LAB MANUAL

SUBJECT:	**DATA WAREHOUSING AND MINING**
CLASS:	**T.E (Computer Engineering)**
SEMESTER:	**VI**

INDEX

Problem 1:

Title:

Design data warehouse for auto sales analysis

1. star,
2. snowflake &
3. Galaxy schema for the same.

Perform following for the above warehouse:

4. Maximum/minimum sale in first quarter w.r.t. location
5. Maximum/minimum sale of item “vehicles” throughout the year
6. Maximum/minimum sale of item throughout the year
7. Maximum/minimum sale during the second & third quarter w.r.t. location and item
8. List out the items in increasing order w.r.t. sales amount & quantity sold
9. List out the suppliers who supply maximum number of “bikes/cars” during year
10. Find out the customer who purchase maximum number of items and also find out all the details of customer along with region.

Notes:

- Design a data cube which contain one fact table and design item, time, supplier, location, customer dimension table, also identify measures for sales. Insert minimum 4 items like bikes, small cars, mid segment cars, car consumables items etc. Also enter minimum 10-12 records
- Region/location, enter minimum 2 cities from each state also enter minimum 2 states. Keep track of sales quarter wise.
- Perform and implement above fact & dimension tables in oracle10g which are same as relational table of database, perform analyze above with the help of SQL tool.
- You have to use concepts of OLAP operation like slice, dice, roll-up, drill-down etc

Objective:

- To learn fundamental of data warehousing
- To learn concepts of dimensional modeling
- To learn star, snowflake & Galaxy schema

Reference:

- SQL-PL/SQL by Ivan Bayrose
- Data Mining Concept and Technique By Han & Kamber
- Data Warehousing Fundamentals By Paulraj

- Data warehousing & Mining By Reema Thereja

Pre-requisite:

- Fundamental Knowledge of Database Management
- Fundamental Knowledge of SQL

Theory:

Dimensional modeling (DM) is the name of a logical design technique often used for data warehouses.

Dimensional modeling always uses the concepts of facts, measures, and dimensions.

Facts are typically (but not always) numeric values that can be aggregated,

Dimensions are groups of hierarchies and descriptors that define the facts. For example, sales amount is a fact; timestamp, product, register#, store#, etc. are elements of dimensions. Dimensional models are built by business process area, e.g. store sales, inventory, claims, etc.

Fact table

The fact table is not a typical relational database table as it is de-normalized on purpose - to enhance query response times. The fact table typically contains records that are ready to explore, usually with ad hoc queries. Records in the fact table are often referred to as events, due to the time-variant nature of a data warehouse environment.
The primary key for the fact table is a composite of all the columns except numeric values / scores (like QUANTITY, TURNOVER, exact invoice date and time).
Typical fact tables in a global enterprise data warehouse are (usually there may be additional company or business specific fact tables):

Sales fact table - contains all details regarding sales
Orders fact table - in some cases the table can be split into open orders and historical orders. Sometimes the values for historical orders are stored in a sales fact table.
Budget fact table - usually grouped by month and loaded once at the end of a year.
Forecast fact table - usually grouped by month and loaded daily, weekly or monthly.
Inventory fact table - report stocks, usually refreshed daily

Dimension table

Nearly all of the information in a typical fact table is also present in one or more dimension tables. The main purpose of maintaining Dimension Tables is to allow browsing the categories quickly and easily.

The primary keys of each of the dimension tables are linked together to form the composite primary key of the fact table. In a star schema design, there is only one de-normalized table for a given dimension.

Typical dimension tables in a data warehouse are:

Time dimension table
Customers dimension table
Products dimension table
Key account managers (KAM) dimension table
Sales office dimension table

Star schema architecture

Star schema architecture is the simplest data warehouse design. The main feature of a star schema is a table at the center, called the **fact table** and the **dimension tables** which allow browsing of specific categories, summarizing, drill-downs and specifying criteria.
Typically, most of the fact tables in a star schema are in database third normal form, while dimensional tables are de-normalized (second normal form).
Despite the fact that the star schema is the simplest data warehouse architecture, it is most commonly used in the data warehouse implementations across the world today (about 90-95% cases).

Snowflake Schema architecture

Snowflake schema architecture is a more complex variation of a star schema design. The main difference is that **dimensional tables in a snowflake schema are normalized**, so they have a typical relational database design.

Snowflake schemas are generally used when a dimensional table becomes very big and when a star schema can't represent the complexity of a data structure. For example if a PRODUCT dimension table contains millions of rows, the use of snowflake schemas should significantly improve performance by moving out some data to other table (with BRANDS for instance).

The problem is that the more normalized the dimension table is, the more complicated SQL joins must be issued to query them. This is because in order for a query to be answered, many tables need to be joined and aggregates generated.

Fact constellation/Galaxy schema Architecture

For each star schema or snowflake schema it is possible to construct a fact **constellation schema**.

This schema is more complex than star or snowflake architecture, which is because it contains multiple fact tables. This allows dimension tables to be shared amongst many fact tables.
In a fact constellation schema, different fact tables are explicitly assigned to the dimensions, which are for given facts relevant. This may be useful in cases when some facts are associated with a given dimension level and other facts with a deeper dimension level.

Use of that model should be reasonable when for example, there is a sales fact table (with details down to the exact date and invoice header id) and a fact table with sales forecast which is calculated based on month, client id and product id.

These dimensions allow us to answer questions such as

- In what regions of the country are pleated pants most popular? (fact table joined with the product and ship-to dimensions)
- What percentage of pants were bought with coupons and how has that varied from quarter to quarter? (fact table joined with the promotion and time dimensions)
- How many pants were sold on holidays versus non-holidays? (fact table joined with the time dimension)

Post lab assignment:

1. Describe OLAP operation like slice, dice, roll-up, drill-down with example
2. Star schema vs snowflake schema
3. Dimensional table Vs. Relational Table
4. Advantages of snowflake schema

Problem 2:

Title:

Choose any system for which, data warehouse is suitable, design all the dimension & fact table for the same also perform any **three analytical queries**.

Perform and implement above fact & dimension tables in oracle10g which are same as relational table of database, perform analyze above with the help of SQL tool.

Note: Student has to perform above case study in group of 3-4

Objective:

- To learn fundamental of data warehousing
- To learn concepts of dimensional modeling
- To learn star, snowflake & Galaxy schema
- Team work

Reference:

- SQL-PL/SQL by Ivan Bayrose
- Data Mining Concept and Technique By Han & Kamber
- Data Warehousing Fundamentals By Paulraj
- Data warehousing & Mining By Reema Thereja

Pre-requisite:

- Fundamental Knowledge of Database Management
- Fundamental Knowledge of SQL

Theory:

Please refer supportive material /case study of previous problem for analytical query & schema designing.

Post lab assignment:

1. Describe OLAP vs OLTP
2. Multi dimensional data cube
3. DMQL

Problem 3

Title:

Implement classification using K nearest neighbor classification

Objective:

- To learn how to classify data by K nearest neighbor algorithm for classification

Reference:

- Data Mining Introductory & Advanced Topic by Margaret H. Dunham
- Data Mining Concept and Technique By Han & Kamber

Pre-requisite:

- Fundamental Knowledge of Database Management

Theory:

In k-nearest-neighbor classification, the training dataset is used to classify each member of a "target" dataset.

The structure of the data is that there is a classification (categorical) variable of interest ("buyer," or "non-buyer," for example), and a number of additional predictor variables (age, income, location...).

Algorithm:

1. For each row (case) in the target dataset (the set to be classified), locate the k closest members (the k nearest neighbors) of the training dataset. A Euclidean Distance measure is used to calculate how close each member of the training set is to the target row that is being examined.
2. Examine the k nearest neighbors - which classification (category) do most of them belong to? Assign this category to the row being examined.
3. Repeat this procedure for the remaining rows (cases) in the target set.
4. also lets the user select a maximum value for k, builds models parallelly on all values of k upto the maximum specified value and scoring is done on the best of these models.

The **computing time** goes up as k goes up, but the **advantage** is that higher values of k provide smoothing that reduces vulnerability to noise in the training data.

In practical applications, typically, k is in units or tens rather than in hundreds or thousands.

Input:

Name	Gender	Height(m)
Kristina	F	1.6
Jim	M	2
Maggie	F	1.9
Bob	M	1.85
Dave	F	1.7
Kimm	M	1.9
Todd	M	1.9
Amy	F	1.85
Kathy	F	1.6

2m<=Tall,   1.7m< H<2m Medium,   H<=1.7m Short

New Tuple <Pat,F,1.6> , suppose K=5 is given than K nearest neighbors to input tuple {(Kristina,F,1.6) , (Kathy,F,1.6),(Dave,F,1.7)}

Output: Pat - Short

Post lab assignment:

1. Difference between simple approach of distance-based classification vs. K nearest neighborhood classification.

Problem 4

Title:

Implement decision tree based algorithm for classification

Objective:

- To learn decision tree based algorithm for classification

Reference:

- Data Mining Introductory & Advanced Topic by Margaret H. Dunham
- Data Mining Concept and Technique By Han & Kamber

Pre-requisite:

- Fundamental Knowledge of Database Management

Theory:

Decision tree learning, used in data mining and machine learning, uses a decision tree as a predictive model which maps observations about an item to conclusions about the item's target value. More descriptive names for such tree models are **classification trees** or **regression trees**. In these tree structures, leaves represent classifications and branches represent conjunctions of features that lead to those classifications.

In decision analysis, a decision tree can be used to visually and explicitly represent decisions and decision making. In data mining, a decision tree describes data but not decisions; rather the resulting classification tree can be an input for decision making.

- Basic steps
- Building tree
- Applying the tree to database

Internal node-test on attribute
Branch-outcome of test
Leaf node-class
Topmost-root node

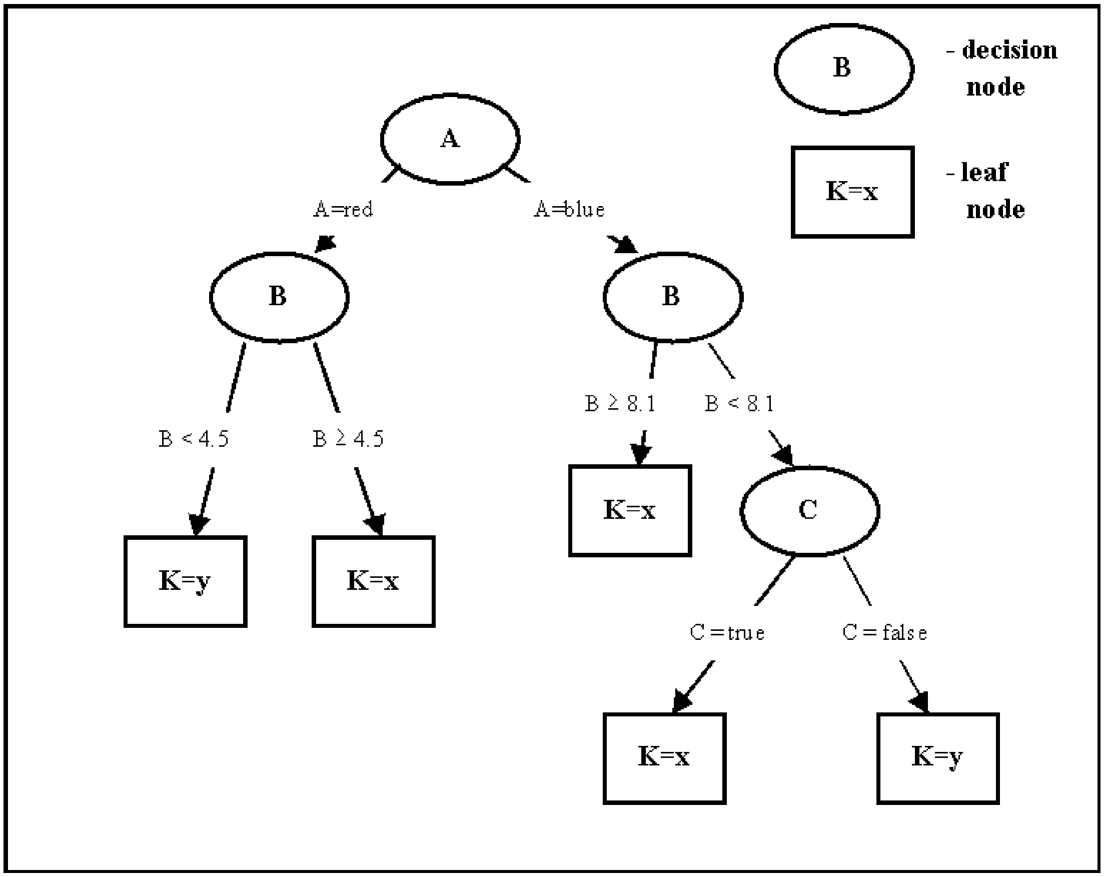
Algorithm

```
Create node N
If all the samples are from same class, C then
        return N as leaf node-labeled C

If attribute list is empty
   return N as leaf node assign common class name

Select test-attribute from attribute list which having highest information
Label N with test-attribute
(Determine the best splitting criteria)

For each value of attribute a of test-attribute
   grow ai branch from node N to condition test attribute
Assign test-value to arch
Si is set of samples of test-attribute ai
Repeat above
```



Post lab assignment:

1. What are the issue of Classification? Explain with example
2. Explain NN-based Algorithm.

Hierarchical clustering creates a hierarchy of clusters which may be represented in a tree structure called a dendrogram. The root of the tree consists of a single cluster containing all observations, and the leaves correspond to individual observations.

Algorithms for hierarchical clustering are generally either *agglomerative*, in which one starts at the leaves and successively merges clusters together; or *divisive*, in which one starts at the root and recursively splits the clusters.

Any valid metric may be used as a measure of similarity between pairs of observations. The choice of which clusters to merge or split is determined by a linkage criterion, which is a function of the pair wise distances between observations.

Cutting the tree at a given height will give a clustering at a selected precision. In the following example, cutting after the second row will yield clusters {a} {b c} {d e} {f}. Cutting after the third row will yield clusters {a} {b c} {d e f}, which is a coarser clustering, with a smaller number of larger clusters.

The algorithm forms clusters in a bottom-up manner, as follows:

1. Initially, put each article in its own cluster.
2. Among all current clusters, pick the two clusters with the smallest distance.
3. Replace these two clusters with a new cluster, formed by merging the two original ones.
4. Repeat the above two steps until there is only one remaining cluster in the pool.

Problem 5:

Title:

Implement K-means algorithm for clustering

Objective:

- To learn K-means algorithm for clustering

Reference:

- Data Mining Introductory & Advanced Topic by Margaret H. Dunham
- Data Mining Concept and Technique By Han & Kamber

Pre-requisite:

- Fundamental Knowledge of Database Management

Theory:

In statistics and machine learning, ***k*-means clustering** is a method of cluster analysis which aims to partition *n* observations into *k* clusters in which each observation belongs to the cluster with the nearest mean.

Here is step by step k means clustering algorithm:

Step 1. Begin with a decision on the value of k = number of clusters

Step 2. Put any initial partition that classifies the data into k clusters. You may assign the training samples randomly, or systematically as the following:

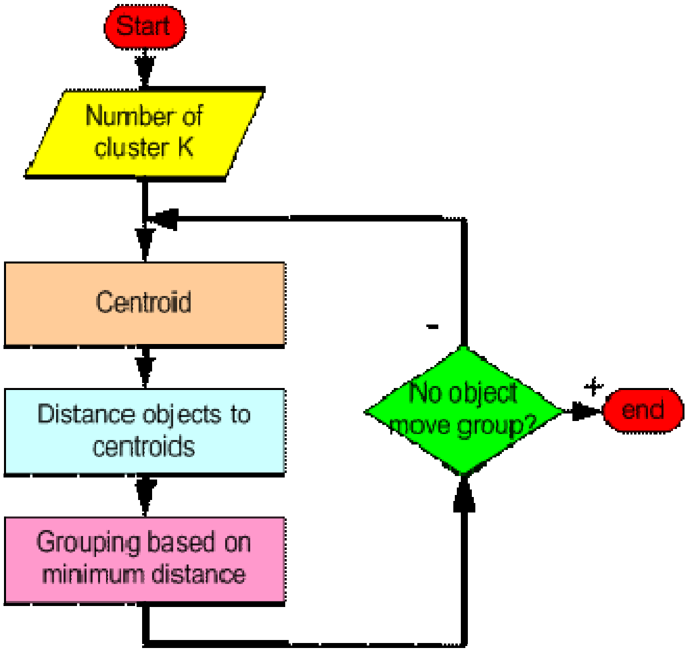
1. Take the first k training sample as single-element clusters

Assign each of the remaining (N-k) training sample to the cluster with the nearest centroid. After each assignment, recomputed the centroid of the gaining cluster.

Step 3 . Take each sample in sequence and compute its distance from the centroid of each of the clusters. If a sample is not currently in the cluster with the closest centroid, switch this sample to that cluster and update the centroid of the cluster gaining the new sample and the cluster losing the sample.

Step 4 . Repeat step 3 until convergence is achieved, that is until a pass through the training sample causes no new assignments.

If the number of data is less than the number of cluster then we assign each data as the centroid of the cluster. Each centroid will have a cluster number. If the number of data is bigger than the number of cluster, for each data, we calculate the distance to all centroid and get the minimum distance. This data is said belong to the cluster that has minimum distance from this data.

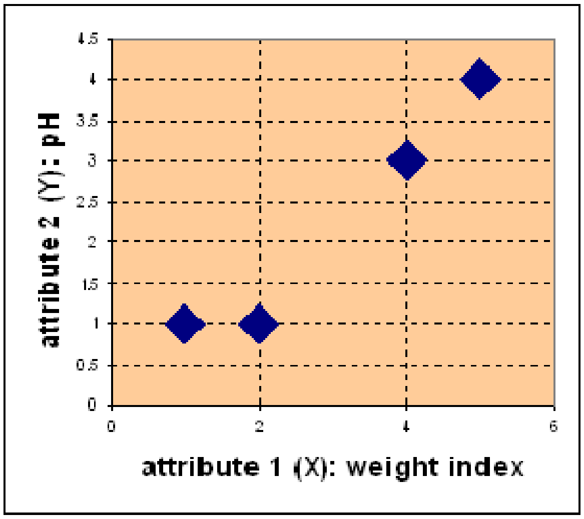

Start
Number of cluster K
Centroid
Distance objects to centroids
Grouping based on minimum distance
No object move group?
-
+
end

<u>Input</u>

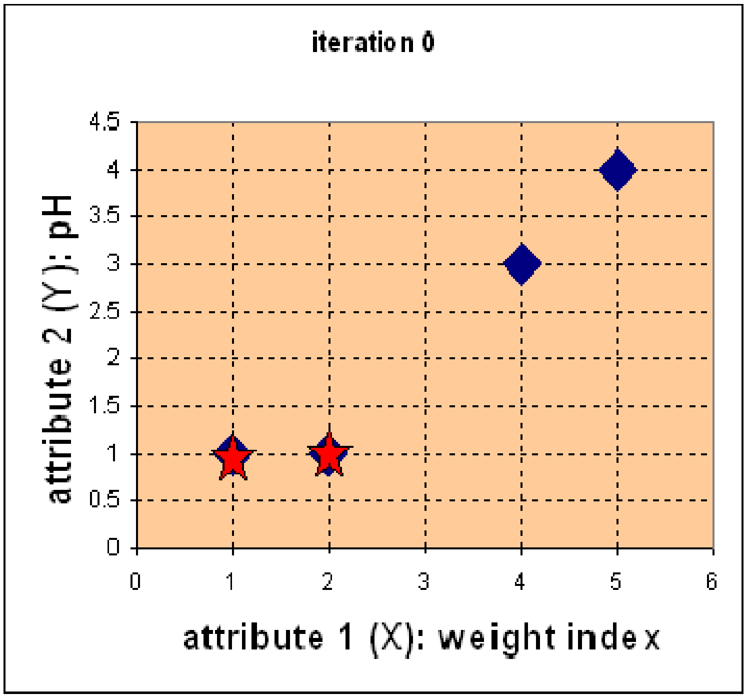
Suppose we have several objects (4 types of medicines) and each object have two attributes or features as shown in table below. Our goal is to group these objects into K=2 group of medicine based on the two features (pH and weight index).

Object	attribute 1 (X): weight index	attribute 2 (Y): pH
Medicine A	1	1
Medicine B	2	1
Medicine C	4	3
Medicine D	5	4

Each medicine represents one point with two attributes (X, Y) that we can represent it as coordinate in an attribute space as shown in the figure below.



1. *Initial value of centroids* : Suppose we use medicine A and medicine B as the first centroids. Let c_1 and c_2 denote the coordinate of the centroids, then $c_1 = (1,1)$ and $c_2 = (2,1)$



2. *Objects-Centroids distance* : we calculate the distance between cluster centroid to each object. Let us use Euclidean distance, then we have distance matrix at iteration 0 is

$$\mathbf{D}^0 = \begin{bmatrix} 0 & 1 & 3.61 & 5 \\ 1 & 0 & 2.83 & 4.24 \end{bmatrix} \quad \begin{matrix} c_1 = (1,1) & group-1 \\ c_2 = (2,1) & group-2 \end{matrix}$$

$$\begin{array}{l} \quad \begin{matrix} A & B & C & D \end{matrix} \\ \begin{bmatrix} 1 & 2 & 4 & 5 \\ 1 & 1 & 3 & 4 \end{bmatrix} \begin{matrix} X \\ Y \end{matrix} \end{array}$$

Each column in the distance matrix symbolizes the object. The first row of the distance matrix corresponds to the distance of each object to the first centroid and the second row is the distance of each object to the second centroid. For example, distance from medicine C = (4, 3) to the first centroid $c_1 = (1,1)$ is $\sqrt{(4-1)^2+(3-1)^2} = 3.61$, and its distance to the second centroid $c_2 = (2,1)$ is $\sqrt{(4-2)^2+(3-1)^2} = 2.83$, etc.

3. *Objects clustering* : We assign each object based on the minimum distance. Thus, medicine A is assigned to group 1, medicine B to group 2, medicine C to group 2 and medicine D to group 2. The element of Group matrix below is 1 if and only if the object is assigned to that group.

$$\begin{array}{l} \mathbf{G}^0 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 \end{bmatrix} \begin{matrix} group-1 \\ group-2 \end{matrix} \\ \qquad \begin{matrix} A & B & C & D \end{matrix} \end{array}$$

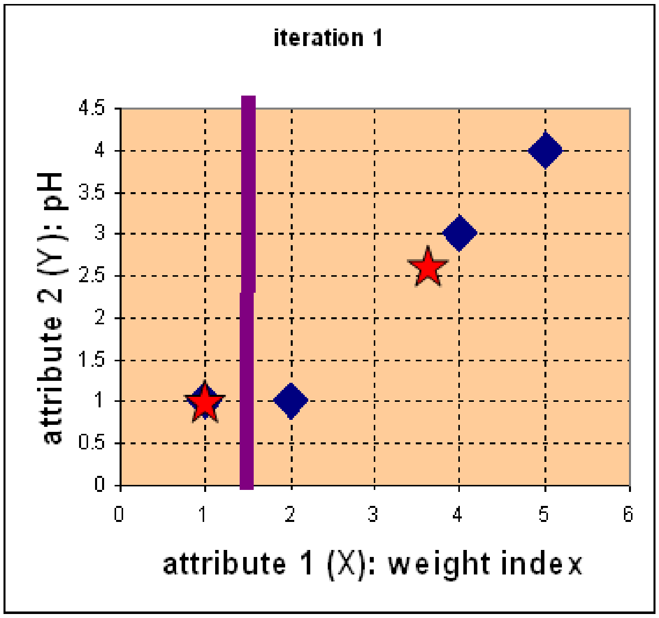
4. *Iteration-1, determine centroids* : Knowing the members of each group, now we compute the new centroid of each group based on these new memberships. Group 1 only has one member thus the centroid remains in $c_1 = (1,1)$. Group 2 now has three members, thus the centroid is the average coordinate among the three members:

$$c_2 = \left(\frac{2+4+5}{3}, \frac{1+3+4}{3}\right) = \left(\tfrac{11}{3}, \tfrac{8}{3}\right).$$

5. *Iteration-1, Objects-Centroids distances* : The next step is to compute the distance of all objects to the new centroids. Similar to step 2, we have distance matrix at iteration 1 is

$$\mathbf{D}^1 = \begin{bmatrix} 0 & 1 & 3.61 & 5 \\ 3.14 & 2.36 & 0.47 & 1.89 \end{bmatrix} \quad \begin{matrix} c_1 = (1,1) & group-1 \\ c_2 = (\frac{11}{3}, \frac{8}{3}) & group-2 \end{matrix}$$

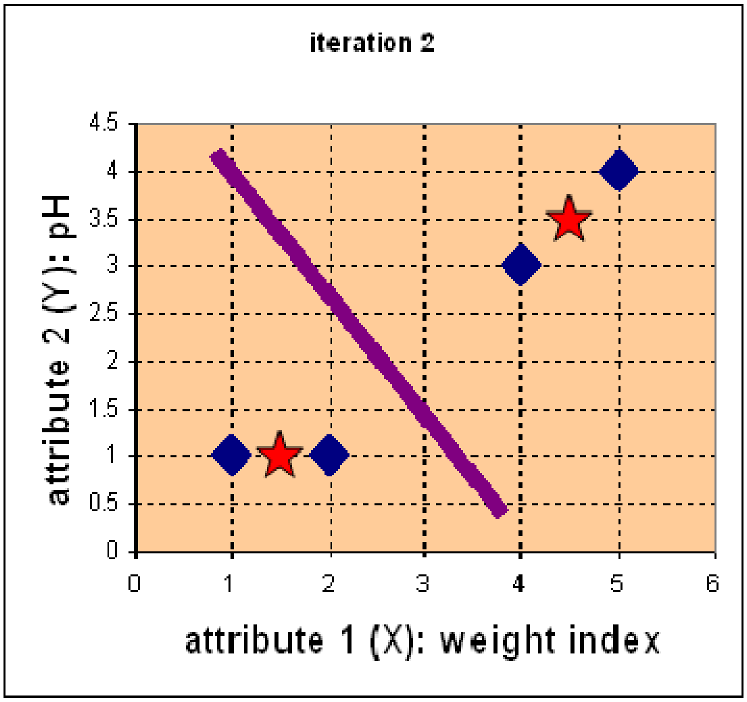
$$\begin{array}{l} \quad \begin{matrix} A & B & C & D \end{matrix} \\ \begin{bmatrix} 1 & 2 & 4 & 5 \\ 1 & 1 & 3 & 4 \end{bmatrix} \begin{matrix} X \\ Y \end{matrix} \end{array}$$



6. *Iteration-1, Objects clustering:* Similar to step 3, we assign each object based on the minimum distance. Based on the new distance matrix, we move the medicine B to Group 1 while all the other objects remain. The Group matrix is shown below

$$G^1 = \begin{array}{cc} \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix} & \begin{array}{l} group-1 \\ group-2 \end{array} \\ \begin{array}{cccc} A & B & C & D \end{array} & \end{array}$$

7. *Iteration 2, determine centroids:* Now we repeat step 4 to calculate the new centroids coordinate based on the clustering of previous iteration. Group1 and group 2 both has two members, thus the new centroids are $c_1 = (\frac{1+2}{2}, \frac{1+1}{2}) = (1\frac{1}{2}, 1)$ and $c_2 = (\frac{4+5}{2}, \frac{3+4}{2}) = (4\frac{1}{2}, 3\frac{1}{2})$



8. *Iteration-2, Objects-Centroids distances* : Repeat step 2 again, we have new distance matrix at iteration 2 as

$$\mathbf{D}^2 = \begin{bmatrix} 0.5 & 0.5 & 3.20 & 4.61 \\ 4.30 & 3.54 & 0.71 & 0.71 \end{bmatrix} \begin{array}{l} \mathbf{c}_1 = (1\frac{1}{2}, 1) \quad group-1 \\ \mathbf{c}_2 = (4\frac{1}{2}, 3\frac{1}{2}) \quad group-2 \end{array}$$

$$\begin{array}{cccc|c} A & B & C & D & \\ \end{array}$$

$$\begin{array}{l} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 1 & 2 & 4 & 5 \\ 1 & 1 & 3 & 4 \end{bmatrix} \begin{array}{l} X \\ Y \end{array} \end{array}$$

9. *Iteration-2, Objects clustering:* Again, we assign each object based on the minimum distance.

$$\mathbf{G}^2 = \begin{array}{l} \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix} \begin{array}{l} group-1 \\ group-2 \end{array} \\ \begin{array}{cccc} A & B & C & D \end{array} \end{array}$$

We obtain result that $\mathbf{G}^2 = \mathbf{G}^1$. Comparing the grouping of last iteration and this iteration reveals that the objects does not move group anymore. Thus, the computation of the k-mean clustering has reached its stability and no more iteration is needed.

Output:

We get the final grouping as the results

Object	Feature 1 (X): weight index	Feature 2 (Y): pH	Group (result)
Medicine A	1	1	1
Medicine B	2	1	1
Medicine C	4	3	2
Medicine D	5	4	2

Note: You can implement above problem no 3 to 6 in C/C++/JAVA

Problem 6:

Title:

Implement Apriori algorithm for association rule

Objective:

- To learn association rule for Apriori algorithm

Reference:

- Data Mining Introductory & Advanced Topic by Margaret H. Dunham
- Data Mining Concept and Technique By Han & Kamber

Pre-requisite:

- Fundamental Knowledge of Database Management

Theory:

Association rule mining is to find out association rules that satisfy the predefined minimum support and confidence from a given database. The problem is usually decomposed into two sub problems.

- Find those item sets whose occurrences exceed a predefined threshold in the database; those item sets are called frequent or large item sets.
- Generate association rules from those large item sets with the constraints of minimal confidence.

Suppose one of the large item sets is $L_k = \{I_1,I_2,...,I_k\}$; association rules with this item sets are generated in the following way: the first rule is $\{I_1,I_2,...,I_{k-1}\} => \{I_k\}$. By checking the confidence this rule can be determined as interesting or not. Then, other rules are generated by deleting the last items in the antecedent and inserting it to the consequent, further the confidences of the new rules are checked to determine the interestingness of them. This process iterates until the antecedent becomes empty.

Since the second sub problem is quite straight forward, most of the research focuses on the first sub problem. The Apriori algorithm finds the frequent sets *L* in Database *D*.

- Find frequent set L_{k-1}.
- Join Step.

    - C_k is generated by joining $L_{k-1}$with itself
- Prune Step.
    - Any (k − 1) -itemset that is not frequent cannot be a subset of a frequent k -itemset, hence should be removed.

where

- (C_k: Candidate itemset of size k)
- (L_k: frequent itemset of size k)

Apriori Pseudocode

```
Apriori $(T, \varepsilon)$
        $L_1 \leftarrow \{$large 1-itemsets that appear in more than $\varepsilon$ transactions $\}$
        $k \leftarrow 2$
while $L_{k-1} \neq \varnothing$
        $C_k \leftarrow$ Generate($L_{k-1}$)
        for transactions $t \in T$
        $C_t \leftarrow$ Subset($C_k$,$t$)
        for candidates $c \in C_t$
        $\mathrm{count}[c] \leftarrow \mathrm{count}[c] + 1$
        $L_k \leftarrow \{c \in C_k | \ \mathrm{count}[c] \geq \varepsilon\}$
        $k \leftarrow k + 1$
        return $\bigcup_k L_k$
```

Input :

A large supermarket tracks sales data by ***SKU(Stoke Keeping Unit)*** (item), and thus is able to know what items are typically purchased together. Apriori is a moderately efficient way to build a list of frequent purchased item pairs from this data. **Let the database of transactions consist of the sets {1,2,3,4}, {2,3,4}, {2,3}, {1,2,4}, {1,2,3,4}, and {2,4}.**

Output

Each number corresponds to a product such as "butter" or "water". The first step of Apriori to count up the frequencies, called the supports, of each member item separately:

Item	Support
1	3
2	6
3	4
4	5

We can define a minimum support level to qualify as "frequent," which depends on the context. For this case, let min support = 3. Therefore, all are frequent. The next step is to generate a list of all 2-pairs of the frequent items. Had any of the above items not been frequent, they wouldn't have been included as a possible member of possible 2-item pairs. In this way, Apriori *prunes* the tree of all possible sets..

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

This is counting up the occurrences of each of those pairs in the database. Since minsup=3, we don't need to generate 3-sets involving {1,3}. This is because since they're not frequent, no supersets of them can possibly be frequent. Keep going:

Item	Support

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

Post lab assignment:

1. Give an example for Apriori with transaction and explain Apriori-gen-algorithm

Problem 7:

Title:

Bayesian Classification

Objective:

- To implement classification using Bayes theorm.

Reference:

- Data Mining Introductory & Advanced Topic by Margaret H. Dunham
- Data Mining Concept and Technique By Han & Kamber

Pre-requisite:

- Fundamental Knowledge of probability and Bayes theorm

Theory:

The simple baysian classification assumes that the effect of an attribute value of a given class membership is independent of other attribute.

The Bayes theorm is as follows –

Let X be an unknown sample. Let it be hypothesis such that X belongs to particular class C. We need to determine P(H/X).

The probability that hypothesis it holds is given that all values of X are observed.

$$P(H/X) = \frac{P(X/H).P(H)}{P(X)}$$

In this program, we initially take the number of tuples in training data set in variable L.

The string array's name, gender, hight, output to store the details and output respectfully. Therefore, the tuple details are taken from user using 'for' loops.

Bayesian classification has an expected classification. Now using the counter variables for various attributes i.e. (male/female) for gender and (short/medium/tall) for hight.

The tuples are scanned and the respective counter is incremented accordingly using if-else-if structure.

Therefore variables pshort, pmed, plong are used to convert the counter variables to corresponding values.

Algorithm –

1. START
2. Store the training data set
3. Specify ranges for classifying the data
4. Calculate the probability of being tall, medium, short
5. Also, calculate the probabilities of tall, short, medium according to gender and classification ranges
6. Calculate the likelihood of short, medium and tall
7. Calculate P(t) by summing up of probable likelihood
8. Calculate actual probabilities

Input :

Training data set

Name	**Gender**	**Height**	**Output**
Christina	F	1.6m	Short
Jim	M	1.9m	Tall
Maggie	F	1.9m	Medium
Martha	F	1.88m	Medium
Stephony	F	1.7m	Medium
Bob	M	1.85m	Short
Dave	M	1.7m	Short
Steven	M	2.1m	Tall
Amey	F	1.8m	Medium

Output

The tuple belongs to the class having highest probability. Thus new tuple is classified.

Post lab assignment:

Problem 8:

Title:

Linear Regression

Objective:

- To write a program to classify the tuples using linear regression.

Reference:

- Data Mining Introductory & Advanced Topic by Margaret H. Dunham
- Data Mining Concept and Technique By Han & Kamber

Pre-requisite:

- Knowledge of regression techniques

Theory:

Regression problem deals with estimation of output values based on input values. In the method we estimate the formula of straight line, which partitions data into 2 classes

- by defining the regression coefficient C, the relation between output parameter Y and input parameter X1, X2, X3 ….. Xn can be estimated

Input :

Training data set

Name	Gender	Height
Christina	F	1.6m
Jim	M	1.9m
Maggie	F	1.9m
Martha	F	1.88m
Stephony	F	1.7m
Bob	M	1.85m
Dave	M	1.7m
Steven	M	2.1m
Amey	F	1.8m

Output

The tuple is being classified using linear regression technique. Having value > 0.5 is classified as medium else < 0.5 then tuple is classified as short.

Post lab assignment:

Problem 9:

Title:

Minimum Spanning based clustering

Objective:

- To write a program to implement Minimum Spanning based clustering.

Reference:

- Data Mining Introductory & Advanced Topic by Margaret H. Dunham
- Data Mining Concept and Technique By Han & Kamber

Pre-requisite:

- Knowledge of single link technique and tree data structure called dendogram.

Theory:

Algorithm –

1. Store the dendogram structure
2. Store the clusters at each level
3. Search the adjacency matrix for smallest distance among clusters
4. Form the cluster
5. Add newly formed cluster to the vector
6. Form dendogram
7. Merge clusters based on their single link distance

Input :

Adjacency matrix

Output

Minimum Spanning tree

Post lab assignment:

Problem 10:

Title:

Introduction to the Weka machine learning toolkit

Objective

To learn to use the Weak machine learning toolkit

References

Witten, Ian and Eibe, Frank. *Data Mining: Practical Machine Learning Tools and Techniques.* Springer.

Requirements

How do you load Weka?

1. What options are available on main panel?
2. What is the purpose of the the following in Weka:
   1. The Explorer
   2. The Knowledge Flow interface
   3. The Experimenter
   4. The command-line interface
3. Describe the arff file format.
4. Press the Explorer button on the main panel and load the weather dataset and answer the following questions
   1. How many instances are there in the dataset?
   2. State the names of the attributes along with their types and values.
   3. What is the class attribute?
   4. In the histogram on the bottom-right, which attributes are plotted on the X,Y-axes? How do you change the attributes plotted on the X,Y-axes?
   5. How will you determine how many instances of each class are present in the data
   6. What happens with the Visualize All button is pressed?
   7. How will you view the instances in the dataset? How will you save the changes?
5. What is the purpose of the following in the Explorer Panel?
   1. The Preprocess panel
      1. What are the main sections of the Preprocess panel?
      2. What are the primary sources of data in Weka?
   2. The Classify panel
   3. The Cluster panel

    4. The Associate panel
    5. The Select Attributes panel
    6. The Visualize panel.
6. Load the iris dataset and answer the following questions:
    1. How many instances are there in the dataset?
    2. State the names of the attributes along with their types and values.
    3. What is the class attribute?
    4. In the histogram on the bottom-right, which attributes are plotted on the X,Y-axes? How do you change the attributes plotted on the X,Y-axes?
    5. How will you determine how many instances of each class are present in the data
    6. What happens with the Visualize All button is pressed?
7. Load the weather dataset and perform the following tasks:
    1. Use the unsupervised filter RemoveWithValues to remove all instances where the attribute 'humidity' has the value 'high'?
    2. Undo the effect of the filter.
    3. Answer the following questions:
        1. What is meant by filtering in Weka?
        2. Which panel is used for filtering a dataset?
        3. What are the two main types of filters in Weka?
        4. What is the difference between the two types of filters? What is the difference between and attribute filter and an instance filter?
8. Load the iris dataset and perform the following tasks:
    1. Press the Visualize tab to view the Visualizer panel.
    2. What is the purpose of the Visualizer?
    3. Select one panel in the Visualizer and experiment with the buttons on the panel.

Postlab

Provide answers to all the questions given above.

Problem 11:

Title:

Classification using the Weka toolkit – Part 1

Objective

To perform classification on data sets using the Weka machine learning toolkit

References

Witten, Ian and Eibe, Frank. *Data Mining: Practical Machine Learning Tools and Techniques.* Springer.

Requirements

1. Load the 'weather.nominal.arff' dataset into Weka and run Id3 classification algorithm**.** Answer the following questions
    1. List the attributes of the given relation along with the type details
    2. Create a table of the weather.nominal.arff data
    3. Study the classifier output and answer the following questions
        1. Draw the decision tree generated by the classifier
        2. Compute the entropy values for each of the attributes
        3. What is the relationship between the attribute entropy values and the nodes of the decision tree?
    4. Draw the confusion matrix? What information does the confusion matrix provide?
    5. Describe the Kappa statistic?
    6. Describe the following quantities:
        1. TP Rate
        2. FP Rate
        3. Precision
        4. Recall
2. Load the 'weather.arff' dataset in Weka and run the Id3 classification algorithm. What problem do you have and what is the solution?
3. Load the 'weather.arff' dataset in Weka and run the OneR rule generation algorithm. Write the rules that were generated.
4. Load the 'weather.arff' dataset in Weka and run the PRISM rule generation algorithm. Write down the rules that are generated.

Postlab

Provide answers to all the questions given above.

Problem 12:

Title

Classification using the Weka toolkit – Part 2

Objective

To perform classification on datasets using the Weka toolkit

References

Witten, Ian and Eibe, Frank. *Data Mining: Practical Machine Learning Tools and Techniques.* Springer.

Requirements

1. Load the glass.arff dataset and perform the following tasks?
   1. How many items are there in the dataset?
   2. List the attributes are there in the dataset.
   3. List the classes in the dataset along with the count of instances in the class.
   4. How will you determine the color assigned to each class?
   5. By examining the histogram, how will you determine which attributes should be the most important in classifying the types of glass?
2. Perform the following classification tasks:
   1. Run the 1Bk classifier for various values of K?
   2. What is the accuracy of this classifier for each value of K?
   3. What type of classifier is the 1Bk classifier?
3. Perform the following classification tasks:
   1. Run the J48 classifier
   2. What is the accuracy of this classifier?
   3. What type of classifier is the J48 classifier?
4. Compare the results of the 1Bk and the J48 classifiers. Which is better?
5. Run the J48 and 1Bk classifiers using
   1. the cross-validation strategy with various fold levels. Compare the accuracy results.
   2. holdout strategy with three percentage levels. Compare the accuracy results.
6. Perform following tasks:
   1. Remove instances belonging to the following classes:
      1. build wind float
      2. build wind non-float
   2. Perform classification using the 1Bk and J48 classifiers. What is the effect of this filter on

the accuracy of the classifiers?

7. Perform the following tasks:
    1. Run the J48 and the NaiveBayes classifiers on the following datasets and determine the accuracy:
        1. vehicle.arff
        2. kr-vs-kp.arff
        3. glass.arff
        4. wave-form-5000.arff

        On which datasets does the NaiveBayes perform better? Why?

8. Perform the following tasks
    1. Use the results of the J48 classifier to determine the most important attributes
    2. Remove the least important attributes
    3. Run the J48 and 1Bk classifiers and determine the effect of this change on the accuracy of these classifiers. What will you conclude from the results?

Postlab

Provide answers to all the questions given above.

Problem 13:

Title

Performing data preprocessing tasks for data mining in Weka

Objective

To learn how to use various data preprocessing methods as a part of the data mining

References

Witten, Ian and Eibe, Frank. *Data Mining: Practical Machine Learning Tools and Techniques.* Springer.

Requirements

Part A: Application of Discretization Filters

Perform the following tasks

1. Load the 'sick.arff' dataset
2. How many instances does this dataset have?
3. How many attributes does it have?
4. Which is the class attribute and what are the characteristics of this attribute?
5. How many attributes are numeric? What are the attribute indexes of the numerica attributes?
6. Apply the Naive Bayes classifier. What is the accuracy of the classifier?

2. Perform the following tasks:
   1. Load the 'sick.arff' dataset.
   2. Apply the supervised discretization filter.
   3. What is the effect of this filter on the attributes?
   4. How many distinct ranges have been created for each attribute?
   5. Undo the filter applied in the previous step.
   6. Apply the unsupervised discretization filter. Do this twice:
      1. In this step, set 'bins'=5
      2. In this step, set 'bins'=10
      3. What is the effect of the unsupervised filter filter on the datset?
   7. Run the the Naive Bayes classifier after apply the following filters
      1. Unsupervised discretized with 'bins'=5
      2. Unsupervised discretized with 'bins'=10

        3. Unsupervised discretized with 'bins''=20.
    8. Compare the accuracy of the following cases
        1. Naive Bayes without discretization filters
        2. Naive Bayes with a supervised discretization filter
        3. Naive Bayes with an unsupervised discretization filter with different values for the 'bins' attributes.

Part B: Attribute Selection

1. Perform the following tasks:
    1. Load the 'mushroom.arff' dataset
    2. Run the J48, 1Bk, and the Naive Bayes classifiers.
    3. What is the accuracy of each of these classifiers?
2. Perform the following tasks:
    1. Go to the 'Select Attributes' panel
    2. Set attribute evaluator to CFSSubsetEval
    3. Set the search method to 'Greedy Stepwise'
    4. Analyze the results window
    5. Record the attribute numbers of the most important attributes
    6. Run the meta classifier AttributeSelectedClassifier using the following:
        1. CFSSubsetEval
        2. GreedStepwise
        3. J48, 1Bk, and NaiveBayes
    7. Record the accuracy of the classifiers
    8. What are the benefits of attribute selection?

Part C

1. Perform the following tasks:
    1. Load the 'vote.arff' dataset.
    2. Run the J48, 1Bk, and Naive Bayes classifiers.
    3. Record the accuracies.
2. Perform the following tasks:
    1. Go to the 'Select Attributes' panel
    2. Set attribute evaluator to 'WrapperSubsetEval'
    3. Set search method to ''RankSearch'
    4. Set attribute evaluator to 'InfoGainAttributeEval'
    5. Analyze the results
    6. Run the metaclassifier AttributeSelectedClassifier using the following:
        1. WrapperSubsetEval
        2. RankSearch
        3. InfoGainAttributeEval
    7. Sampling
        1. Load the 'letter.arff' dataset
        2. Take any attribute and record the min, max, mean, and standard deviation of the attribute

3. Apply the Resample filter with 'sampleSizePercent' set to 50 percent
4. What is the size of the filtered dataset. Observe the min, max, mean, and standard deviation of the attribute that was selected in step 2. What is the percentage change in the values?
5. Give the benefit of sampling a large dataset.

Postlab

Provide answers to all the questions given above.