Chapter 20: Data Analysis

Database System Concepts, 6th Ed.

Chapter 20: Data Analysis

- 20.1 Decision Support Systems
- 20.2 Data Warehousing
- 20.3 Data Mining
- 20.4 Classification
- 20.5 Association Rules
- 20.6 Other Types of Associations
- 20.7 Clustering
- 20.8 Other Forms of Data Mining

Decision Support Systems

- **Decision-support systems** are used to make business decisions, often based on data collected by on-line transaction-processing systems

- Examples of business decisions:
  - What items to stock?
  - What insurance premium to change?
  - To whom to send advertisements?

- Examples of input data used for making decisions
  - Retail sales transaction details
  - Customer profiles (income, age, gender, etc.)

Decision-Support Systems: Overview

Components of DSS

- **Data analysis** tasks are simplified by specialized tools and SQL extensions
  - Example tasks
    - For each product category and each region, what were the total sales in the last quarter and how do they compare with the same quarter last year
    - As above, for each product category and each customer category
- **Statistical analysis** packages (e.g., : S++) can be interfaced with databases
  - Statistical analysis is a large field, but not covered here
- **Data mining** seeks to discover knowledge automatically in the form of statistical rules and patterns from large databases
- A **data warehouse** archives information gathered from multiple sources, and stores it under a unified schema, at a single site
  - Important for large businesses that generate data from multiple divisions, possibly at multiple sites
  - Data may also be purchased externally

Chapter 20: Data Analysis

- 20.1 Decision Support Systems
- 20.2 Data Warehousing
- 20.3 Data Mining
- 20.4 Classification
- 20.5 Association Rules
- 20.6 Other Types of Associations
- 20.7 Clustering
- 20.8 Other Forms of Data Mining

Data Warehousing

- Data sources often store only current data, not historical data
- Corporate decision making requires a unified view of all organizational data, including historical data
- A **data warehouse** is a repository (archive) of information gathered from multiple sources, stored under a unified schema, at a single site
  - Greatly simplifies querying, permits study of historical trends
  - Shifts decision support query load away from transaction processing systems

Data Warehousing

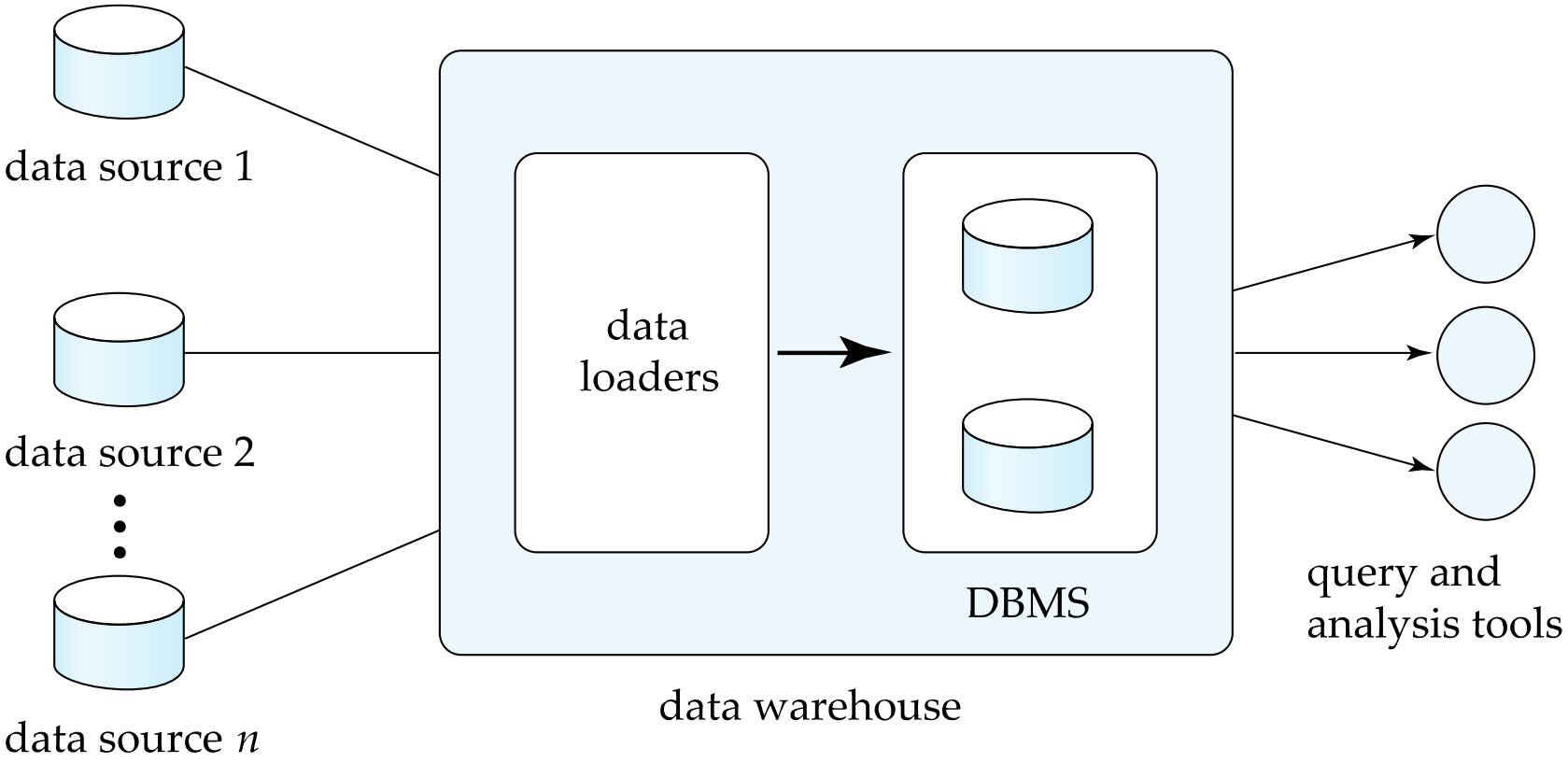


Fig 20.01

Data Warehouse Design Issues

- ***When and how to gather data***
  - **Source driven architecture**: data sources transmit new information to warehouse, either continuously or periodically (e.g., at night)
  - **Destination driven architecture**: warehouse periodically requests new information from data sources
  - Keeping warehouse exactly synchronized with data sources (e.g., using two-phase commit) is too expensive
    - Usually OK to have slightly out-of-date data at warehouse
    - Data/updates are periodically downloaded form online transaction processing (OLTP) systems
- ***What schema to use***
  - Schema integration

More Warehouse Design Issues

- ***Data cleansing***
  - E.g., correct mistakes in addresses (misspellings, zip code errors)
  - **Merge** address lists from different sources and **purge** duplicates
- ***How to propagate updates***
  - Warehouse schema may be a (materialized) view of schema from data sources
- ***What data to summarize***
  - Raw data may be too large to store on-line
  - Aggregate values (totals/subtotals) often suffice
  - Queries on raw data can often be transformed by query optimizer to use aggregate values

Warehouse Schemas

- Dimension values are usually encoded using small integers and mapped to full values via dimension tables
- Resultant schema is called a **star schema**
  - More complicated schema structures
    - **Snowflake schema**: multiple levels of dimension tables
    - **Constellation**: multiple fact tables

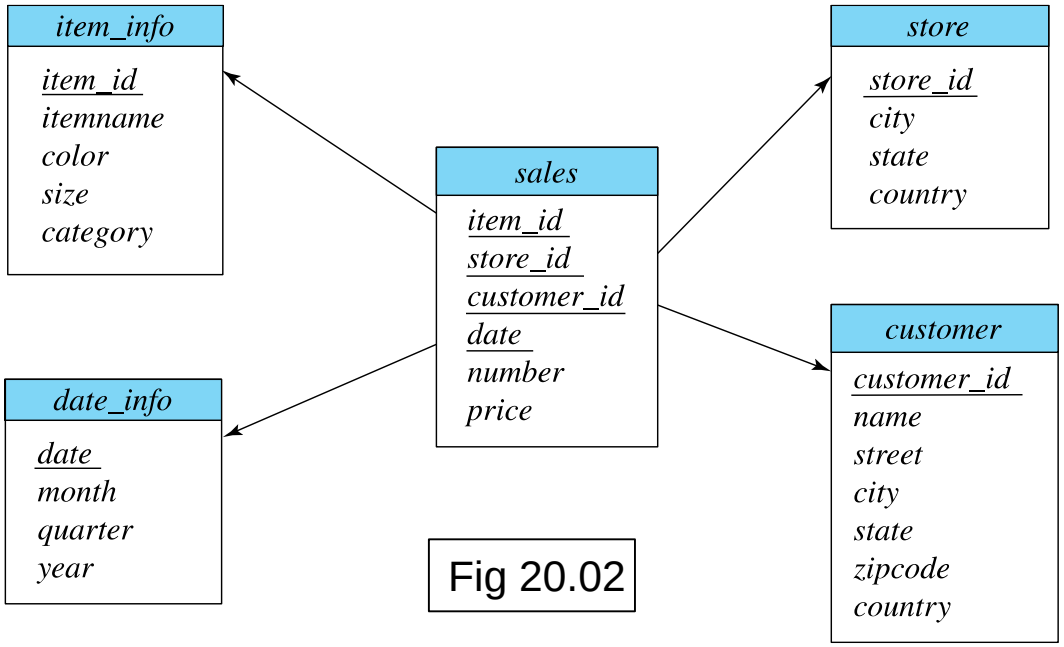


Fig 20.02

Chapter 20: Data Analysis

- 20.1 Decision Support Systems
- 20.2 Data Warehousing
- 20.3 Data Mining
- 20.4 Classification
- 20.5 Association Rules
- 20.6 Other Types of Associations
- 20.7 Clustering
- 20.8 Other Forms of Data Mining

Data Mining [1/2]

- Data mining is the process of semi-automatically analyzing large databases to find useful patterns
- **Prediction** based on past history
  - Predict if a credit card applicant poses a good credit risk, based on some attributes (income, job type, age, ..) and past history
  - Predict if a pattern of phone calling card usage is likely to be fraudulent
  - **Classification**
    - Given a new item whose class is unknown, predict to which class it belongs
    - 종속변수(y)를 독립변수(x)들의 함수 (f) 로 적합
    - 즉, 데이터 {(x,y)} 로 부터 y=f(x)의 f를 찾는다
    - 예: y 무엇을 예측할 수 있는가?
      - 소비자가 마케팅 캠페인에 반응할 확률
      - 휴대폰 고객이 향후 6개월 내에 이탈할 확률
      - 와인의 품질 => 품질 등급
  - **Regression** formulae
    - Given a set of mappings for an unknown function, predict the function result for a new parameter value

Data Mining [2/2]

- **Descriptive Patterns**
  - **Associations**
    - Find books that are often bought by “similar” customers. If a new such customer buys one such book, suggest the others too
    - Associations may be used as a first step in detecting **causation**
      - E.g., association between exposure to chemical X and cancer
  - **Clusters**
    - Form groups (clusters) of similar records
      - Segmenting markets into groups of similar customers
      - Detection of clusters remains important in detecting epidemics
        - » E.g. typhoid cases were clustered in an area surrounding a contaminated well

Chapter 20: Data Analysis

- 20.1 Decision Support Systems
- 20.2 Data Warehousing
- 20.3 Data Mining
- 20.4 Classification
- 20.5 Association Rules
- 20.6 Other Types of Associations
- 20.7 Clustering
- 20.8 Other Forms of Data Mining

Classification Rules

- Classification rules help assign new objects to classes.
  - E.g., given a new automobile insurance applicant, should he or she be classified as low risk, medium risk or high risk?

- Classification rules for above example could use a variety of data, such as educational level, salary, age, etc.
  - $\forall$ person P, P.degree = masters **and** P.income > 75,000
    $\Rightarrow$ P.credit = excellent
  - $\forall$ person P, P.degree = bachelors **and**
    (P.income $\geq$ 25,000 and P.income $\leq$ 75,000)
    $\Rightarrow$ P.credit = good

- Rules are not necessarily exact: there may be some misclassifications
- Classification rules can be shown compactly as a decision tree

Decision Tree

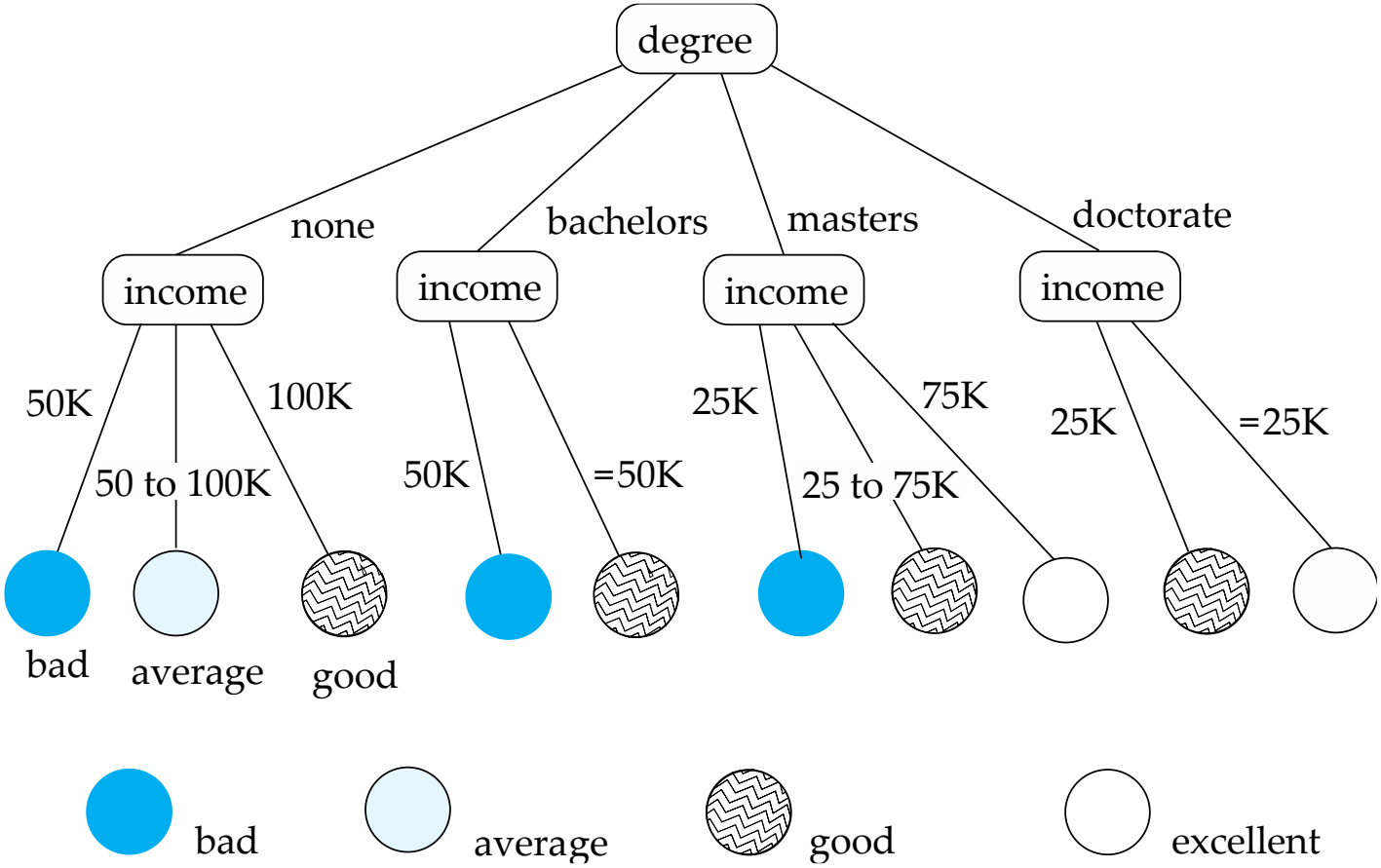


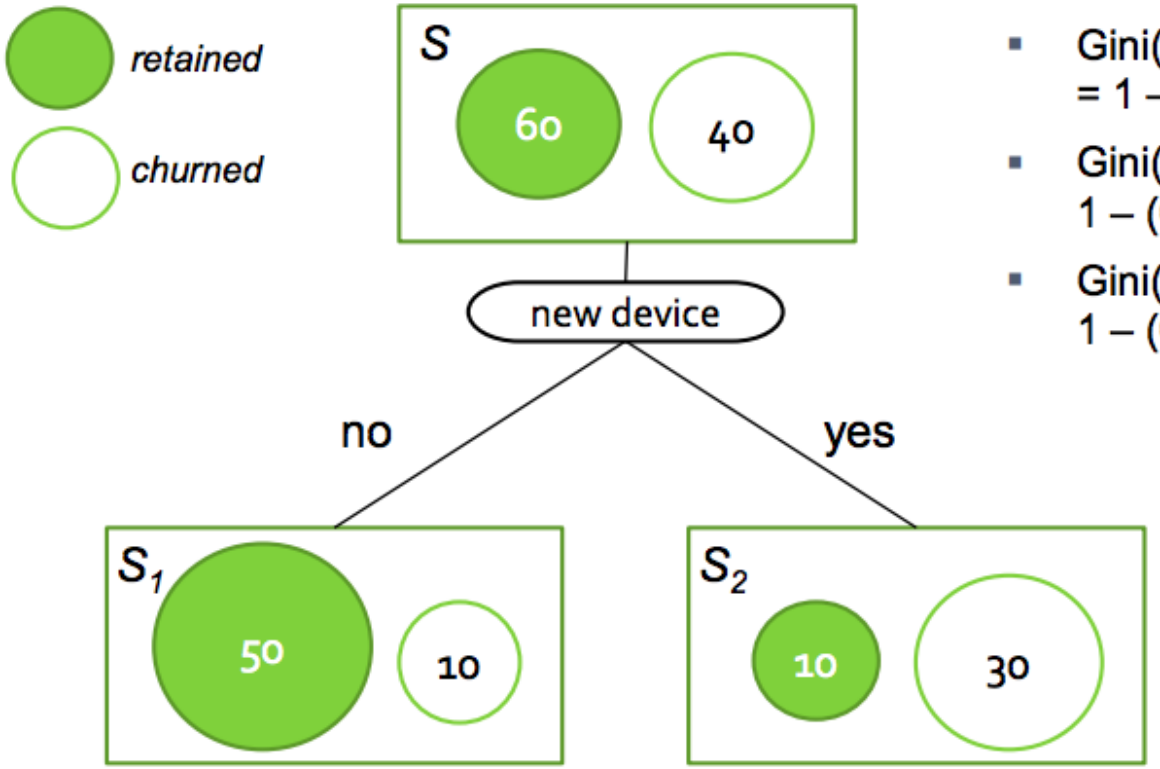
Fig 20.03

Construction of Decision Trees

- **Training set**: a data sample in which the classification is already known.
- **Greedy** top down generation of decision trees.
  - Each internal node of the tree partitions the data into groups based on a **partitioning attribute**, and a **partitioning condition** for the node
  - **Leaf** node:
    - all (or most) of the items at the node belong to the same class, or
    - all attributes have been considered, and no further partitioning is possible

Decision Tree Example

- Compute *Gini measures* and the *information gain*.



- Gini(S) = 1 – ((60/100)^2 + (40/100)^2)
  = 1 – (0.36 + 0.16) = 1 – 0.52 = 0.48
- Gini(S_1) = 1 – ((50/60)^2 + (10/60)^2) =
  1 – (0.69 + 0.03) = 1 – 0.72 = 0.28
- Gini(S_2) = 1 – ((10/40)^2 + (30/40)^2) =
  1 – (0.06 + 0.56) = 1 – 0.62 = 0.38

Information Gain

= purity(S) – (0.6*purity(S_1) +
0.4*purity(S_2))

= 0.48 – (0.6*0.28 + 0.4*0.38)

= 0.48 – 0.32 = 0.16

Best Splits [1/3]

- Pick best attributes and conditions on which to partition
- The purity of a set S of training instances can be measured quantitatively in several ways
  - Notation
    - number of classes = k
    - number of instances = |S|
    - fraction of instances in class i = p_i
- The **Gini measure of purity** is defined as

$$\text{Gini}(S) = 1 - \sum_{i-1}^{k} p^2_i$$

  - When all instances are in a single class, the Gini value is 0
  - It reaches its maximum (of 1 –1 /k) if each class the same number of instances

Best Splits [2/3]

- Another measure of purity is the **entropy** measure, which is defined as

$$\text{entropy (S)} = -\sum_{i-1}^{k} p_i log_2 p_i$$

  - The entropy value is 0 if all instances are in a single class
  - It reaches its maximum when each class has the same number of instances

- When a set S is split into multiple sets Si, I=1, 2, …, r, we can measure the purity of the resultant set of sets as:

$$\text{purity}(S_1, S_2, \ldots, S_r) = \sum_{i=1}^{r} \frac{|S_i|}{|S|} \text{purity}(S_i)$$

- Either Gini or Entropy can be used as purity function

Best Splits [3/3]

- The information gain due to particular split of S into S_i, i = 1, 2, …., r

  Information-gain (S, {S_1, S_2, …., S_r) = purity(S) – purity (S_1, S_2, … S_r)

- Measure of "cost" of a split:

  Information-content (S, {S_1, S_2, ….., S_r})) = $-\sum_{i-1}^{r} \frac{|S_i|}{|S|} \log_2 \frac{|S_i|}{|S|}$

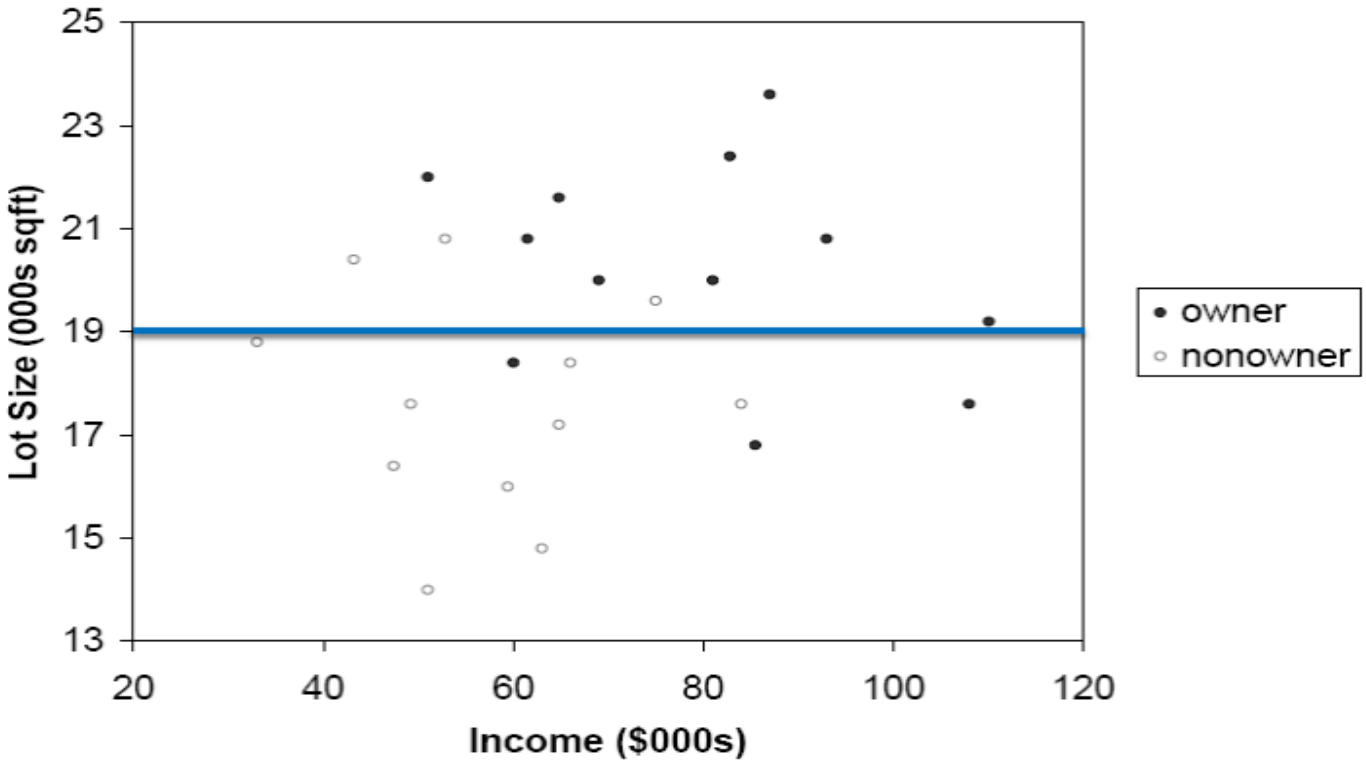
- **Information-gain ratio** = $\frac{\text{Information-gain } (S, \{S_1, S_2, \ldots\ldots, S_r\})}{\text{Information-content } (S, \{S_1, S_2, \ldots\ldots, S_r\})}$

- The best split is the one that gives the maximum information gain ratio

Decision Tree for Lawn Owner [1/4]

참고자료

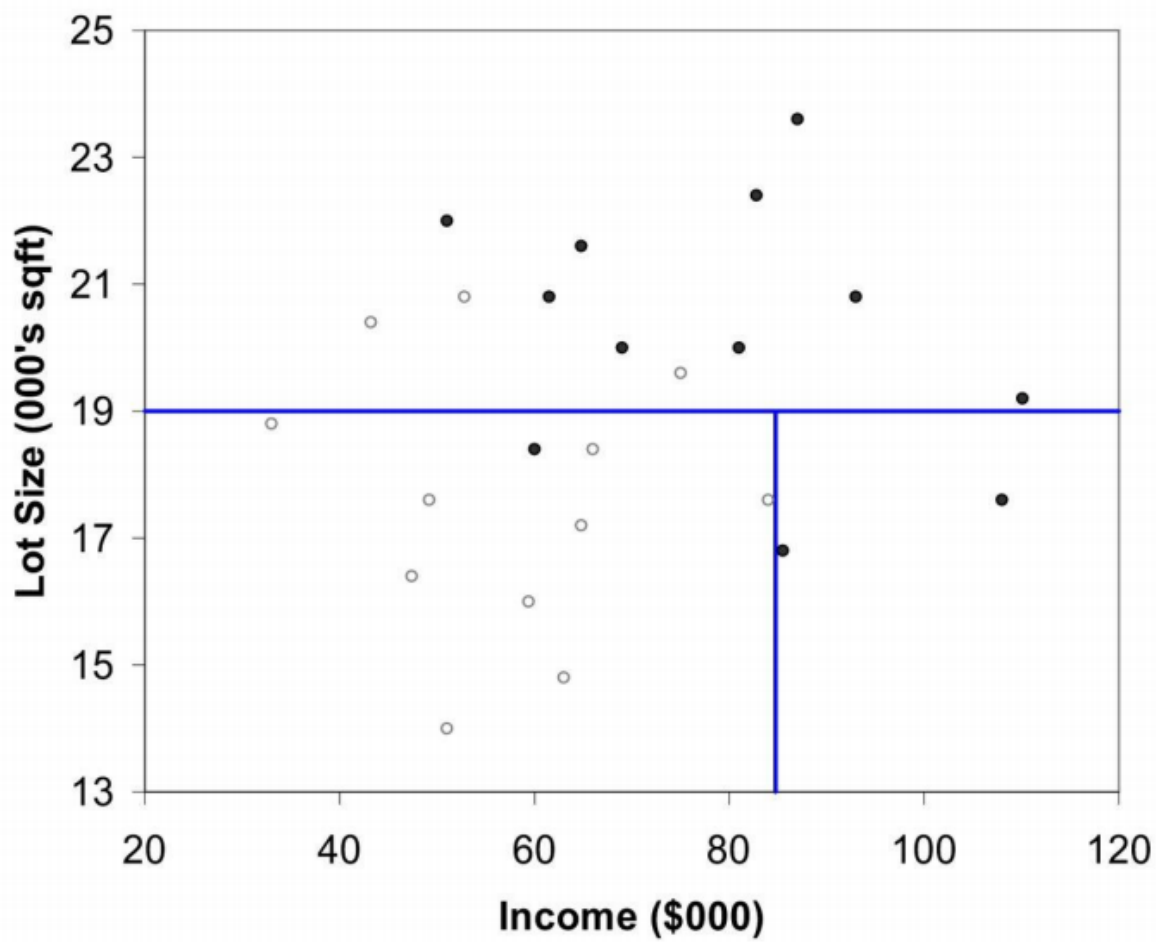
첫 번째 분할: Lot Size = 19,000



Decision Tree for Lawn Owner [2/4]

참고자료

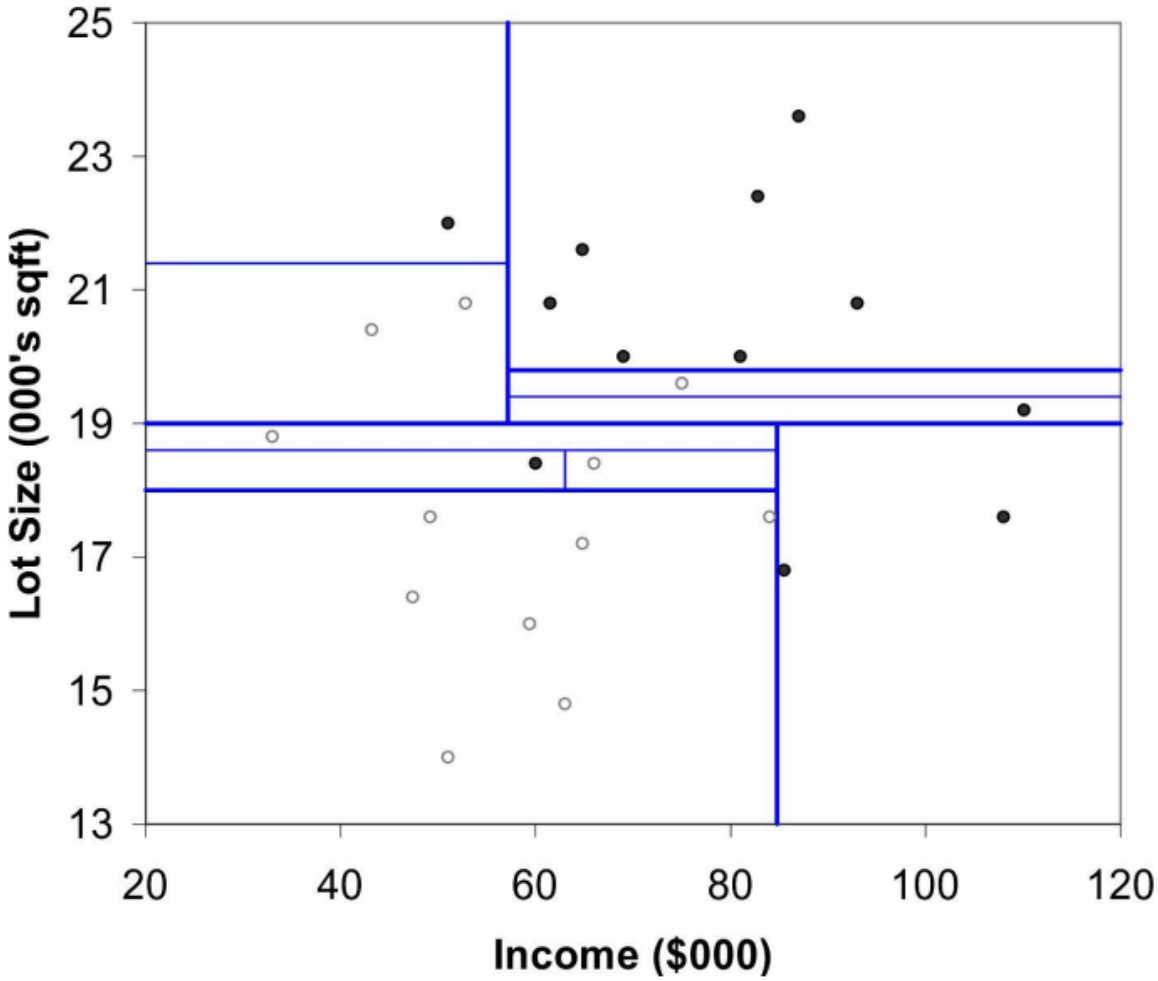
두 번째 분할: Income = $84,000



Decision Tree for Lawn Owner [3/4]

참고자료

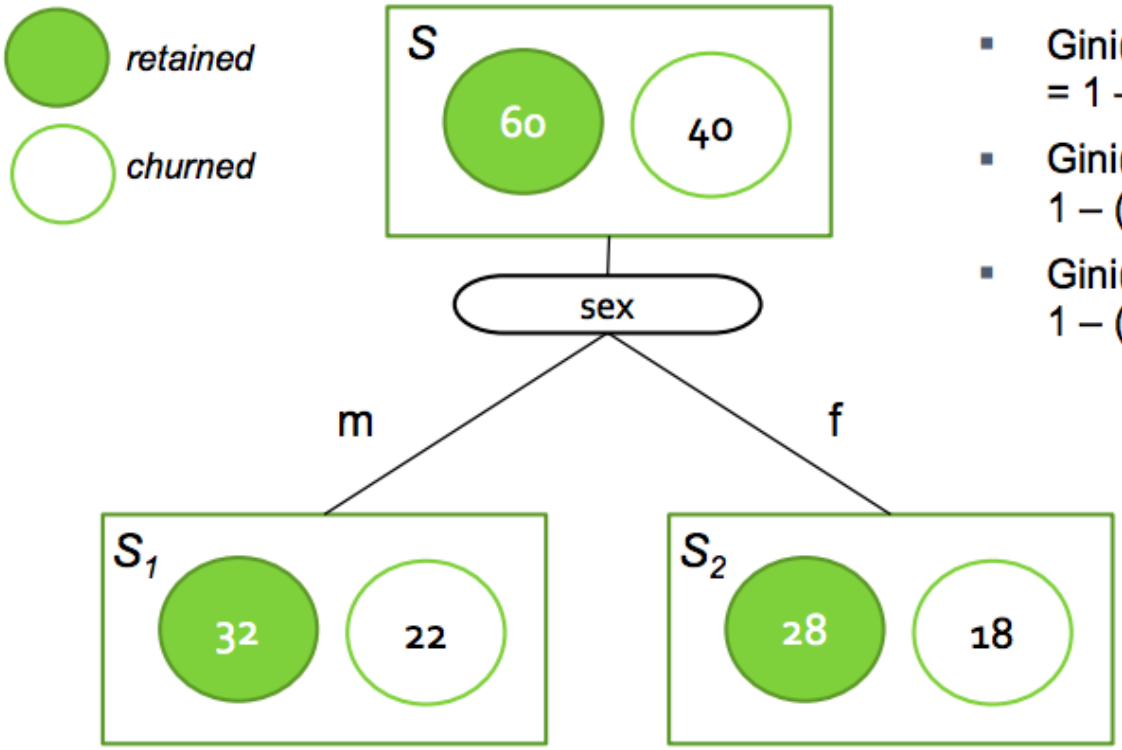
모든 분할 후



Decision Tree for Lawn Owner [4/4]

참고자료

- Compute *Gini measures* and the *information gain*.



- Gini(S) = 1 – ((60/100)^2 + (40/100)^2) = 1 – (0.36 + 0.16) = 1 – 0.52 = 0.48
- Gini(S_1) = 1 – ((32/54)^2 + (22/54)^2) = 1 – (0.35 + 0.17) = 1 – 0.52 = 0.48
- Gini(S_2) = 1 – ((28/46)^2 + (18/46)^2) = 1 – (0.37 + 0.15) = 1 – 0.52 = 0.48

Information Gain

= purity(S) – (0.54*purity(S_1) + 0.46*purity(S_2))

= 0.48 – 0.48 = 0

Finding Best Splits

- Categorical attributes (with no meaningful order):
  - Multi-way split, one child for each value
  - Binary split: try all possible breakup of values into two sets, and pick the best

- Continuous-valued attributes (can be sorted in a meaningful order)
  - Binary split:
    - Sort values, try each as a split point
      - E.g., if values are 1, 10, 15, 25, split at ≤ 1, ≤ 10, ≤ 15
    - Pick the value that gives best split
  - Multi-way split:
    - A series of binary splits on the same attribute has roughly equivalent effect

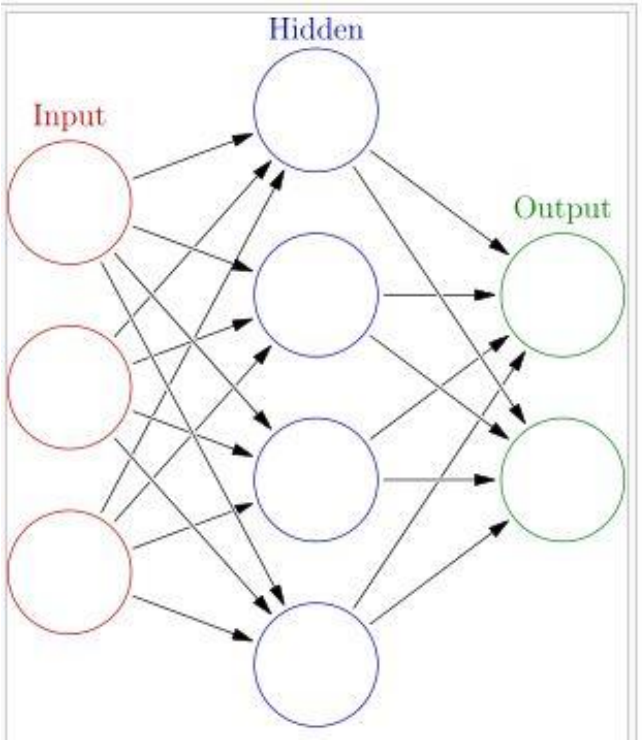
Decision-Tree Construction Algorithm

Procedure *GrowTree* (*S*)
  Partition (*S*);

Procedure Partition (*S*)
  if (*purity* (*S*) > δ_p or |*S*| < δ_s) **then**
    return;
  for each attribute *A*
    evaluate splits on attribute *A*;
  Use best split found (across all attributes) to partition
    S into $S_1, S_2, \ldots, S_r$,
  for *i* = 1, 2, ….., *r*
    Partition (S_i);

Neural Network Classifier

- Neural net classifiers are studied in artificial intelligence and are not covered here



An artificial neural network is an interconnected group of nodes, akin to the vast network of neurons in a brain. Here, each circular node represents an artificial neuron and an arrow represents a connection from the output of one neuron to the input of another.

Naïve Bayesian Classifiers [1/3]

Bayesian classifiers use **Bayes theorem**, which says $p(c_j \mid d) = \frac{p(d \mid c_j)\, p(c_j)}{p(d)}$

where $p(c_j \mid d)$ = probability of instance d being in class c_j,

$p(d \mid c_j)$ = probability of generating instance d given class c_j,

$p(c_j)$ = probability of occurrence of class c_j, and

$p(d)$ = probability of instance d occuring

- The class with the maximum probability becomes the predicated class of instance d
- Example: Suppose there are 2 classes Excellent and Good
  - The income bucket has values in the range (75000, 80000)
  - The probability of "Excellent" income being in (75000, 80000) is 0.1
  - The probability of "Good" income being in (75000, 80000) is 0.05
  - The overall 0.1 fraction of person are classified as Excellent
  - The overall 0.3 fraction of person are classified as Good
  - If John's income is 76000
    - $p(d \mid c_j)\, p(c_j)$ for class Excellent is 0.01
    - $p(d \mid c_j)\, p(c_j)$ for class Good is 0.015
    - So, John would be classified in class Good

Naïve Bayesian Classifiers [2/3]

- Bayesian classifiers require
  - computation of $p(d \mid c_j)$
  - precomputation of $p(c_j)$
  - $p(d)$ can be ignored since it is the same for all classes

- To simplify the task, **naïve Bayesian classifiers** assume attributes have independent distributions, and thereby estimate

$$p(d \mid c_j) = p(d_1 \mid c_j) * p(d_2 \mid c_j) * \ldots * (p(d_n \mid c_j)$$

  - Each of the $p(d_i \mid c_j)$ can be estimated from a histogram on d_i values for each class c_j
    - the histogram is computed from the training instances
  - Histograms on multiple attributes are more expensive to compute and store

Naïve Bayesian Classifiers [3/3]

- Divide the range of values of attribute i into equal intervals and store the fraction of instances of class c_j that fall in each interval

- Given a value d_i for attribute i, the value of $p\ (d_i \mid c_j)$ is simply the fraction belonging to class c_j that fall in the interval to which d_i belongs

- Class C1: Attribute A (10--20: 0.3, 20--30: 0.7), Attribute B (a--c: 0.2, d--f: 0.8)
  Class C2: Attribute A (10--20: 0.6, 20--30: 0.4), Attribute B (a--c: 0.7, d--f: 0.3)
  instance d (25, e) ➔ compute p(d, C1) and p(d, C2)

Naïve Bayesian Classifier Example [1/2]

Class:

C1:buys_computer = 'yes'

C2:buys_computer = 'no'

Data sample

X = (age <=30,

Income = medium,

Student = yes

Credit_rating = Fair)

age	income	student	credit_rating	com
<=30	high	no	fair	no
<=30	high	no	excellent	no
31…40	high	no	fair	yes
>40	medium	no	fair	yes
>40	low	yes	fair	yes
>40	low	yes	excellent	no
31…40	low	yes	excellent	yes
<=30	medium	no	fair	no
<=30	low	yes	fair	yes
>40	medium	yes	fair	yes
<=30	medium	yes	excellent	yes
31…40	medium	no	excellent	yes
31…40	high	yes	fair	yes
>40	medium	no	excellent	no

Naïve Bayesian Classifier Example [2/2]

- $P(C_i)$: P(buys_computer = "yes") = 9/14 = 0.643
  P(buys_computer = "no") = 5/14= 0.357
- Compute $P(X|C_i)$ for each class
  P(age = "<=30" | buys_computer = "yes") = 2/9 = 0.222
  P(age = "<= 30" | buys_computer = "no") = 3/5 = 0.6
  P(income = "medium" | buys_computer = "yes") = 4/9 = 0.444
  P(income = "medium" | buys_computer = "no") = 2/5 = 0.4
  P(student = "yes" | buys_computer = "yes) = 6/9 = 0.667
  P(student = "yes" | buys_computer = "no") = 1/5 = 0.2
  P(credit_rating = "fair" | buys_computer = "yes") = 6/9 = 0.667
  P(credit_rating = "fair" | buys_computer = "no") = 2/5 = 0.4
- **X = (age <= 30 , income = medium, student = yes, credit_rating = fair)**

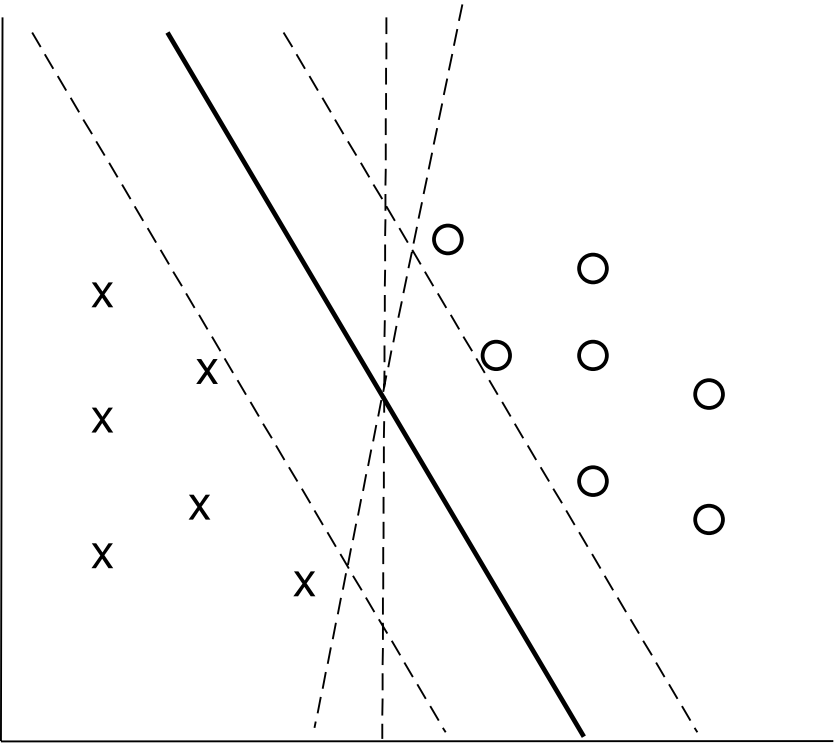
$P(X|C_i)$: P(X|buys_computer = "yes") = 0.222 x 0.444 x 0.667 x 0.667 = 0.044
P(X|buys_computer = "no") = 0.6 x 0.4 x 0.2 x 0.4 = 0.019

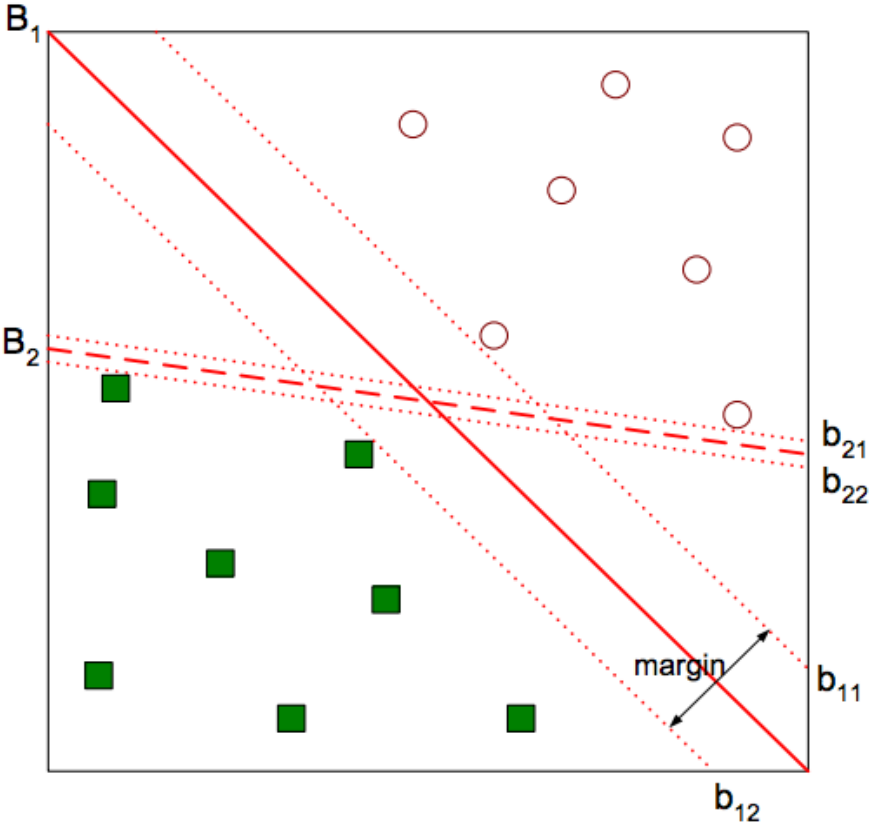
$P(X|C_i)*P(C_i)$: P(X|buys_computer = "yes") * P(buys_computer = "yes") = 0.028
P(X|buys_computer = "no") * P(buys_computer = "no") = 0.007

Therefore, X belongs to class ("buys_computer = yes")

Support Vector Machine Classifier

- The original SVM algorithm was invented by V.N. Vapnik and A.Y. Chervonenkis in 1963
- The current standard incarnation (soft margin) was proposed by C.Cortes and V.N. Vapnik in 1993 and published in 1995



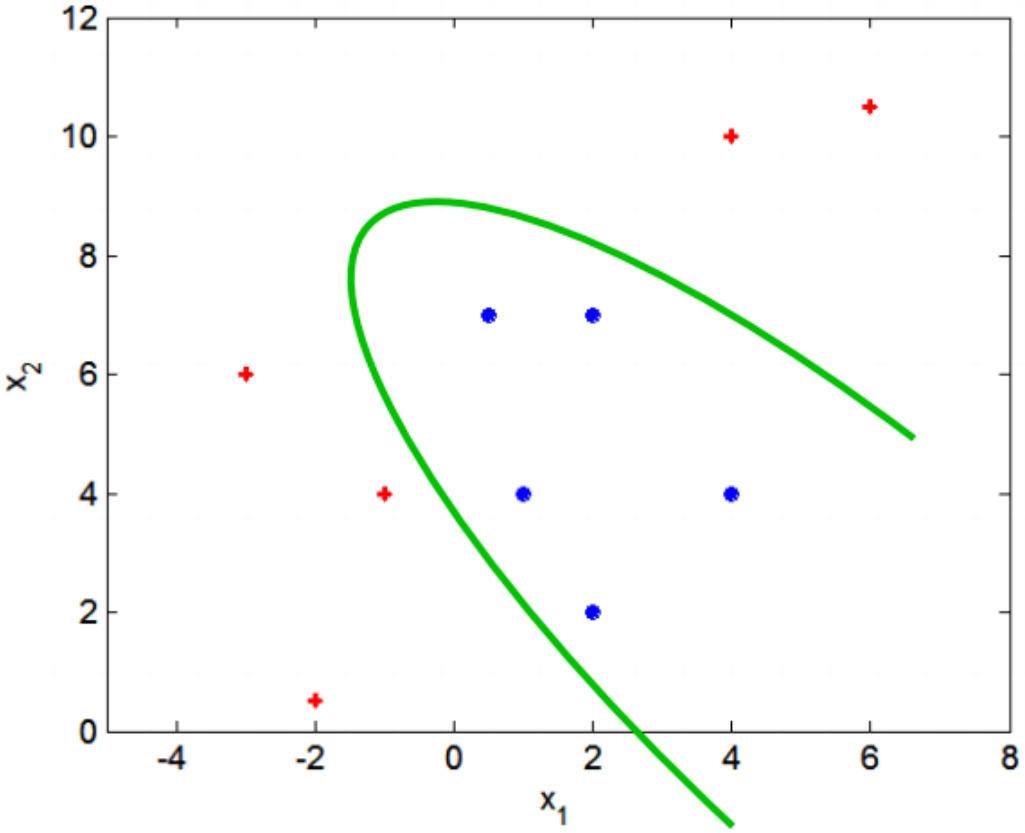


Find hyperplane maximizes the margin => B1 is better than B2

SVM can find nonlinear curves using kernel functions [1/2]

참고자료

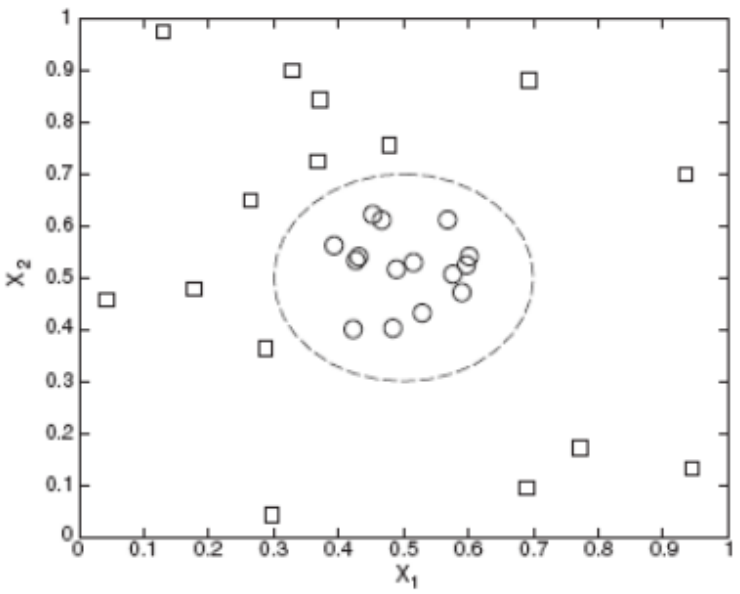
- What if decision boundary is not linear?



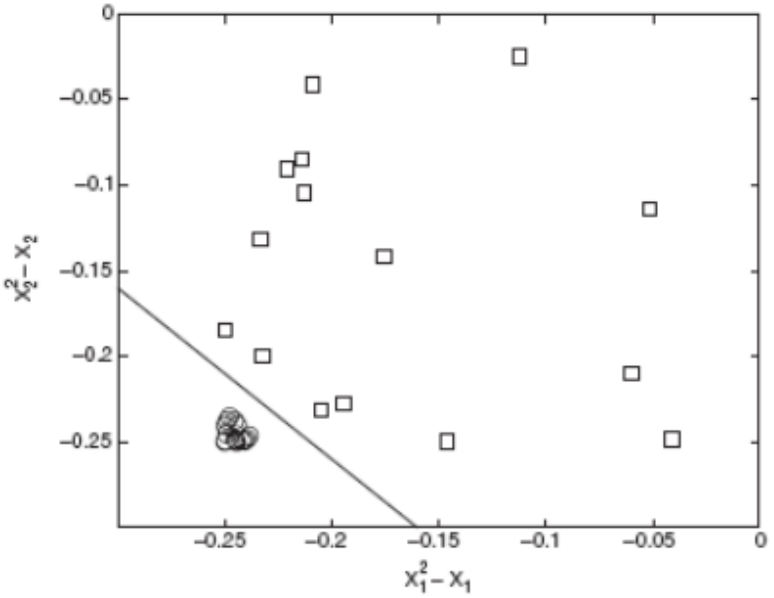
SVM can find nonlinear curves using kernel functions [2/2]

참고자료

- Nonlinear SVM



(a) Decision boundary in the original two-dimensional space.

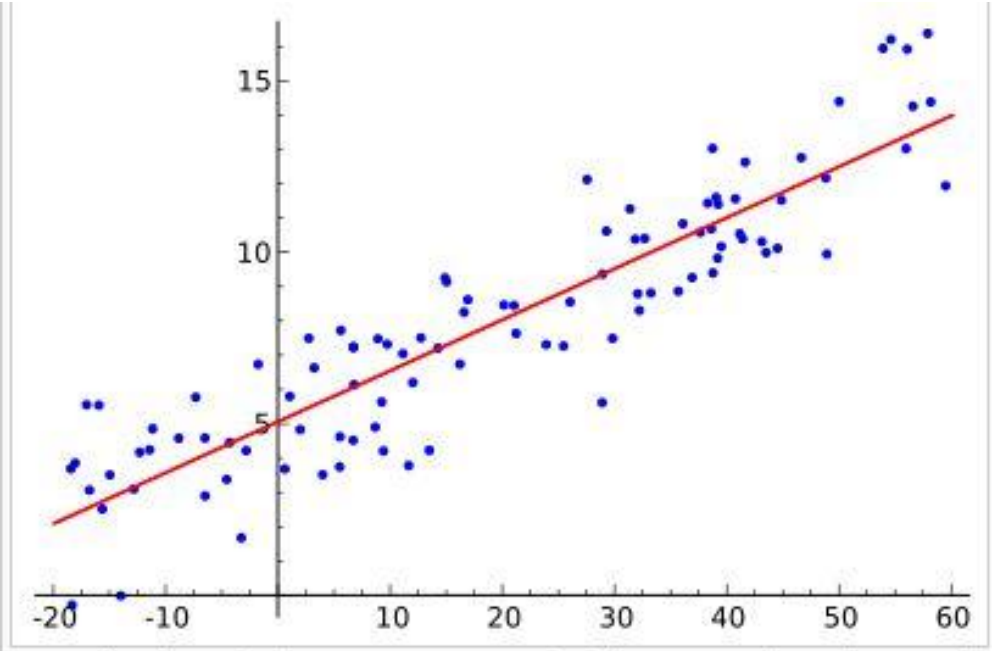


(b) Decision boundary in the transformed space.

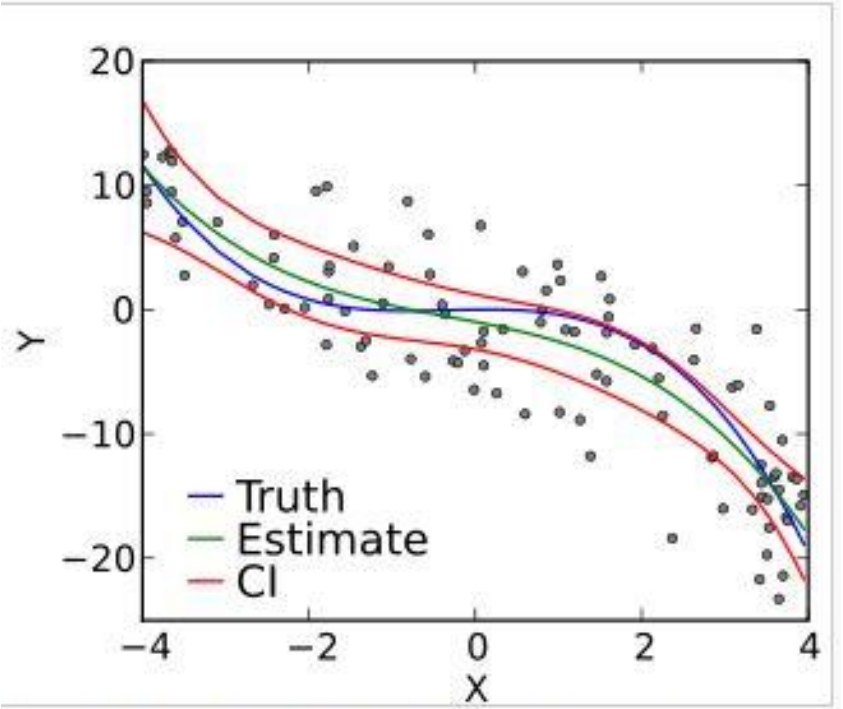
Regression Classifier

- Regression deals with the prediction of a value, rather than a class.
  - Given values for a set of variables, X_1, X_2, …, X_n, we wish to predict the value of a variable Y
- One way is to infer coefficients a_0, a_1, a_1, …, a_n such that
  $$Y = a_0 + a_1 * X_1 + a_2 * X_2 + \ldots + a_n * X_n$$
- Finding such a linear polynomial is called **linear regression**.
  - In general, the process of finding a curve that fits the data is also called **curve fitting**
- The fit may only be approximate
  - because of noise in the data, or
  - because the relationship is not exactly a polynomial
- Regression aims to find coefficients that give the best possible fit
  - Standard techniques in statistics

Regression Example



Example of simple linear regression, which has one independent variable



Example of a cubic polynomial regression, which is a type of linear regression.

Chapter 20: Data Analysis

- 20.1 Decision Support Systems
- 20.2 Data Warehousing
- 20.3 Data Mining
- 20.4 Classification
- 20.5 Association Rules
- 20.6 Other Types of Associations
- 20.7 Clustering
- 20.8 Other Forms of Data Mining

Association Rules [1/2]

- Retail shops are often interested in associations between different items that people buy
  - Someone who buys bread is quite likely also to buy milk
  - A person who bought the book *Database System Concepts* is quite likely also to buy the book *Operating System Concepts*
- Associations information can be used in several ways
  - E.g., when a customer buys a particular book, an online shop may suggest associated books
- **Association rules:**

  bread $\Rightarrow$ *milk* *DB-Concepts, OS-Concepts* $\Rightarrow$ Networks

  - Left hand side: **antecedent**, right hand side: **consequent**
  - An association rule must have an associated **population**; the population consists of a set of **instances**
    - E.g., each transaction (sale) at a shop is an instance, and the set of all transactions is the population

Association Rules [2/2]

- Rules have an associated support, as well as an associated confidence
- **Support** is a measure of what fraction of the population satisfies both the antecedent and the consequent of the rule
  - E.g., suppose only 0.001 percent of all purchases include milk and screwdrivers
    - The support for the rule is *milk* $\Rightarrow$ *screwdrivers* is low
- **Confidence** is a measure of how often the consequent is true when the antecedent is true
  - E.g., the rule *bread* $\Rightarrow$ *milk* has a confidence of 80 percent if 80 percent of the purchases that include bread also include milk

- If A $\Rightarrow$ B
  - Support (A) = count(A) / population
  - Support of rule = count (*A union B) / population*
  - Confidence of rule = support (*B*) / support (*A*)

Finding Association Rules

- We are generally only interested in association rules with reasonably high support (e.g., support of 2% or greater)
- Naïve algorithm
  1. Consider all possible sets of relevant items
  2. For each set find its support (i.e., count how many transactions purchase all items in the set)
     - **Large itemsets**: sets with sufficiently high support
  3. Use large itemsets to generate association rules.
     1. From itemset A generate the rule $A - \{b\} \Rightarrow b$ for each $b \in A$.
        if the rule has sufficient confidence
        - Support of rule = support (A).
        - Confidence of rule = support (A) / support ($A - \{b\}$)

Finding Support

- Determine support of itemsets via a single pass on set of transactions
  - Large itemsets: sets with a high count at the end of the pass
- If memory not enough to hold all counts for all itemsets use multiple passes, considering only some itemsets in each pass
- Optimization: Once an itemset is eliminated because its count (support) is too small none of its supersets needs to be considered
- Given a, b, c: counts would be incremented for {a}, {b}, {c}, {a,b}, {b,c}, {a,c}, {a,b,c}
- The **a priori technique** to find large itemsets:
  - Pass 1: count support of all sets with just 1 item. Eliminate those items with low support
  - Pass *i*: **candidates**: every set of *i* items such that all its *i-1* item subsets are large
    - Count support of all candidates
    - Stop if there are no candidates
- Apriori Algorithm
  - 1994 VLDB – Fast Algorithms for Mining Association Rules by R. Agrawal and R. Srikant

Apriori Algorithm for Association Rule

- Find all large itemsets
  - itemsets with minimum support
- Use the large itemsets to generate desired rules
  - (ex) ABCD and AB are large itemsets
  - conf = support(ABCD) / support(AB)
  - If conf >= minconf
  - AB ➔ CD

- Discovering Large Itemsets
  - Any subset of large itemset is large
  - To find large k-itemset
    - Create candidates by combining large (k-1) itemsets
    - Delete those that contain subset that is not large

Apriori Algorithm Example [1/2]

$Sup_{min} = 2$

Database TDB

Tid	Items
10	A, C, D
20	B, C, E
30	A, B, C, E
40	B, E

1st scan →

C_1

Itemset	sup
{A}	2
{B}	3
{C}	3
{D}	1
{E}	3

L_1

Itemset	sup
{A}	2
{B}	3
{C}	3
{E}	3

C_2

Itemset
{A, B}
{A, C}
{A, E}
{B, C}
{B, E}
{C, E}

2nd scan ←

C_2

Itemset	sup
{A, B}	1
{A, C}	2
{A, E}	1
{B, C}	2
{B, E}	3
{C, E}	2

L_2

Itemset	sup
{A, C}	2
{B, C}	2
{B, E}	3
{C, E}	2

C_3

Itemset
{B, C, E}

3rd scan →

L_3

Itemset	sup
{B, C, E}	2

참고자료

Apriori Algorithm Example [2/2]

- Apriori Candidate Generation
  - 1. Join Step

insert into C_k

select $p.item_1, p.item_2, p.item_{k-1}, q.item_{k-1}$

from $L_{k-1}p, L_{k-1}q$

where $p.item_1 = q.item_1, \ldots, p.item_{k-2} = q.item_{k-2}, p.item_{k-1} < q.item_{k-1}$

  - 2. Prune Step
    - Check all the subsets, remove a candidate with “small” subset

- Example

L_3 = { {1 2 3}, {1 2 4}, {1 3 4}, {1 3 5}, {2 3 4} }

After joining

{ {1 2 3 4}, {1 3 4 5} }

After pruning

{1 2 3 4}

Chapter 20: Data Analysis

- 20.1 Decision Support Systems
- 20.2 Data Warehousing
- 20.3 Data Mining
- 20.4 Classification
- 20.5 Association Rules
- 20.6 Other Types of Associations
- 20.7 Clustering
- 20.8 Other Forms of Data Mining

Other Types of Associations [1/2]

- Basic association rules have several limitations
- Deviations from the expected probability are more interesting
  - E.g., if many people purchase bread, and many people purchase cereal, quite a few would be expected to purchase both
  - We are interested in **positive** as well as **negative correlations** between sets of items
    - Positive correlation: co-occurrence is higher than predicted
    - Negative correlation: co-occurrence is lower than predicted

- Limitation : A=>B 에서 B가 원래 frequent한 것일 수 있음
- Confidence (콜라 => 햄버거) high
- Confidence (다이어트콜라 => 햄버거) high
- P(햄버거) 가 높아서 별 의미가 없음

Other Types of Associations [2/2]

- **Sequence associations / sequence correlations**
  - Sequence data = Time series data
  - E.g., whenever bonds go up, stock prices go down in 2 days
- **Deviations from temporal patterns**
  - E.g., deviation from a steady growth
  - E.g., sales of winter wear go down in summer
    - Not surprising, part of a known pattern
  - Look for deviation from value predicted using past patterns

Chapter 20: Data Analysis

- 20.1 Decision Support Systems
- 20.2 Data Warehousing
- 20.3 Data Mining
- 20.4 Classification
- 20.5 Association Rules
- 20.6 Other Types of Associations
- 20.7 Clustering
- 20.8 Other Forms of Data Mining

Clustering

- Clustering: Intuitively, finding clusters of points in the given data such that similar points lie in the same cluster
- Group points into *k* sets (for a given *k*) using distance metrics
  - the average distance of points from the centroid of their assigned group is minimized
  - the average distance between every pair of points in a cluster is minimized
- Known as K-means clustering algorithm
  - Has been studied extensively in statistics, but on small data sets

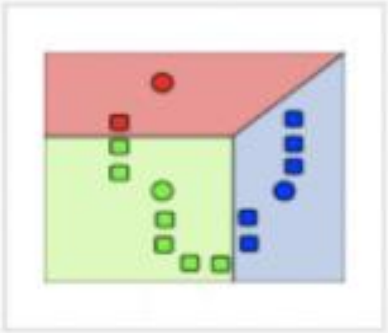
- Data mining systems aim at clustering techniques that can handle very large data sets
  - E.g., the Birch clustering algorithm (more shortly)

참고자료

K-means [1/3]

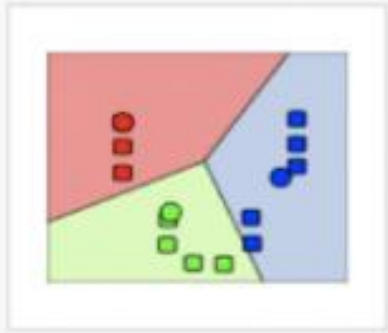
- Partition the input points into k initial random generates clusters
- Build a new partition by associating each point with the closest centroid
- Computes the mean point, or centroid, of each set
- Repeat these two steps until convergence

Shows the initial randomized point and a number of points.



Points are associated with the nearest initial randomized point.

Now the initial randomized points are moved to the center of their respective clusters(the centroids).



Steps 2 & 3 are repeated until a suitable level of convergence has been reached.

참고자료

K-means [2/3]

Pro

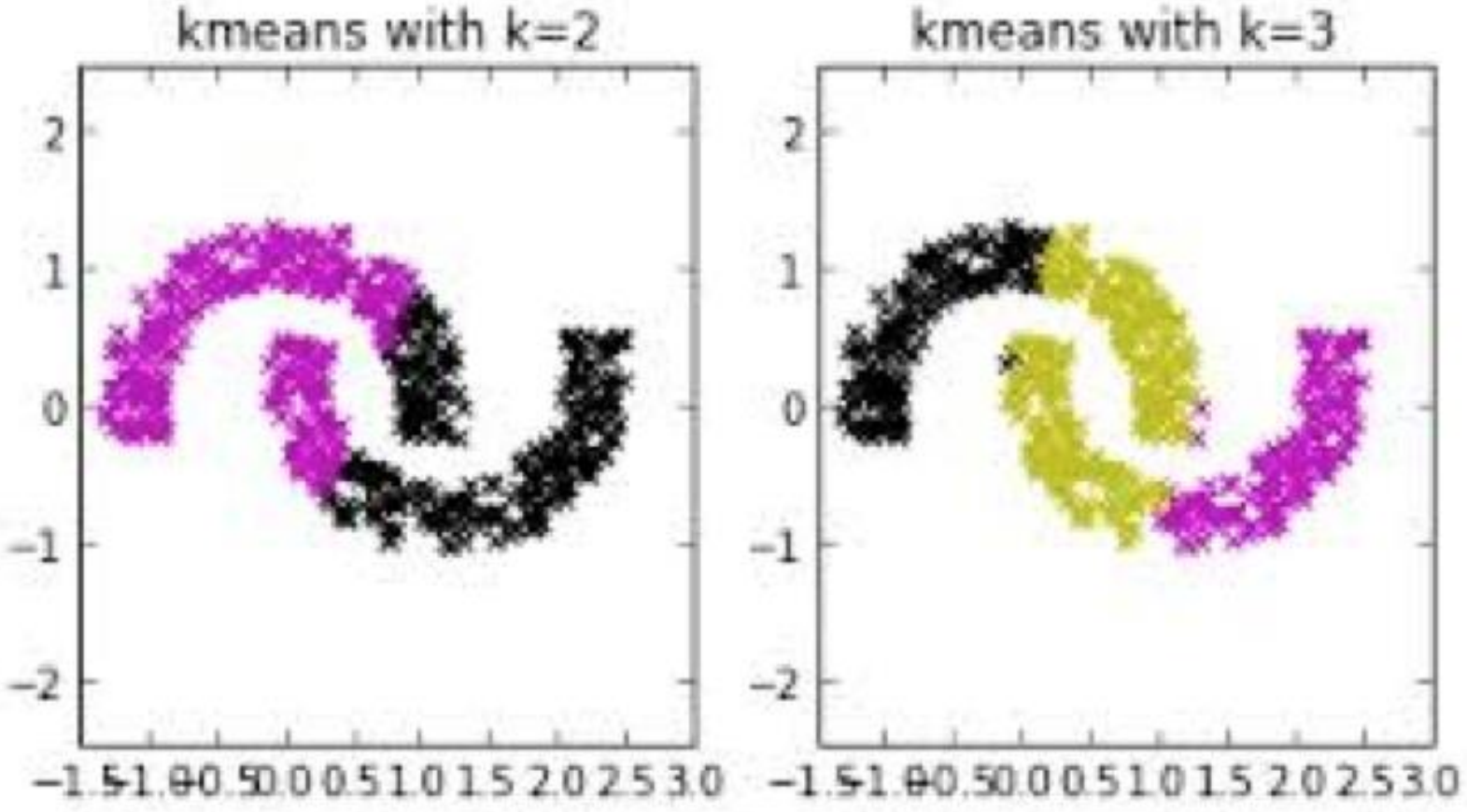
- Simple
- Fast

Cons

- does not yield to the same result with each run
- need to choose a priori the number of clusters
- it is not guaranteed to find the optimal configuration
  - Trick: place the first centroid on a data point, place the second centroid on a point that is far away as possible from the first one and so on...

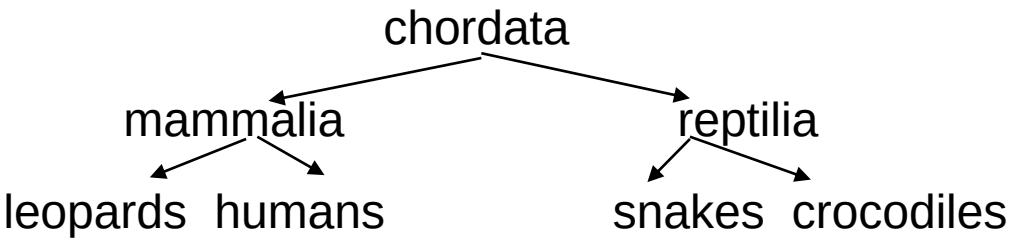
참고자료

K-means [3/3]



Hierarchical Clustering

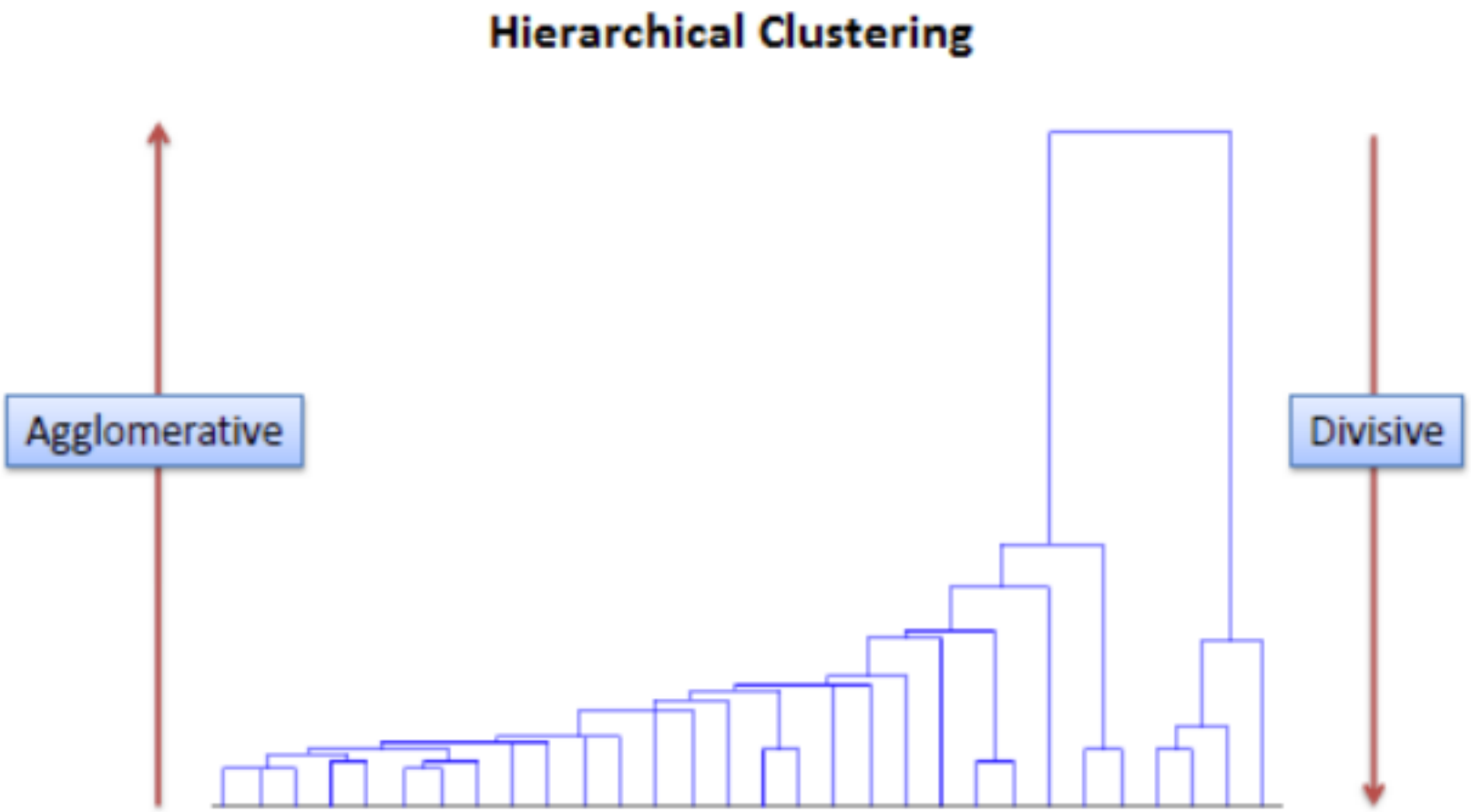
- Example from biological classification
  - (the word classification here does not mean a prediction mechanism)



- Other examples: Internet directory systems (e.g., Yahoo, more on this later)
- **Agglomerative (덩어리가 되는) clustering algorithms**
  - Build small clusters, then cluster small clusters into bigger clusters, and so on
- **Divisive clustering algorithms**
  - Start with all items in a single cluster, repeatedly refine (break) clusters into smaller ones

참고자료

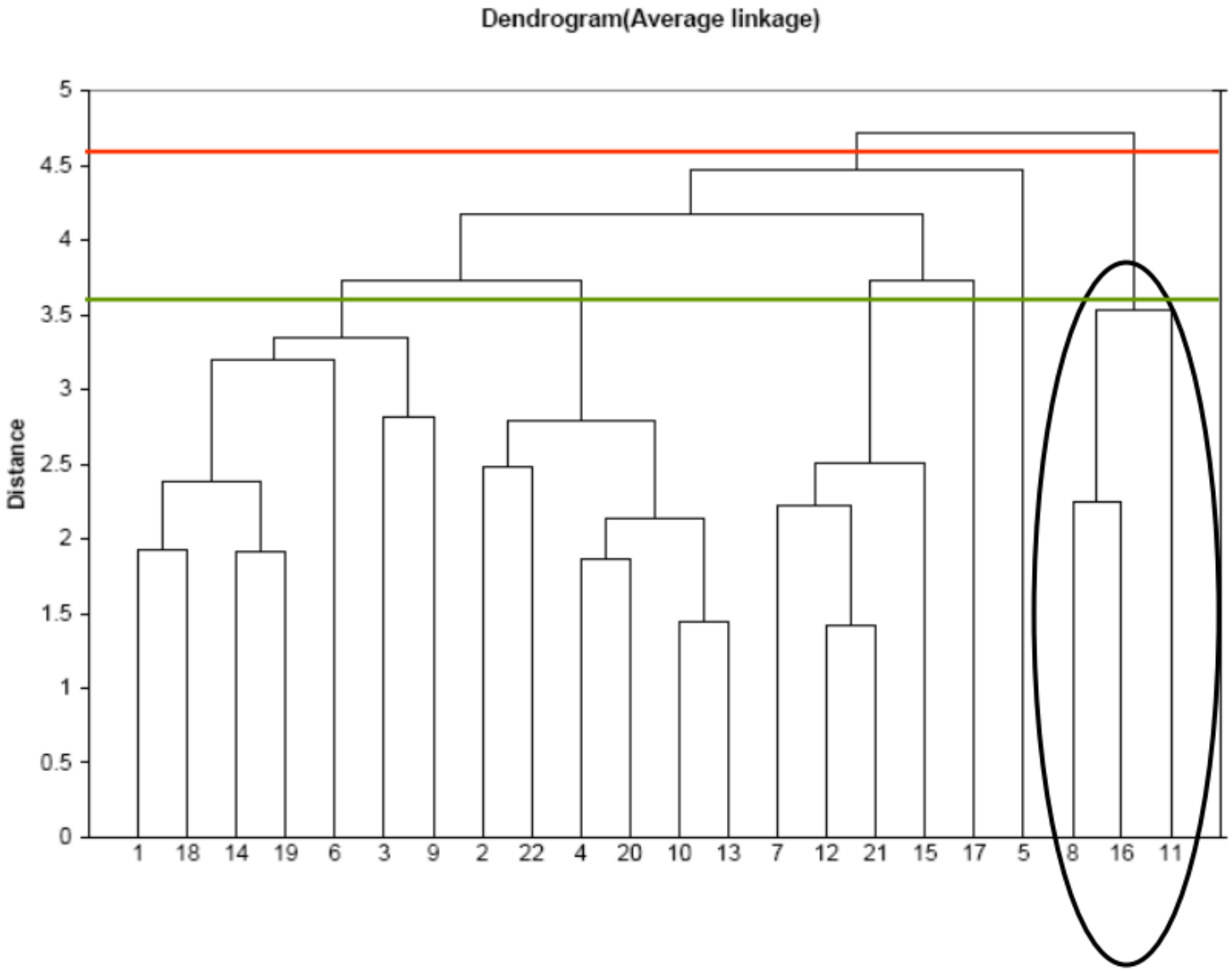
Hierarchical Clustering Example [1/2]



참고자료

Hierarchical Clustering Example [2/2]

- Determining Number of Clusters



Large Data Clustering

- Clustering algorithms have been designed to handle very large datasets
- **Birch Clustering Algorithm by** Zhang, Ramakrishnan, and Livny (1996 Sigmod)

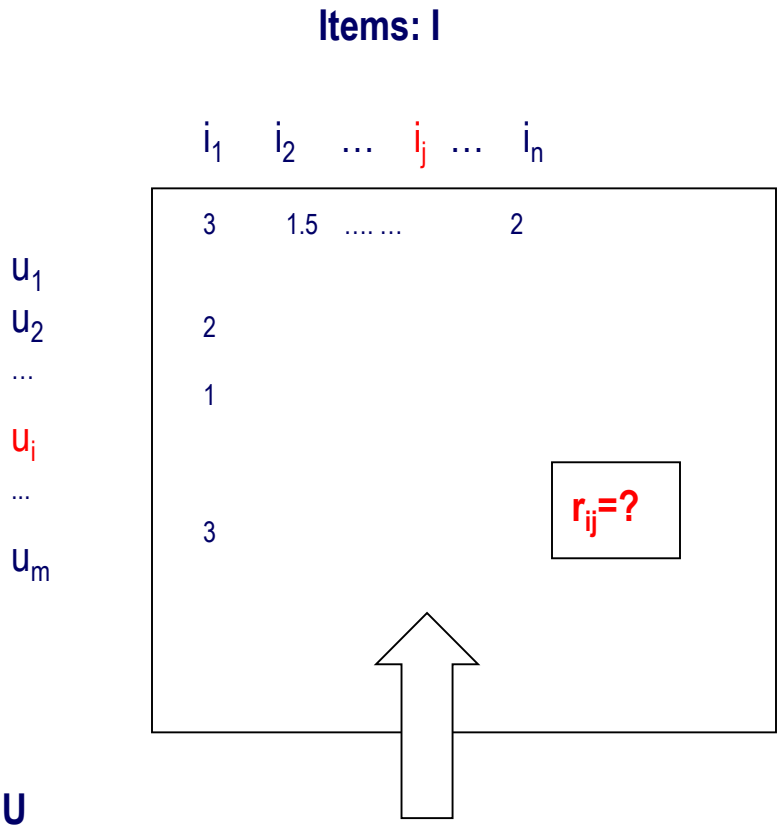
  (balanced iterative reducing and clustering using hierarchies)
  - Main idea: use an in-memory R-tree to store points that are being clustered
  - Insert points one at a time into the R-tree, merging a new point with an existing cluster if is less than some δ distance away
  - If there are more leaf nodes than fit in memory, merge existing clusters that are close to each other
  - At the end of first pass we get a large number of clusters at the leaves of the R-tree
    - Merge clusters to reduce the number of clusters

DBSCAN

Collaborative Filtering

- Goal: predict what movies/books/… a person may be interested in, on the basis of
  - Past preferences of the person
  - Other people with similar past preferences
  - The preferences of such people for a new movie/book/…
- One approach based on repeated clustering
  - Cluster people on the basis of preferences for movies
  - Then cluster movies on the basis of being liked by the same clusters of people
  - Again cluster people based on their preferences for (the newly created clusters of) movies
  - Repeat above till equilibrium
  - Suppose 4 persons & 5 movies
    - P1(m1,m2, m5), P2(m2, m4), P3(m1, m5), P4(m2)
- Above problem is an instance of **collaborative filtering**, where users collaborate in the task of filtering information to find information of interest

Collaborative Filtering: A Framework



The task:

Q1: Find Unknown ratings?

Q2: Which items should we recommend to this user?

.

.

.

Chapter 20: Data Analysis

- 20.1 Decision Support Systems
- 20.2 Data Warehousing
- 20.3 Data Mining
- 20.4 Classification
- 20.5 Association Rules
- 20.6 Other Types of Associations
- 20.7 Clustering
- 20.8 Other Forms of Data Mining

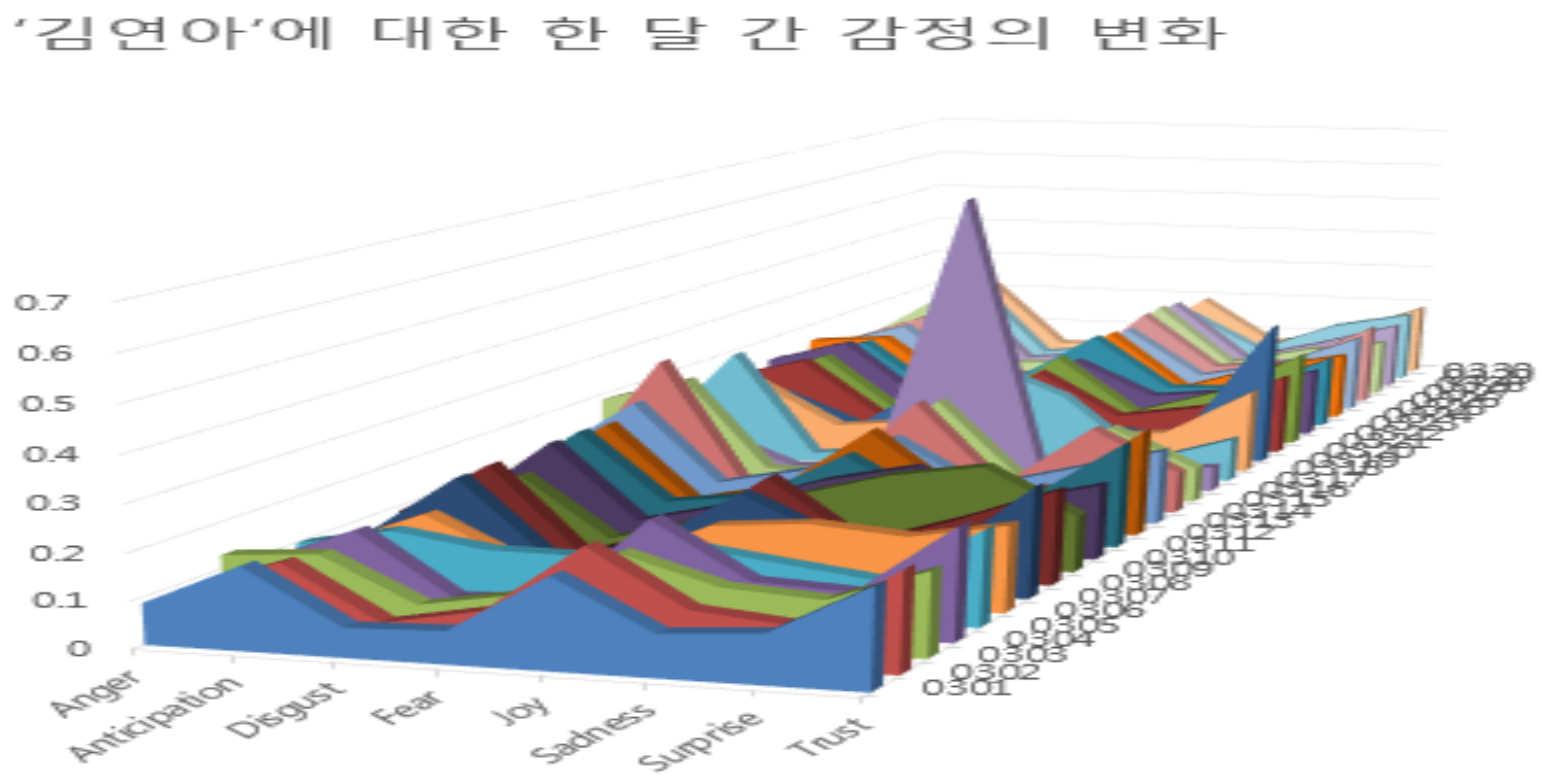
Other Forms of Data Mining

- **Text mining**: application of data mining to textual documents
  - cluster Web pages to find related pages
  - cluster pages a user has visited to organize their visit history
  - classify Web pages automatically into a Web directory

- **Data visualization** systems help users examine large volumes of data and detect patterns visually
  - Can visually encode large amounts of information on a single screen
  - Humans are very good a detecting visual patterns

Text Mining in Twitter

- 2013년 3월 동안 '김연아' 관련 트윗 분석 (Sentiment Analysis)
  - 임의로 선정된 토픽에 대해 단기일정기간 분석된 감정 패턴
  - 데이터 전처리, 감정 분석을 위한 전처리 등에 오랜 시간 소요



Spotfire by Tibco