The central tendency of an ordinal attribute can be represented by its mode and its median (the middle value in an ordered sequence), but the mean cannot be defined.

  Note: Nominal, Binary, and Ordinal attributes are *qualitative (i.e.* they *describe* a feature of an object without giving an actual size or quantity)

Numeric Attributes

- A **numeric attribute** is *quantitative* i.e. it is a measurable quantity, represented in integer or real values. Types of numeric attributes *interval-scaled* or *ratio-scaled*.
-
  1. Interval-Scaled Attributes
  2. Ratio-Scaled Attributes

1. Interval-Scaled Attributes

- **Interval-scaled attributes** are measured on a scale of equal-size units.
- The values of interval-scaled attributes have order and can be positive, 0, or negative.
- These attributes to provide a ranking of values, such attributes allow us to compare and quantify the *difference* between values.

Example:

1. Attribute: Temperature
   Values: 15^0 C/F 20^0 C/F 25^0 C/F (temp. values on different days)
2. Calendar dates For instance, the years 2002 and 2010 are eight years apart.

- Interval-scaled attributes are numeric, so we can compute their mean. median and mode measures of central tendency
  Note: Interval-scaled attributes cannot speak of the values in terms of ratios (i.e. true zero point does not exist)

2. **Ratio-Scaled Attributes**

- A **ratio-scaled attribute** is a numeric attribute with an inherent zero-point i.e. if a measurement is ratio-scaled, we can speak of a value as being a multiple (or ratio) of another value
- The values are ordered, and we can also compute the difference between values, as well as the mean, median, and mode.

Example:

1. Kelvin (K) temperature scale has what is considered a true zero-point (0^0 K=-273.15^0C)
2. *Years of experience* (e.g. employees)
3. *Number of words in a documents*

Discrete versus Continuous Attributes

- D**iscrete attribute: It** has a finite or countably infinite set of values, which may or may not be represented as integers. The attributes *hair color*, *smoker*, *medical test*, and *drink size* each have a finite number of values, and so are discrete

  Example for discrete attributes: Numeric values: 0,1 , Customer_ID, Age : 0 -110 ZIP codes (finite but countably infinite discrete values)

- **Continuous attribute:** If an attribute is not discrete, it is **continuous**. The terms *numeric attribute* and *continuous attribute* are often used interchangeably

  Note: Continuous values are real numbers, whereas numeric values can be either integers or real numbers.) In practice, real values are represented using a finite number of digits. Continuous attributes are typically represented as floating-point variables.

Data Visualization (DV)

Question: Why we require data visualization?
Answer: To convey data to users effectively

Definition: Data visualization aims to communicate data clearly and effectively through graphical representation

Applications of DV:

- Work reporting
- Managing business operations
- Tracking progress of tasks
- Discover the data relationships between the data that are not easily observable by looking at the raw data
- To create fun and interesting graphics etc

Data visualization techniques:

1. Pixel-oriented techniques
2. Geometric projection techniques
3. Icon-based techniques
4. Hierarchical and graph-based techniques

1. Pixel-oriented techniques :

- A simple way to visualize the value of a dimension is to use a pixel where the color of the pixel reflects the dimension's value.
- For a data set of *m* dimensions, **pixel-oriented techniques** create *m* windows on the screen, one for each dimension.
- The *m* dimension values of a record are mapped to *m* pixels at the corresponding positions in the windows.
- The colors of the pixels reflect the corresponding values.
- Inside a window, the data values are arranged in some global order shared by all Windows.
- The global order may be obtained by sorting all data records in a way that's meaningful for the task at hand.

Example:

- A customer information table, which consists of four dimensions: *income*, *credit limit*, *transaction volume*, and *age*.
- We analyze the correlation between *income* and the other attributes by visualization
- Sort all customers in income-ascending order, and use this order to lay out the customer data in the four visualization windows (*Shown in below figure 1*)
- The pixel colors are chosen so that the smaller the value, the lighter the shading
- Using pixel based visualization, we can easily observe the following: *credit limit* increases as *income* increases;
- Customers whose income is in the middle range are more likely to purchase the products
- There is no clear correlation between *income* and *age*.

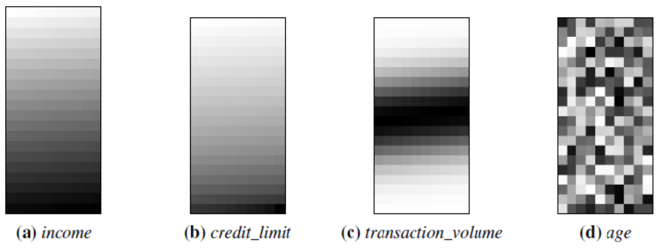


Figure I' Pixel-oriented visualization of four attributes by sorting all customers in *income* ascending order.

- In pixel-oriented techniques, data records can also be ordered in a query-dependent way.

  Example: Given a point query, we can sort all records in descending order of similarity to the point query.

- Filling a window by laying out the data records in a linear way may not work well for a wide window.
- The first pixel in a row is far away from the last pixel in the previous row, though they are next to each other in the global order.
- A pixel is next to the one above it in the window, even though the two are not next to each other in the global order.
- To solve this problem, we can lay out the data records in a space-filling curve to fill the windows.

A space-filling curve:

It is a curve with a range that covers the entire *n*-dimensional unit hypercube. Since the visualization windows are 2-D, we can use any 2-D space-filling curve

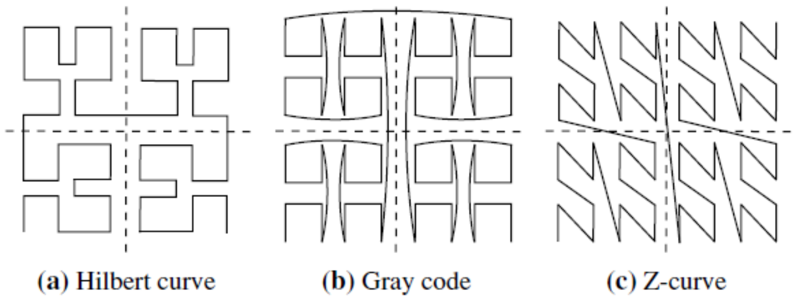


Figure II Some frequently used 2-D space-filling curves.

Note that the windows do not have to be rectangular.

Example:

- The *circle segment technique* uses windows in the shape of segments of a circle shown in Figure 2.1II.
- This technique can ease the comparison of dimensions because the dimension windows are located side by side and form a circle.

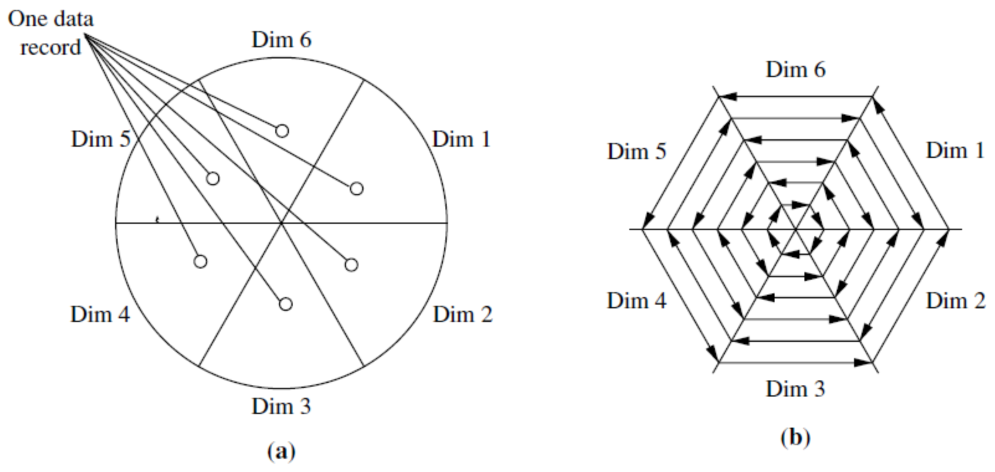


Figure III The circle segment technique. (a) Representing a data record in circle segments. (b) Laying out pixels in circle segments.

Disadvantage of Pixel-oriented techniques:

A drawback of pixel-oriented visualization techniques is that they cannot help us much in understanding the distribution of data in a multidimensional space.

Example: They do not show whether there is a dense area in a multidimensional subspace

2. Geometric Projection Visualization Techniques

Geometric Projection Visualization Techniques: They help users find interesting projections of multidimensional data sets. The central challenge the geometric projection techniques try to address is how to visualize a high-dimensional space on a 2-D display.

Types of Geometric **Projection Visualization Techniques:**

I. Scatter plot
II. Scatter-plot matrix technique:
III. Parallel coordinates technique:

I. Scatter plot: It displays 2-D data points using Cartesian coordinates. A third dimension can be added using different colors or shapes to represent different data points.

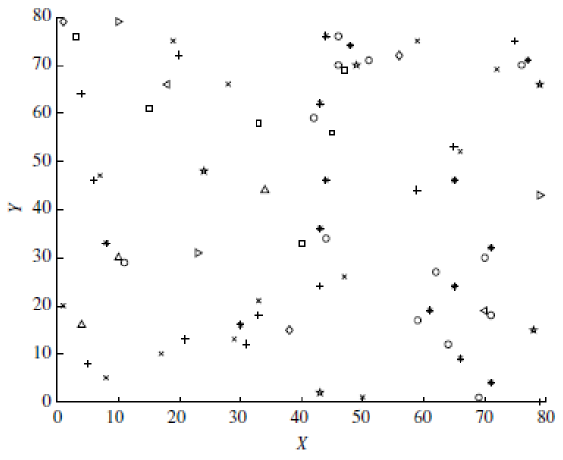


Figure 2.13 Visualization of a 2-D data set using a scatter plot.

The above Figure 2.13 is an example 2-D scatter plot , where *X* and *Y* are two spatial attributes and the third dimension is represented by different shapes. Through this visualization, we can see that points of types "+" and "X" tend to be colocated.

A 3-D scatter plot uses three axes in a Cartesian coordinate system. If it also uses color, it can display up to 4-D data points (Figure 2.14).

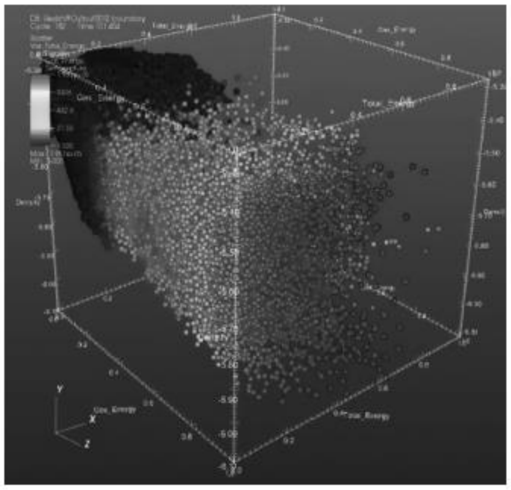


Figure 2.14 Visualization of a 3-D data set using a scatter plot.

II. **Scatter-plot matrix technique:** It is a useful extension to the scatter plot. For an *n* dimensional data set, a scatter-plot matrix is an *nxn* grid of 2-D scatter plots that provides a visualization of each dimension with every other dimension

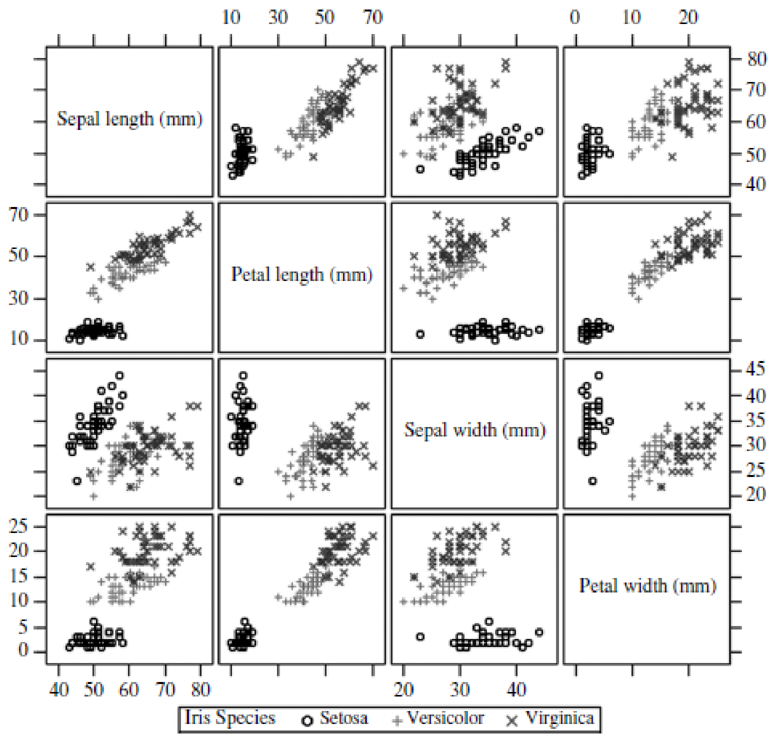


Figure 2.15 Visualization of the Iris data set using a scatter-plot matrix.

Figure 2.15: It shows an example, which visualizes the Iris data set. The data set consists of 450 samples from each of three species of Iris flowers. There are five dimensions in the data set: length and width of sepal and petal, and species.

Note: The scatter-plot matrix becomes less effective as the dimensionality increases

III. Parallel coordinates technique: It draws *n* equally spaced axes, one for each dimension, parallel to one of the display axes.

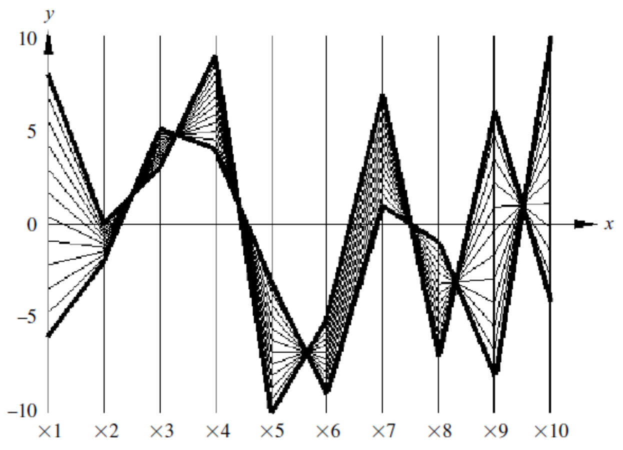


Figure 2.16 Here is a visualization that uses parallel coordinates.

In the above figure 2.16 a data record is represented by **a polygonal line** that intersects each axis at the point corresponding to the associated dimension value

Note: A major limitation of the parallel coordinates technique is that it cannot effectively show a data set of many records. Even for a data set of several thousand records, visual clutter and overlap often reduce the readability of the visualization and make the patterns hard to find.

3. Icon-Based Visualization Techniques

Icon-based visualization techniques use small icons to represent multidimensional data values.
Two popular icon-based techniques are:

1. *Chernoff faces and Asymmetric Chernoff faces*
2. *Stick figures*.

1. **Chernoff faces :** They were introduced in 1973 by statistician Herman Chernoff. They display multidimensional data of up to 18 variables (or dimensions) as a cartoon human face

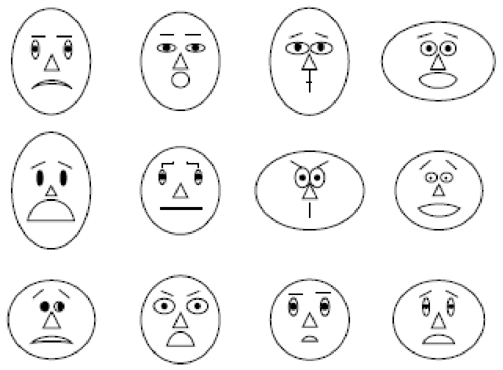


Figure 2.17 Chernoff faces. Each face represents an n-dimensional data point ($n \leq 18$).

- Chernoff faces help reveal trends in the data.
- Components of the face, such as the eyes, ears, mouth, and nose, represent values of the dimensions by their shape, size, placement, and orientation.

  Example:
  Dimensions can be mapped to the following *facial characteristics*:
  Eye size, eye spacing, nose length, nose width, mouth curvature, mouth width, mouth openness, pupil size, eyebrow slant, eye eccentricity, and head eccentricity.

- Chernoff faces make use of the ability of the human mind to recognize small differences in facial characteristics and to assimilate many facial characteristics at once
- Viewing large tables of data can be tedious
- By condensing the data, Chernoff faces make the data easier for users to digest
- They facilitate visualization of regularities and irregularities present in the data

Limitation/Disadvantages of Chernoff faces :

- Relating multiple relationships is limited
- Specific data values are not shown
- Facial features vary in perceived importance (i.e. similarity of two faces (representing two multidimensional data points) can vary depending on the order in which dimensions are assigned to facial characteristics

***Asymmetrical Chernoff* faces** : These are the extension to the original technique(chernoff faces). Since a face has vertical symmetry (along the *y*-axis), the left and right side of a face are identical, which wastes space. Asymmetrical Chernoff faces double the number of facial characteristics, thus allowing up to 36 dimensions to be displayed.

2. **Stick figure:** maps multidimensional data to five-piece stick figures, where each figure has four limbs and a body. Two dimensions are mapped to the display (*x* and *y*) axes and the remaining dimensions are mapped to the angle and/or length of the limbs.

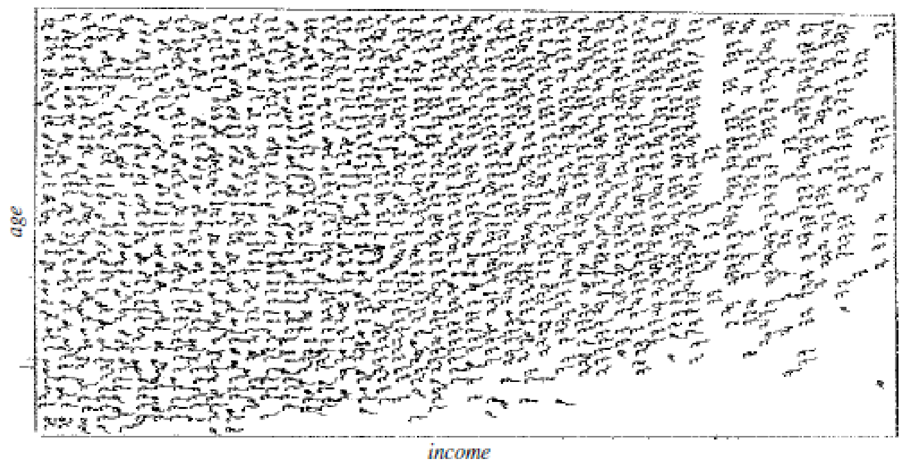


Figure 2.18 Census data represented using stick figures.

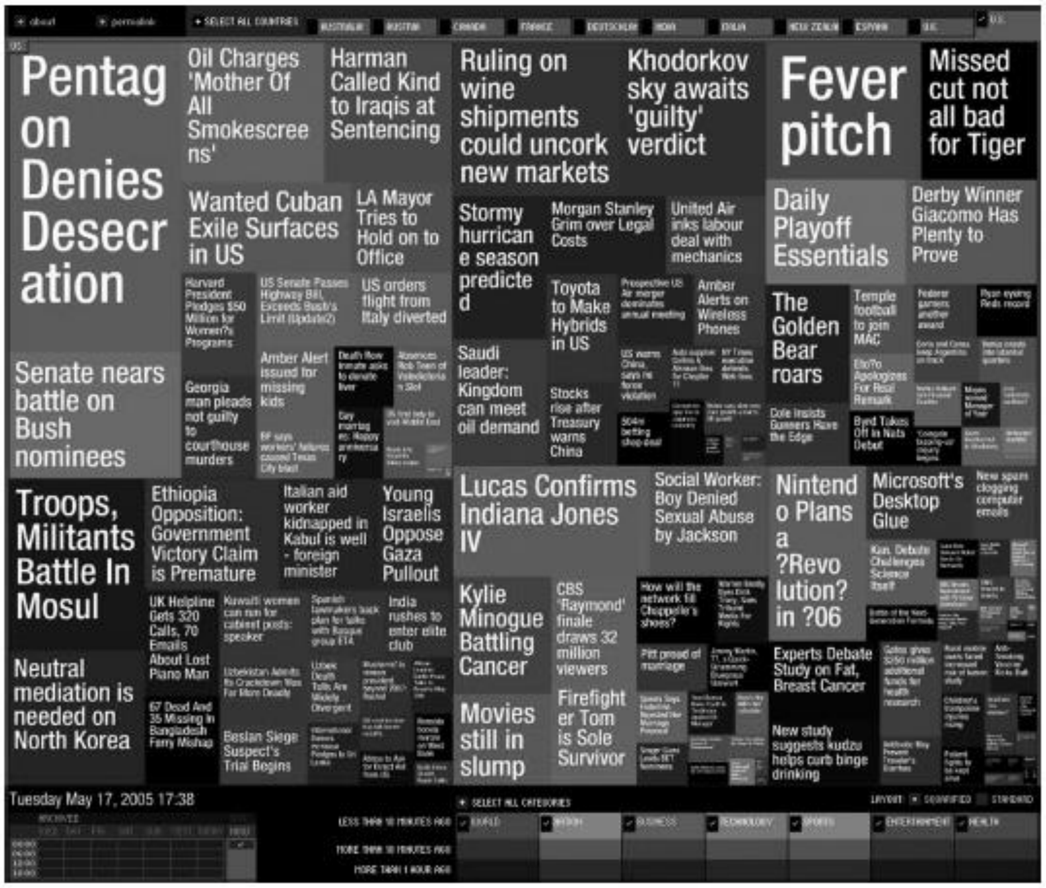
The above Figure 2.18 shows census data, where *age* and *income* are mapped to the display axes, and the remaining dimensions (*gender, education*, and so on) are mapped to stick figures. If the data items are relatively dense with respect to the two display dimensions, the resulting visualization shows texture patterns, reflecting data trends.

Note: The visualization techniques (Pixel, Geometric projection and Icon Based) discussed so far focus on *visualizing multiple dimensions simultaneously*. However, *for a large data set of high dimensionality, it would be difficult to visualize all dimensions at the same time*

3. Hierarchical Visualization Techniques

Hierarchical visualization techniques partition all dimensions into subsets (i.e., subspaces). The subspaces are visualized in a hierarchical manner.

Example: Tree-maps display hierarchical data as a set of nested rectangles



Pentagon Denies Desecration
Oil Charges 'Mother Of All Smokescreens'
Harman Called Kind to Iraqis at Sentencing
Ruling on wine shipments could uncork new markets
Khodorkovsky awaits 'guilty' verdict
Fever pitch
Missed cut not all bad for Tiger
Wanted Cuban Exile Surfaces in US
LA Mayor Tries to Hold on to Office
Stormy hurricane season predicted
Morgan Stanley Grim over Legal Costs
United Air inks labour deal with mechanics
Daily Playoff Essentials
Derby Winner Giacomo Has Plenty to Prove
US orders flight from Italy diverted
Toyota to Make Hybrids in US
The Golden Bear roars
Senate nears battle on Bush nominees
Saudi leader: Kingdom can meet oil demand
Stocks rise after Treasury warns China
Troops, Militants Battle In Mosul
Ethiopia Opposition: Government Victory Claim is Premature
Italian aid worker kidnapped in Kabul is well - foreign minister
Young Israelis Oppose Gaza Pullout
Lucas Confirms Indiana Jones IV
Social Worker: Boy Denied Sexual Abuse by Jackson
Nintendo Plans a ?Revolution? in ?06
Microsoft's Desktop Glue
Kylie Minogue Battling Cancer
CBS 'Raymond' finale draws 32 million viewers
Neutral mediation is needed on North Korea
Movies still in slump
Firefighter Tom is Sole Survivor
Experts Debate Study on Fat, Breast Cancer
New study suggests kudzu helps curb binge drinking
Tuesday May 17, 2005 17:38

Figure 2.20 Newsmap: Use of tree-maps to visualize Google news headline stories.

In above Figure 2.20 it shows a tree-map visualizing Google news stories. All news stories are organized into *seven categories*, each shown in a large rectangle of a unique color. Within each category (i.e., each rectangle at the top level), the news stories are further partitioned into smaller subcategories.

4. Visualizing Complex Data and Relations

- In early days, visualization techniques were mainly for *numeric data*
- Recently, more and more non-numeric data, such as text and social networks, have become available
- Visualizing and analyzing such data attracts a lot of interest
- There are many new visualization techniques dedicated to these kinds of data for example, many people on the Web tag various objects such as pictures, blog entries, and product reviews.

- A **tag cloud** is a visualization of statistics of user-generated tags. Often, in a tag cloud, tags are listed alphabetically or in a user-preferred order. The importance of a tag is indicated by font size or color

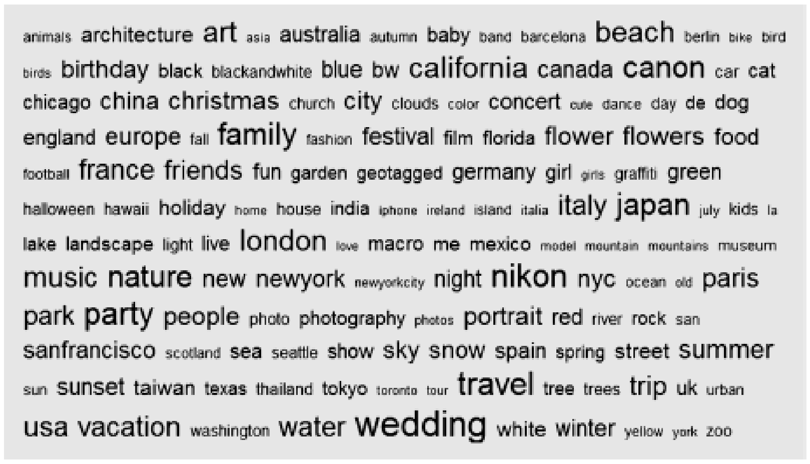

animals architecture art asia australia autumn baby band barcelona beach berlin bike bird
birds birthday black blackandwhite blue bw california canada canon car cat
chicago china christmas church city clouds color concert cute dance day de dog
england europe fall family fashion festival film florida flower flowers food
football france friends fun garden geotagged germany girl girls graffiti green
halloween hawaii holiday home house india iphone ireland island italia italy japan july kids la
lake landscape light live london love macro me mexico model mountain mountains museum
music nature new newyork newyorkcity night nikon nyc ocean old paris
park party people photo photography photos portrait red river rock san
sanfrancisco scotland sea seattle show sky snow spain spring street summer
sun sunset taiwan texas thailand tokyo toronto tour travel tree trees trip uk urban
usa vacation washington water wedding white winter yellow york zoo

Figure 2.21 Using a tag cloud to visualize popular Web site tags.

The above figure 2.21 shows a tag cloud for visualizing the popular tags used in a Web site.
Tag clouds are often used in two ways:

1. A tag cloud for a single item, we can use the size of a tag to represent the number of times that the tag is applied to this item by different users
2. A tag statistics on multiple items, we can use the size of a tag to represent the number of items that the tag has been applied to, that is, the popularity of the tag.

In addition to complex data, complex relations among data entries also raise challenges for visualization

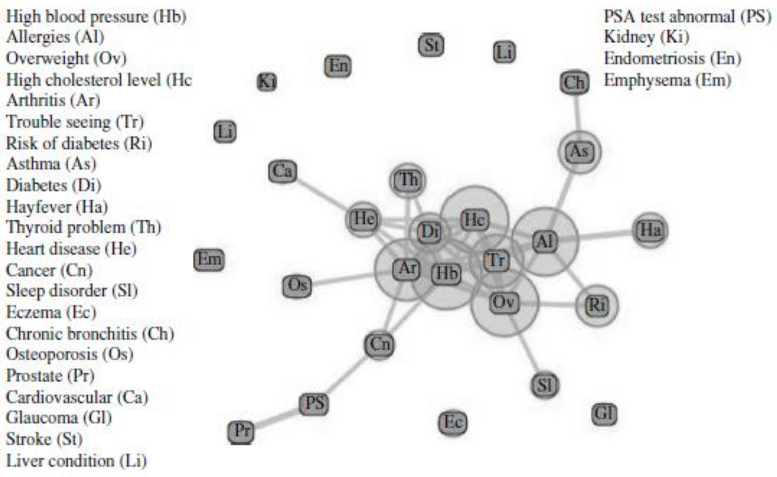


Figure 2.22 Disease influence graph of people at least 20 years old in the NHANES data set.

The above Figure 2.22 uses a disease influence graph to visualize the correlations between diseases. The nodes in the graph are diseases, and the size of each node is proportional to the prevalence of the corresponding disease. Two nodes are linked by an edge if the corresponding diseases have a strong correlation. The width of an edge is proportional to the strength of the correlation pattern of the two corresponding diseases.

Measuring Data Similarity and Dissimilarity

Data mining applications are:

- Clustering,
- Outlier analysis, and nearest-neighbor classification,

We need ways to assess how alike or unalike objects are in comparison to one another.
For example: A store may want to search for clusters of *customer* objects, resulting in groups of customers with similar characteristics (e.g., similar income, area of residence, and age). Such information can then be used for marketing

A **cluster** is a collection of data objects such that the objects within a cluster are *similar* to one another and *dissimilar* to the objects in other clusters

Outlier analysis also employs clustering-based techniques to identify potential outliers as objects that are highly dissimilar to others

Knowledge of **object similarities** can also be used in ***nearest-neighbor classification*** schemes where a given object (e.g., a *patient*) is assigned a class label (relating to, say, a *diagnosis*) based on its similarity toward other objects in the model.

Similarity and **dissimilarity** measures, which are referred to as **measures of *proximity*.** Similarity and dissimilarity are related

A **similarity measure** for two objects, *i* and *j*, will typically return the value 0 if the objects are unalike. The higher the similarity value, the greater the similarity between objects. (Typically, a value of 1 indicates complete similarity, that is, the objects are identical.)

A **dissimilarity measure** works the opposite way. It returns a value of 0 if the objects are the same and therefore, far from being dissimilar). The higher the dissimilarity value, the more dissimilar the two objects are.

Two **data structures** that are commonly used in the Similarity and **dissimilarity** measures are:

- ***Data matrix*** (used to store the data objects)
- ***Dissimilarity matrix*** (used to store dissimilarity values for pairs of objects).
-

Object dissimilarity can be computed for objects described by *nominal* attributes, by *binary* attributes by *numeric* attributes , by *ordinal* attributes , or by combinations of these attribute types

Data Matrix versus Dissimilarity Matrix

Data matrix (or *object-by-attribute structure*): This structure stores the *n* data objects in the form of a relational table, or *n*-by-*p* matrix (*n* objects x *p* attributes):

Each row corresponds to an object. As part of our notation, we may use *f* to index through the *p* attributes.

$$\begin{bmatrix} x_{11} & \cdots & x_{1f} & \cdots & x_{1p} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ x_{i1} & \cdots & x_{if} & \cdots & x_{ip} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ x_{n1} & \cdots & x_{nf} & \cdots & x_{np} \end{bmatrix}$$

Dissimilarity matrix (or *object-by-object structure*): This structure stores a collection of proximities that are available for all pairs of *n* objects. It is often represented by an *n*-by-*n* table:

$$\begin{bmatrix} 0 & & & & \\ d(2,1) & 0 & & & \\ d(3,1) & d(3,2) & 0 & & \\ \vdots & \vdots & \vdots & & \\ d(n,1) & d(n,2) & \cdots & \cdots & 0 \end{bmatrix}$$

where *d*.(*i*, *j*)is the measured **dissimilarity** or "difference" between objects *i* and *j*. In general, *d*.(*i*, *j*)is a non-negative number that is close to 0 when objects *i* and *j* are highly similar or "near" each other, and becomes larger the more they differ.

Note: that *d*.(*i*, *i*) D= 0; i.e, the difference between an object and itself is 0.
Furthermore, *d*.(*i*, *j*) = *d*.(*j*,i *)*

Measures of similarity can often be expressed as a function of measures of dissimilarity
For example, for nominal data,

$$sim(i,j) = 1 - d(i,j)$$

where *sim*.(*i*, *j*) is the similarity between objects *i* and *j*.

A data matrix is made up of **two entities or "things**," namely rows (for objects) and columns (for attributes). Therefore, the data matrix is often called a **two-mode** matrix. The dissimilarity matrix contains one kind of entity (dissimilarities) and so is called a **one-mode** matrix. Many clustering and nearest-neighbor algorithms operate on a dissimilarity matrix. Data in the form of a data matrix can be transformed into a dissimilarity matrix before applying such algorithms

Proximity Measures for Nominal Attributes

A nominal attribute can take on two or more states

For example: *map color* is a nominal attribute that may have, say, five states: *red, yellow, green, pink*, and *blue*. Let the number of states of a nominal attribute be *M*. The states can be denoted by letters, symbols, or a set of integers, such as 1, 2, : : : , *M*. Notice that such integers areused just for data handling and do not represent any specific ordering.

The dissimilarity between two objects *i* and *j* can be computed based on the **ratio of mismatches**: where ***m*** is the **number of *matches*** (i.e., the number of attributes for which *i* and *j* are in the same state), and ***p*** is the **total number of attributes** describing the objects. Weights can be assigned to increase the effect of *m* or to assign greater weight to the matches in attributes having a larger number of states

$$d(i,j) = \frac{p - m}{p},$$

Dissimilarity between nominal attributes. Suppose that we have the sample data of Table 2.2, except that only the *object-identifier* and the attribute *test-1* are available, where *test-1* is nominal. Let's compute the dissimilarity matrix that is,

$$\begin{bmatrix} 0 & & & \\ d(2,1) & 0 & & \\ d(3,1) & d(3,2) & 0 & \\ d(4,1) & d(4,2) & d(4,3) & 0 \end{bmatrix}.$$

Since here we have one nominal attribute, *test-1*, we set $p = 1$ so that $d.(i, j)$ evaluates to 0 if objects *i* and *j* match, and 1 if the objects differ. Thus, we get

$$\begin{bmatrix} 0 & & & \\ 1 & 0 & & \\ 1 & 1 & 0 & \\ 0 & 1 & 1 & 0 \end{bmatrix}.$$

From this, we see that all objects are dissimilar except objects 1 and 4 (i.e., $d.(4,1) = 0$)

Table 2.2 A Sample Data Table Containing Attributes of Mixed Type

Object Identifier	***test-1 (nominal)***	***test-2 (ordinal)***	***test-3 (numeric)***
1	code A	excellent	45
2	code B	fair	22
3	code C	good	64
4	code A	excellent	28

Alternatively, similarity can be computed as

$$sim(i, j) = 1 - d(i, j) = \frac{m}{p}.$$

Proximity between objects described by nominal attributes can be computed using an alternative encoding scheme. Nominal attributes can be encoded using asymmetric binary attributes by creating a new binary attribute for each of the *M* states. For an object with a given state value, the binary attribute representing that state is set to 1, while the remaining binary attributes are set to 0.

For example: To encode the nominal attribute *map color*, a binary attribute can be created for each of the five colors previously listed. For an object having the color *yellow*, the *yellow* attribute is set to 1, while the remaining four attributes are set to 0.