Data Types and Representation

- Relational databases and relational data representation
- Types of data objects and their attributes
- Data dimensionality
- Typical search and analysis tasks
- Measures of data quality
- Data preprocessing
- Similarity between data objects

Relational Data Model

Basic Structure of Relational Databases

- Formally:
  - given sets D_1, D_2, …. D_n (attribute domains)
  - a **relation** r is a subset of D_1 x D_2 x … x D_n
- relation schema: attributes and their domains
- relation instance: tuples (records)

attributes
(or columns)

customer-name	*customer-street*	customer-city
Jones	Main	Harrison
Smith	North	Rye
Curry	North	Rye
Lindsay	Park	Pittsfield

tuples
(or rows)

customer

Relational Attribute Types

- Each attribute of a relation has a name
- The set of allowed values for each attribute is called the **domain** of the attribute
- Examples of simple domain types:
  - integer
  - string
  - date

Keys

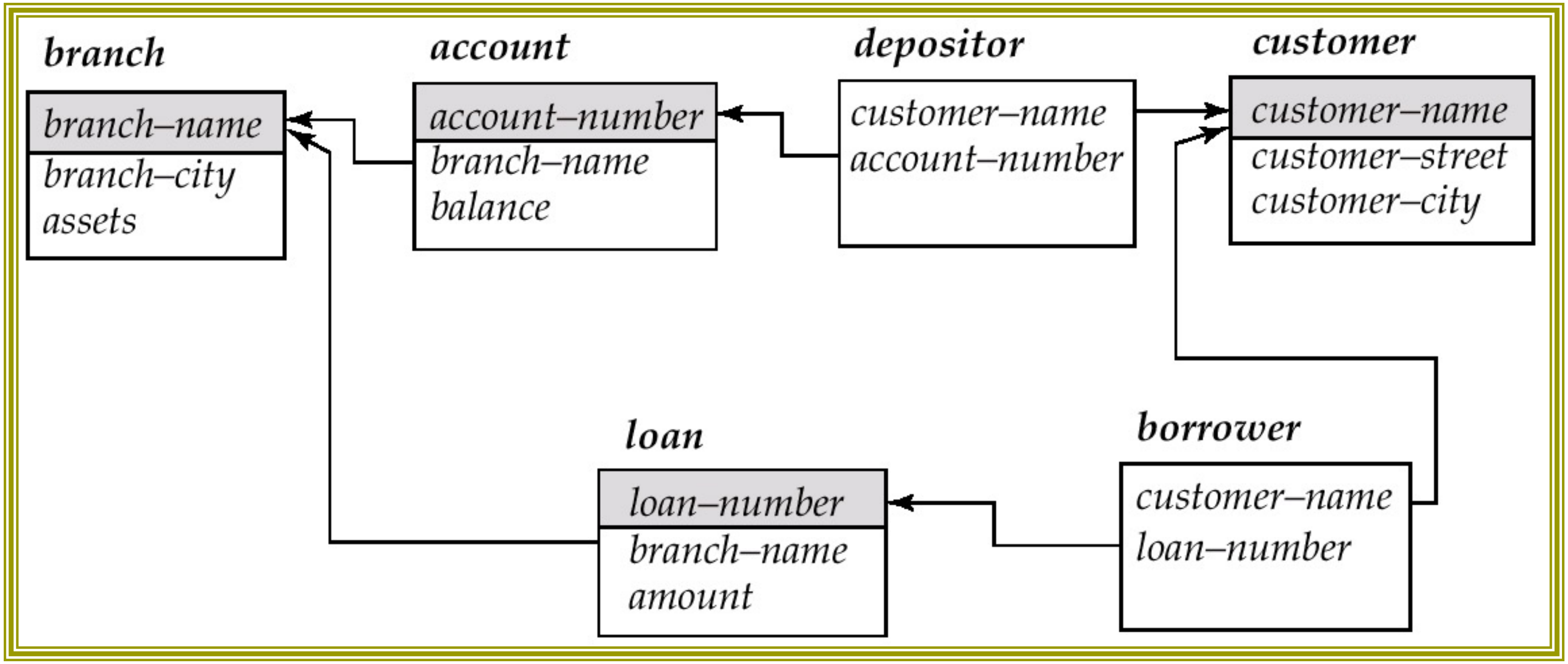
- An attribute or a set of attributes is a key for the relation if there cannot be two tuples in the relation with exactly the same values in these attributes
- The primary key of the relation is a designated key
  - E.g. an identifier attribute

Database

- A database consists of multiple relations which are inter-related
- Information about an enterprise is broken up into parts, with each relation storing one part of the information

  E.g.:
  - *account* : stores information about accounts
  - *depositor* : stores information about which customer owns which account
  - *customer* : stores information about customers

Schema Diagram for a Banking Enterprise



Join operations can be used to bring together information which is split to multiple relations

Natural Join Operation – Example

- Relation *loan*

loan-number	*branch-name*	*amount*
L-170	Downtown	3000
L-230	Redwood	4000
L-260	Perryridge	1700

- Relation *borrower*

customer-name	*loan-number*
Jones	L-170
Smith	L-230

- Relation loan $\bowtie$ borrower

loan-number	*branch-name*	*amount*	*customer-name*
L-170	Downtown	3000	Jones
L-230	Redwood	4000	Smith

Aggregate Operation – Example

- Relation *account* grouped by *branch-name*:

branch-name	*account-number*	*balance*
Perryridge	A-102	400
Perryridge	A-201	900
Brighton	A-217	750
Brighton	A-215	750
Redwood	A-222	700

$$_{branch\text{-}name}\,\mathcal{G}_{\,sum(balance)}\,(account)$$

branch-name	*balance*
Perryridge	1300
Brighton	1500
Redwood	700

SQL: Basic Structure

- SQL is based on set and relational operations with certain modifications and enhancements
- A typical SQL query has the form:

  select $A_1, A_2, \ldots, A_n$
  from $r_1, r_2, \ldots, r_m$
  where P

  - A_is represent attributes
  - r_is represent relations
  - P is a predicate.
- This query is equivalent to the relational algebra expression:

$$\Pi_{A1,\ A2,\ \ldots,\ An}(\sigma_P\,(r_1 \times r_2 \times \ldots \times r_m))$$

- The result of an SQL query is a multiset (bag) of tuples

Aggregate Functions – Group By

- Find the number of depositors for each branch.

select *branch-name,* **count (distinct** *customer-name)*
from *depositor, account*
where *depositor.account-number = account.account-number*
group by *branch-name*

- Note: In the select clause outside of aggregate functions we must have:
  - attributes that appear in the group by list and/or
  - aggregate functions on attributes of each group

Aggregate Functions – Having Clause

- Find the names and average account balances of all branches where the average account balance is more than $1,200.

select *branch-name,* **avg** *(balance)*
from *account*
group by *branch-name*
having avg *(balance)* > 1200

Note: predicates in the **having** clause are applied after the formation of groups whereas predicates in the **where** clause are applied before forming groups

Data Types

Types of data

- Record data (each object is a row)
  - Data Matrix (dense vectors, all attributes have values)
  - Document Data (sparse vectors, some attributes have values)
  - Set-valued Data (sparse binary vectors)
- Graphs (each object is a node, edges are relationships)
  - World Wide Web (each object is a web page, edges are links)
  - Molecular Structures (each object is an atom, edges are bonds)
- Ordered data (objects are sequences of simple data types)
  - Temporal Data
  - Sequential Data
  - Genetic Sequence Data
- Unstructured Data (objects have no structure)
  - Text documents
- Semistructured Data (objects may have different structure)
  - XML, JSON, etc.

Record Data

- Data that consists of a collection of records, each of which has a fixed set of attributes
- As in a relational table

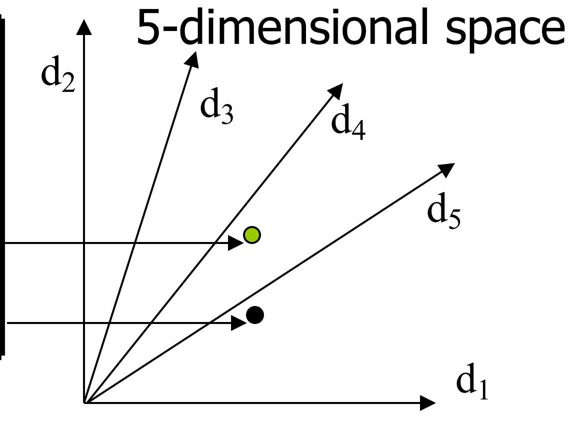
tuples (records, rows)

sid	name	login	age	gpa
53821	Jones	jones@math	18	1.8
53832	Smith	smith@math	19	3.2
53927	Parker	parker@cs	21	2.5
53111	Smith	smith@eee	20	2.8
53267	Black	black@me	19	3.1
53542	Dave	dave@phy	18	3.6

Data Matrix Mapping

- If data objects have the same fixed set of numeric attributes, then the data objects can be thought of as points in a multi-dimensional space, where each dimension represents a distinct attribute

Projection of x Load	Projection of y load	Distance	Load	Thickness
10.23	5.27	15.22	2.7	1.2
12.65	6.25	16.22	2.2	1.1



Relational Data

- Multiple tables logically connected to each other
- Some tables represent the relationships between entities appearing in other tables
- A **database schema** :
  - Employees(*ssn*: char(11), *name*: char(30), *lot*: integer)
  - Departments(*did*: integer, *dname*: char(20), *budget*: real)
  - Works_in(*ssn*: char(11), *did*: integer, *since*: date)
- A **database instance** :

Employees

ssn	***name***	***lot***
13-324	Jones	22
13-322	Smith	45
12-824	Parker	125
21-397	Smith	12

Departments

did	***dname***	***budget***
34	Toys	122000
12	Tools	239000
76	Food	100000

Works_in

ssn	***did***	***since***
13-324	34	1/1/01
13-322	34	1/4/02
13-322	12	2/2/05
12-824	76	1/1/03
21-397	76	1/1/05

Document Data (unstructured to structured data)

- Each document becomes a 'term' vector,
  - each term is a component (attribute) of the vector,
  - the value of each component is the number of times the corresponding term occurs in the document.

	team	coach	play	ball	score	game	win	lost	timeout	season
Document 1	3	0	5	0	2	6	0	2	0	2
Document 2	0	7	0	2	1	0	0	3	0	0
Document 3	0	1	0	0	1	2	2	0	3	0

Set-valued Data (Transactional Data)

- A special type of record data, where:
  - Each record (e.g., transaction) involves a set of items.
  - For example, consider a grocery store. The set of products purchased by a customer during one shopping trip constitute a transaction, while the individual products that were purchased are the items.
  - Instead of modeling them as sparse binary vectors, we can use the original set representation, which saves us a lot of space.

sparse binary vector representation

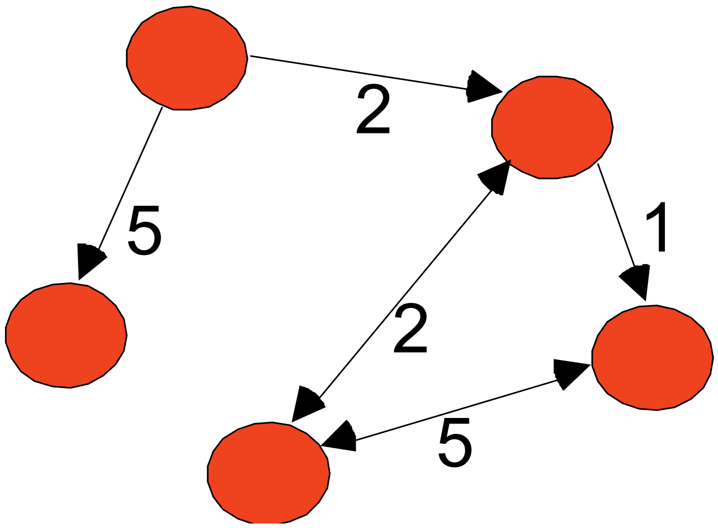
TID	Bread	Coke	Milk	Beer	Diaper
1	1	1	1	0	0
2	1	0	0	1	0
3	0	1	1	1	1
4	1	0	1	1	1
5	0	1	1	0	1

original set representation

TID	Items
1	Bread, Coke, Milk
2	Beer, Bread
3	Beer, Coke, Diaper, Milk
4	Beer, Bread, Diaper, Milk
5	Coke, Diaper, Milk

Graph Data

- Example: Generic graph and HTML Links



Data Mining

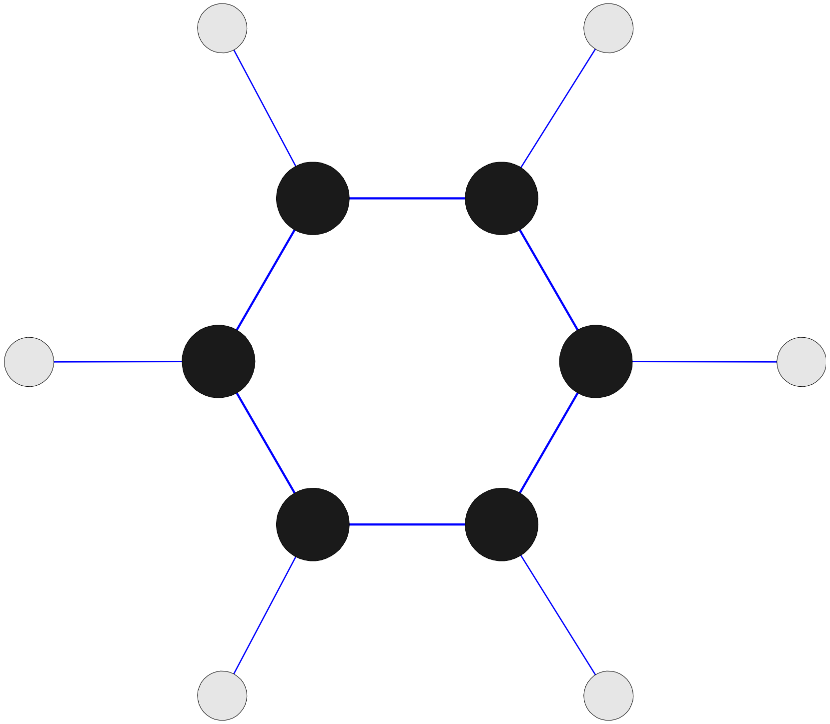
Graph Partitioning

Parallel Solution of Sparse Linear System of Equations

<a href="papers/papers.html#ffff">
N-Body Computation and Dense Linear System Solvers

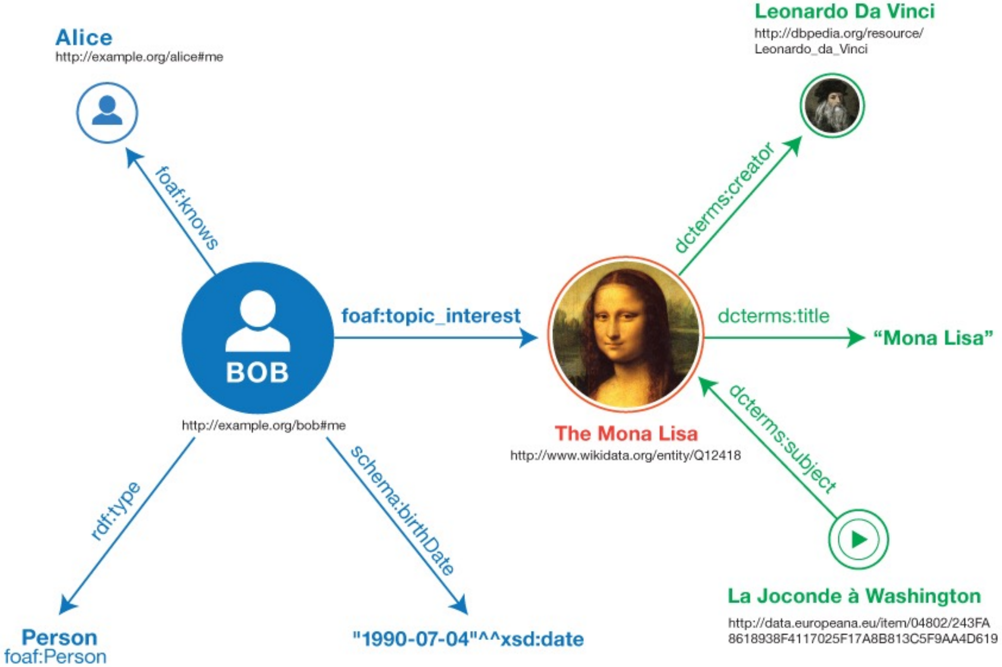
Graph Data

- Example: Chemical Data
- Benzene Molecule: C_6H_6

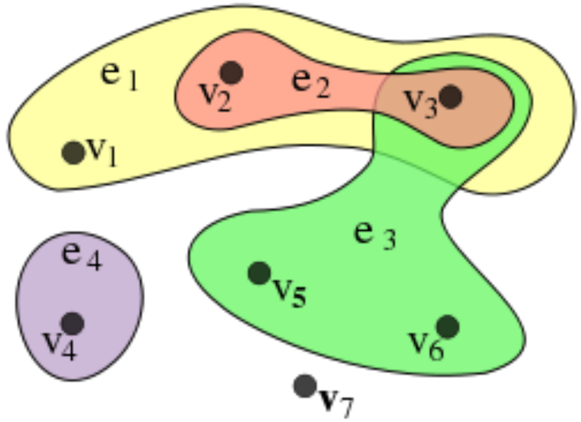


Graph Data

- Directed or undirected
- Nodes and edges can be labeled or not (information networks, knowledge bases)
- Molecular structures are 3D arrangements (the angles between edges matter)
- Hypergraphs: edges can join any number of vertices



example RDF graph taken from w3.org



example hypergraph taken from Wikipedia

Ordered Data

- Sequences of purchase transactions

Items/Events

(A B) (D) (C E)

(B D) (C) (E)

(C D) (B) (A E)

An element of the sequence

Ordered Data

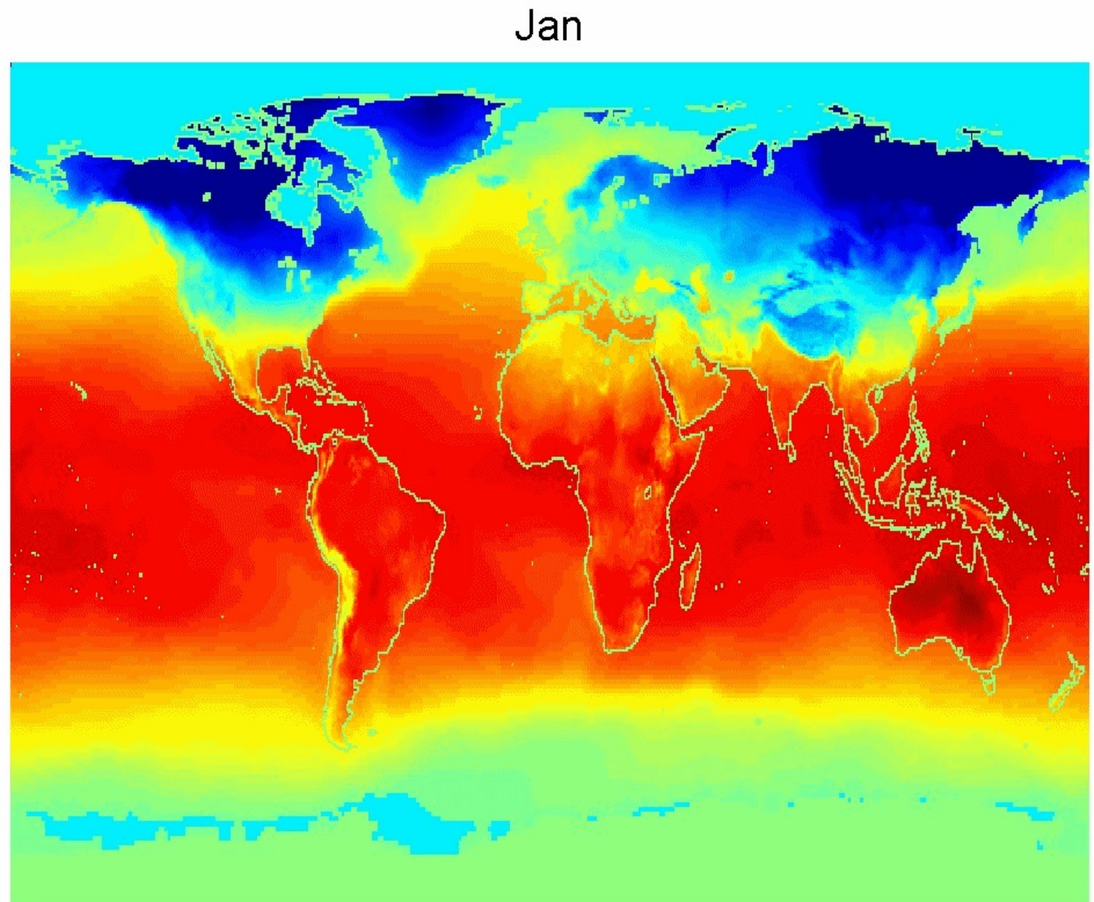
- Genomic sequence data

GGTTCCGCCTTCAGCCCCGCGCC
CGCAGGGCCCGCCCCGCGCCGTC
GAGAAGGGCCCGCCTGGCGGGCG
GGGGGAGGCGGGGCCGCCCGAGC
CCAACCGAGTCCGACCAGGTGCC
CCCTCTGCTCGGCCTAGACCTGA
GCTCATTAGGCGGCAGCGGACAG
GCCAAGTAGAACACGCGAAGCGC
TGGGCTGCCTGCTGCGACCAGGG

Ordered Data

- Spatio-Temporal Data

Average Monthly Temperature of land and ocean



Unstructured Data

- Searching unstructured data is the main subject of Information Retrieval
- Each "object" is a document, or a doc segment
  - Ex: searching for paragraphs or excerpts
- Documents are converted to a structured format that facilitates search
- Information Retrieval is approximate (subjective) by nature
  - Correct results are those that make the user happy

Semistructured Data

- XML documents embed structure and content
- Different documents may have different structure

```
<atricle>
    <title> The Importance of Evergreen </title>
    <author id=“smith”>
        <name>
            <firstname> John </firstname>
            <lastname> Smith </lastname>
        </name>
        <address> Smithsville </address>
    </author>
    <author id=“jones”>
        <name>
            <lastname> Jones </lastname>
        </name>
        <address> Jonesville </address>
    </author>
    <contactauthor idref=“smith”>
</article>
```

Important Characteristics of Structured Data

- Dimensionality: number of attributes each object has
  - Data of large dimensionality are hard to handle

- Sparsity
  - Only presence counts

- Resolution
  - Retrieved information depends on scale

Record Data: Typical Structured Data

- Collection of data objects and their attributes
- An attribute is a property or characteristic of an object
  - Examples: eye color of a person, temperature, etc.
  - Attribute is also known as variable, field, characteristic, or feature
- A collection of attribute values describe an object
  - Object is also known as record, point, case, sample, entity, or instance

Attributes

Objects

ssn	*name*	*lot*
13-324	Jones	22
13-322	Smith	45
12-824	Parker	125
21-397	Smith	12

Properties of Data Types

Attribute Values

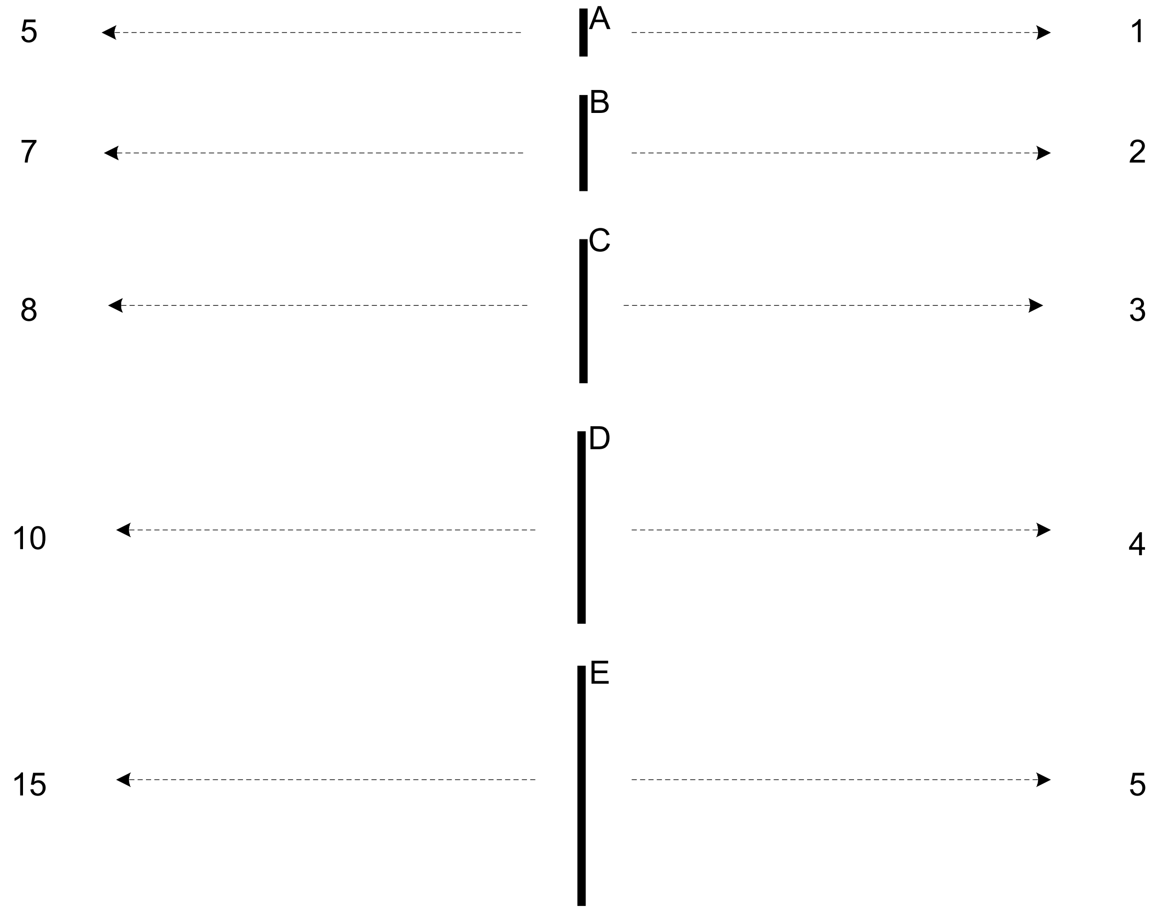
- Attribute values are numbers or symbols assigned to an attribute

- Distinction between attributes and attribute values
  - Same attribute can be mapped to different attribute values
    - Example: height can be measured in feet or meters

  - Different attributes can be mapped to the same set of values
    - Example: Attribute values for ID and age are integers
    - But properties of attribute values can be different
      - ID has no limit but age has a max and minvalue

Measurement of Length

- **The way you measure an attribute may not capture all the attributes properties.**



Types of Attributes

- There are different types of attributes
  - Nominal
    - Examples: ID numbers, eye color, zip codes
  - Ordinal
    - Examples: rankings (e.g., taste of potato chips on a scale from 1-10), grades, height in {tall, medium, short}
  - Interval
    - Examples: calendar dates, temperatures in Celsius or Fahrenheit.
  - Ratio
    - Examples: temperature in Kelvin, length, time, counts

Properties of Attribute Values

- The type of an attribute depends on which of the following properties it possesses:
  - Distinctness: $= \quad \neq$
  - Order: $< \quad >$
  - Addition: $+ \quad -$
  - Multiplication: $* \quad /$

  - Nominal attribute: distinctness
  - Ordinal attribute: distinctness & order
  - Interval attribute: distinctness, order & addition
  - Ratio attribute: all 4 properties

Attribute Type	Description	Examples	Statistics
Nominal	The values of a nominal attribute are just different names, i.e., nominal attributes provide only enough information to distinguish one object from another. (=, ≠)	zip codes, employee ID numbers, eye color, sex: {*male, female*}	mode, entropy, contingency, correlation, χ^2 test
Ordinal	The values of an ordinal attribute provide enough information to order objects. (<, >)	hardness of minerals, {*good, better, best*}, grades, street numbers	median, percentiles, rank correlation, run tests, sign tests
Interval	For interval attributes, the differences between values are meaningful, i.e., a unit of measurement exists. (+, -)	calendar dates, temperature in Celsius or Fahrenheit	mean, standard deviation, Pearson's correlation
Ratio	For ratio variables, both differences and ratios are meaningful. (*, /)	temperature in Kelvin, monetary quantities, counts, age, mass, length, electrical current	geometric mean, harmonic mean

Attribute Level	Allowed transformation	Comments
Nominal	Any permutation of values	If all employee ID numbers were reassigned, would it make any difference?
Ordinal	An order preserving change of values, i.e., *new_value = f(old_value)* where *f* is a monotonic function.	An attribute encompassing the notion of good, better best can be represented equally well by the values {1, 2, 3} or by { 0.5, 1, 10}.
Interval	*new_value =a * old_value + b* where a and b are constants	Thus, the Fahrenheit and Celsius temperature scales differ in terms of where their zero value is and the size of a unit (degree).
Ratio	*new_value = a * old_value*	Length can be measured in meters or feet.

Discrete and Continuous Attributes

- Discrete Attribute
  - Has only a finite or countably infinite set of values
  - Examples: zip codes, counts, or the set of words in a collection of documents
  - Often represented as integer variables.
  - Note: binary attributes are a special case of discrete attributes

- Continuous Attribute
  - Has real numbers as attribute values
  - Examples: temperature, height, or weight.
  - Practically, real values can only be measured and represented using a finite number of digits.
  - Continuous attributes are typically represented as floating-point variables.

Asymmetric Attributes

- Binary attributes
- Only presence (non-zero) values are regarded as important
- Examples:
  - Items purchased in a supermarket transaction
  - Words that appear in a document
  - Symptoms of a patient at a hospital
- When comparing the values of an asymmetric attribute in two objects, only non-zero values are important

Non-record data

- Data in a non-record format can be searched and analyzed by
  1. Extracting features from them (e.g., existence of known common substructures)
     - E.g., color features, texture, shape, objects from images
  2. Representing them as record data based on the features they contain
- Alternatively, use other representations:
  - Graph, semistructured
- Avoid using inappropriate representation!

Working with Data

Data Search and Analysis

- The reason behind preprocessing, storing, and indexing the data is the efficient support of data search and analysis
- Database queries (e.g., SQL) have a well-defined structure and a well-defined result
- IR queries (e.g., keyword search) are unstructured and the quality of the result is not ensured
  - Search based on content
  - Result depends on similarity function used
- Similarity queries retrieve the most similar objects to a given query object
- Data mining operations aim at finding statistically interesting patterns in data

Information Retrieval

- Typically refers to searching for unstructured data (typically documents)
- Search is expressed by means of a keyword query
  - A set of keywords
  - Documents that are the most relevant to these keywords are retrieved
  - Relevance can either be based on boolean containment or on similarity
  - Other factors are considered in the ranking (e.g. popularity of documents)
  - Quality of search is subjective (i.e., personalized)
- Web keyword search is the dominant IR paradigm
  - Others: email search

Similarity Search

- A generalization of keyword search in IR
- Input 1: a collection of objects that have the same format (e.g., same schema)
  - e.g., collection of documents, collection of images, records in a database table, vertices in a graph
- Input 2: a query object of the same format
- Output: the most similar object (or set of most similar objects) in the collection to the query object
- Problem 1: how to define an appropriate similarity function
  - Use quality measures to assess the effectiveness of alternative definitions
- Problem 2: how to process the query efficiently

Data Mining

- General definition:
  - Find statistically interesting patterns in data
  - Data can be of any format, can even be dirty
- Popular data mining tasks:
  - Classification (induction in Machine Learning): learn a set of rules from known factual data for classifying objects to a set of predefined categories (class labels)
  - Clustering: divide a collection of objects into a set of clusters, such that objects in the same cluster are similar to each other, while objects in different clusters are dissimilar
  - Association Analysis: discover interesting relations between variables in large databases, or statistically significant co-existences of entities in sets (e.g., market basket data)
  - Others: anomaly detection, regression, summarization

Data Preprocessing

Data Quality

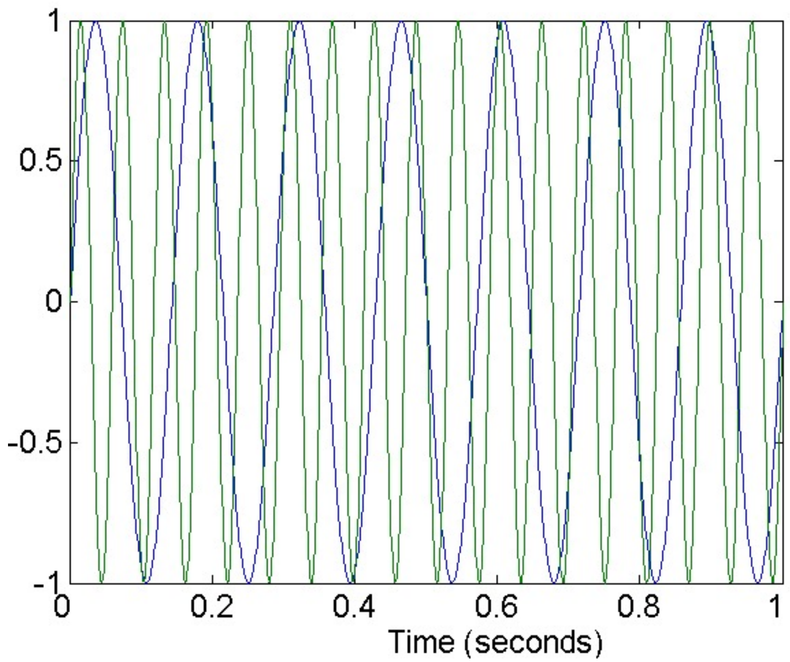
- Data should be of good quality for correct data analysis
  - Depends on intended queries or mining tasks

- What kinds of data quality problems?
- How can we detect problems with the data?
- What can we do about these problems?

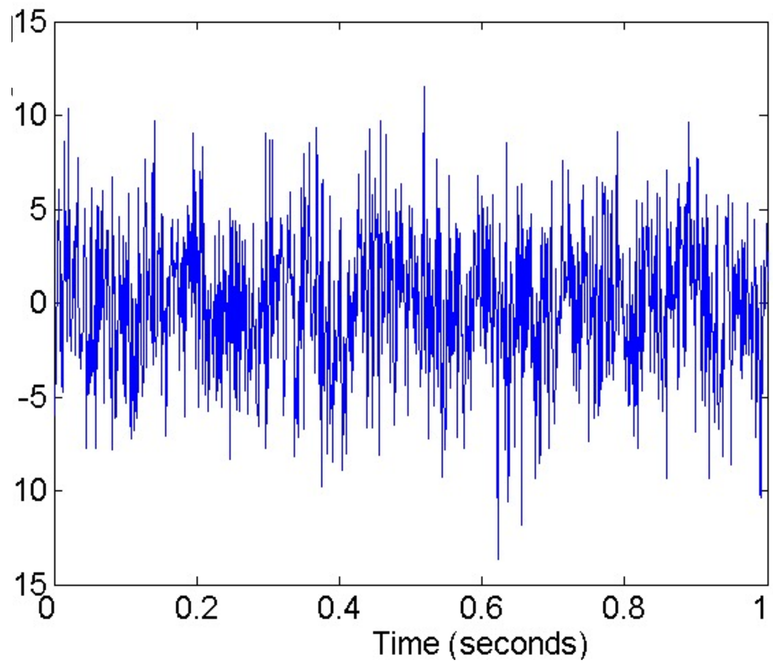
- Examples of data quality problems:
  - Noise and outliers
  - missing values
  - duplicate data

Noise

- Noise refers to modification of original values



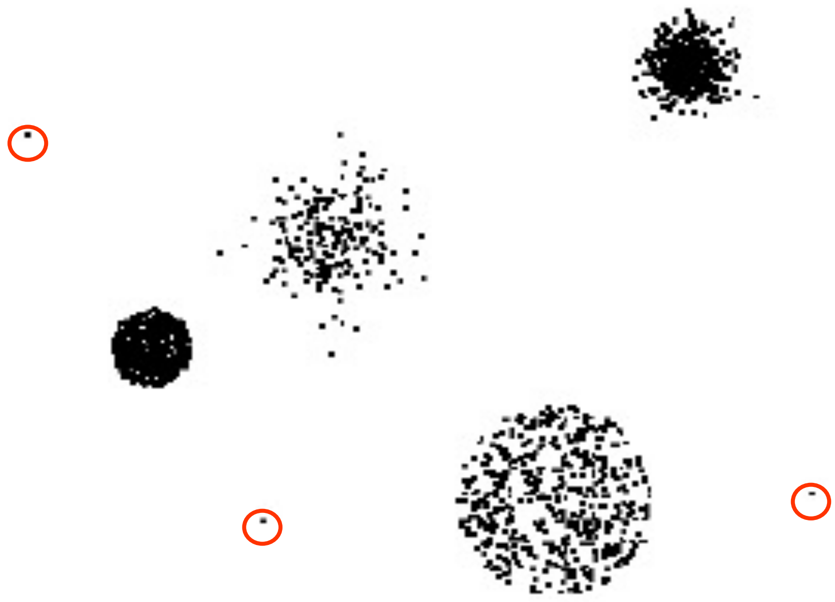
Two Sine Waves



Two Sine Waves + Noise

Outliers

- Outliers are data objects with characteristics that are considerably different than most of the other data objects in the data set



Missing Values

- Reasons for missing values
  - Information is not collected
    (e.g., people decline to give their age and weight)
  - Attributes may not be applicable to all cases
    (e.g., annual income is not applicable to children)

- Handling missing values
  - Eliminate Attributes with Missing Values
  - Eliminate Data Objects
  - Ignore the Missing Value During Analysis
  - Replace with all possible values (weighted by their probabilities)

Inconsistent or Duplicate Data

- Data set may include inconsistent values
  - E.g., negative age
  - Detect and treat them as missing values
  - Sometimes error-correction codes can be used to find correct value

- Data set may include data objects that are duplicates, or almost duplicates of one another
  - Major issue when merging data from heterogeneous sources
  - E.g., Same person with multiple email addresses

- Data cleaning
  - Process of dealing with inconsistent/missing/duplicate data issues

Data Preprocessing

- Aggregation
- Sampling
- Dimensionality Reduction
- Feature subset selection
- Feature creation
- Discretization and Binarization
- Attribute Transformation

Aggregation

- Combining two or more attributes (or objects) into a single attribute (or object)

- Purpose
  - Data reduction
    - Reduce the number of attributes or objects
    - Introduce new “summary” attributes
  - Change of scale
    - Cities aggregated into regions, states, countries, etc
  - More “smooth” data
    - Aggregated data tends to have less variability

Sampling

- Sampling is the main technique employed for data selection.
  - It is often used for both the preliminary investigation of the data and the final data analysis.

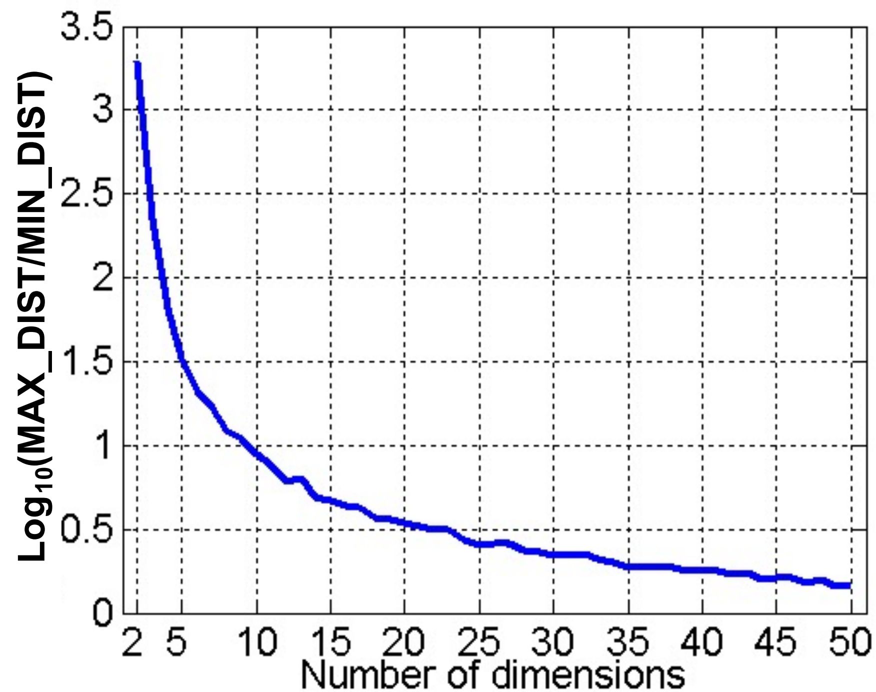
- Statisticians sample because obtaining the entire set of data of interest is too expensive or time consuming.

- Sampling is used because managing and processing the entire set of data of interest is too expensive or time consuming.

Curse of Dimensionality

- When dimensionality increases, data becomes increasingly sparse in the space that it occupies

- Definitions of density and distance between points, which is critical for clustering and outlier detection, become less meaningful



- **Randomly generate 500 points**
- **Compute difference between max and min distance between any pair of points**

Dimensionality Reduction

- Purpose:
  - Avoid curse of dimensionality
  - Reduce amount of time and memory required by data mining algorithms
  - Allow data to be more easily visualized
  - May help to eliminate irrelevant features or reduce noise
- Techniques
  - Principal Component Analysis
  - Singular Value Decomposition
  - Others: supervised and non-linear techniques

Feature Subset Selection

- Another way to reduce dimensionality of data
- Redundant features
  - duplicate much or all of the information contained in one or more other attributes
  - Example: purchase price of a product and the amount of sales tax paid
- Irrelevant features
  - contain no information that is useful for the data mining task at hand
  - Example: students' ID is often irrelevant to the task of predicting students' GPA

Discretization and Binarization

- Some search/analysis algorithms require that data are nominal
- Continuous-domain data should be converted to categorical
- Transformation should not affect quality of analysis
- Discretization: Convert continuous attribute to categorical
- Binarization: Convert continuous/discrete attribute to binary

Attribute Transformation

- A function that maps the entire set of values of a given attribute to a new set of replacement values such that each old value can be identified with one of the new values
  - Simple functions: x^k, $\log(x)$, e^x, $|x|$
  - Standardization and Normalization
    - Traditional standardization in stats: $x' = (x-\mu)/\sigma$
    - Min-max normalization: $x' = (x-min)/(max-min)$
    - If outliers exist, replace mean by median and standard deviation by absolute standard deviation

Similarity and Distance

Similarity and Dissimilarity

- Similarity
  - Numerical measure of how alike two data objects are.
  - Is higher when objects are more alike.
  - Often falls in the range [0,1]
- Dissimilarity
  - Numerical measure of how different are two data objects
  - Lower when objects are more alike
  - Minimum dissimilarity is often 0
  - Upper limit varies
- Proximity refers to a similarity or dissimilarity

Similarity/Dissimilarity for Simple Attributes

p and *q* are the attribute values for two data objects.

Attribute Type	Dissimilarity	Similarity
Nominal	$d = \begin{cases} 0 & \text{if } p = q \\ 1 & \text{if } p \neq q \end{cases}$	$s = \begin{cases} 1 & \text{if } p = q \\ 0 & \text{if } p \neq q \end{cases}$
Ordinal	$d = \frac{\lvert p-q \rvert}{n-1}$ (values mapped to integers 0 to $n-1$, where n is the number of values)	$s = 1 - \frac{\lvert p-q \rvert}{n-1}$
Interval or Ratio	$d = \lvert p - q \rvert$	$s = -d$, $s = \frac{1}{1+d}$ or $s = 1 - \frac{d - min\_d}{max\_d - min\_d}$

Table 5.1. Similarity and dissimilarity for simple attributes

Euclidean Distance

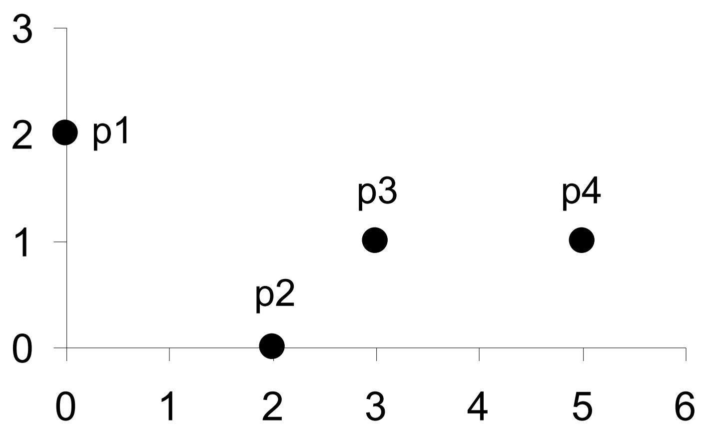
- Euclidean Distance

$$dist = \sqrt{\sum_{k=1}^{n} (p_k - q_k)^2}$$

Where n is the number of dimensions (attributes) and p_k and q_k are, respectively, the k^th attributes (components) or data objects p and q.

- Applicable if objects have multiple attributes and none of them is nominal
- Standardization is necessary, if scales differ
  - Typically min-max normalization at each dimension
    - x' = (x-min)/(max-min)

Euclidean Distance



point	x	y
p1	0	2
p2	2	0
p3	3	1
p4	5	1

	p1	p2	p3	p4
p1	0	2.828	3.162	5.099
p2	2.828	0	1.414	3.162
p3	3.162	1.414	0	2
p4	5.099	3.162	2	0

Distance Matrix

Minkowski Distance

- Minkowski Distance is a generalization of Euclidean Distance

$$dist = \left(\sum_{k=1}^{n} | p_k - q_k |^r \right)^{\frac{1}{r}}$$

Where r is a parameter, n is the number of dimensions (attributes) and p_k and q_k are, respectively, the kth attributes (components) or data objects p and q.

Minkowski Distance: Examples

- $r = 1$. City block (Manhattan, taxicab, L_1 norm) distance.
  - A common example of this is the Hamming distance, which is just the number of bits that are different between two binary vectors
- $r = 2$. Euclidean distance
- $r \to \infty$. "supremum" (L_{max} norm, L_∞ norm) distance.
  - This is the maximum difference between any component of the vectors
- Do not confuse *r* with *n*, i.e., all these distances are defined for all numbers of dimensions.

Minkowski Distance

point	x	y
p1	0	2
p2	2	0
p3	3	1
p4	5	1

L1	p1	p2	p3	p4
p1	0	4	4	6
p2	4	0	2	4
p3	4	2	0	2
p4	6	4	2	0

L2	p1	p2	p3	p4
p1	0	2.828	3.162	5.099
p2	2.828	0	1.414	3.162
p3	3.162	1.414	0	2
p4	5.099	3.162	2	0

L_∞	p1	p2	p3	p4
p1	0	2	3	5
p2	2	0	1	3
p3	3	1	0	2
p4	5	3	2	0

Distance Matrix

Common Properties of a Distance

- Distances, such as the Euclidean distance, have some well-known properties.

  1. $d(p, q) \geq 0$ for all p and q and $d(p, q) = 0$ only if $p = q$. (Positivity)
  2. $d(p, q) = d(q, p)$ for all p and q. (Symmetry)
  3. $d(p, r) \leq d(p, q) + d(q, r)$ for all points p, q, and r. (Triangle Inequality)

  where $d(p, q)$ is the distance (dissimilarity) between points (data objects), p and q.

- A distance that satisfies these properties is a metric
- For metrics we can apply some generalized indexing techniques, without caring about their exact definition (we only use their properties)

Common Properties of a Similarity

- Similarities, also have some well-known properties.

  1. $s(p, q) = 1$ (or maximum similarity) only if $p = q$.
  2. $s(p, q) = s(q, p)$ for all p and q. (Symmetry)

  where $s(p, q)$ is the similarity between points (data objects), p and q.

Similarity Between Binary Vectors

- Common situation is that objects, *p* and *q*, have only binary attributes
  - Bought items in transaction records, symptoms of patients, etc.

- Compute similarities using the following quantities

  M_{01} = the number of attributes where p was 0 and q was 1
  M_{10} = the number of attributes where p was 1 and q was 0
  M_{00} = the number of attributes where p was 0 and q was 0
  M_{11} = the number of attributes where p was 1 and q was 1

- Simple Matching and Jaccard Coefficients

  SMC = number of matches / number of attributes
  $= (M_{11} + M_{00}) / (M_{01} + M_{10} + M_{11} + M_{00})$

  J = number of 1-matches / number of not-both-zero attributes values
  $= (M_{11}) / (M_{01} + M_{10} + M_{11})$

SMC versus Jaccard: Example

p = 1 0 0 0 0 0 0 0 0 0
q = 0 0 0 0 0 0 1 0 0 1

$M_{01} = 2$ (the number of attributes where p was 0 and q was 1)
$M_{10} = 1$ (the number of attributes where p was 1 and q was 0)
$M_{00} = 7$ (the number of attributes where p was 0 and q was 0)
$M_{11} = 0$ (the number of attributes where p was 1 and q was 1)

$$SMC = (M_{11} + M_{00})/(M_{01} + M_{10} + M_{11} + M_{00}) = (0+7) / (2+1+0+7) = 0.7$$

$$J = (M_{11}) / (M_{01} + M_{10} + M_{11}) = 0 / (2 + 1 + 0) = 0$$

Document data: Cosine Similarity

- If d_1 and d_2 are two document vectors, then

$$\cos(d_1, d_2) = (d_1 \bullet d_2) / ||d_1||\ ||d_2|| ,$$

where $\bullet$ indicates vector dot product and $||d||$ is the length of vector d.

- Example:

d_1 = **3 2 0 5 0 0 0 2 0 0**
d_2 = **1 0 0 0 0 0 0 1 0 2**

$d_1 \bullet d_2$= 3*1 + 2*0 + 0*0 + 5*0 + 0*0 + 0*0 + 0*0 + 2*1 + 0*0 + 0*2 = 5
$||d_1||$ = $(3*3+2*2+0*0+5*5+0*0+0*0+0*0+2*2+0*0+0*0)^{0.5}$ = $(42)^{0.5}$ = 6.481
$||d_2||$ = $(1*1+0*0+0*0+0*0+0*0+0*0+0*0+1*1+0*0+2*2)^{0.5}$ = $(6)^{0.5}$ = 2.45

$\cos(d_1, d_2)$ = .3150

	team	coach	play	ball	score	game	win	lost	timeout	season
Document 1	3	0	5	0	2	6	0	2	0	2
Document 2	0	7	0	2	1	0	0	3	0	0
Document 3	0	1	0	0	1	2	2	0	3	0

General Approach for Combining Similarities

- Sometimes the different attributes of objects in the same collection are of many different types, but an overall similarity is needed.

1. For the k^{th} attribute, compute a similarity, s_k, in the range $[0, 1]$.
2. Define an indicator variable, δ_k, for the k_{th} attribute as follows:

$$\delta_k = \begin{cases} 0 & \text{if the } k^{th} \text{ attribute is a binary asymmetric attribute and both objects have} \\ & \text{a value of } 0 \text{, or if one of the objects has a missing values for the } k^{th} \text{ attribute} \\ 1 & \text{otherwise} \end{cases}$$

3. Compute the overall similarity between the two objects using the following formula:

$$similarity(p, q) = \frac{\sum_{k=1}^{n} \delta_k s_k}{\sum_{k=1}^{n} \delta_k}$$

Why similarity/distance?

- Object retrieval
  - Range and nearest neighbor search in spatial data
  - Image retrieval based on feature vector similarity
  - Search for similar time-series
  - Information (document) retrieval
  - Find similar transaction records
- Recommender systems
  - Content-based recommendation
  - Find similar users in collaborative filtering
- Data mining
  - Nearest neighbor classification
  - Cluster analysis based on distances between objects

Summary

- Relational Databases is a mature and well used technology for data representation
  - however, there are limitations w.r.t. the data types that can be stored in relational databases
- Data are inherently complex
  - multiple data types
  - structured and less structured
- Before data management, we should bring data to the desired format
  - Assess data quality
  - Apply data preprocessing
  - Convert to structured data or another format
  - Depends on desired query operations on data, which in turn depends on application
- Defining a proper similarity function between data objects is a key issue in many search/analytics tasks