3.1 Classification and Prediction

Classification and Prediction:

- Used to extract models describing important data classes or to predict future data trends.
- Eg. Classification Model → Built to categorize bank loan applications as either safe or risky.
- Eg. Prediction Model → Built to predict the expenditures for potential customers on computer equipment given their income and occupation.

What is classification? What is Prediction?

Classification—A Two-Step Process

1) First Step: (Model Construction)

- Model built describing a pre-determined set of data classes or concepts.
- A class label attribute determines which tuple belongs to which pre-determined class.
- This model is built by analyzing a training data set.
- Each tuple in a training data set is called a training sample.
- This model is represented in the form of:
  - Classification Rules o Decision Trees o Mathematical Formulae
- Eg: Database having customer credit info
  - Classification Rule – identifies customers with 'Excellent credit rating' or 'Fair credit rating'.

2) Second Step: (Using the model in prediction)

- Model is used for classification
  - Predictive accuracy of the model is estimated. - There are several methods for this.
    - Holdout Method is the simple technique.
      - Uses test set of class labeled samples.
      - These are randomly selected and are independent of training samples.
      - The Accuracy of the Model on a given test set is the % of test set samples that are correctly classified by the model.
      - If the accuracy of the model is acceptable the model can be used to classify future data tuples for which the class label is not known.

Supervised vs. Unsupervised Learning

- *Supervised learning (classification)*
  - Supervision: The training data (observations, measurements, etc.) are accompanied by labels indicating the class of the observations
  - New data is classified based on the training set

Unsupervised learning (clustering)

- The class labels of training data is unknown
- Given a set of measurements, observations, etc. with the aim of establishing the existence of classes or clusters in the data

3.2 Issues Regarding Classification and Prediction

Issues regarding Classification and Prediction:

1) Prepare the data for classification and Prediction: - Steps Applied are:
    a. Data Cleaning: - To reduce Confusion
        - a.1 Remove / Reduce Noise → By Smoothing
        - a.2 Treatment of missing values → Replace with common / probable values
    b. Relevance Analysis:
        - Removing redundant & irrelevant attributes – called as Feature Selection
        - Otherwise they slow down / mislead the classification process.
        - Improves classification efficiency & scalability
    c. Data Transformation:
        - c.1 Data Generalized to higher level concepts
          → Concept Hierarchies used for this purpose
          → Compress the original training data
        - c.2 Data Normalized – Scaling all values for a given attribute
          → They fall within a range say (-1.0 to 1.0) / (0 to 1.0)
2) Comparing Classification Methods:
    - Compared and Evaluated according to the following criteria:
    a. Predictive Accuracy: - Ability of the model to correctly predict the class label.
    b. Speed: - Computational cost in generating & using the model.
    c. Robustness: - Ability of the model to make correct predictions even with noisy / missing data.
    d. Scalability: - Ability of the model to perform efficiently on large volumes of data.
    e. Interpretability: - Level of understanding and insight provided by the model.

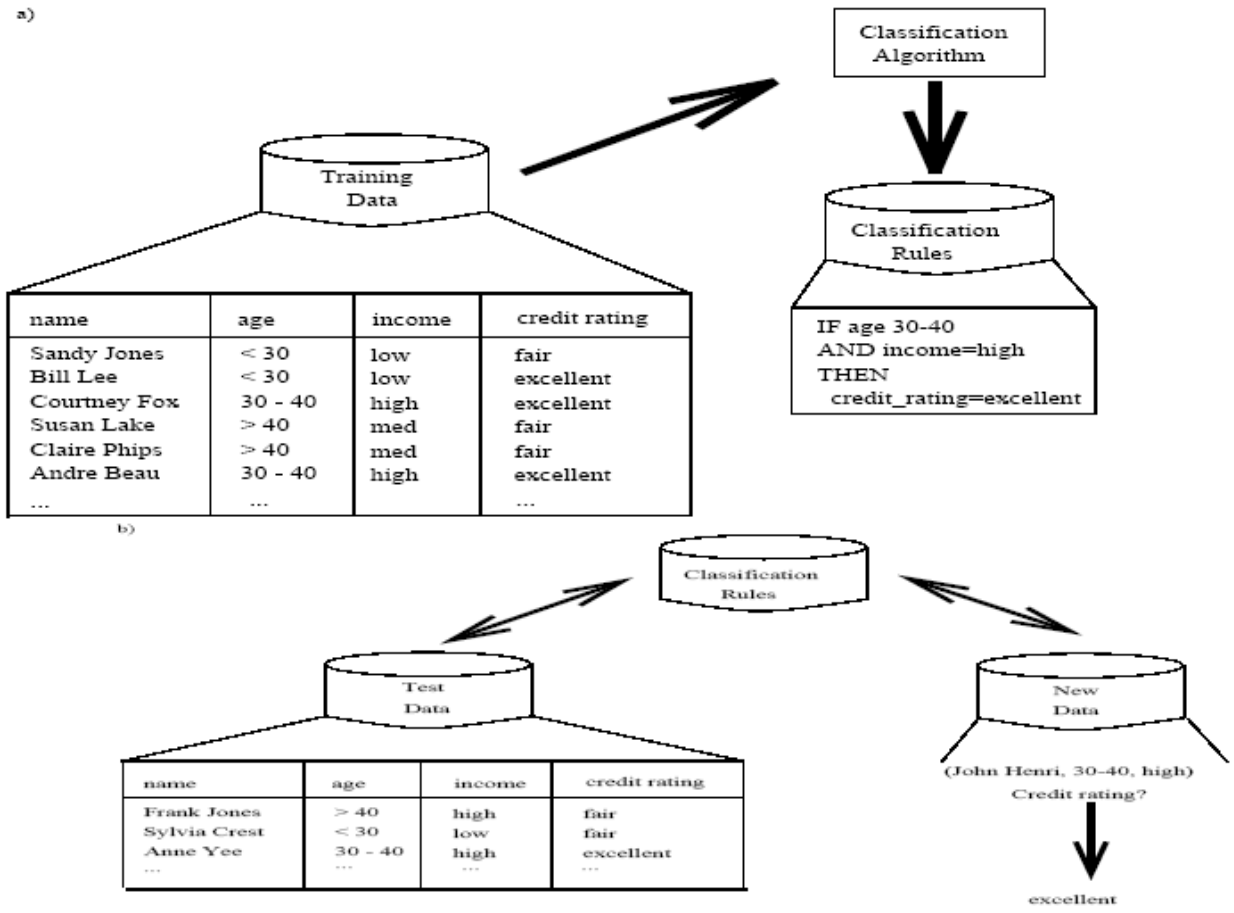


Figure 7.1: The data classification process: a) *Learning:* Training data are analyzed by a classification algorithm. Here, the class label attribute is *credit_rating*, and the learned model or classifier is represented in the form of classification rules. b) *Classification:* Test data are used to estimate the accuracy of the classification rules. If the accuracy is considered acceptable, the rules can be applied to the classification of new data tuples.

3.3 Classification By Decision Tree Induction

Decision Tree Induction:

- A decision tree is a flow chart like tree structure.
  - Each internal node denotes test on an attribute
  - Each branch = outcome of the test; Leaf Nodes = Classes
- Internal nodes denoted by rectangles & leaf nodes denoted by ovals.
- To classify the new tuples the attribute values of the tuples are tested against the decision tree.
- Path is traced from root to leaf node to find the class of the tuple.
- Decision trees can be easily converted to classification rules.
- Decision trees are used in many application areas ranging from medicine to business.
- Decision trees are the basis of several commercial rule induction systems.

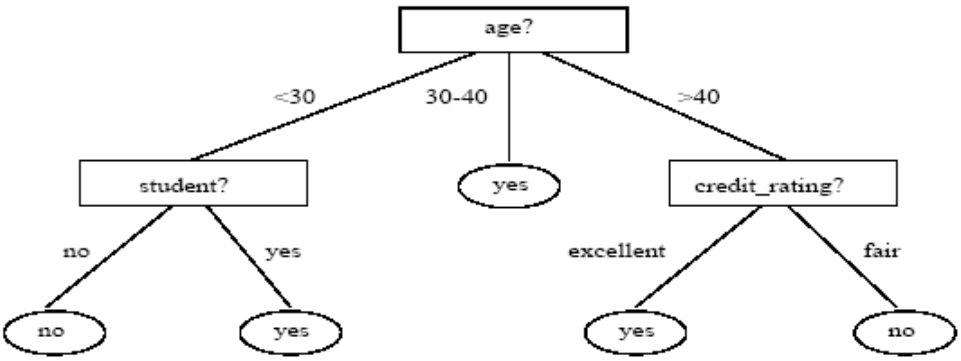


Figure 7.2: A decision tree for the concept *buys_computer*, indicating whether or not a customer at *AllElectronics* is likely to purchase a computer. Each internal (non-leaf) node represents a test on an attribute. Each leaf node represents a class (either *buys_computer = yes* or *buys_computer = no*).

Algorithm for generating a decision tree from training samples:

(Generate decision tree)

Input: Set of training samples, attribute list (set of candidate attributes)

Output: A decision tree

Method:

1) create a node N;
2) **if** *samples* are all of the same class, C **then**
3) return N as a leaf node labeled with the class C;
4) **if** *attribute-list* is empty **then**
5) return N as a leaf node labeled with the most common class in *samples*; // majority voting
6) select *test-attribute*, the attribute among *attribute-list* with the highest information gain;
7) label node N with *test-attribute*;
8) **for each** known value a_i **of** *test-attribute* // partition the samples
9) grow a branch from node N for the condition *test-attribute*=a_i;
10) let s_i be the set of samples in *samples* for which *test-attribute*=a_i; // a partition
11) **if** s_i is empty **then**
12) attach a leaf labeled with the most common class in *samples*;
13) **else** attach the node returned by Generate_decision_tree(s_i, *attribute-list* - *test-attribute*);

This algorithm constructs decision trees in top-down recursive divide-and-conquer manner.

Basic Strategy:

1) Tree starts as a single node having all training samples.
2) If samples are all of same class then
   Node = leaf-labeled with the class
3) This uses entropy-based heuristic (information gain) for selecting the attribute which can be the class label. This attribute is the 'test' or 'decision' attribute at the node. In this all attributes are categorical.
4) A branch created for known value of test attribute – Samples partitioned accordingly.
5) Uses the above process recursively to form a decision tree.
6) Recursive partitioning stops when any one condition is true:
   a. All samples belong to same Class.
   b. No remaining attributes to partition the samples.
   c. Majority Voting: Convert node in to leaf, label it with class in majority of samples.
   d. No sample for test-attribute = ai
      Leaf is created with majority class in samples.

Attribute Selection Measure / Information Gain Measure / Measure of the goodness of split:

- Used to select the test attribute at each node in the tree
- The attribute with the highest information gain is chosen as the test attribute for the current node.
- Let S = set of data samples; |S| = s
- Let the class label attribute have m distinct values and these distinct values define distinct classes
  Ci (for i = 1…m)
- Let si = Number of samples in Ci
- Expected Information needed to classify a given sample is

$$I(s_1, s_2, \ldots, s_m) = -\sum_{i=1}^{m} p_i log_2(p_i)$$

Where pi = probability that any sample belongs to Ci = si/s

- Choose an attribute A that has v distinct values {a1,a2,…av}
- Then attribute A can partition S in v subsets {S1,S2,…Sv} where Sj has samples with value aj of A
- Let sij = Number of samples of class Ci in a subset Sj
- Expected information need if the samples are to be partitioned based on the attribute A.

$$E(A) = \sum_{j=1}^{v} \frac{s_{1j} + \cdots + s_{mj}}{s} I(s_{1j}, \ldots, s_{mj}).$$

- Information gained by branching on A is: $Gain(A) = I(s_1, s_2, \ldots, s_m) - E(A)$
- Similarly information gain computed for each attribute;
- Attribute with highest information gain is chosen as test attribute.
- A node is created and labeled with the attribute; Branches are created for each value of the attribute
- Samples are partitioned based on the attributes values.

Example:

- Table below presents a training set of data tuples; |S| = s = 14;
- Class label attribute = buys_computer;
- Distinct values of class label = {yes, no}
- Distinct classes C1 = Class of ‘yes’; C2 = Class of ‘no’;

rid	age	income	student	credit_rating	Class: buys_computer
1	<30	high	no	fair	no
2	<30	high	no	excellent	no
3	30-40	high	no	fair	yes
4	>40	medium	no	fair	yes
5	>40	low	yes	fair	yes
6	>40	low	yes	excellent	no
7	30-40	low	yes	excellent	yes
8	<30	medium	no	fair	no
9	<30	low	yes	fair	yes
10	>40	medium	yes	fair	yes
11	<30	medium	yes	excellent	yes
12	30-40	medium	no	excellent	yes
13	30-40	high	yes	fair	yes
14	>40	medium	no	excellent	no

- s1 = Number of samples in class C1 = 9; s2 = Number of samples in class C2 = 5
- Expected Information need to classify the sample using the class-label attribute:

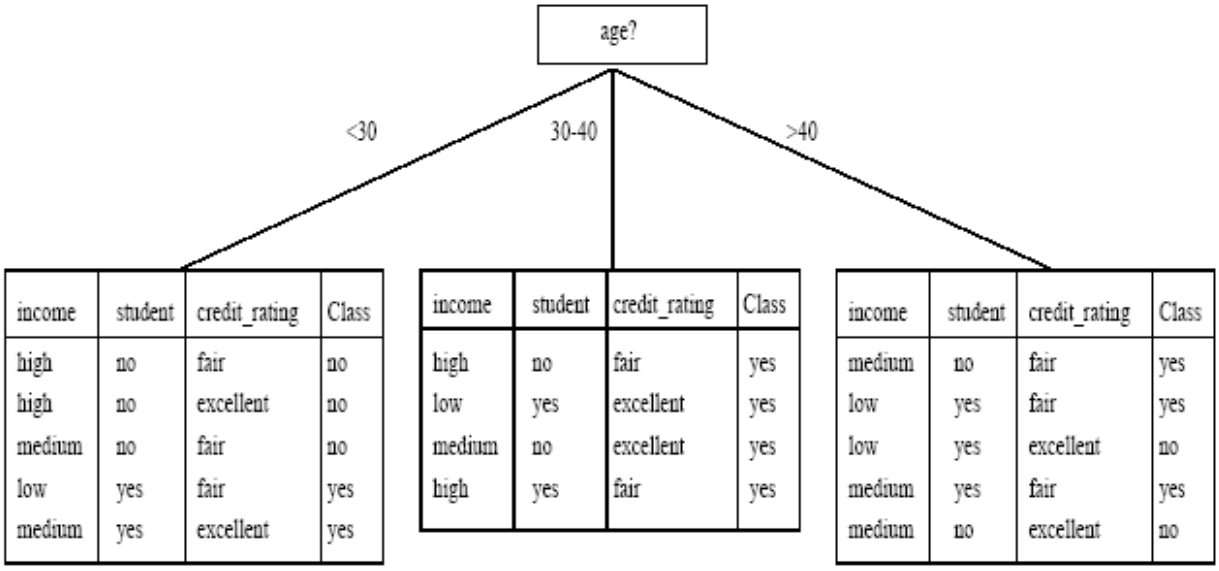
$$I(s_1, s_2) = I(9,5) = -\frac{9}{14}\log_2\frac{9}{14} - \frac{5}{14}\log_2\frac{5}{14} = 0.940$$

- First choose attribute 'age' that has 3 distinct values {<30, 30-40, >40}
- Now S is partitioned into 3 sets of samples
  - Samples for which age <30 = S1 = 5
  - Samples for which age = 30-40 = S2 = 4
  - Samples for which age >40 = S3 = 5
- s11 = Samples of class C1 in S1 = Samples with age <30 and buys_computer = 'yes' = 2
- s21 = Samples of class C2 in S1 = Samples with age <30 and buys_computer = 'no' = 3
- s12 = Samples of class C1 in S2 = Samples with age =30-40 and buys_computer = 'yes' = 4
- s22 = Samples of class C2 in S2 = Samples with age =30-40 and buys_computer = 'no' = 0
- s13 = Samples of class C1 in S3 = Samples with age >40 and buys_computer = 'yes' = 3
- s23 = Samples of class C2 in S3 = Samples with age >40 and buys_computer = 'no' = 2
- $I(s_{11}, s_{21}) = 0.971$; $I(s_{12}, s_{22}) = 0$; $I(s_{13}, s_{23}) = 0.971$;
- Expected information needed to classify the given sample using the attribute 'age'.

$$E(age) = \frac{5}{14}I(s_{11}, s_{21}) + \frac{4}{14}I(s_{12}, s_{22}) + \frac{5}{14}I(s_{13}, s_{23}) = 0.694.$$

$$Gain(age) = I(s_1, s_2) - E(age) = 0.246$$

- Similarly, compute $Gain(income) = 0.029, Gain(student) = 0.151$
- Age has highest information gain among the attributes.
- Create a node & label it with age; Branches created for age < 30; age = 30-40; age > 40
- Samples partitioned based on the value of age; Samples in the branch age = 30-40 belong to C1 = yes
- Create a leaf to the end of this branch and label it with 'yes'



income	student	credit_rating	Class
high	no	fair	no
high	no	excellent	no
medium	no	fair	no
low	yes	fair	yes
medium	yes	excellent	yes

income	student	credit_rating	Class
high	no	fair	yes
low	yes	excellent	yes
medium	no	excellent	yes
high	yes	fair	yes

income	student	credit_rating	Class
medium	no	fair	yes
low	yes	fair	yes
low	yes	excellent	no
medium	yes	fair	yes
medium	no	excellent	no

- Decision tree (i) used in wide range of applications (ii) Fast generation (iii) High accuracy.

Tree Pruning:

- Done to avoid over fitting of data & to handle noise / outliers
- Least reliable branches removed using statistical measures → Faster classification and efficient
- Two approaches – (i) Pre-pruning Approach (ii) Post-pruning Approach

Pre-Pruning Approach:

- Halt tree construction early; Upon halting node becomes leaf
- Leaf = most frequent class (or) Probability distribution of samples in the class
- If partitioning the samples at a node results in a split that is below the pre-specified threshold
- → Further partitioning halted.
- Choice of threshold is a challenge; High thresholds → Over simplified trees;
- Low thresholds → Complex trees

Post-Pruning Approach:

- Removes branches from a "fully grown" tree.
- Tree node pruned by removing its branches.
- Pruned node becomes a leaf & labeled by most frequent class among its former branches.
- Uses the cost complexity pruning algorithm.
- For each node the algorithm calculates the expected error rate for if the sub tree at the node is pruned, it calculates the expected error rate if the node is not pruned.
- If pruning the node has greater expected error rate than not pruning, then the sub tree is kept.
- Otherwise the node is pruned.
- Likewise the tree is pruned progressively for each node.
- The decision tree with minimum expected error rate is preferred.
- Pre-pruning and post-pruning can be combined for best approach.
- Post-pruning requires more computations than pre-pruning
- But post-pruning is more reliable & accurate than pre-pruning.

Extracting classification rules from decision trees:

- Decision trees can also be represented in the form of IF THEN Classification rules.
- One rule is created for each path from root to leaf node.
- IF part is called as Rule Antecedent; THEN part is called as Rule Consequent
- Each (attribute=value) pair along a path forms a conjunction in the rule antecedent.
- Class prediction of each leaf node forms the Rule Consequent.
- In case of very large tree the classification rule representation in the form of IF THEN rules is easy for humans to interpret.
- Example: The rules extracted from the above decision tree are:
    - IF age = '<30' AND student = 'no' THEN buys_computer = 'no'
    - IF age = '<30' AND student = 'yes' THEN buys_computer = 'yes'
    - IF age = '30-40' THEN buys_computer = 'yes'
    - IF age = '>40' AND credit_rating = 'excellent' THEN buys_computer = 'yes'
    - IF age = '>40' AND credit_rating = 'fair' THEN buys_computer = 'no'

3.4 Bayesian Classification

Bayesian Classification:

- "What are Bayesian Classifiers?" They are statistical classifiers
- Predicts class membership probabilities = Probability that a given sample belongs to a particular class
- This is based on Bayes Theorem
- Exhibits high accuracy and speed when applied to large databases.
- One type of Bayesian Classification is "Naïve Bayesian Classifier"
- This method has performance comparable to Decision Tree induction.
- This method is based on the assumption that "The effect of an attribute value on a given class is independent of the values of the other attributes"
- This assumption is called "Class Conditional Independence"
- Another type of Bayesian Classification is 'Bayesian Belief Networks'
- Bayesian Classifiers have minimum error rate when compared to all other classifiers

Bayes Theorem:

- Let X be a data sample whose class label is unknown
- **Ex:** Data Samples consists of fruits described by their colour and shape.
- Say, X is red and round
- Let H be some hypothesis such that the data sample X belongs to a class C
- **Ex:** H is the hypothesis that X is an apple.
- Then, we have to determine P(H/X) = Probability that the hypothesis H holds for the data sample X
- P(H/X) is called as the Posterior Probability of H on X
- **Ex:** Probability that X is an apple given that X is red and round.
- P(X/H) = Posterior Probability of X on H
- **Ex:** Probability that X is red and round given that X is an apple.
- P(H) = Prior Probability of H
- **Ex:** Probability that any given data sample is an apple regardless of its colour and shape.
- P(X) = Prior Probability of X
- **Ex:** Probability that X is red and round given that X is an apple.
- Bayes Theorem is => P(H/X) = P(X/H) P(H) / P(X)

Naïve Bayesian Classification:

1. Each data sample is represented by an n-dimensional vector X = (x1, x2, … xn) [Where, we have n-attributes => A1, A2, … An].
2. Say there are m classes, C1, C2, … Cm; Let X be an unknown data sample having no class label.
   Then Naïve Bayesian Classifier assigns X to the class Ci; Where Ci is the class having highest Posterior Probability Conditioned on X.
   i.e. $P(C_i|X) > P(C_j|X) \text{ for } 1 \leq j \leq m,\ j \neq i.$ = **Maximum Posteriori Hypothesis**
   Here we calculate P(Ci/X) by $P(C_i|X) = \frac{P(X|C_i)P(C_i)}{P(X)}.$ = **Bayes Theorem**
3. P(X) is constant for all classes. Hence only P(X/Ci) & P(Ci) need to be maximized.

Here P(Ci) = si/s; Where si = Samples in class Ci; s = Total number of samples in all classes.

4. To evaluate P(X/Ci) we use the Naïve assumption of "Class Conditional Independence".

$$P(X|C_i) = \prod_{k=1}^{n} P(x_k|C_i).$$

Hence

The probabilities P(x1/Ci), P(x2/Ci), … P(xn/Ci) can be estimated from the training samples;

Where P(xk/Ci) = sik/si; Where sik = No. Of samples in Class Ci that has value = xk for Ak;

si = No. Of samples in class Ci.

5. Evaluate P(X/Ci) P(Ci) for each class Ci; [i = 1, … m]

X is assigned to the class Ci for which P(X/Ci) P(Ci) is maximum.

Example- Predicting a class label using Naïve Bayesian Classification:

rid	age	income	student	credit_rating	Class: buys_computer
1	<30	high	no	fair	no
2	<30	high	no	excellent	no
3	30-40	high	no	fair	yes
4	>40	medium	no	fair	yes
5	>40	low	yes	fair	yes
6	>40	low	yes	excellent	no
7	30-40	low	yes	excellent	yes
8	<30	medium	no	fair	no
9	<30	low	yes	fair	yes
10	>40	medium	yes	fair	yes
11	<30	medium	yes	excellent	yes
12	30-40	medium	no	excellent	yes
13	30-40	high	yes	fair	yes
14	>40	medium	no	excellent	no

- Let C1 = Class buys_computer = yes; C2 = Class buys_computer = no
- Let us try to classify an unknown sample:
- X = (age = "<30", income = medium, student = yes, credit-rating = fair)
- We need to maximize P(X/Ci) P(Ci) for i = 1,2; So, Compute P(C1) & P(C2)
- P(buys_computer = yes) = 9/14 = 0.643 P(buys_computer = no) = 5/14 = 0.357
- Next, Compute P(X/C1) & P(X/C2)
- P(age = "<30" / buys_computer = yes) = 2/9 = 0.222
- P(income = medium / buys_computer = yes) = 4/9 = 0.444
- P(student = yes / buys_computer = yes) = 6/9 = 0.667
- P(credit-rating = fair / buys_computer = yes) = 6/9 = 0.667
- P(age = "<30" / buys_computer = no) = 3/5 = 0.600
- P(income = medium / buys_computer = no) = 2/5 = 0.400
- P(student = yes / buys_computer = no) = 1/5 = 0.200
- P(credit-rating = fair / buys_computer = no) = 2/5 = 0.400
- Hence P(X/buys_computer = yes) = 0.222 * 0.444 * 0.667 * 0.667 = 0.044
- P(X/buys_computer = no) = 0.600 * 0.400 * 0.200 * 0.400 = 0.019
- Finally P(X/buys_computer = yes) P(buys_computer = yes) = 0.044 * 0.643 = 0.028
- P(X/buys_computer = no) P(buys_computer = no) = 0.019 * 0.357 = 0.007
- Hence Naïve Bayesian Classifier predicts "buys_computer = yes" for sample X.

3.5 Other Classification Methods

Other classification methods

In this section, we give a brief description of a number of other classi3cation methods. These methods include k-nearest neighbor classi3cation, case-based reasoning, genetic algorithms, rough set and fuzzy set approaches. In general, these methods are less commonly used for classi3cation in commercial data mining systems than the methods described earlier in this chapter. Nearest-neighbor classi3cation, for example, stores all training samples, which may present di3culties when learning from very large data sets. Furthermore, many applications of case-based reasoning, genetic algorithms, and rough sets for classi3cation are still in the prototype phase. These methods, however, are enjoying increasing popularity, and hence we include them here.

k-nearest neighbor classifiers

Nearest neighbor classi3ers are based on learning by analogy. The training samples are described by n-dimensional numeric attributes. Each sample represents a point in an n-dimensional space. In this way, all of the training samples are stored in an n-dimensional pattern space. When given an unknown sample, a k-nearest neighbor classi3er searches the pattern space for the k training samples that are closest to the unknown sample.

> The unknown sample is assigned the most common class among its k nearest neighbors. When k = 1, the

unknown sample is assigned the class of the training sample that is closest to it in pattern space.

> Nearest neighbor classi3ers are instance-based since they store all of the training samples. They can incur

expensive computational costs when the number of potential neighbors (i.e., stored training samplesÑ with which to compare a given unlabeled sample is great. Therefore, e3cient indexing techniques are required. Unlike decision tree induction and backpropagation, nearest neighbor classi3ers assign equal weight to each attribute. This may cause confusion when there are many irrelevant attributes in the data.

Nearest neighbor classi3ers can also be used for prediction, i.e., to return a real-valued prediction for a given unknown sample. In this case, the classi3er returns the average value of the real-valued labels associated with the k nearest neighbors of the unknown sample.

Case-based reasoning

Case-based reasoning (CBRÑ classi3ers are instanced-based. Unlike nearest neighbor

classi3ers, which store train-ing samples as points in Euclidean space, the samples or \cases" stored by CBR are complex symbolic descriptions. Business applications of CBR include problem resolution for customer service help desks, for example, where cases describe product-related diagnostic problems. CBR has also been applied to areas such as engineering and law, where cases are either technical designs or legal rulings, respectively.

When given a new case to classify, a case-based reasoner will 3rst check if an identical training case exists. If one is found, then the accompanying solution to that case is returned. If no identical case is found, then the case-based reasoner will search for training cases having components that are similar to those of the new case. Conceptually, these training cases may be considered as neighbors of the new case. If cases are represented as graphs, this involves searching for subgraphs which are similar to subgraphs within the new case. The case-based reasoner tries to combine the solutions of the neighboring training cases in order to propose a solution for the new case. If incompatibilities arise with the individual solutions, then backtracking to search for other solutions may be necessary. The case-based reasoner may employ background knowledge and problem-solving strategies in order to propose a feasible combined solution.

Challenges in case-based reasoning include 3nding a good similarity metric (e.g., for matching subgraphsÑ, devel-oping e3cient techniques for indexing training cases, and methods for combining solutions.

Genetic algorithms

Genetic algorithms attempt to incorporate ideas of natural evolution. In general, genetic learning starts as follows. An initial population is created consisting of randomly generated rules. Each rule can be represented by a string of bits. As a simple example, suppose that samples in a given training set are described by two Boolean attributes, A1 and A2, and that there are two classes, C1 and C2. The rule \IF A1 and not A2 THEN C2 " can be encoded as the bit string \100", where the two leftmost bits represent attributes A1 and A2, respectively, and the rightmost bit represents the class. Similarly, the rule \if not A1 and not A2 then C1" can be encoded as \001". If an attribute has k values where k ó 2, then k bits may be used to encode the attribute's values. Classes can be encoded in a similar fashion.

Based on the notion of survival of the 3ttest, a new population is formed to consist of the 3ttest rules in the current population, as well as o−spring of these rules. Typically, the 3tness of a rule is assessed by its classi3cation accuracy on a set of training samples.

O−spring are created by applying genetic operators such as crossover and mutation. In crossover, substrings from pairs of rules are swapped to form new pairs of rules. In mutation, randomly selected bits in a rule's string are inverted.

The process of generating new populations based on prior populations of rules continues until a population P \evolves" where each rule in P satis3es a prespeci3ed 3tness threshold.

Genetic algorithms are easily parallelizable and have been used for classi3cation as well as

other optimization problems. In data mining, they may be used to evaluate the ɜtness of other algorithms.

Rough set theory

Rough set theory can be used for classiɜcation to discover structural relationships within imprecise or noisy data. It applies to discrete-valued attributes. Continuous-valued attributes must therefore be discretized prior to its use.

Rough set theory is based on the establishment of equivalence classes within the given training data. All of the data samples forming an equivalence class are indiscernible, that is, the samples are identical with respect to the attributes describing the data. Given real-world data, it is common that some classes cannot be distinguished

3.6 Prediction

Prediction:

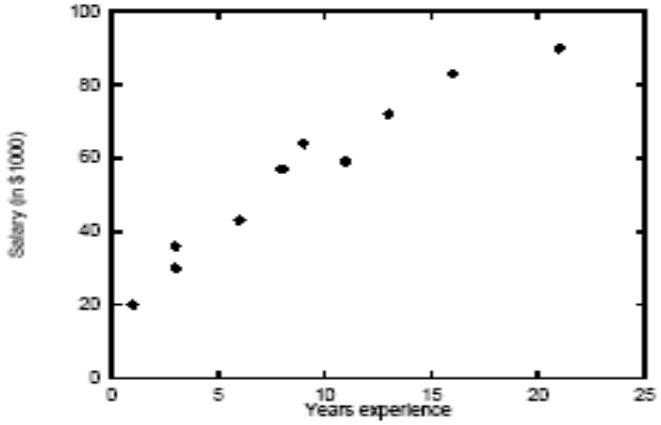
- Data values two types – Continuous, Categorical
- Prediction of Continuous values done using statistical technique – regression.
- Example: Predict the salary of graduates with 10 years of work experience
- Many problems can be solved using linear regression and for the remaining problems we can apply transformation of variables so that nonlinear problem can be converted to a linear one.

Linear and Multiple Regressions:

Linear Regression:

- Data are modeled using a straight line. This is the simplest form of regression.
- Expressed as $Y = \alpha + \beta X,$ where Y is the Response Variable and X is the Predictor Variable and α and β are regression coefficients corresponding to Y-intercept and slope of the line respectively.
- These regression coefficients can be solved by the method of least squares.
- Let us consider s samples of data points (x1, y1), (x2, y2), … (xs, ys)
- Then regression coefficients can be estimated using the below equations.
- $$\beta = \frac{\sum_{i=1}^{s}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{s}(x_i - \bar{x})^2}, \qquad \alpha = \bar{y} - \beta\bar{x},$$
- Where $\bar{x}$ is the average of $x_1, x_2, .., x_s,$ and $\bar{y}$ is the average of $y_1, y_2, .., y_s.$
- Example – Salary data plotted on a graph shows that there is linear relationship between X (years of experience) and Y (Salary).

X *years experience*	Y *salary (in $1000)*
3	30
8	57
9	64
13	72
3	36
6	43
11	59
21	90
1	20
16	83



- From the above data we compute $\bar{x} = 9.1$ and $\bar{y} = 55.4$.
- Hence using the above equations we have:

$$\beta = \frac{(3-9.1)(30-55.4)+(8-9.1)(57-55.4)+\ldots+(16-9.1)(83-55.4)}{(3-9.1)^2+(8-9.1)^2+\ldots+(16-9.1)^2} = 3.7$$

- $\alpha = 55.4 - (3.7)(9.1) = 21.7$
- Substituting these values in the linear regression equation, we have:
- $Y = 21.7 + 3.7X.$
- Hence the salary of a graduate with 10 years of experience is $\$58.7\text{K}.$

Multiple Regressions:

- Extension of Linear Regression. Involves more than one predictor variable (X).
- Response variable Y is expressed as a linear function of multi dimensional feature vector.
- Consider that we have two predictor variables X1 & X2. Then the equation is:
- $Y = \alpha + \beta_1 X_1 + \beta_2 X_2$
- Same method of least squares applied to solve $\alpha,\ \beta_1,\ \text{and}\ \beta_2.$

Non-Linear Regression (or) Polynomial Regression:

- Add Polynomial terms to the basic linear model.
- 1. Apply transformations to the variables.
- 2. Convert non-linear model into a linear one.
- 3. Then solve the same by method of least squares.
- Example: Consider the cubic polynomial relationship given the below equation:
- $Y = \alpha + \beta_1 X + \beta_2 X^2 + \beta_3 X^3$
- 1. Apply transformations to variables: $X_1 = X$; $X_2 = X^2$; $X_3 = X^3$
- 2. Convert non-linear model into a linear one: $Y = \alpha + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_3.$
- 3. Solve the above equation by the method of least squares.

Other Regression Models:
Generalized Linear Models:

- Used for Prediction of Categorical data values.
- Here the variance of the response variable Y is a function of the mean value of Y.
- Types:
    - Logistic Regression
    - Poisson Regression

Log-Linear Models:

- Used for Prediction of Categorical data values. If the values are continuous those values are discretized before applying log-linear models.
- Estimates the probability value associated with data cube cells.
- Example: Say, we have data for the attributes city, item, year and sales.
- In this, sales is a continuous valued attribute. Hence its values are discretized to make it categorical.
- Now estimate the probability of each cell in a 4D base cuboid for the above attributes.
- This is done based on the 2D cuboids for city & item, city & year, city & sales and 3D cuboid for item, year and sales.
- Thus, such iterative techniques can be used to build higher order data cubes from lower order ones.

3.7 Clusters Analysis: Types Of Data In Cluster Analysis

What is Cluster Analysis?

The process of grouping a set of physical objects into classes of similar objects is called clustering.

Cluster – collection of data objects
– Objects within a cluster are similar and objects in different clusters are dissimilar.

Cluster applications – pattern recognition, image processing and market research.

- helps marketers to discover the characterization of customer groups based on purchasing patterns
- Categorize genes in plant and animal taxonomies
- Identify groups of house in a city according to house type, value and geographical location
- Classify documents on WWW for information discovery

Clustering is a preprocessing step for other data mining steps like classification, characterization.
Clustering – Unsupervised learning – does not rely on predefined classes with class labels.

Typical requirements of clustering in data mining:

1. Scalability – Clustering algorithms should work for huge databases
2. Ability to deal with different types of attributes – Clustering algorithms should work not only for numeric data, but also for other data types.
3. Discovery of clusters with arbitrary shape – Clustering algorithms (based on distance measures) should work for clusters of any shape.

4. Minimal requirements for domain knowledge to determine input parameters – Clustering results are sensitive to input parameters to a clustering algorithm (example – number of desired clusters). Determining the value of these parameters is difficult and requires some domain knowledge.
5. Ability to deal with noisy data – Outlier, missing, unknown and erroneous data detected by a clustering algorithm may lead to clusters of poor quality.
6. Insensitivity in the order of input records – Clustering algorithms should produce same results even if the order of input records is changed.
7. High dimensionality – Data in high dimensional space can be sparse and highly skewed, hence it is challenging for a clustering algorithm to cluster data objects in high dimensional space.
8. Constraint-based clustering – In Real world scenario, clusters are performed based on various constraints. It is a challenging task to find groups of data with good clustering behavior and satisfying various constraints.
9. Interpretability and usability – Clustering results should be interpretable, comprehensible and usable. So we should study how an application goal may influence the selection of clustering methods.

Types of data in Clustering Analysis

1. Data Matrix: (object-by-variable structure)
   Represents n objects, (such as persons) with p variables (or attributes) (such as age, height, weight, gender, race and so on. The structure is in the form of relational table or n x p matrix as shown below:

$$\begin{bmatrix} x_{11} & \cdots & x_{1f} & \cdots & x_{1p} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ x_{i1} & \cdots & x_{if} & \cdots & x_{ip} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ x_{n1} & \cdots & x_{nf} & \cdots & x_{np} \end{bmatrix}$$ → called as "two mode" matrix

2. Dissimilarity Matrix: (object-by-object structure)
   This stores a collection of proximities (closeness or distance) that are available for all pairs of n objects. It is represented by an n-by-n table as shown below.

$$\begin{bmatrix} 0 & & & & \\ d(2,1) & 0 & & & \\ d(3,1) & d(3,2) & 0 & & \\ \vdots & \vdots & \vdots & & \\ d(n,1) & d(n,2) & \cdots & \cdots & 0 \end{bmatrix}$$ → called as "one mode" matrix

   Where d (i, j) is the dissimilarity between the objects i and j; d (i, j) = d (j, i) and d (i, i) = 0

   Many clustering algorithms use Dissimilarity Matrix. So data represented using Data Matrix are converted into Dissimilarity Matrix before applying such clustering algorithms.

   Clustering of objects done based on their similarities or dissimilarities.
   Similarity coefficients or dissimilarity coefficients are derived from correlation coefficients.

3.8 Categorization of Major Clustering Methods

Categorization of Major Clustering Methods

The choice of many available clustering algorithms depends on type of data available and the application used.

Major Categories are:

1. Partitioning Methods:
   - Construct k-partitions of the n data objects, where each partition is a cluster and k <= n.
   - Each partition should contain at least one object & each object should belong to exactly one partition.
   - Iterative Relocation Technique – attempts to improve partitioning by moving objects from one group to another.
   - Good Partitioning – Objects in the same cluster are "close" / related and objects in the different clusters are "far apart" / very different.
   - Uses the Algorithms:
     - K-means Algorithm: - Each cluster is represented by the mean value of the objects in the cluster.
     - K-mediods Algorithm: - Each cluster is represented by one of the objects located near the center of the cluster.
     - These work well in small to medium sized database.

2. Hierarchical Methods:
   - Creates hierarchical decomposition of the given set of data objects.
   - Two types – Agglomerative and Divisive
   - Agglomerative Approach: (Bottom-Up Approach):
     - Each object forms a separate group
     - Successively merges groups close to one another (based on distance between clusters)
     - Done until all the groups are merged to one or until a termination condition holds. (Termination condition can be desired number of clusters)
   - Divisive Approach: (Top-Down Approach):
     - Starts with all the objects in the same cluster
     - Successively clusters are split into smaller clusters
     - Done until each object is in one cluster or until a termination condition holds (Termination condition can be desired number of clusters)
   - Disadvantage – Once a merge or split is done it can not be undone.
   - Advantage – Less computational cost
   - If both these approaches are combined it gives more advantage.
   - Clustering algorithms with this integrated approach are BIRCH and CURE.

3. Density Based Methods:
   - Above methods produce Spherical shaped clusters.
   - To discover clusters of arbitrary shape, clustering done based on the notion of density.
   - Used to filter out noise or outliers.

- Continue growing a cluster so long as the density in the neighborhood exceeds some threshold.
- Density = number of objects or data points
- That is for each data point within a given cluster; the neighborhood of a given radius has to contain at least a minimum number of points.
- Uses the algorithms: DBSCAN and OPTICS

4. Grid-Based Methods:
   - Divides the object space into finite number of cells to forma grid structure.
   - Performs clustering operations on the grid structure.
   - Advantage – Fast processing time – independent on the number of data objects & dependent on the number of cells in the data grid.
   - STING – typical grid based method
   - CLIQUE and Wave-Cluster – grid based and density based clustering algorithms.

5. Model-Based Methods:
   - Hypothesizes a model for each of the clusters and finds a best fit of the data to the model.
   - Forms clusters by constructing a density function that reflects the spatial distribution of the data points.
   - Robust clustering methods
   - Detects noise / outliers.

Many algorithms combine several clustering methods.

3.9 Partitioning Methods

Partitioning Methods

Database has n objects and k partitions where k<=n; each partition is a cluster.

Partitioning criterion = Similarity function:

Objects within a cluster are similar; objects of different clusters are dissimilar.

Classical Partitioning Methods: k-means and k-mediods:

(A) Centroid-based technique: The k-means method:

- Cluster similarity is measured using mean value of objects in the cluster (or clusters center of gravity)
- Randomly select k objects. Each object is a cluster mean or center.
- Each of the remaining objects is assigned to the most similar cluster – based on the distance between the object and the cluster mean.
- Compute new mean for each cluster.
- This process iterates until all the objects are assigned to a cluster and the partitioning criterion is met.
- This algorithm determines k partitions that minimize the squared error function.
- Square Error Function is defined as: $E = \Sigma_{i=1}^{k} \Sigma_{x \in C_i} |x - m_i|^2,$

- Where x is the point representing an object, mi is the mean of the cluster Ci.

- Algorithm:

Algorithm 8.4.1 (*k*-means) The *k*-means algorithm for partitioning based on the mean value of the objects in the cluster.

Input: The number of clusters *k*, and a database containing *n* objects.

Output: A set of *k* clusters which minimizes the squared-error criterion.

Method: The *k*-means algorithm is implemented as follows.

```
1)  arbitrarily choose k objects as the initial cluster centers;
2)  repeat
3)      (re)assign each object to the cluster to which the object is the most similar,
            based on the mean value of the objects in the cluster;
4)      update the cluster means, i.e., calculate the mean value of the objects for each cluster;
5)  until no change;
```

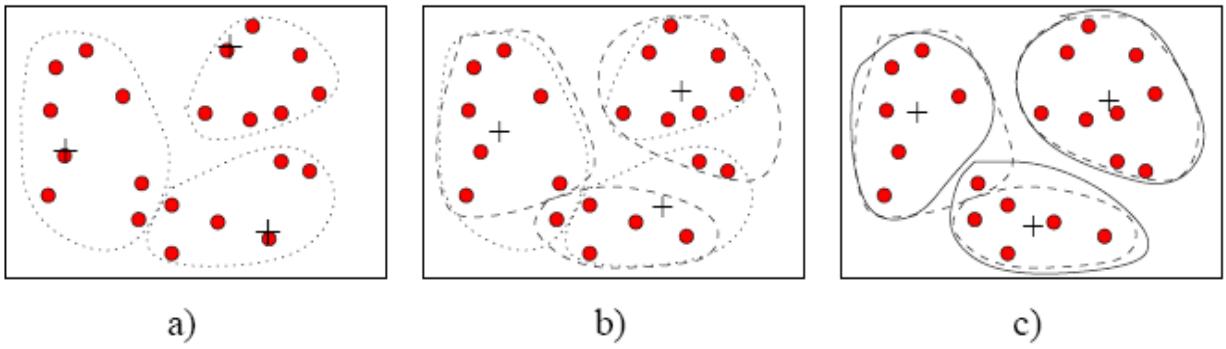


Figure 8.2: Clustering of a set of points based on the *k*-means method

- Advantages: Scalable; efficient in large databases
- Computational Complexity of this algorithm:
  - O(nkt); n = number of objects, k number of partitions, t = number of iterations
  - k << n and t << n
- Disadvantage:
  - Cannot be applied for categorical data – as mean cannot be calculated.
  - Need to specify the number of partitions – k
  - Not applicable for clusters of different size.
  - Noise and outliers cannot be detected

(B) Representative Point-based technique: The k-mediods method:

- Mediod – most centrally located point in a cluster – Reference point
- Partitioning is based on the principle of minimizing the sum of the dissimilarities between each object with its corresponding reference point.

- PAM – Partitioning Around Mediods – k-mediods type clustering algorithm.
- Finds k clusters in n objects by finding mediod for each cluster.
- Initial set of k mediods are arbitrarily selected.
- Iteratively replaces one of the mediods with one of the non-mediods so that the total distance of the resulting clustering is improved.

- After initial selection of k-mediods, the algorithm repeatedly tries to make a better choice of mediods by analyzing all the possible pairs of objects such that one object is the mediod and the other is not.
- The measure of clustering quality is calculated for each such combination.
- The best choice of points in one iteration is chosen as the mediods for the next iteration.
- Cost of single iteration is O(k(n-k)2).
- For large values of n and k, the cost of such computation could be high.

Algorithm 8.4.2 The k-medoids algorithm for partitioning based on the central objects.

Input: The number of clusters k, and a database containing n objects.

Output: A set of k clusters which minimizes the sum of the dissimilarities of all the objects to their nearest medoid.

Method: The k-medoids algorithm is implemented as follows.

```
1)    arbitrarily choose k objects as the initial medoids;
2)    repeat
3)        assign each object to the cluster corresponding to the nearest medoid;
4)        calculate the objective function, which is the sum of dissimilarities
              of all the objects to their nearest medoid;
5)        swap the medoid x by an object y if such a swap reduces the objective function;
6)    until no change;
```

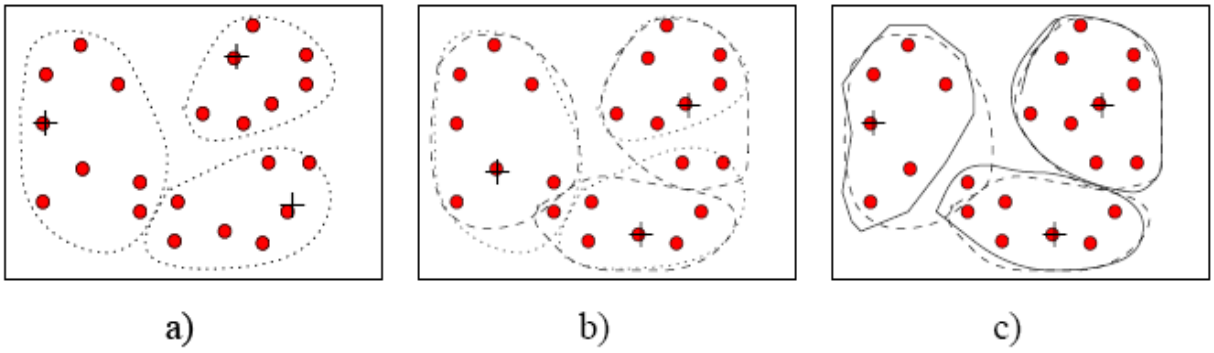


Figure 8.4: Clustering of a set of points based on the k-medoids method

- Advantage: - k-mediods method is more robust than k-means method.
- Disadvantage: - k-mediods method is more costly than k-means method.
- User needs to specify k the number of clusters in both these methods.

(C) Partitioning method in large databases: from k-mediods to CLARANS:

- (i) CLARA – Clustering LARge Applications – Sampling based method.
- In this method, only a sample set of data is considered from the whole dataset and the mediods are selected from this sample using PAM. Sample selected randomly.
- CLARA draws multiple samples of the data set, applies PAM on each sample and gives the best clustering as the output. Classifies the entire dataset to the resulting clusters.
- Complexity of each iteration in this case is: O(kS2 + k(n-k)); S = size of the sample; k = number of clusters; n = total number of objects.
- Effectiveness of CLARA depends on sample size.
- Good clustering of samples does not imply good clustering of the dataset if the sample is biased.

- (ii) CLARANS – Clustering LARge Applications based on RANdomized Search – To improve quality and scalability of CLARA.
-
- This is similar to PAM & CLARA
- It does not consider a sample or does not consider the entire database.
- Begins like PAM – selects k-mediods by applying Randomized Iterative Optimization
- Then randomly selects few pairs (k, j) = "maxneighbour" number of pairs for swapping.
-
- If the pair with minimum cost found then updates the mediod set and continues.
- Else current selections of mediods are considered as the local optimum set.
- Now repeat by randomly selecting new mediods – search for another local optimum set.
-
- Stops after finding "num local" number of local optimum sets.
- Returns best of local optimum sets.
- CLARANS enables detection of outliers – Best mediod based method.
- Drawbacks – Assumes object fits into main memory; Result is based on input order.

3.10 Hierarchical Methods

Hierarchical Methods

This works by grouping data objects into a tree of clusters. Two types – Agglomerative and Divisive.

Clustering algorithms with integrated approach of these two types are BIRCH, CURE, ROCK and CHAMELEON.

BIRCH – Balanced Iterative Reducing and Clustering using Hierarchies:

- Integrated Hierarchical Clustering algorithm.
- Introduces two concepts – Clustering Feature and CF tree (Clustering Feature Tree)
- CF Trees – Summarized Cluster Representation – Helps to achieve good speed & clustering scalability
- Good for incremental and dynamical clustering of incoming data points.
- Clustering Feature CF is the summary statistics for the cluster defined as: $CF = (N, \vec{LS}, SS)$;
- where N is the number of points in the sub cluster (Each point is represented as $\vec{X}$);
- $\vec{LS}$ is the linear sum of N points = $\sum_{i=1}^{N} \vec{X}_i$; SS is the square sum of data points $\sum_{i=1}^{N} \vec{X}_i^2$.

- CF Tree – Height balanced tree that stores the Clustering Features.
- This has two parameters – Branching Factor B and threshold T
- Branching Factor specifies the maximum number of children.

- Threshold parameter T = maximum diameter of sub clusters stored at the leaf nodes.
- Change the threshold value => Changes the size of the tree.
- The non-leaf nodes store sums of their children's CF's – summarizes information about their children.

- BIRCH algorithm has the following two phases:
    - Phase 1: Scan database to build an initial in-memory CF tree – Multi-level compression of the data – Preserves the inherent clustering structure of the data.
        - CF tree is built dynamically as data points are inserted to the closest leaf entry.
        - If the diameter of the subcluster in the leaf node after insertion becomes larger than the threshold then the leaf node and possibly other nodes are split.
        - After a new point is inserted, the information about it is passed towards the root of the tree.
        - Is the size of the memory to store the CF tree is larger than the the size of the main memory, then a smaller value of threshold is specified and the CF tree is rebuilt.
        - This rebuild process builds from the leaf nodes of the old tree. Thus for building a tree data has to be read from the database only once.
    - Phase 2: Apply a clustering algorithm to cluster the leaf nodes of the CF-tree.
- Advantages:
    - Produces best clusters with available resources.
    - Minimizes the I/O time
- Computational complexity of this algorithm is – O(N) – N is the number of objects to be clustered.
- Disadvantage:
    - Not a natural way of clustering;
    - Does not work for non-spherical shaped clusters.

CURE – Clustering Using Representatives:

- Integrates hierarchical and partitioning algorithms.
- Handles clusters of different shapes and sizes; Handles outliers separately.
- Here a set of representative centroid points are used to represent a cluster.
- These points are generated by first selecting well scattered points in a cluster and shrinking them towards the center of the cluster by a specified fraction (shrinking factor)
- Closest pair of clusters are merged at each step of the algorithm.
-

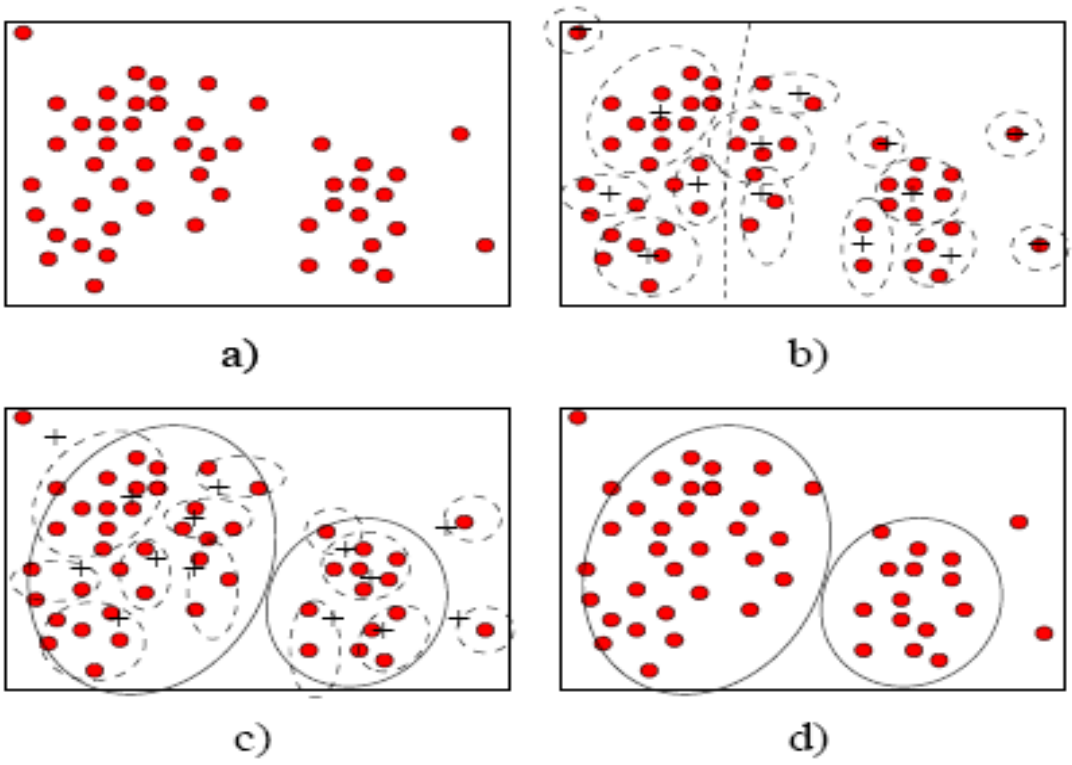


Figure 8.6: Clustering of a set of points by CURE

- Having more than one representative point in a cluster allows BIRCH to handle clusters of non-spherical shape.
- Shrinking helps to identify the outliers.
- To handle large databases – CURE employs a combination of random sampling and partitioning.
- The resulting clusters from these samples are again merged to get the final cluster.

- CURE Algorithm:
  - Draw a random sample s
  - Partition sample s into p partitions each of size s/p
  - Partially cluster partitions into s/pq clusters where q > 1
  - Eliminate outliers by random sampling – if a cluster is too slow eliminate it.
  - Cluster partial clusters
  - Mark data with the corresponding cluster labels
  -
- Advantage:
  - High quality clusters
  - Removes outliers
  - Produces clusters of different shapes & sizes
  - Scales for large database
- Disadvantage:
  - Needs parameters – Size of the random sample; Number of Clusters and Shrinking factor
  - These parameter settings have significant effect on the results.

ROCK:

- Agglomerative hierarchical clustering algorithm.
- Suitable for clustering categorical attributes.
- It measures the similarity of two clusters by comparing the aggregate inter-connectivity of two clusters against a user specified static inter-connectivity model.
- Inter-connectivity of two clusters C1 and C2 are defined by the number of cross links between the two clusters.
- link(pi, pj) = number of common neighbors between two points pi and pj.

- Two steps:
  - First construct a sparse graph from a given data similarity matrix using a similarity threshold and the concept of shared neighbors.
  - Then performs a hierarchical clustering algorithm on the sparse graph.

CHAMELEON – A hierarchical clustering algorithm using dynamic modeling:

- In this clustering process, two clusters are merged if the inter-connectivity and closeness (proximity) between two clusters are highly related to the internal interconnectivity and closeness of the objects within the clusters.
- This merge process produces natural and homogeneous clusters.
- Applies to all types of data as long as the similarity function is specified.

- This first uses a graph partitioning algorithm to cluster the data items into large number of small sub clusters.
- Then it uses an agglomerative hierarchical clustering algorithm to find the genuine clusters by repeatedly combining the sub clusters created by the graph partitioning algorithm.
- To determine the pairs of most similar sub clusters, it considers the interconnectivity as well as the closeness of the clusters.

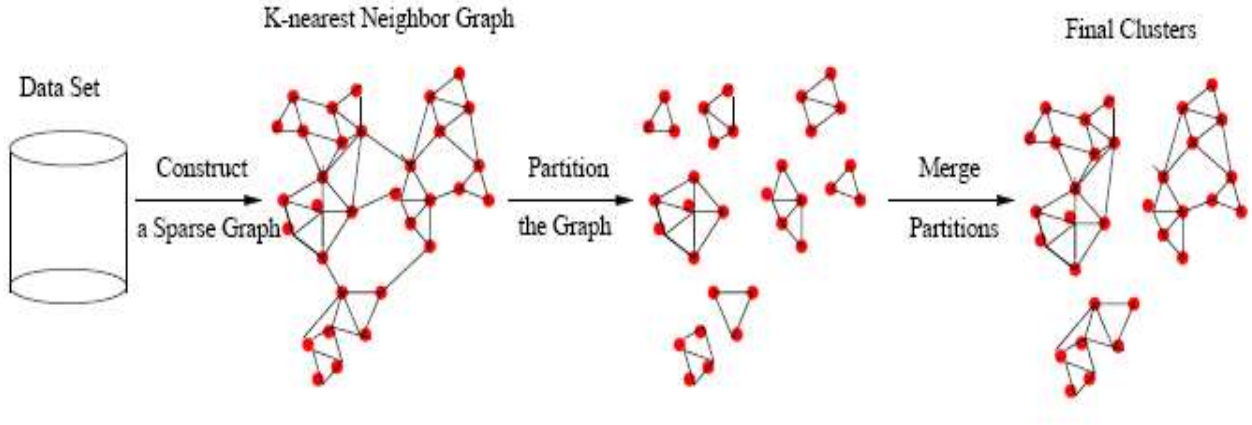


Figure 8.7: CHAMELEON: Hierarchical clustering based on k-nearest neighbors and dynamic modeling

- In this objects are represented using k-nearest neighbor graph.
- Vertex of this graph represents an object and the edges are present between two vertices (objects)

- Partition the graph by removing the edges in the sparse region and keeping the edges in the dense region. Each of these partitioned graph forms a cluster
- Then form the final clusters by iteratively merging the clusters from the previous cycle based on their interconnectivity and closeness.

- CHAMELEON determines the similarity between each pair of clusters Ci and Cj according to their **relative inter-connectivity RI(Ci, Cj)** and their **relative closeness RC(Ci, Cj)**.

$$RI(C_i, C_j) = \frac{|EC_{\{C_i,C_j\}}|}{\frac{1}{2}(|EC_{C_i}| + |EC_{C_j}|)}$$

-
- $EC_{\{C_i,C_j\}}$ = edge-cut of the cluster containing both Ci and Cj
- EC_{C_i} (or EC_{C_j}) = size of min-cut bisector

$$RC(C_i, C_j) = \frac{\overline{S}_{EC_{\{C_i,C_j\}}}}{\frac{|C_i|}{|C_i|+|C_j|}\overline{S}_{EC_{C_i}} + \frac{|C_j|}{|C_i|+|C_j|}\overline{S}_{EC_{C_j}}}$$

-
- $\overline{S}_{EC_{\{C_i,C_j\}}}$ = Average weight of the edges that connect vertices in Ci to vertices in Cj
- $\overline{S}_{EC_{C_i}}$ = Average weight of the edges that belong to the min-cut bisector of cluster Ci.
- Advantages:
  - More powerful than BIRCH and CURE.
  - Produces arbitrary shaped clusters
- Processing cost:
- $O(n^2)$ - n = number of objects.

Review Questions

Two Marks:

1. Write about the two step process of classification.
2. Distinguish between Supervised and Unsupervised learning.
3. List down the issues regarding classification and prediction.
4. What is meant by decision tree rule induction?
5. Explain about the two tree pruning approaches.
6. Show how to extract classification rules from decision trees using suitable example.
7. Write notes on linear regression and explain how to solve the linear equation.
8. Detail on (i) Multiple regression (ii) Polynomial regression.
9. Write about (i) Generalized Linear Models (ii) Log-Linear Models
10. What is cluster Analysis?
11. What are the typical requirements of clustering in data mining?
12. Write about the types of data used in clustering analysis.
13. What are the major categories of clustering methods?
14. Write about the partitioning algorithm CLARA.
15. Write about the partitioning algorithm CLARANS.

Sixteen Marks:

1. Explain about the Decision tree induction algorithm with an example.
2. (i) Write notes on Bayes Classification. (2)
   (ii) Define the Bayes Theorem with example. (4)
   (iii) Explain in detail about Naïve Bayesian Classifiers with suitable example. (10)
3. (i) Describe on the k-means classical partitioning algorithm. (8)
   (ii) Describe on the k-mediods / Partitioning Around Mediods algorithm. (8)
4. (i) Describe on the BIRCH hierarchical algorithm. (8)
   (ii) Describe on the CURE hierarchical algorithm. (8)
5. (i) Describe on the ROCK hierarchical algorithm. (8)
   (ii) Describe on the CHAMELEON hierarchical algorithm. (8)

Assignment Topic:

1. Write in detail about “Other Classification Methods”.