Part 1

1 Introduction to Data Mining

1.1 Why data mining?

- Data explosion
- Rich data, but poor information

1.2 What is data mining?

- Discovering interesting patterns in huge amounts of data

1.3 Knowledge discovery from data (KDD)

1. Data cleaning: remove noise and inconsistent data
2. Data integration: where multiple data source may be combined
3. Data selection: where relevant data are retrieved from database.
4. Data transformation: where data are transformed into forms appropriate for mining by performing summary or aggregation operations.
5. Data mining: intelligent methods are applied to extract data patterns.
6. Pattern evaluation: identify interesting patterns.
7. Knowledge presentation: visualization and knowledge representation techniques are used to present mined knowledge to users.

1.4 Various views

- Data view: kinds of data to be mined
  - 3Vs, 4Vs and 5Vs: Volume, Variety, Velocity, Veracity, Value.
  - Database-oriented: relational database, data warehouse, transnational database.
  - Sequence, stream, temporal, time-series data
  - Spatial, spatial-temporal data
  - Text, multimedia, Web data
  - Graph, social networks data
- Knowledge view: kinds of knowledge to be discovered

  - Concept/class description
  - Frequent patterns, associations, correlations
  - Classification and prediction
  - Cluster, outlier and evolution analysis
- Method view: kinds of techniques utilized
- Application view: kinds of applications adapted

1.5 Major issues in data mining

- Mining technology: mining different knowledge from diverse data types
- User interaction
- Applications and social impacts

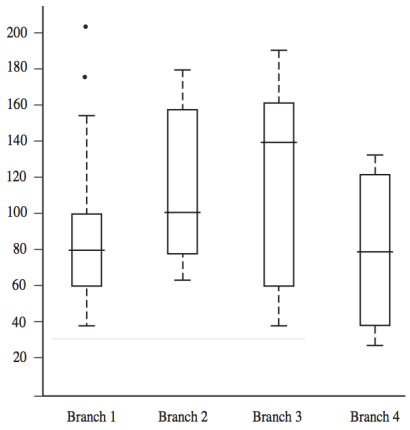
2 Getting to know your data

2.1 Data objects and attribute types

- Data object: an entity with certain attribute(s). Also called feature, dimension, variable.
- Attribute types:
  - Nominal (categorical): e.g. hair color, major, occupation.
  - Binary (boolean, symmetric or asymmetric): e.g. gender, smoker, disease
  - Ordinal: e.g. drink size, grade, professional rank
  - Numeric (quantitative):
    - Interval-scaled
    - Ratio-scaled
    - Discrete vs. continuous

2.2 Basic statistical description of data

- Central Tendency:
  - Mean: sample, weighted and trimmed mean.
  - Median: middle value if N is off, otherwise average value of the middle two values.
    - Estimation: $median = L_1 + (\frac{N/2+(\sum freq)_l}{freq_{median}})width$
  - Mode: value that occurs most frequently
    - $mean - mode = 3 * (mean - median)$
  - Midrange: average of min and max
- Data Dispersion:
  - Range: difference between mac and min
  - Interquartile range: $IQR = Q_3 - Q_1$
  - Five number summary: min, Q1, median, Q3, max
  - Outlier: value higher/lower than 1.5 * IQR of Q3/Q1
  - Variance: $s^2 = \frac{1}{N}\sum_{i=1}^{N}(x_i - \bar{x})^2$
  - Standard deviation: square root of variance.
  - Boxplot:



* Box: Q1, Median, Q3, IQR
* Whiskers: min, max, 1.5 * IQR

2.3 Data visualization

- Why data visualization?
  - Gain insights, qualitative overview and explore.
- Visualization methods:
  - Pixel-oriented
  - Geometric projection
  - Icon-based
  - Hierarchical
  - Visualizing complex data and relations

2.4 Measuring data similarity and dissimilarity

- Data martix: object-by-attribute (two modes)

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

- Dissimilarity matrix: object-by-object (one-mode)

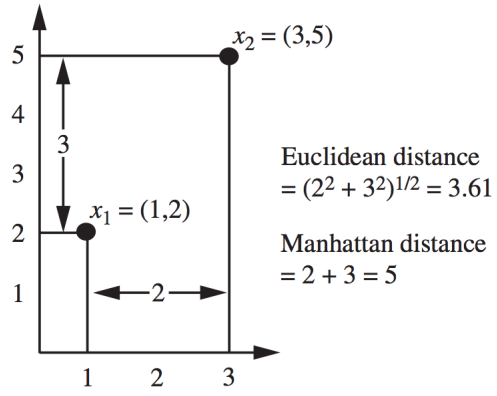
$$\begin{bmatrix} 0 & & & & \\ d(2,1) & 0 & & & \\ d(3,1) & d(3,2) & 0 & & \\ \vdots & \vdots & \vdots & & \\ d(n,1) & d(n,2) & \ldots & \ldots & 0 \end{bmatrix}$$

2.4.1 Object similarity/dissimilarity

- Minkowski distance (L_p norm): $d(i,j) = (|x_{i1} - x_{j1}|^p + |x_{i2} - x_{j2}|^p + ... + |x_{in} - x_{jn}|^p)^{1/p}$
- Euclidean distance (L_2 norm): $d(i,j) = \sqrt{(x_{i1} - x_{j1})^2 + (x_{i2} - x_{j2})^2 + ... + (x_{in} - x_{jn})^2)}$
- Manhattan distance (L_1 norm): $d(i,j) = |x_{i1} - x_{j1}| + |x_{i2} - x_{j2}| + ... + |x_{in} - x_{jn}|$
- Weighted distance

2.4.2 Distance measure

- Euclidean distance vs. Manhattan distance:
  - Properties:
    - $d(i,j) \geq 0$
    - $d(i,i) = 0$
    - $d(i,j) = d(j,i)$
    - $d(i,j) \leq d(i,k) + d(k,j)$ (triangular inequality)



- Nominal Attributes: e.g. hair color, occupation
  - Method 1: simple matching
    - $d(i,j) = (p - m)/p$
    - m: number of matches, p: total number of variables
  - Method 2: view each state as a binary variable

    * e.g. colors(red, green, yellow, blue); then (0, 1, 0, 0) means color green

- Binary variables:
  - Contingency table:

		obj_j	obj_j	
		1	0	sum
obj_i	1	q	r	q+r
obj_i	0	s	t	s+t
	sum	q+s	r+t	q+r+s+t

  - Distance measure for symmetric binary variables: $d(i,j) = \frac{r+s}{q+r+s+t}$
  - Distance measure for asymmetric binary variables: $d(i,j) = \frac{r+s}{q+r+s}$

3 Data Preprocessing

3.1 Overview

- Measure of data quality: accuracy, completeness, consistency, timeliness, believability, interpretability, accessibility

3.2 Data Cleaning

- Why data cleaning?
  - Imperfect data in real word
  - Incomplete: missing attributes
  - Containing errors or outliers
  - Containing discrepancies
- How to handle missing data?
  - Ignore to tuple
  - Fill in the missing value manually
  - Fill in it automatically with:
    - A global constant
    - The attribute mean
    - The most probably value (e.g. regression, Bayesian inference, decision tree)
- How to handle noisy data?
  - Binning:
    - First sort and partition data into bins (buckets)
    - Then smooth by: bin means, bin median, bin boundaries
  - Regression: fit data into regression function
  - Clustering: detect and remove outliers

3.3 Data Integration

- Definition: Combines data from multiple sources
- Entity identification:
  - Schema integration, object matching
  - E.g. customer id vs. customer number
  - Bill Clinton vs. William Clinton
- Redundant data:

  - Different naming, derived data
  - May be detected by correlation analysis
  - Correlation Analysis: related variables
    * Correlation coefficient (numerical data):
      · $r_{A,B} = \frac{\sum_{i=1}^{N}(a_i - \bar{A})(b_i - \bar{B})}{N_{\sigma A \sigma B}} = \frac{\sum_{i=1}^{N}(a_i b_i) - N\bar{A}\bar{B}}{N_{\sigma A \sigma B}}$
    * X^2 (chi-square) test (categorical data):
      · $X^2 = \sum_{i=1}^{c}\sum_{j=1}^{r}\frac{(o_{ij} - e_{ij})^2}{e_{ij}}$
      · $e_{ij} = \frac{count(A=a_i) * count(B=b_j)}{N}$
    * Correlation does not imply causality

3.4 Data Reduction

- Why data reduction?
  - Massive data sets
  - Mining takes a long time
- Goal of data reduction:
  - Data set is much smaller in volume
  - produces almost the same mining results

`Data reduction strategies:`

- Dimensionality reduction:
  - attribute subset selection
  - Principal component analysis (PCA): given N data vectors of n dimensions, find $k \leq n$ orthogonal vectors that can best represent the data
- Numerosity reduction:
  - Parametric methods: 1. Assume the data fits some model, 2. Estimate model parameters, 3. Store the parameters and discard the data
    * Regression and long-linear models
  - Non-parametric methods: do not assume models
    * Histograms: divide data into buckets and store average or sum for each bucket.
    * Clustering: partition data intro cluster based on similarity
    * Sampling: use a small sample to represent whole data, then choose a representative subset of the data.
- Data compression:
  - String compression
  - Audio/video compression

3.5 Data Transformation and Discretization

3.5.1 Transformation

- Smoothing: remove noise from data.
- Aggregation: summarization (e.g. daily sales => monthly, annual sales)
- Generalization: concept hierarchy climbing (e.g. street => city => state)
- Normalization: scale to fill within a range
  - Min-max normalization: $v' = \frac{v - min_A}{max_A - min_A}(new\_max_A - new\_min_A) + new\_min_A$
  - Z-score normalization: $v' = \frac{v - mean}{stdev}$
  - Normalization by decimal scaling: $v' = \frac{v}{10^j}$ where j is the smallest integer s.t. $MAX(|v'|) < 1$
    - e.g. range [-986, 917], $j = 3$ because divide by 1000
- Attribute/feature construction: new attributes constructed from existing ones.

3.5.2 Discretization

- Three types of attributes: nominal(unordered est), ordinal(ordered set), continuous(integer or real number)
- Discretization: divide continuous range into intervals
  `Methods:`
- Binning: split, unsupervised(means: how you split it does not depend on data)
- Histogram analysis: split, unsupervised
- Clustering analysis: split/merge, unsupervised
- Entropy-based discretization: split, supervised
- Interval merging by X^2 analysis: merge, supervised
- Intuitive partitioning: split, unsupervised

4 Data Warehouse and OLAP

4.1 What is a Data Warehouse?

- A decision support database that is maintained `separately` from an organization's operational database.
- Support `information processing` by providing a solid platform of consolidated, historical data for analysis.
- A data warehouse is a `subject-oriented`, `integrated`, `time-variant`, and `nonvolatile` collection of data in support of management's decision-making process.

4.2 Subject-Oriented

- Organized around major subjects (e.g. customers, product, sales)
- Focus on the modeling and analysis of data decision making, not on daily operations or transaction processing.
- Provide a simple and concise view around particular subject issues by excluding data that are not useful in the decision support process.

4.3 Integrated

- Integrate multiple, heterogeneous data sources.
- Data cleaning and data integration techniques applied.

4.4 Time-Variant

- Significantly longer time span
  - Operational database: current data
  - Data warehouse: historical perspective (e.g. past 5-10 years)
- Every key structure in a data warehouse contains time info, explicitly or implicitly.

4.5 Nonvolatile (not changed)

- A physically separate store of data transformed from operational environments.
- No operational update of data
- Only two operations in data accessing: initial loading of data, access of data.

4.6 Data warehouse vs. Operational database

- OTLP (on-line transaction processing): major task of traditional relational DBMS
- OLAP (on-line analytical processing): major task of data warehouse system

	OLTP	**OLAP**
users	clerk, IT professional	knowledge worker
function	day to day operations	decision support
DB design	application-oriented	subject-oriented
data	current, up-to-date detailed, flat relational isolated	historical, summarized, multidimensional integrated, consolidated
usage	repetitive	ad-hoc
access	read/write index/hash on prim. key	lots of scans
unit of work	short, simple transaction	complex query
# records accessed	tens	millions
#users	thousands	hundreds
DB size	100MB-GB	100GB-TB
metric	transaction throughput	query throughput, response

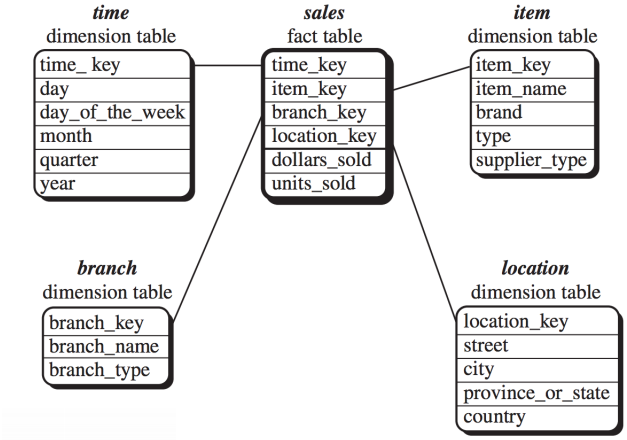
5 Data Cube

5.1 What is a Data Cube?

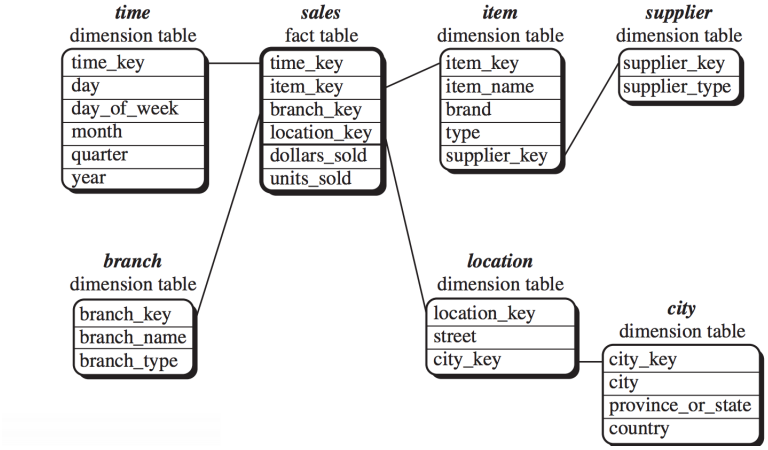
- Allow data to be modeled and views in multiple dimensions (e.g. sales)
- Dimensions: e.g. time, item, branch, location
- Facts: numerical measures (e.g. sold items, sold dollars

5.2 Conceptual Modeling

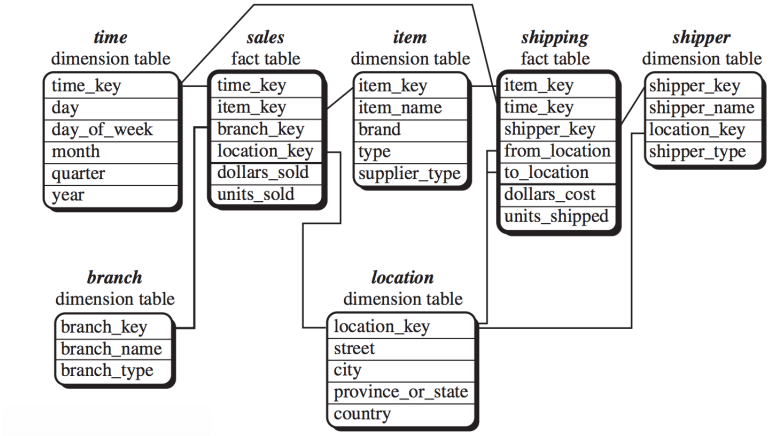
- Star schema: a fact table, a set of dimension



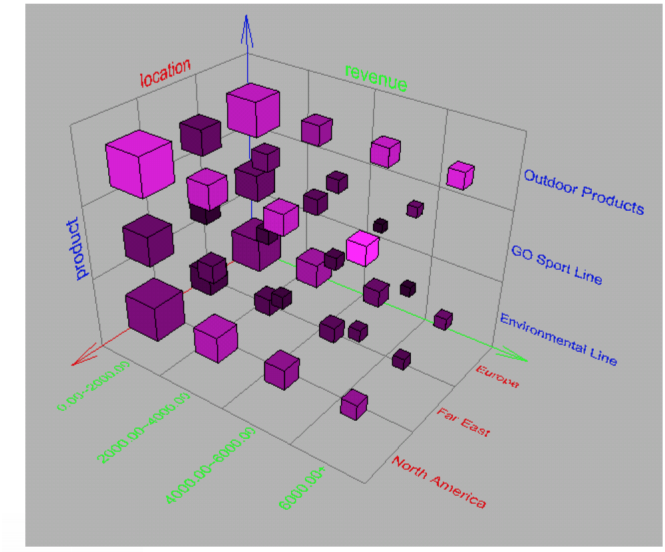
- Snowflake schema: a fact table, a hierarchy of dimension tables



- Fact constellations: multiple fact tables share dimension tables



5.3 Browsing a Data Cube



5.4 Typical OLAP Operations

- Roll-up (drill-up): summerization
- Drill-down: reverse of roll-up
- Slice and dice: project and select (sub-cube)
- Pivot (rotate): visualization, 3D or 2Ds
- Drill-across: more than one fact tables
- Drill-through: to the back-end relational tables

Part 2

6 Mining Frequent Patterns, Associations & Correlations

6.1 Market Basket Analysis

- Example: Which items are frequently purchased together by my customers?

6.2 Frequent Pattern Analysis

- Frequent patterns in a data set: a set of items, subsequences, substructures.
- Frequent item set: $X = \{x_1, x_2, ..., x_k\}$
- Association rule: $X \rightarrow Y$
  - Support: probability that a transaction contains $X \cup Y$
  - Confidence: conditional probability that a transaction containing X also contains Y
- Mining association rules:
  - Two-step process:
    - Find all frequent itemsets (w/ minimum support)
    - Generate strong association rules from the frequent itemsets (min support, min confidence)
  - A long pattern contains a combinatorial number of subpatterns
    - E.g. 100 items: $\binom{100}{1} + \binom{100}{2} + ... + \binom{100}{100} = 2^{100} - 1$
- Closed & Max Patterns
  - Solution: mine closed patterns & max-patterns
  - Closed pattern X: no super-pattern $Y \supset X$ w/ the same support
  - Max-pattern X: no super-pattern $Y \supset X$
  - Closed pattern is a lossless compression of frequent patterns
    - Reducing the number of patterns and rules

6.3 Apriori Algorithm

- Apriori property: subset of a freq. itemset is also frequent
- Apriori pruning: if X is infrequent, then superset of X is pruned.
- Procedure:
  - Scan DB to get frequent 1-itemset.

- Generate candidate (k+1)-itemsets from frequent k-itemsets.
- Test candidate (k+1)-itemsets against DB.
- Stop when no freq. or candidate itemsets can be generated.

- Important details:
  - Self-joining of k-itemsets to generate (k+1)-itemsets
    - Two k-itemsets are joined if their first (k-1) items are the same.
  - Pruning: remove if subset not frequent.

6.4 Interestingness Measure

- Association rule: $A \rightarrow B$ [support, confidence]
- A strong association rule: play basketball $\rightarrow$ eat cereal [40%, 66.7%]
- The rule is misleading:
  - Overall, 75% of students eat cereal.
  - play basketball $\rightarrow$ not eat cereal [20%, 33.3%]

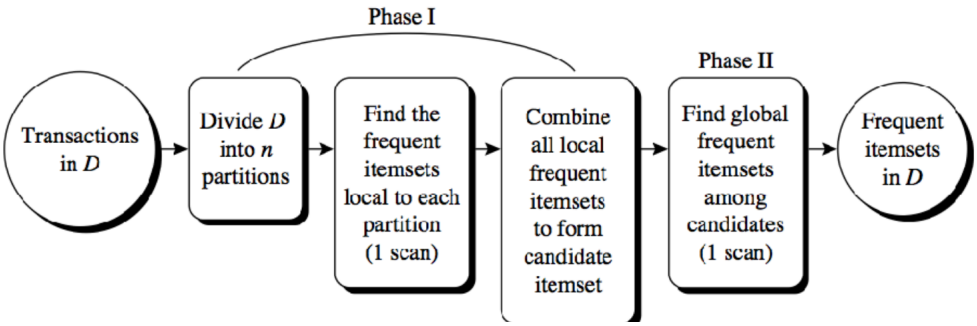
6.5 Correlation Rules

- $A \rightarrow B$ [support, confidence, correlation]
- Measure of dependent/correlated events: $lift(A, B) = \frac{P(A \cup B)}{P(A)P(B)}$
- lift = 1? No correlation
- lift < 1? Negatively dependent
- lift >1? Positively dependent

6.6 Improving Apriori

6.6.1 Reduce data scans

- Partition: Two data scans



- Sampling for frequent patterns:
  - Select a sample dataset.
  - Mine frequent patterns within sample.
    - May use a lower min support
  - Scan whole dataset for actual support
    - Only check closed patterns
    - E.g. check abcd instead of ab, abc, ..., etc.
  - Scan again to fins missed frequent patterns.
- Transaction reduction:
  - If a transaction T does not contain any frequent k-itemset, then for any h > k, no need to check T when searching for frequent h-itemset.

6.6.2 Reduce number of candidates

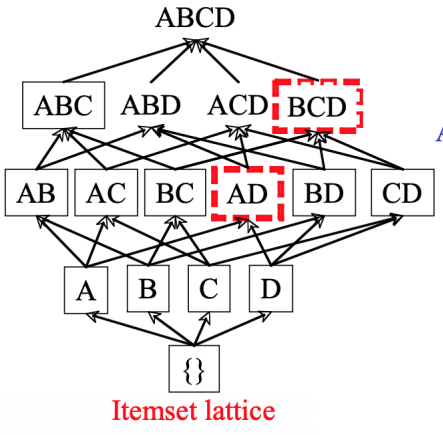
- Hash items to buckets
  - If a hash bucket count is below support threshold, then itemsets in that hash bucket are not frequent itemsets.

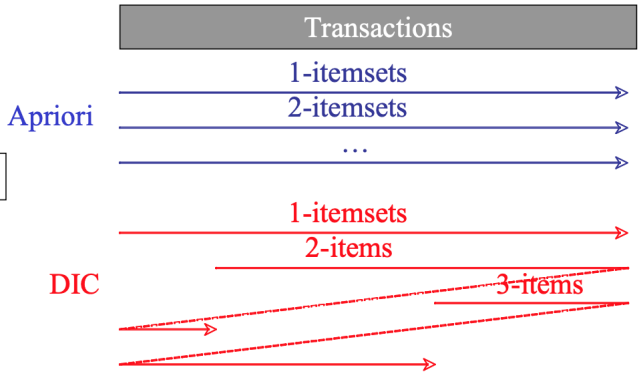
Create hash table H_2 using hash function $h(x, y) = ((order\ of\ x) \times 10 + (order\ of\ y))\ mod\ 7$

H_2

bucket address	0	1	2	3	4	5	6
bucket count	2	2	4	2	2	4	4
bucket contents	{I1, I4} {I3, I5}	{I1, I5} {I1, I5}	{I2, I3} {I2, I3} {I2, I3} {I2, I3}	{I2, I4} {I2, I4}	{I2, I5} {I2, I5}	{I1, I2} {I1, I2} {I1, I2} {I1, I2}	{I1, I3} {I1, I3} {I1, I3} {I1, I3}

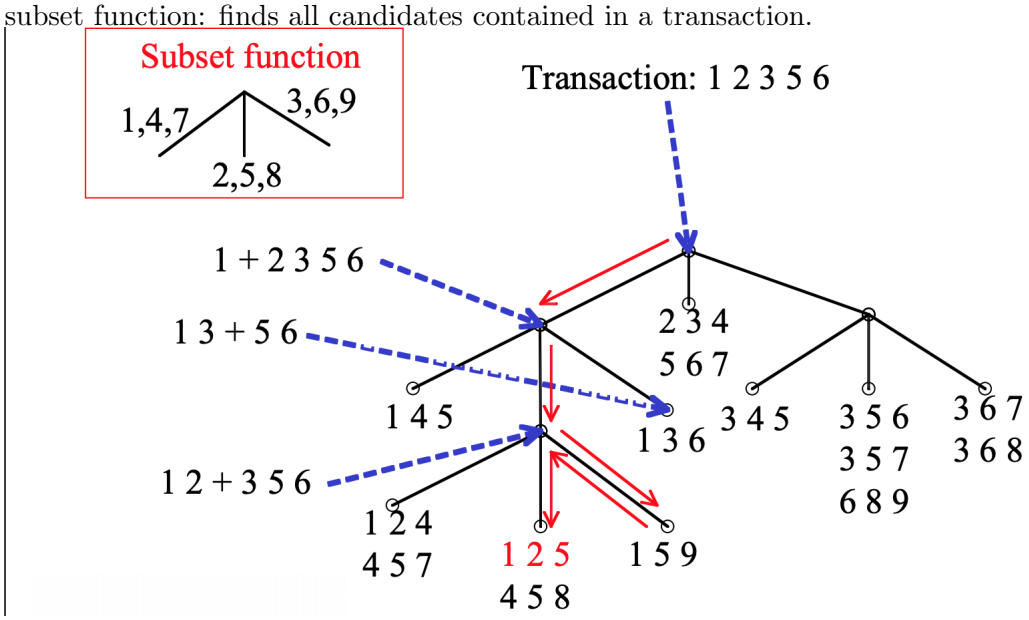
- Dynamic itemset counting
  - If A and D are frequent, start count for AD
  - If BC, BD, CD are frequent, start count for BCD





6.6.3 Count support of candidates

- First method:
  - Store candidate itemsets in a hash-tree
  - leaf-node contains a list of itemsets and counts.
  - interior node contains a hash table
  - subset function: finds all candidates contained in a transaction.



- Second method: vertical data format
  - Example: organize by datasets not be transactions
    - Horizontal data format: T1:{A, D, E, F}
    - Vertical data format: t(AD) = {T1, T6, ...}
  - Derive closed patter bia vertical intersection
    - t(X) = {T1, T2, T3} and t(Y) = {T1, T3, T3}
    - t(XY) = {T1, T3}

6.7 FP-growth

- Find frequent itemsets without candidate generation
- Grow long patterns from short ones using local frequent items
- Example:
  1. `abc` is a frequent itemset
  2. get all transactions with abc: DB | abc
  3. `d` is a local frequent item in DB | abc
  4. then `abcd` is a frequent itemset

7 Advanced Pattern Mining

7.1 Road Map

- Kinds of patterns: set, sequential, structural
- Completeness: all, closed, maximal, constrained, approximate, near-match, top-k
- Levels of abstraction:
  - Computer → printer
  - Laptop → HP printer (more specific)
- Number of data dimensions
  - Computer → printer
  - (age:30-39, income:42K-38K) → HDTV
- Types of value
  - Boolean: presence or absence
  - Quantitative: e.g. age, income
- Types of rules
  - Association, correlation, gradient

7.2 Multi-level association rules

- Support: uniform, reduced, group-based
- Redundancy filtering:
  - Milk → wheat bread [8%, 70%]
  - 2% milk → wheat bread [2%, 72%]

7.3 Multi-dimensional association

- Single-dimensional (intra-dimensional) rules:
  - buys(X, "milk") → buys(X, "bread")
- Multi-dimensional rules: ≥ predicates
  - Inter-dimensional (no repeated predicates)
    - age(X, "19-25") ∧ occupation(X, "student") → buys(X, "coke")
  - Hybrid-dimensional (repeated predicates)
    - age(X, "19-25") ∧ buys(X, "popcorn") → buys(X, "coke")

7.4 Categorical vs. Quantitative

- Categorical attributes
  - Nominal, finite number of possible values, no ordering among values
  - E.g. occupation, brand, color
- Quantitative attributes
  - Numeric, implicit ordering among values
  - E.g. age, income, price

7.5 Mining Quantitative Association

- Techniques categorized by how numerical attributes (e.g. age, salary) are treated
  - Static discretization: predefined concepts
  - Dynamic discretization: data distribution
  - Clustering: distance-based association
  - Deviation: from normal data
    - Sex=female $\rightarrow$ wage:mean=\$7/hr
    - Overall mean = \$9/hr

7.6 Constraint-based mining

- User flexibility: proved constrains on what to be mined.
- System optimization: more efficient mining
- Constraints in data mining:
  - Knowledge type constraint
  - Data constraint
  - Dimension/level constraint
  - Interestingness constraint
  - Rules (or pattern) constraint

7.7 Metarule-guided mining

- $P_1 \wedge P_2 \wedge ... \wedge P_a \rightarrow Q_1 \wedge Q_2 \wedge ... \wedge Q_n$
- n = a + b, find all n-predicate sets L_n
- Compute the support of all a-predicate subsets of L_n
- Compute the confidence of rules

7.8 Anti-monotonicity

- If an itemset S violates the constraint, so does any of its supersets.

7.9 Monotonicity

- If an itemset S satisfies the constraint, so does any of its supersets.

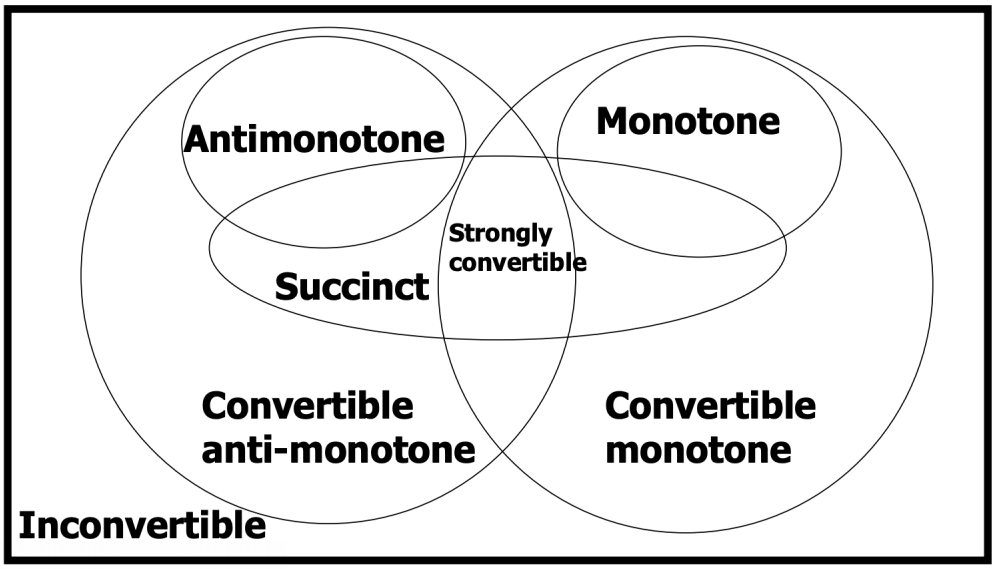
7.10 Succinctness

- Enumerate all and only those sets that are guaranteed to satisfy the constraint

7.11 Convertible constraints

- Convert tough constraints into int-monotonic or monotonic by properly ordering items

7.12 Classification of constrains



8 Classification: Basic Concepts

8.1 Basic Concepts

8.1.1 Classification vs. Prediction

- Classification
  - Determines categorical class labels (categorical values)
  - Raise the question: is the classification correct or not?
  - E.g. safe vs. risky, weather condition(sunny, cloudy, etc)
- Prediction
  - Models continuous-values functions (numerical values)
  - Raise the question: how close is your prediction?
- Typical applications:
  - Loan approval, target marketing, medical diagnosis, fraud detection, etc.

8.1.2 Classification

- Step 1: Learning
  - Model construction
  - Training set
  - Class labels
- Step 2: Classification result
  - Test set
  - Accuracy

8.1.3 Supervised vs. Unsupervised

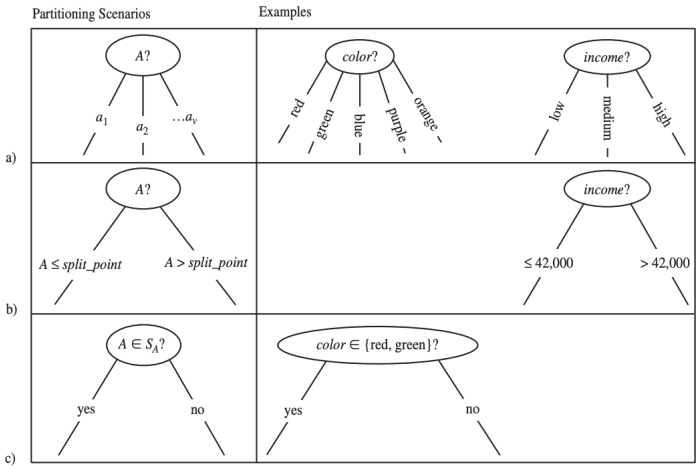
- Supervised learning (classification)
  - Supervision: training data accompanied by class labels
  - New data is classified based on training
- Unsupervised learning (clustering)
  - Class labels of training data is unknown
  - Aims to establish the existence of classes or clusters in the data

8.1.4 Issues: Evaluation Criteria

- Accuracy: classification vs. prediction
- Speed: time to construct / use the model
- Robustness: handling noise and missing values
- Scalability: large amounts of data
- Interpretability: understanding and insight
- Goodness of rules: E.g. decision tree size, compactness of classification rules

8.2 Decision Tree Induction

- Basic algorithm (a greedy algorithm)
  - Top-down, recursive, divide-and-conquer
  - Attribute selection
  - Attribute split
    a Discrete-valued
    b Continuous-valued: split_point
    c Discrete-valued: binary tree, splitting_subset



- Stopping conditions
  - All samples belong to the same class
  - No remaining attributes: majority voting
  - No samples left
- Attribute selection measures:
  - Information gain

    * D, m classes C_i: $p_i = |C_{i,D}|/|D|$
    * Expected information (entropy) needed to classify D: $Info(D) = -\sum_{i=1}^{m} p_i log_2(p_i)$
    * Information needed to classify D using A $(a_1, a_2, ..., a_v)$: $Info_A(D) = \sum_{j=1}^{v} \frac{|D_j|}{|D|} \times Info(D_j)$
    * Information gain: $Gain(A) = Info(D) - Info_A(D)$
    * Continuous-values attribute A
        · `Determine the best split point for A`
        · Sort A values in increasing order
        · Consider the midpoint of adjacent values: $(a_i + ai + 1)/2$
        · Pick the midpoint with minimum Info$_A$(D)
        · `Split`
        · D1:A $\leq$ split point; D2:A $>$ split point
  - Gain Ratio
    * Information gain measure biased toward attributes with a large number of values (e.g. customerID, productID)
    * Select attribute with maximum gain ratio
        · $SplitInfo_A(D) = -\sum_{j=1}^{v} \frac{|D_j|}{|D|} \times log_2(\frac{|D_j|}{|D|})$
        · $gainRatio(A) = \frac{Gain(A)}{SplitInfo(A)}$
  - Gini Index
    * $Gini(D) = 1 - \sum_{i=1}^{m} p_i^2$
    * Binary split: using attribute A
        · $Gini_A(D) = \frac{|D_1|}{|D|} Gini(D_1) + \frac{|D_2|}{|D|} Gini(D_2)$
    * Reduction in impurity
        · $\Delta Gini(A) = Gini(D) - Gini_A(D)$
    * Select attribute with largest impurity reduction
  - Comparison of the tree measures:
    * Information gain: multi-valued attributes
    * Gain ratio: unbalanced splits
    * Gini index: multi-values, equal-sized and pure partition, not good when number of classes is large

- Overfitting and Tree Pruning
  - Overfitting of the training data
    * Too many branches, reflect anomalies due to noise or outliers
    * Poor accuracy for unseen data
  - Tree pruning to avoid overfitting
    * Pre-pruning: halt tree construction early
    * Post-pruning: remove branches from a "fully-grown" tree

8.3 Bayesian classification

- Basics
  - A statistical classifier: predicts class membership probabilities
  - Foundation: based on Bayes' Theorem
  - Performance(naive Bayesian classifier): comparable to decision tree and some neural network classifiers
  - Incremental
- Bayes' Theorem
  - $P(H|X) = \frac{P(X|H)P(H)}{P(X)}$
- Naive Bayesian Classifier
  - $X = (x_1, x_2, ..., x_n)$
  - m classes: $C_1, C_2, ..., C_m$
  - Classification: maximal $P(C_i|X)$
  - Based on Bayes' Theorem: $P(C_i|X) = \frac{P(X|C_i)P(C_i)}{P(X)}$
  - Since P(X) is constant for all classes, only need to maximize $P(X|C_i)P(C_i)$
  - Naive assumption: class conditional independence
    - $P(X|C_i) = \prod_{k=1}^{n} P(X_k|C_i) = P(x_1|C_i) \times P(x_2|C_i) \times ... \times P(x_n|C_i)$
  - If A_k is categorical, $P(x_k|C_i)$
  - If A_k if continuous-valued, assume Gaussian distribution, $P(x_k|C_i) = g(x_k, \mu_{C_i}, \sigma C_i)$
    - $g(x, \mu, \sigma) = \frac{i}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$
  - Advantages
    - Easy to compute
    - Good results in most cases
  - Disadvantages
    - Assumption: class conditional independence
    - Dependencies exist in practice

8.4 Rule-based classification

- IF-THEN rules
  - Assessment of a rule
    - coverage(R) = $n_{covers}/|D|$
    - accuracy(R) = $n_{correct}/n_{covers}$
- Rules-based classification
  - Rule R is triggered (precondition satisfied)

  - R is the only rule triggered
  - No rule is triggered
  - More than one rule is triggered
    * Size ordering: most attribute tests
    * Class-based ordering: importance (e.g. prevalence, misclassification cost)
    * Rule-based ordering: (decision list) priority list ordered by rule quality or by experts
- Rule extraction
  - From a decision tree
  - Each root to leaf path
  - Leaf: class predication
  - Rules are exhaustive and mutually exclusive
- Rule quality measures
  - Consider both coverage and accuracy
  - number of positive or negative tuple covered by a rules
    * $FOIL\_Gain = pos' \times (log_2 \frac{pos'}{pos'+neg'} - log_2 \frac{pos}{pos+neg})$
  - Observed frequency vs expected frequency
    * $Likelihood\_Ratio = 2 \sum_{i=1}^{m} f_i log(\frac{f_i}{e_i})$
  - Rule pruning: $FOIL\_Prune(R) = \frac{pos-neg}{pos+neg}$

8.5 Model evaluation and selection

- Classifier accuracy measures
  - Partition: training data and testing data
  - Accuracy, recognition rate
  - Error rate, misclassification rate
  - Confusion matrix
  - Sensitivity: t_pos / pos
  - Specificity: t_neg / neg
  - Precision: t_post / (t_pos + f_pos)
  - $accuracy = sensitivity \frac{pos}{pos+neg} + specificity \frac{neg}{pos+neg}$
- Classifier/Predictor evaluation
  - Holdout, random sampling
  - Cross-validation
    * Divide into k subsamples
    * Use k-1 subsamples for training, one for testing
  - Bootstrapping

        * Sample with replacement → training data

- Model selection
  - Choose between two models M_1 and M_2
  - Mean error rate: estimated error on future data
  - Difference between error rates of M_1 and M_2
  - T-test
    * $t = \frac{\bar{err}(M_1) - \bar{err}(M_2)}{\sqrt{var(M_1 - M_2)/k}}$
    * $var(M_1 - M_2) = \frac{1}{k}\sum_{i=1}^{k}[err(M_1)_i - err(M_2)_i - (\bar{err}(M_1) - \bar{err}(M_2))]^2$

8.6 Improve classification accuracy

- Ensemble methods
  - Use a combination of multiple model to increase accuracy
  - Popular ensemble methods: bagging and boosting
- Bagging: bootstrap aggregation
  - Analogy: diagnosis by multiple doctors
  - Training: given a dataset D of d tuples
    * Training set D_i: d tuples sampled randomly with replacement
    * A classifier M_i is learned for D_i
  - Classification: majority vote
  - Predication: average of multiple predictions
- Boosting
  - Analogy: diagnosis by multiple doctors
    * weighted by previous diagnosis accuracy
  - Weights are assigned to each training tuple
  - A series of k classifiers is iteratively learned
  - After classifier M_i learned, adjust weights so M_{i+1} pays more attention to tuples that were misclassified by M_i
  - M* combines the votes of all k classifiers, weighted by individual accuracy
- Adaboost
  - D: $(X_1, y_1), (X_2, y_2), ..., (X_d, y_d)$
  - Initial weight of each tuple: I/d
  - Round i (i=1, ..., k)
    * D_i: sample d tuples with replacement from D
    * Pr(choose tuple$_j$ based on tuple$_j$'s weight

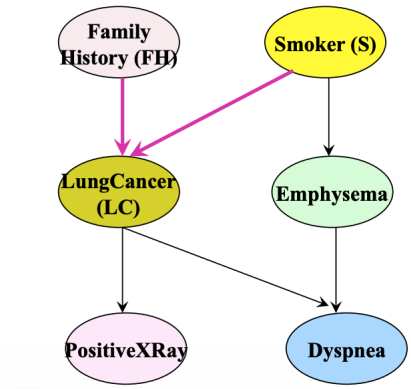
        * Learn M_i from D_i compute its error rate
            · $error(M_i) = \sum_{j=1}^{d} w_j \times err(X_j)$
        * Reduce weights of correctly classified tuples: $w_j = w_j \times \frac{error(M_i)}{1-error(M_i)}$
        * Normalize tuples weights so sum is 1
            · E.g. (0.1, 0.3, 0.1) → (0.2, 0.6, 0.2)
        * Classification: weighted votes for k classifiers
            · $weight(M_i) = log\frac{i-error(M_i)}{error(M_i)}$

- Ensemble methods: accuracy
    - Bagging
        * Often significantly better than single classifier
        * Noise: note considerably worse, more robust
        * Prediction: proved improved accuracy
    - Boosting
        * Generally better than bagging
        * May overfit the model to misclassified data

9 Classification: Advanced Methods

9.1 Bayesian belief networks

- Subset of variables conditionally independent (creating a network that captures the dependency of variables)
- Causal model: a directed, acyclic graph
  - node, link, parent, CPTs



CPT: Conditional Probability Table for variable LungCancer:

	(FH, S)	(FH, ~S)	(~FH, S)	(~FH, ~S)
LC	0.8	0.5	0.7	0.1
~LC	0.2	0.5	0.3	0.9

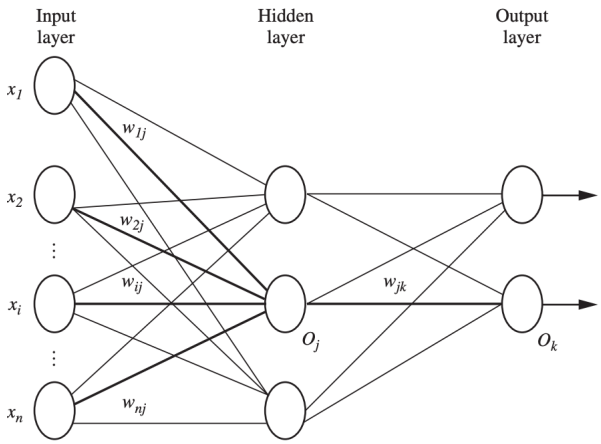
$$P(x_1, \cdots, x_n) = \prod_{i=1}^{n} P(x_i | Parents(Y_i))$$

9.2 Neural networks

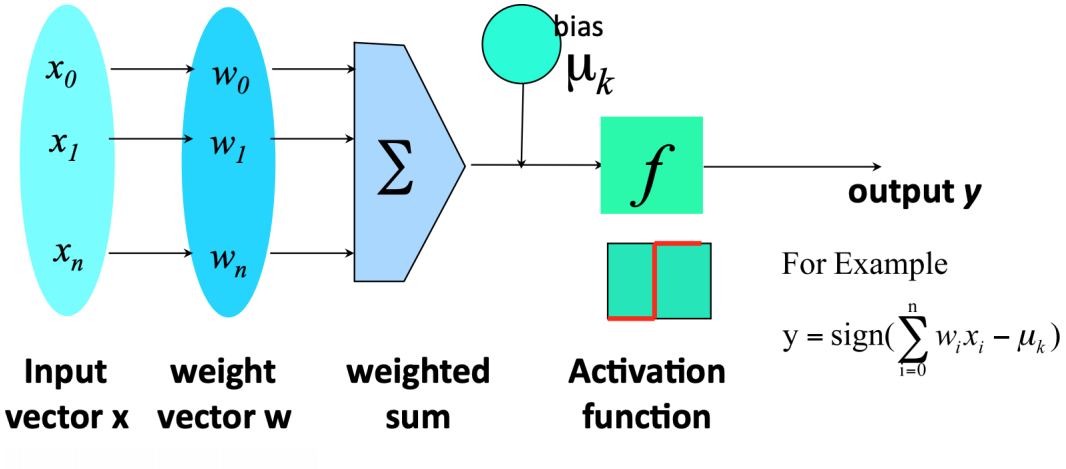
- A set of connected input/out units
- Features: supervised learning, weighted connections, multi-layer, feed-forward, fully connected, backpropagation, adjust weights

9.3 Network topology

- Hidden layers
- Units in each layer
- Input values:
  - Continuous: normalize to [0.0, 1.0]
  - Discrete: one unit per attribute value, initially 0
- Output: 1 unit per class if more than two classes
- Trial-and-error (you do not know what is better, try and see what happens)
  - different topology, different initial weights



- Neuron
  - A hidden/output layer unit



9.4 Backpropagation

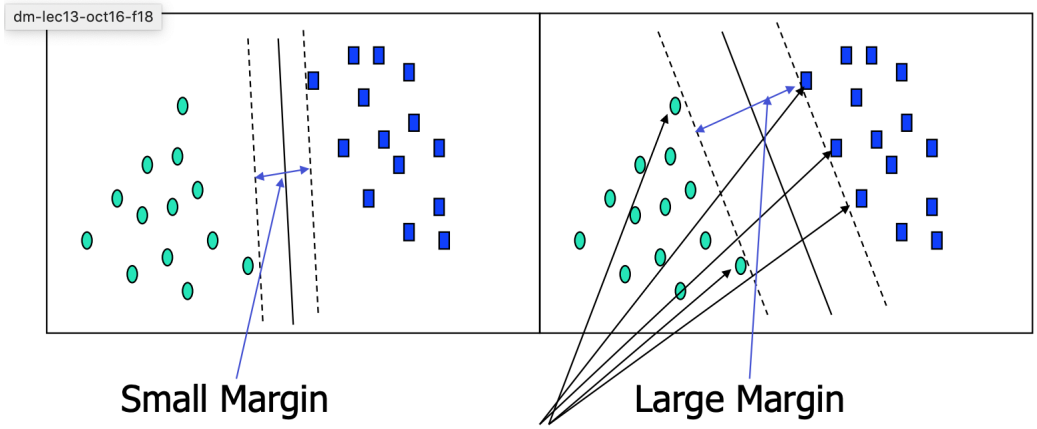
- Initialize weights, biases: small random numbers
  - $I_j = \sum_i w_{ij} O_i + \theta_j$
- Propagate the inputs forward
  - $O_j = \frac{1}{1+e^{-I_j}}$
- Backpropagate the error
  - $Err_j = O_j(1 - O_j)(T_j - O_j)$
  - $Err_j = O_j(1 - O_j) \sum_k Err_k W_{jk}$
- Termination condition
  - $w_{ij} = w_{ij} + (l) Err_j O_i$
  - $\theta_j = \theta_j + (l) Err_j$

9.5 Neural network as classifier

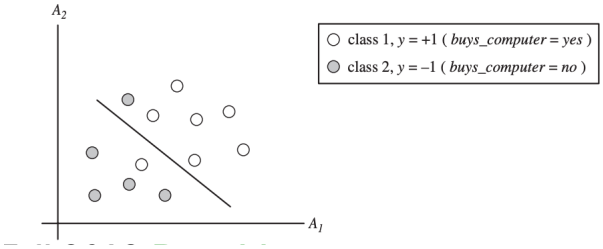
- Weakness: long training time; parameters determined empirically; pool interpretability
- Strength: high tolerance to noisy data; can classify untrained patterns; well-suited for continuous-valued inputs and outputs; success on a wide array of real-world data; inherently parallel; rule extraction

9.6 Support vector machines

- Linearly separable data (larger margin is more robust and means better separation)
  - Example:



  - D:$(x_1, y_1), ..., (x_{|D|}, y_{|D|})$, x_i is n-dimensional
  - A separating hyperplane: $W * x + b = 0$
  - Find maximum marginal hyperplane (MMH)
  - Hyperplanes H_1 and H_2: sides of margin
  - Support vectors: training tuples fall on H_1, H_2
- Linearly inseparable
  - Example:



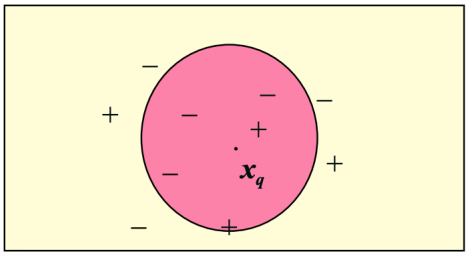
  - Transform data into a higher dimension
  - Search for optimal linear separating hyperplane in the new space
  - Computing dot product on the transformed data mathematically equivalent to applying a kernel function to original data

* $K(x_i, x_j$
* $= \phi(x_i).\phi(x_j)$

- Support vector machines
  - Classification for both linear and nonlinear data
  - Transforms data to higher dimension using nonlinear mapping
  - Search for optimal linear separation hyperplane in the new dimension
  - SVM finds this hyperplane using support vectors("essential" training tuples) and margins(defined by the support vectors)

9.7 Lazy learners(learning from your neighbors)

- Eager learning: constructs classification model on training data before receiving test data
- Lazy learning: stores training data and delays processing until a new instance must be classified
- Lazy: less time in training, more time in predicting
- Lazy: require efficient storage techniques
- Lazy: richer hypothesis space using many local linear functions for implicit global approximation
- k-nearest-neighbor classifiers
  - Find k-nearest-neighbors of a test tuple
  - Discrete-valued: most common values
  - Continuous-valued: average of k values
  - Five greater weight to closer neighbors
  - Indexing for nearest-neighbor search



9.8 Additional topic regarding classification

- Multiclass classification
  - One-versus-all (OVA)
  - All-versus-all (AVA)

- Semi-supervised training
  - Self-training
  - Co-training
- Active learning
  - achieve high accuracy using as few labeled instances as possible
  - Uncertainty sampling, vision space
- Transfer learning
  - E.g. sentiment classification (camera reviews $\rightarrow$ TV reviews
  - E.g. Transfer AdaBoost

10 Cluster Analysis

10.1 What is cluster analysis

- Cluster: a collection of data objects
  - Similar to one another withing a cluster
  - Dissimilar to objects in other clusters
- Cluster analysis
  - Group similar objects into cluster
  - Similarity measure and clustering algorithms
- Unsupervised learning: no predefined classes
- Requirements of clustering: scalability, different types of attributes, clusters with arbitrary shape, minimal domain knowledge for parameters, noisy data, incremental and insensitive to input order, high dimensionality, constraint-based clustering, interpretability and usability

10.2 Major clustering methods

- Partitioning methods
  - Construct k partitions, iterative relocation
  - E.g. k-means, k-medoids, CLARANS
- Hierarchical methods
  - Hierarchical decomposition, split/merge
  - E.g. BIRCH, ROCK, Chameleon
- Density-based methods
  - Connectivity and density functions
  - E.g. DBSCAN, OPTICS, DENCLUE
- Grid-based methods
  - Quantize into cells, multi granularity grid
  - E.g. STING, WaveCluster
- Model-based methods
  - Hypothesized cluster model, best fit
  - E.g. EM, COBWEB, SOM
- Clustering high-dimensional data
  - Subsapce clustering: CLIQUE, PROCLUS

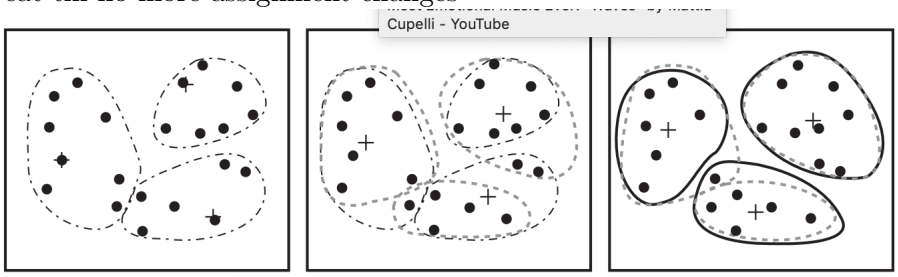
  - Frequent-pattern-based clustering: pCLuster
- Constraint-based clustering
  - User-specified or application-oriented constraints
  - E.g. COD (obstacles), user-constrained clustering, semi-supervised clustering

10.3 Partitioning methods

- Given a dataset D of n objects
- Given k, find a partition of k clusters that optimized the chosen partition criterion
- Global optimal: enumerate all partitions (slow)
- Heuristic methods (faster):

10.3.1 K-means

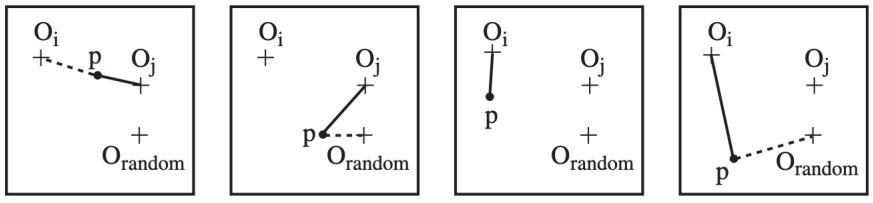
  - Idea: cluster represented by mean (centroid)
  - Partition objects into k nonempty cluster
  - Computer mean (centroid) of each cluster
  - Assign each object to closes centroid
  - Repeat till no more assignment changes



  - Complexity
    - Relatively efficient: O(nkt)
      - n: #objects, k: #clusters, t: #iterations
  - Important points:
    - Often terminates at local optimal
    - Applicable only when centroid is defined
    - Need to specify k in advance
    - Not suitable for discovering clusters with non-convex shapes
    - Sensitive to noise and outliers

10.3.2 K-medoids

- Idea: cluster represented by medoid (object closest to centroid)
- K-means methods is sensitive to outliers. Substantially distort the distribution of data
- K-medoids: find representative objects (medoids)

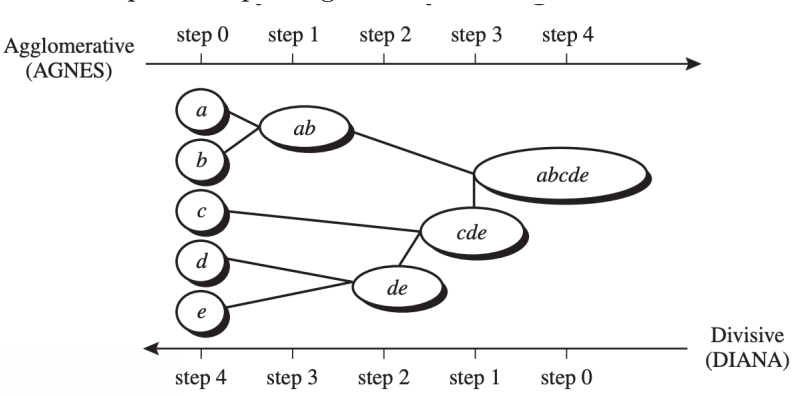


- data object
+ cluster center
— before swapping
--- after swapping

- PAM (Partitioning Around Medoids)
  - Starts from an initial set of medoids
  - Iteratively replace a medoid with a non-medoid if it reduces the total distance
  - Effective for small datasets, does not scale
  - $O(k(n-k)^2)$ for each iteration
- CLARA: apply PAM on multiple sampled sets
- CLARANS: use randomized sample to search for neighboring solutions

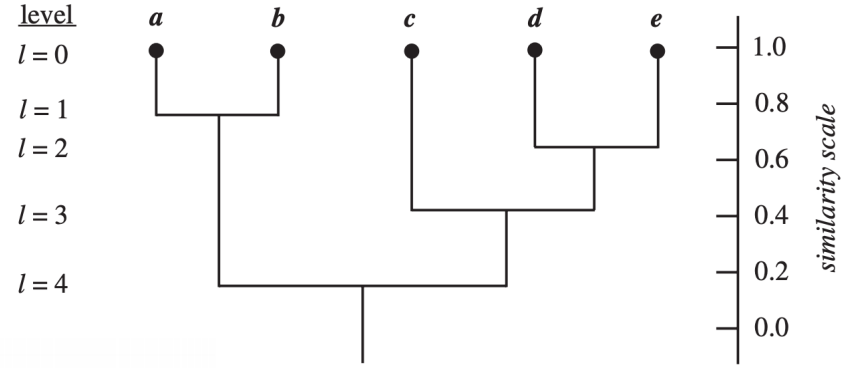
10.4 Hierarchical methods

- Group data objects into a tree of clusters
- Agglomerative: bottom-up merging
- Decisive: top-down splitting

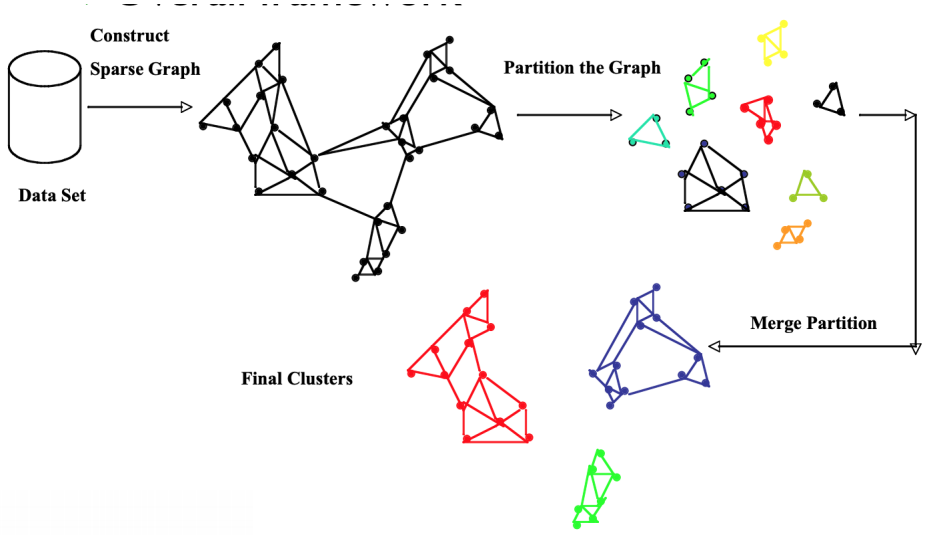


- Dendrogram
  - Represents the process of hierarchical clustering

  - Clustering: cut dendrogram at a certain level



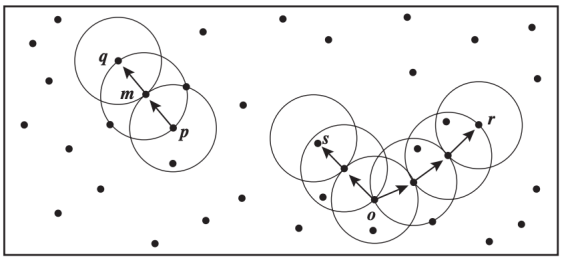
- Distance between clusters
  - Minimum distance: $d_{min}(C_i, C_j)$
  - Maximum distance: $d_{max}(C_i, C_j)$
  - Mean distance: $d_{mean}(C_i, C_j) = |m_i - m_j|$
  - Average distance: $d_{avf}(C_i, C_j) = \frac{1}{n_i n_j} \sum_{p \in C_i} \sum_{p' \in C_j} |p - p'|$
- Uses distance matrix ad clustering criteria
- Does not need to specify k (# clusters)
- Termination condition
  - E.g. cluster distance exceeds a threshold
- Cannot undo previous merge/split decisions
- Algorithms
  - BIRCH
    - Clustering large amount of numerical data
    - Multi-phase clustering
      - Phase 1: microclustering (hierarchical clustering)
      - Phase 2: macroclustering (iterative partitioning)
    - Clustering feature: $CF =< n, LS, SS >$ (capturing how similar the clusters are)
  - CHAMELEON
    - Overall framework



* k-nearest-neighbor graph

10.5 Density-based methods

- Clustering based on density (local cluster criterion), e.g. density-connected points
- Major features
  - Clusters of arbitrary shape, handles noise, single scan, density parameter for termination
- Typical methods
  - DBSCAN
    * ϵ-neighborhood of p: withing radius ϵ of p
    * Core object p: at least MinPts points in the ϵ-neighborhood of p
    * Directly density reachable, density reachable, density connected



  - DENCLUE
    * Uses statistical density functions
    * Major features:
      · Solid mathematical foundation
      · Good for datasets with large amounts of noise
      · Compact description of arbitrarily-shaped clusters in high-dimensional datasets
      · Significantly faster than existing algorithm

        - Needs a large number of parameters
    - Influence function: impact of a data point within its neighborhood
    - Overall density: sum of the influence function of all data points
    - Density attractors: local maximal of overall density function
    - Clusters can be determined mathematically by identifying density attractors

10.6 Grid-based methods

- How it works:
  - Uses multi-resolution grid data structure
  - Quantizes object space to cells in the grid
  - Fast processing time: depends on the number of cells, not the number of objects
- Typical methods:
  - STING (STatistical INfromation Grid)
    - Idea: take spatial area and divide it to rectangular cells
    - Pros:
      - O(g): number of grid cells at bottom level
      - Query-independent, easy to parallelize, incremental update (because no need to shuffle and change the shape, static.)
    - Cons:
      - Finer granularity vs. coarser granularity
      - Only horizontal or vertical cluster boundaries, no diagonal boundaries
  - CLIQUE
    - Dimension-growth subspace clustering. Grows from single dimensions to high dims
    - Both density-based and grid-based
      - Each dimension $\rightarrow$ equal-width intervals
      - Non-overlapping rectangular units
      - Cluster: a set of connected dense units

10.7 Evaluation of clustering

- Major tasks:
  - Clustering tendency (whether the data has clusters or not)
  - Number of clusters
  - Clustering quality (ground truth means we know the actual cluster that certain data belong to)
    - Extrinsic methods (with ground truth)
    - Intrinsic methods (without ground truth)

11 Advanced Cluster Analysis

11.1 Probabilistic model-based clustering

- Cluster membership of each object
  - belongs to a single cluster
  - Weighted distribution in multiple clusters
- Fuzzy clusters (soft clusters)
  - n objects, k clusters
  - w_ij: probability of object i belonging to cluster j
  - $0 \leq w_ij \leq 1$
- Hidden categories (probabilistic clusters)
  - Each represented by a probability density function over the data space
- Mixture model: observed data instances drawn independently from multiple clusters
- Model-based clustering
  - Assumption: data are generated by mixture of underlying probability distributions
  - $P(D|C) = \prod_{i=1}^{n} P(o_i|C) = \prod_{i=1}^{n} \sum_{j=1}^{k} w_j f_j(o)i)$
  - Attempt to optimize the fit between data and some mathematical model
    - Find a set C of k probabilistic clusters such that P(D—C) is maximized
- EM: Expectation Maximization
  - A popular iterative refinement algorithm
  - An extension to k-means
    - Assign each object to a cluster according to a weight (probability distribution)
    - New means computed on weighted sum
  - Mixture of k distributions
    - Distribution → cluster
    - E.g. Gaussian distribution
    - $\theta_i = (\mu_i, \sigma_i)$
  - Expectation step (E-step)
    - $P(\theta_j|o_i, \theta) = \frac{P(o_i|\theta_j)}{\sum_{l=1}^{k} P(o_i|\theta_l)}$
  - Maximization step (M-step)
    - $\mu_j = \frac{\sum_{i=1}^{n} o_i P(\theta_j|o_i,\theta)}{\sum_{i=1}^{n} P(\theta_j|o_i,\theta)}$
    - $\sigma_j = \sqrt{\frac{\sum_{i=1}^{n} P(\theta_j|o_i,\theta)(o_i-\mu_j)^2}{\sum_{i=1}^{n} P(\theta_j|o_i,\theta)}}$
  - Can be characterized by a few parameters
  - Generally converges quickly but may not reach the global optima
  - Computationally expensive if number of distributions is large or data set contains very few observed data points

11.2 Clustering high-dimensional data

- High-dimensional data: e.g. text documents, DNA micro-array data
- Challenges:
  - Many irrelevant dimensions may mask clusters
  - Distance measure dominated by noises
  - Cluster may exists only in some subspaces
- Methods:
  - Subspace clustering, dimensionality reduction
- The curse of dimensionality
  - Data in only one dimension is relatively packed
  - Adding a dimension "stretch" the points across that dimension, making them further apart
  - Adding more dimensions will make the points further apart
    - High-dimensional data is extremely sparse
  - Distance measure becomes meaningless, due to equi-distance
- Why subspace clustering
  - Clusters may exist only in some subspaces
  - Subspace clustering: find clusters in all the subspaces
  - Dimension-growth: CLIQUE
  - Dimension-reduction: PROCLUS
    - Mediods, subset of dimensions with small distance, k * L dimensions for k clusters
  - Frequent patter-based clustering
    - Frequent term-based document clustering
    - Clustering by pattern similarity in micro-array data (pClustering)
- Bi-Clustering
  - Cluster both objects and attributes simultaneously
  - Examples: micro-array data analysis, customers and products

11.3 Clustering graph and network data

- Applications:
  - Bi-partite graphs: customers and products, authors and conferences
  - Web search engines: web graphs, click through graphs
  - Social networks, friendship/coauthor graphs

- Graph clustering methods
  - Generic clustering or graph specific (E.g. minimum cuts, density based)
- Structure-connected cluster C
  - Connectivity, maximality
- Hubs
  - Not belong to any cluster
  - Bridge to many clusters
- Outliers
  - Connect to few clusters

11.4 Clustering with constrains

- Categorization of constraints
  - Constraints on individual objects: e.g. cluster houses worth over 300K
  - Constraints on the selection of clustering parameters: number of clusters, MinPts, etc.
  - Constraints on distance or similarity function: weighted functions, obstacles
  - User-specific constraints on the properties of individual clusters
  - Semi-supervised clustering based on "partial" supervision

12 Outlier Analysis

12.1 Types of Outliers

- Global outliers (point anomaly)
  - Different from the rest of the dataset
- Contextual outliers (conditional outlier)
  - Contextual vs. behavioral attributes
  - E.g. 30-minute commute to work
- Collective outliers
  - A subset of objects
  - E.g. over 100 delayed flights

12.2 Outlier detection challenges

- Modeling normal objects and outliers: difficult to enumerate all normal cases
- Application-specific outlier detection: e.g. clinic data vs. marketing analysis
- Handling noise in outlier detection: blur the normal vs. outlier distinction
- Understandability: justification, degree of outlier

12.3 Outlier detection methods

- Whether user labels are available: supervised, semi-supervised, unsupervised
- Assumptions about normal data and outliers: statistical (model-based), proximity-based, clustering-based

12.4 Proximity-based approaches

- Intuition: objects that are far away from the other are outliers
- Distance-based:
  - Distance of the k-th nearest neighbor or fraction of objects within distance r
- Density-based:
  - Density around outlier
  - Density of neighbors

12.5 Clustering-based approaches

- An object is outlier if
  - Does not belong to any cluster
  - Far from its closest cluster
  - Belongs to a small or sparse cluster
- Using training set to find patterns of formal data
- Compare new object with cluster
- Strength:
  - No labels required, support many data types
  - Cluster are summaries of data
  - Only need to compare object to clusters (fast)
- Weakness
  - Effectiveness depends on clustering method
  - High computation cost: first find clusters
  - To reduce cost: e.g. fixed-width clustering

12.6 Classification-based methods

- Train classification model that can distinguish normal data from outliers
- Brute-force approach
  - Labeled normal and outlier objects
  - Skewed data, cannot detect unseen anomaly
- One-class model
  - Classifier for normal data

12.7 Semi-supervised learning

- Combine classification and clustering
- Clustering to find large cluster C and small cluster C1
- C: objs w/ normal label
- One-class model from C
- C1 objs w/ outlier label
- Any obj not in C's model $\rightarrow$ identify as outlier

12.8 Contextual outliers

- Transform to conventional outlier detection if the contexts can be clearly identified
  - Identify the context of O
  - Compare O w/ others in the same context
  - Use conventional outlier detection
- Generalize context if too few objects in the same context
  - E.g. customers by age group
  - E.g. by geographic region
- Modeling normal behavior with respect to contexts
- Contextual attributes → behavior attributes
  - Mixture model U on contextual attributes
  - Mixture model V on behavior attributes
  - $p(V_i|U_j)$
- Approaches: regression, Markov models, finite state automation

12.9 Collective outliers

- Objects as a group deviate significantly from the entire dataset
- Need to examine structures, i.e. relationships between multiple data objects (e.g. temporal, spatial, graph/network)
- Different from contextual outlier detection: structures are often not explicitly defined
- Reduce to conventional outlier detection: structure unit → "super" data object
- Model the expected behavior of structure unites directly
- E.g. online social networks of customers
  - Structure unit: subgraphs of the network
  - Small subgraphs with very low frequency
  - Large subgraphs that are surprisingly frequent
- E.g. temporal sequences
  - Learn a Markov model from the sequences; a subsequence that deviates from the model

12.10 Outlier in High-D data

- Data in high-d spaces are often sparse. Object distances dominated by noise
- Interpretation of outliers
  - Too many features, need to identify subspaces of the outliers
- Subspaces that signifying the outliers: capture the local behavior of data
- Scalable: number of subspaces increases exponentially