



SHREE L. R. TIWARI COLLEGE OF ENGINEERING

An Autonomous Institute, Affiliated to University of Mumbai Approved by AICTE & DTE, Maharashtra State
NAAC Accredited, NBA Accredited Program | ISO 9001:2015 Certified | DTE Code No: 3423
Recognized under Section 2(f) of the UGC Act 1956 | Minority Status (Hindi Linguistic)
Miraroad (East), Thane - 401 107 - Maharashtra

EXPERIMENT NO. 1

TITLE: One case study on building Data warehouse/Data Mart

AIM: Write Detailed Problem statement and design dimensional modelling (creation of star and snowflake schema)

THEORY:

Dimensional Modelling:

It is a data structure technique optimized for data storage in a Data warehouse. The purpose of dimensional modeling is to optimize the database for faster retrieval of data. The concept of Dimensional Modelling was developed by Ralph Kimball and consists of “fact” and “dimension” tables. A dimensional model in a data warehouse is designed to read, summarize, analyze numeric information like values, balances, counts, weights, etc. in a data warehouse.

For instance, in the relational mode, normalization and ER models reduce redundancy in data. On the contrary, dimensional models in data warehouses arrange data in such a way that it is easier to retrieve information and generate reports.

Elements of Dimensional Data Model:

1) Fact:

Facts are the measurements/metrics or facts from your business process. For a sales business process, a measurement would be quarterly sales number.

2) Dimension:

Dimension provides the context surrounding a business process event. In simple terms, they give who, what, where of a fact. In the Sales business process, for the fact quarterly sales number, dimensions would be

- Who – Customer Names
- Where – Location
- What – Product Name

In other words, a dimension is a window to view information in the facts.

3) Attributes:

The Attributes are the various characteristics of the dimension in dimensional data modeling. In the Location dimension, the attributes can be

- State
- Country
- Zip code etc.



SHREE L. R. TIWARI COLLEGE OF ENGINEERING

An Autonomous Institute, Affiliated to University of Mumbai Approved by AICTE & DTE, Maharashtra State
NAAC Accredited, NBA Accredited Program | ISO 9001:2015 Certified | DTE Code No: 3423
Recognized under Section 2(f) of the UGC Act 1956 | Minority Status (Hindi Linguistic)
Miraroad (East), Thane - 401 107 - Maharashtra

4) Fact Table:

A fact table is a primary table in dimension modeling. Fact Table contains

- Measurements/facts
- Foreign key to dimension table

5) Dimension Table:

A dimension table contains dimensions of a fact. They are joined to fact table via a foreign key. Dimension tables are de-normalized tables. The Dimension Attributes are the various columns in a dimension table. Dimensions offer descriptive characteristics of the facts with the help of their attributes. No set limit is set for given number of dimensions. The dimension can also contain one or more hierarchical relationships.

Steps of Dimensional Modelling

The accuracy in creating your Dimensional modeling determines the success of your data warehouse implementation. Here are the steps to create a Dimension Model. The model should describe the Why, How much, When/Where/Who and What of your business process.

Step 1: Identifying the actual business process a data warehouse should cover. In our case, The business process covered by the data warehouse is the TOURISM MANAGEMENT SYSTEM. This involves tracking and analyzing **tourist bookings** made through various **tour packages**, including details about **tourists, locations, hotels, transport modes, and booking time**.

The selection of the Business process also depends on the quality of data available for that process. It is the most important step of the Data Modelling process, and a failure here would have cascading and irreparable defects.

Step 2: Identify the Grain-The Grain describes the level of detail for the business problem/solution. It is the process of identifying the lowest level of information for any table in your data warehouse. Each record in the fact table represents a single booking transaction made by a tourist for a particular package, at a specific location, using a certain transport mode, staying in a specific hotel, and booked at a specific time.

During this stage, you answer questions like:

Do we need to store all the available products or just a few types of products? This decision is based on the business processes selected for Data warehouse

Do we store the product sale information on a monthly, weekly, daily or hourly basis? This decision depends on the nature of reports requested by executives

How do the above two choices affect the database size?

Example of Gain:

"One record per booking per tourist, per package, per location, per transport, per hotel, and per time period."



SHREE L. R. TIWARI COLLEGE OF ENGINEERING

An Autonomous Institute, Affiliated to University of Mumbai Approved by AICTE & DTE, Maharashtra State
NAAC Accredited, NBA Accredited Program | ISO 9001:2015 Certified | DTE Code No: 3423
Recognized under Section 2(f) of the UGC Act 1956 | Minority Status (Hindi Linguistic)
Miraroad (East), Thane - 401 107 - Maharashtra

Step 3: Dimensions are nouns like date, store, inventory, etc. These dimensions are where all the data should be stored. For example, the date dimension may contain data like a year, month and weekday.

Example Dimensions in Tourism Management System:

Tourist Dimension

- tourist_key (PK), tourist_name, gender, age, nationality

Package Dimension

- package_key (PK), package_name, duration_days, category, agency_name

Location Dimension

- location_key (PK), city, state_or_province, country, tourist_spot_name

Transport Dimension

- transport_key (PK), transport_mode, transport_company, class_type

Hotel Dimension

- hotel_key (PK), hotel_name, hotel_category, city

Time Dimension

- time_key (PK), day, month, quarter, year

Each dimension provides descriptive information for analysis.

Step 4: Identify the Fact

This step is co-associated with the business users of the system because this is where they get access to data stored in the data warehouse. Most of the fact table rows are numerical values like price or cost per unit, etc.

Example of Facts: -The business wants to find the number of units sold and revenue generated per product, per store, per customer, per day and time.

Step 5: Build Schema-In this step, you implement the Dimension Model. A schema is nothing but the database structure (arrangement of tables). There are two popular schemas

- Star Schema
- Snowflake Schema

Star Schema:

A dimensional model with a fact table in the middle and dimension tables arranged around the fact table is called a star schema. Here, the fact table is at the core of the star and the dimension tables are along the spikes of the star. In this arrangement every attribute in the dimension table has the even chance to be participated in a query to analyze that attribute. Each dimension table is related to the fact table in a one-to- many relationship.

Star schema structure shows how the users normally view their metrics along their business dimensions. It can answer questions of what, when, by whom and whom etc. The answers are produced by combining one or more-dimension tables with the fact table. The relationship of a particular row in the fact tables with the rows in



SHREE L. R. TIWARI COLLEGE OF ENGINEERING

An Autonomous Institute, Affiliated to University of Mumbai Approved by AICTE & DTE, Maharashtra State
NAAC Accredited, NBA Accredited Program | ISO 9001:2015 Certified | DTE Code No: 3423
Recognized under Section 2(f) of the UGC Act 1956 | Minority Status (Hindi Linguistic)
Miraroad (East), Thane - 401 107 - Maharashtra

each dimension table. These relationships are shown as the spikes of the Star schema. Example: Star Schema for sales data

Snowflake schema:

A refinement of star schema where some dimensional hierarchy is further split (normalized) into a set of smaller dimension tables, forming a shape similar to snowflake. However, the snowflake structure can reduce the effectiveness of browsing, since more joins will be needed. The snowflake schema is a more complex data warehouse model than a star schema, and is a type of star schema. It is called a snowflake schema because the diagram of the schema resembles a snowflake.

CONCLUSION: Hence, we have understood the concept of dimensional modelling and we have created the dimensional modelling design using starschema and snowflake schema for the selected case study for building Data Ware house/Data Mart.



SHREE L. R. TIWARI COLLEGE OF ENGINEERING

An Autonomous Institute, Affiliated to University of Mumbai Approved by AICTE & DTE, Maharashtra State
NAAC Accredited, NBA Accredited Program | ISO 9001:2015 Certified | DTE Code No: 3423
Recognized under Section 2(f) of the UGC Act 1956 | Minority Status (Hindi Linguistic)
Miraroad (East), Thane - 401 107 - Maharashtra

EXPERIMENT NO. 2

TITLE: Implementation of all Dimension Table and Fact Table

AIM: Write an SQL query to implement Dimension Table and Fact Table based on Experiment 1 Case Study.

THEORY:

Fact Table:

In data warehousing, a fact table consists of the measurements, metrics or facts of a business process. It is located at the center of a star schema or a snowflake schema surrounded by dimension tables. Where multiple fact tables are used, these are arranged as a fact constellation schema. A fact table typically has two types of columns: those that contain facts and those that are foreign keys to dimension tables. The primary key of a fact table is usually a composite key that is made up of all of its foreign keys. Fact tables contain the content of the data warehouse and store different types of measures like additive, non-additive, and semi-additive measures. Fact tables provide the (usually) additive values that act as independent variables by which dimensional attributes are analyzed. Fact tables are often defined by their grain. The grain of a fact table represents the most atomic level by which the facts may be defined. The grain of a SALES fact table might be stated as "Sales volume by Day by Product by Store". Each record in this fact table is therefore uniquely defined by a day, product and store. Other dimensions might be members of this fact table (such as location/region) but these add nothing to the uniqueness of the fact records. These "affiliate dimensions" allow for additional slices of the independent facts but generally provide insights at a higher level of aggregation (a region contains many stores).

Dimension Table:

In data warehousing, a dimension table is one of the set of companion tables to a fact table. The fact table contains business facts (or measures), and foreign keys which refer to candidate keys (normally primary keys) in the dimension tables.

Contrary to fact tables, dimension tables contain descriptive attributes (or fields) that are typically textual fields (or discrete numbers that behave like text). These attributes are designed to serve two critical purposes: query constraining and/or filtering, and query result set labeling.

Dimension attributes should be:

- Verbose (labels consisting of full words)
- Descriptive
- Complete (having no missing values)
- Discretely valued (having only one value per dimension table row)
- Quality assured (having no misspellings or impossible values)



SHREE L. R. TIWARI COLLEGE OF ENGINEERING

An Autonomous Institute, Affiliated to University of Mumbai Approved by AICTE & DTE, Maharashtra State
NAAC Accredited, NBA Accredited Program | ISO 9001:2015 Certified | DTE Code No: 3423
Recognized under Section 2(f) of the UGC Act 1956 | Minority Status (Hindi Linguistic)
Miraroad (East), Thane - 401 107 - Maharashtra

Dimension table rows are uniquely identified by a single key field. It is recommended that the key field be a simple integer because a key value is meaningless, used only for joining fields between the fact and dimension tables.

OUTPUT:

Fact and Dimension Table of Star and Snowflake Schema

Fact Table:

```
-- Fact Table
CREATE TABLE Fact_Bookings (
    booking_id INTEGER PRIMARY KEY AUTOINCREMENT,
    tourist_key INTEGER,
    package_key INTEGER,
    location_key INTEGER,
    transport_key INTEGER,
    hotel_key INTEGER,
    time_key INTEGER,
    total_cost REAL,
    number_of_people INTEGER,

    FOREIGN KEY (tourist_key) REFERENCES Dim_Tourist(tourist_key),
    FOREIGN KEY (package_key) REFERENCES Dim_Package(package_key),
    FOREIGN KEY (location_key) REFERENCES Dim_Location(location_key),
    FOREIGN KEY (transport_key) REFERENCES Dim_Transport(transport_key),
    FOREIGN KEY (hotel_key) REFERENCES Dim_Hotel(hotel_key),
    FOREIGN KEY (time_key) REFERENCES Dim_Time(time_key)
);
```

● Dim_Tourist

```
CREATE TABLE Dim_Tourist (
    tourist_key INTEGER PRIMARY KEY AUTOINCREMENT,
    tourist_name TEXT,
    gender TEXT,
    age INTEGER,
    nationality TEXT
);
```



SHREE L. R. TIWARI COLLEGE OF ENGINEERING

An Autonomous Institute, Affiliated to University of Mumbai Approved by AICTE & DTE, Maharashtra State
NAAC Accredited, NBA Accredited Program | ISO 9001:2015 Certified | DTE Code No: 3423
Recognized under Section 2(f) of the UGC Act 1956 | Minority Status (Hindi Linguistic)
Miraroad (East), Thane - 401 107 - Maharashtra

- **Dim_Package**

```
CREATE TABLE Dim_Package (
    package_key INTEGER PRIMARY KEY AUTOINCREMENT,
    package_name TEXT,
    duration_days INTEGER,
    category TEXT,
    agency_name TEXT
);
```

- **Dim_Location**

```
CREATE TABLE Dim_Location (
    location_key INTEGER PRIMARY KEY AUTOINCREMENT,
    city TEXT,
    state_or_province TEXT,
    country TEXT,
    tourist_spot_name TEXT
);
```

- **Dim_Transport**

```
CREATE TABLE Dim_Transport (
    transport_key INTEGER PRIMARY KEY AUTOINCREMENT,
    transport_mode TEXT,
    transport_company TEXT,
    class_type TEXT
);
```

- **Dim_Hotel**

```
CREATE TABLE Dim_Hotel (
    hotel_key INTEGER PRIMARY KEY AUTOINCREMENT,
    hotel_name TEXT,
    hotel_category TEXT,
    city TEXT
);
```

- **Dim_Time**

```
CREATE TABLE Dim_Time (
    time_key INTEGER PRIMARY KEY AUTOINCREMENT,
    day INTEGER,
    month INTEGER,
    quarter INTEGER,
    year INTEGER
);
```



SHREE L. R. TIWARI COLLEGE OF ENGINEERING

An Autonomous Institute, Affiliated to University of Mumbai Approved by AICTE & DTE, Maharashtra State
NAAC Accredited, NBA Accredited Program | ISO 9001:2015 Certified | DTE Code No: 3423
Recognized under Section 2(f) of the UGC Act 1956 | Minority Status (Hindi Linguistic)
Miraroad (East), Thane - 401 107 - Maharashtra

Values of the Fact Table:

Fact_Bookings

booking_id	tourist_key	package_key	location_key	transport_key	hotel_key	time_key	total_cost	number_of_people
1	1	1	1	1	1	1	85000	2
2	2	2	2	2	2	2	60000	4

Values of Dimension Table:

Dim_Hotel

hotel_key	hotel_name	hotel_category	city
1	Taj Hotel	5-star	Agra
2	Mountain View Resort	3-star	Manali

Dim_Location

location_key	city	state_or_province	country	tourist_spot_name
1	Agra	Uttar Pradesh	India	Taj Mahal
2	Manali	Himachal Pradesh	India	Solang Valley

Dim_Package

package_key	package_name	duration_days	category	agency_name
1	Golden Triangle Tour	7	Cultural	TravelX Agency
2	Himalayan Adventure	10	Adventure	PeakTravel

Dim_Time

time_key	day	month	quarter	year
1	15	8	3	2025
2	2	1	1	2025

Dim_Tourist

tourist_key	tourist_name	gender	age	nationality
1	John Doe	Male	32	USA
2	Priya Sharma	Female	27	India

Dim_Transport

transport_key	transport_mode	transport_company	class_type
1	Flight	Air India	Economy
2	Bus	Himachal Roadways	Deluxe

CONCLUSION: Hence, we have created and displayed Fact Table & Dimension Table.



SHREE L. R. TIWARI COLLEGE OF ENGINEERING

An Autonomous Institute, Affiliated to University of Mumbai Approved by AICTE & DTE, Maharashtra State
NAAC Accredited, NBA Accredited Program | ISO 9001:2015 Certified | DTE Code No: 3423
Recognized under Section 2(f) of the UGC Act 1956 | Minority Status (Hindi Linguistic)
Miraroad (East), Thane - 401 107 - Maharashtra

EXPERIMENT NO. 3

TITLE: One case study on building OLAP operations

AIM: Write Detailed Problem statement to implementation of OLAP operations: Slice, Dice, Rollup, Drilldown and Pivot based on experiment 1

THEORY:

Online Analytical Processing Server (OLAP) is based on the multidimensional data model. It allows managers, and analysts to get an insight of the information through fast, consistent, and interactive access to information. This chapter covers the types of OLAP, operations on OLAP, differences between OLAP, and statistical databases and OLTP.

Types of OLAP Servers

We have four types of OLAP servers –

- Relational OLAP (ROLAP)
- Multidimensional OLAP (MOLAP)
- Hybrid OLAP (HOLAP)
- Specialized SQL Servers

OLAP Operations

Since OLAP servers are based on a multidimensional view of data, we will discuss OLAP operations in multidimensional data.

Here is the list of OLAP operations – .

Roll-up

· Drill-down

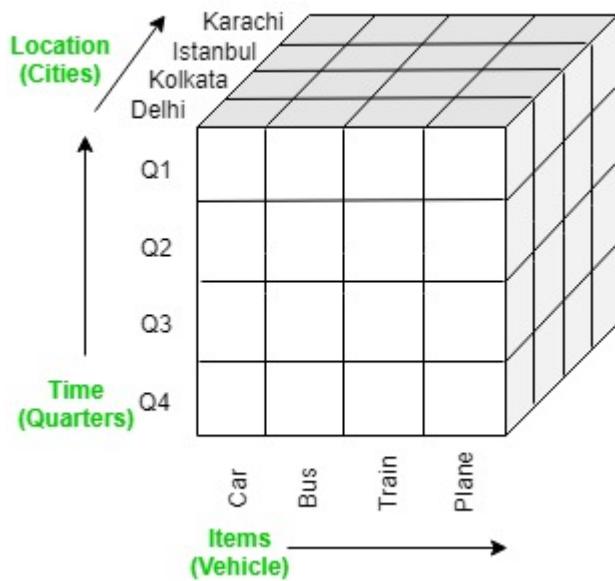
· Slice and dice

· Pivot (rotate)



SHREE L. R. TIWARI COLLEGE OF ENGINEERING

An Autonomous Institute, Affiliated to University of Mumbai Approved by AICTE & DTE, Maharashtra State
NAAC Accredited, NBA Accredited Program | ISO 9001:2015 Certified | DTE Code No: 3423
Recognized under Section 2(f) of the UGC Act 1956 | Minority Status (Hindi Linguistic)
Miraroad (East), Thane - 401 107 - Maharashtra

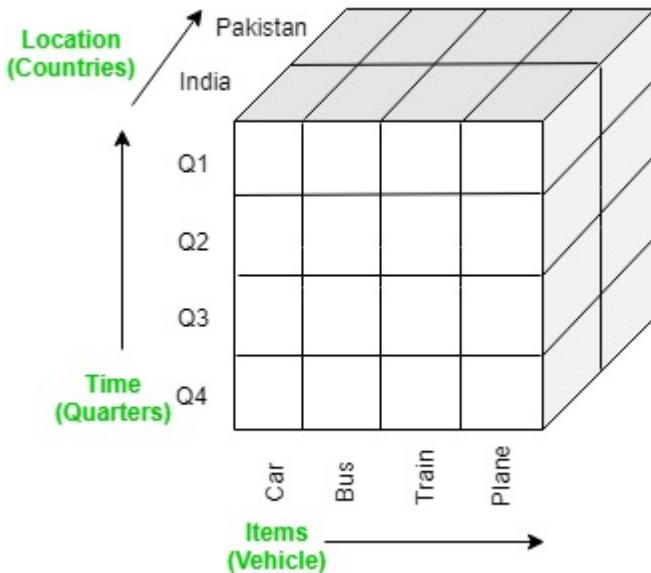


Roll-up

Roll-up performs aggregation on a data cube in any of the following ways –

- By climbing up a concept hierarchy for a dimension ·
- By dimension reduction

The following diagram illustrates how roll-up works.





SHREE L. R. TIWARI COLLEGE OF ENGINEERING

An Autonomous Institute, Affiliated to University of Mumbai Approved by AICTE & DTE, Maharashtra State
NAAC Accredited, NBA Accredited Program | ISO 9001:2015 Certified | DTE Code No: 3423
Recognized under Section 2(f) of the UGC Act 1956 | Minority Status (Hindi Linguistic)
Miraroad (East), Thane - 401 107 - Maharashtra

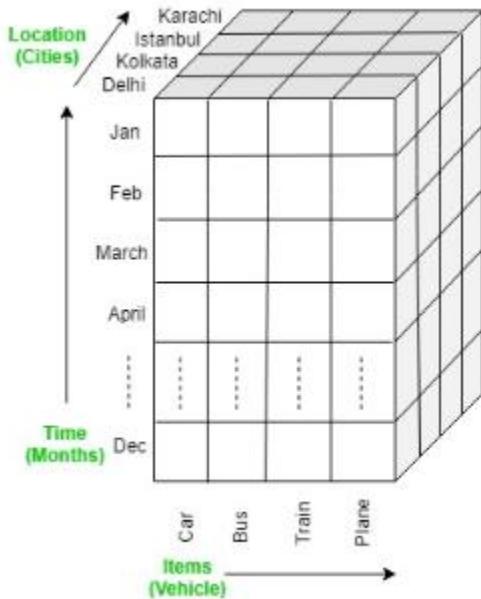
- Roll-up is performed by climbing up a concept hierarchy for the dimension Time. Initially the concept hierarchy was "Quarter < year".
- On rolling up, the data is aggregated by ascending the time hierarchy from the level of quarter to the level of year.
- The data is grouped into years rather than quarters.
- When roll-up is performed, one or more dimensions from the data cube are aggregated to higher level.

Drill-down

Drill-down is the reverse operation of roll-up. It is performed by either of the following ways –

By stepping down a concept hierarchy for a dimension · By introducing a new dimension.

- Drill-down is performed by stepping down a concept hierarchy for the dimension time. · Initially the concept hierarchy was "Year < Quarter < Month < Day."
- On drilling down, the Time dimension is descended from the level of the year to the level of Quarter. When drill-down is performed, one or more dimensions from the data cube are added. · It navigates the data from less detailed data to highly detailed data.





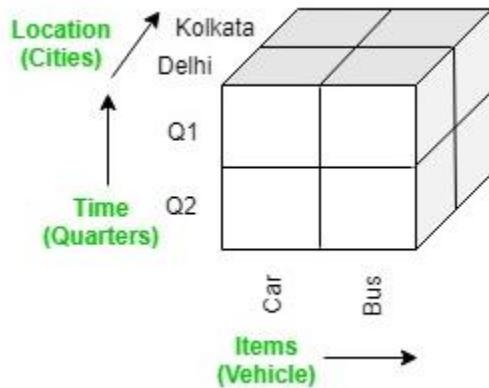
SHREE L. R. TIWARI COLLEGE OF ENGINEERING

An Autonomous Institute, Affiliated to University of Mumbai Approved by AICTE & DTE, Maharashtra State
NAAC Accredited, NBA Accredited Program | ISO 9001:2015 Certified | DTE Code No: 3423
Recognized under Section 2(f) of the UGC Act 1956 | Minority Status (Hindi Linguistic)
Miraroad (East), Thane - 401 107 - Maharashtra

Slice

The slice operation selects one particular dimension from a given cube and provides a new sub-cube. Consider the following diagram that shows how slice works.

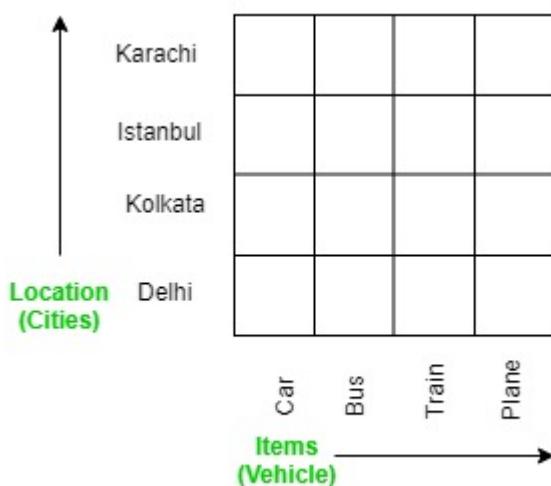
- Here Slice is performed for the Tourism Management System Cube by selecting one value from the third dimension (Customer or Region).
- It will form a new sub-cube by selecting one or more dimensions.



Dice

Dice selects two or more dimensions from a given cube and provides a new sub-cube. Consider the following diagram that shows the dice operation.

The dice operation on the cube based on the following selection criteria involves three dimensions.





SHREE L. R. TIWARI COLLEGE OF ENGINEERING

An Autonomous Institute, Affiliated to University of Mumbai Approved by AICTE & DTE, Maharashtra State
NAAC Accredited, NBA Accredited Program | ISO 9001:2015 Certified | DTE Code No: 3423
Recognized under Section 2(f) of the UGC Act 1956 | Minority Status (Hindi Linguistic)
Miraroad (East), Thane - 401 107 - Maharashtra

Pivot

The pivot operation is also known as rotation. It rotates the data axes in view in order to provide an alternative presentation of data. Consider the following diagram that shows the pivot operation.

Car			
Bus			
Train			
Plane			
Items (Vehicle)		Delhi	Kolkata
		Istanbul	Karachi
Location (Cities)			

CONCLUSION: Hence, we have studied building different OLAP operations.

TITLE: Study on Data Mining tools like WEKA & XLMiner.

AIM: Write a program to implement Naive Bayes classifier using WEKA tool

THEORY:

WEKA:

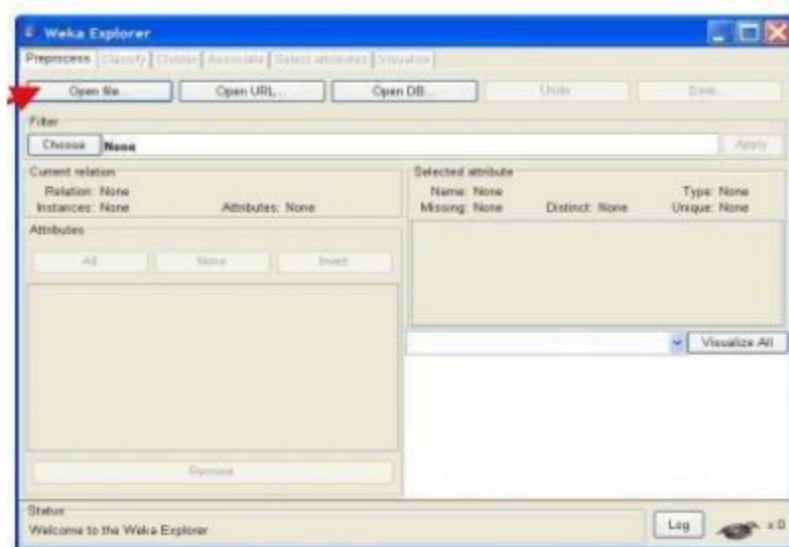
WEKA is a data mining system developed by the University of Waikato in New Zealand that implements data mining algorithms. WEKA is a state-of-the-art facility for developing machine learning (ML) techniques and their application to real-world data mining problems. It is a collection of machine learning algorithms for data mining tasks. The algorithms are applied directly to a dataset. WEKA implements algorithms for data preprocessing, classification, regression, clustering, association rules; it also includes a visualization tools. The new machine learning schemes can also be developed with this package. WEKA is open source software issued under the GNU General Public License. The goal of this Tutorial is to help you to learn WEKA Explorer. The tutorial will guide you step by step through the analysis of a simple problem using WEKA Explorer preprocessing, classification, clustering, association, attribute selection, and visualization tools. At the end of each problem there is a representation of the results with explanations side by side. Each part is concluded with the exercise for individual practice. By the time you reach the end of this tutorial, you will be able to analyze your data with WEKA Explorer using various learning schemes and interpret received results.:.

WEKA is a collection of tools for:

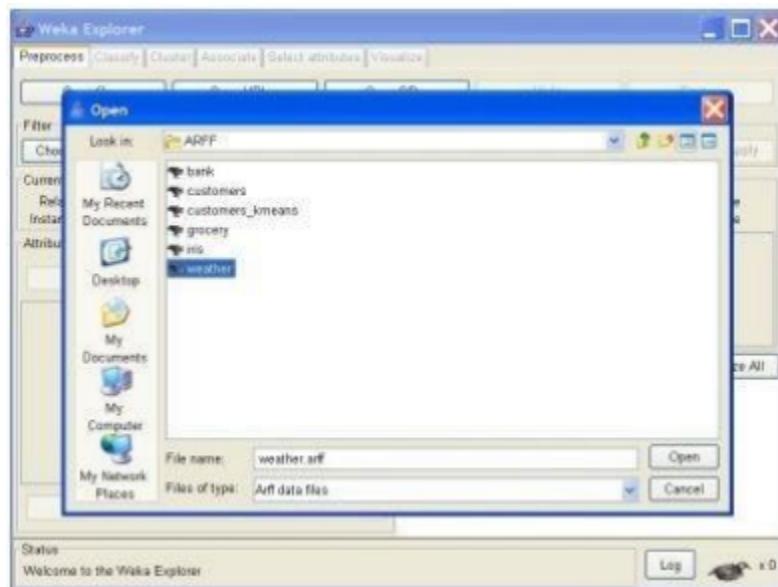
- Regression
- Clustering
- Association
- Data pre-processing
- Classification
- Visualization

Opening file from a local file system:

Click on ‘Open file...’ button.



It brings up a dialog box allowing you to browse for the data file on the local file system, choose “FileName.arff” file.



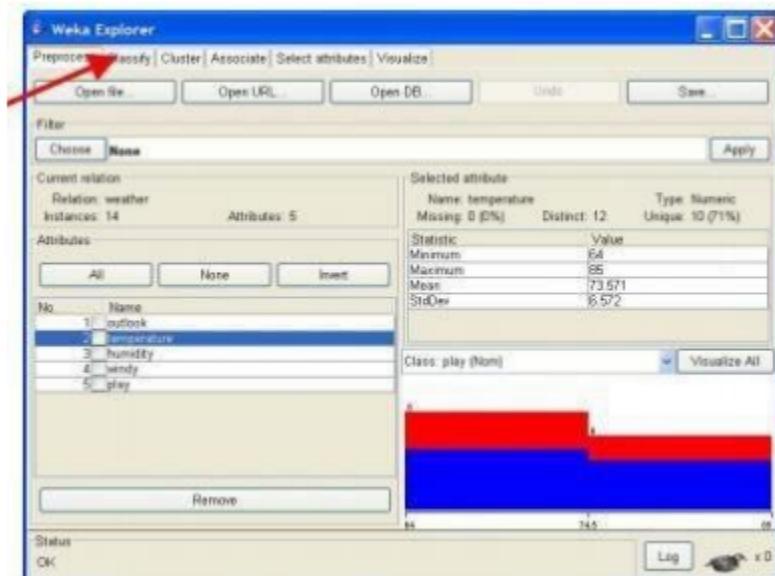
Some databases have the ability to save data in CSV format. In this case, you can select CSV file from the local filesystem. If you would like to convert this file into ARFF format, you can click on ‘Save’ button. WEKA automatically creates ARFF file from your CSV file.



Building “Classifiers”:

Classifiers in WEKA are the models for predicting nominal or numeric quantities. The learning schemes available in WEKA include decision trees and lists, instance-based classifiers, support vector machines, multi-layer perceptrons, logistic regression, and bayes’ nets. “Meta”- classifiers include bagging, boosting, stacking, error-correcting output codes, and locally weighted learning.

Once you have your data set loaded, all the tabs are available to you. Click on the ‘Classify’ tab.



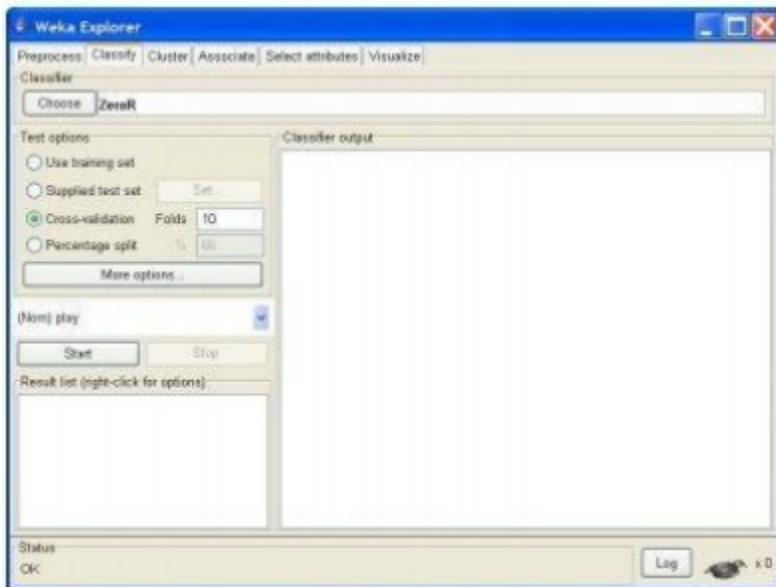


SHREE L. R. TIWARI COLLEGE OF ENGINEERING

An Autonomous Institute Affiliated to University of Mumbai,

Approved by AICTE & DTE, Maharashtra State | NAAC Accredited, NBA Accredited Program | ISO 9001:2015 Certified |
DTE Code No: 3423 | Recognized under Section 2(f) of the UGC Act 1956 | Minority Status (Hindi Linguistic)

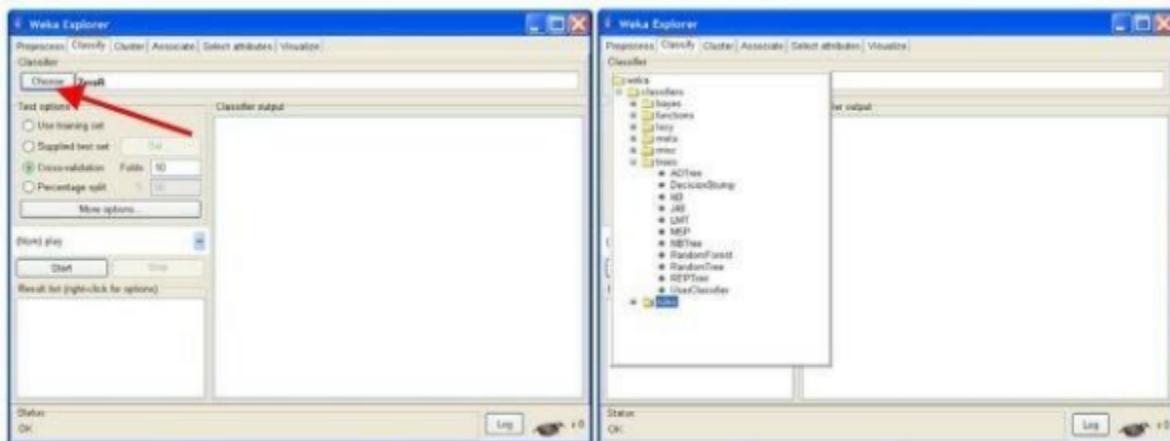
‘Classify’ window comes up on the screen.



Now you can start analyzing the data using the provided algorithms. In this exercise you will analyze the data with C4.5 algorithm using J48, WEKA’s implementation of decision tree learner. The sample data used in this exercise is the weather data from the file “weather.arff”. Since C4.5 algorithm can handle numeric attributes, in contrast to the ID3 algorithm from which C4.5 has evolved, there is no need to discretize any of the attributes. Before you start this exercise, make sure you do not have filters set in the ‘Preprocess’ window. Filter exercise in section 3.6 was just a practice.

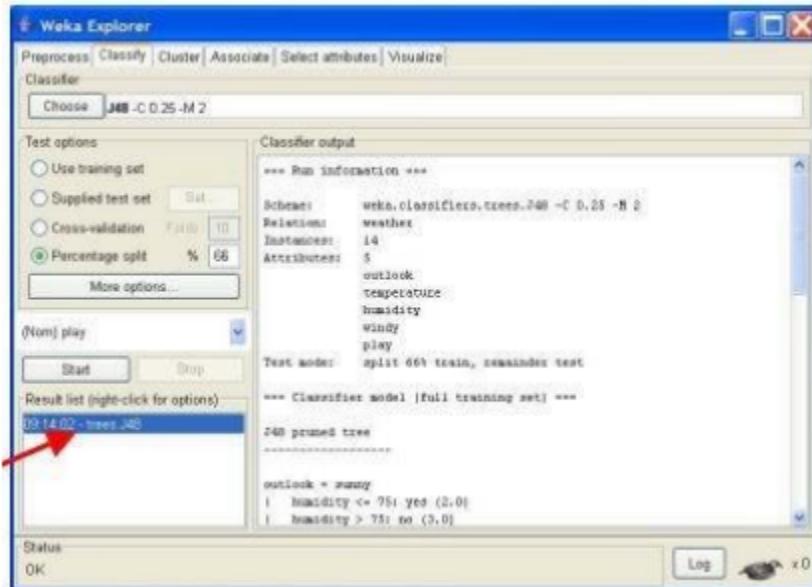
Choosing a Classifier:

Click on ‘Choose’ button in the ‘Classifier’ box just below the tabs and select C4.5 classifier WEKA → Classifiers → Trees → J48.

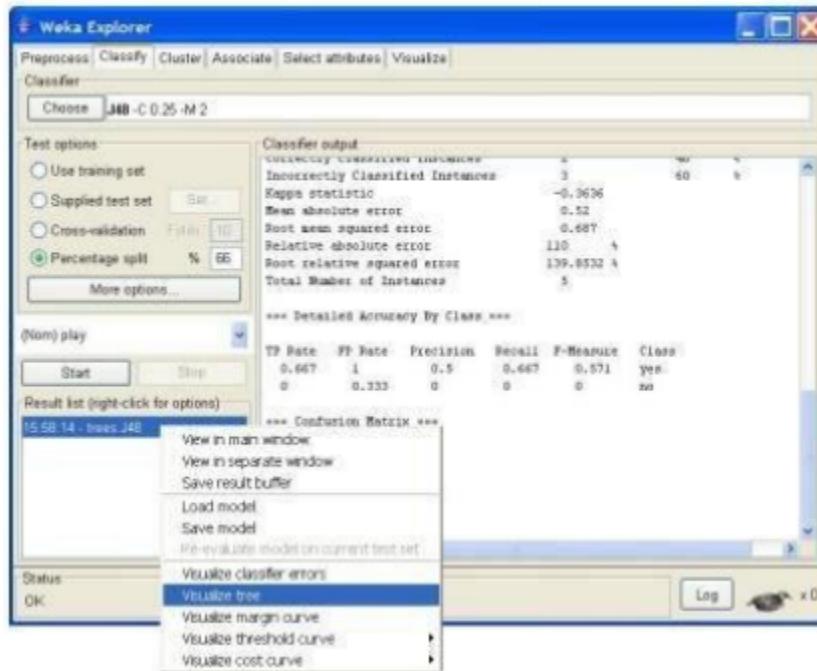


Visualization of Results:

After training a classifier, the result list adds an entry.



WEKA lets you see a graphical representation of the classification tree. Right-click on the entry in ‘Result list’ for which you would like to visualize a tree. It invokes a menu containing the following items:



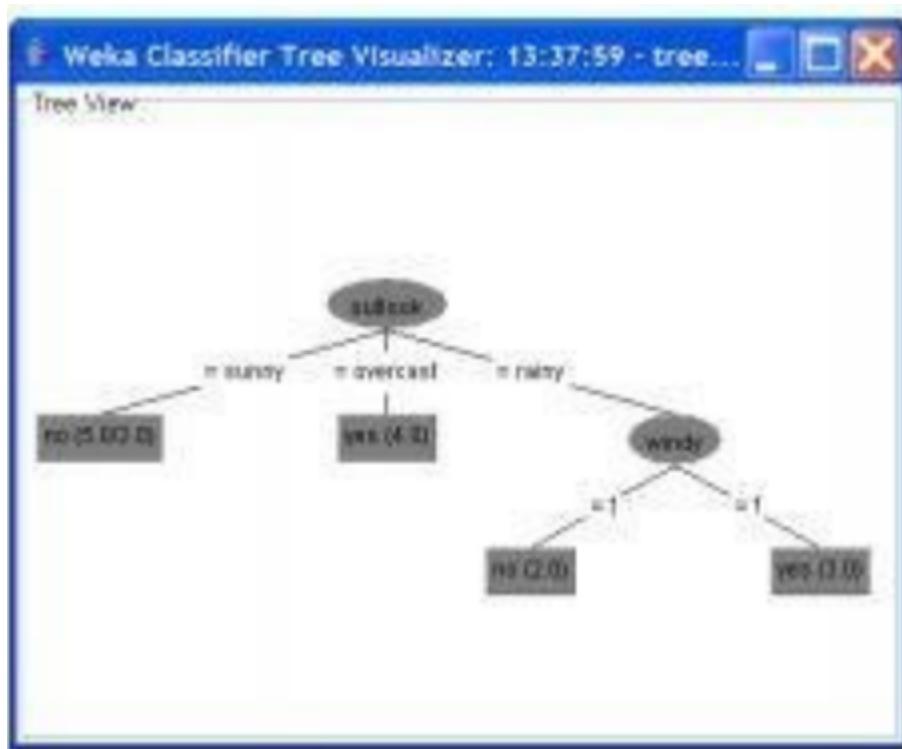
Select the item ‘Visualize tree’; a new window comes up to the screen displaying the tree.



SHREE L. R. TIWARI COLLEGE OF ENGINEERING

An Autonomous Institute Affiliated to University of Mumbai,

Approved by AICTE & DTE, Maharashtra State | NAAC Accredited, NBA Accredited Program | ISO 9001:2015 Certified |
DTE Code No: 3423 | Recognized under Section 2(f) of the UGC Act 1956 | Minority Status (Hindi Linguistic)



WEKA also lets you visualize classification errors. Right-click on the entry in 'Result list' again and select 'Visualize classifier errors' from the menu:

The screenshot shows the Weka Explorer interface. In the center, there is a table titled "Classifier output" with various performance metrics. Below it is a table titled "Detailed Accuracy By Class" and a "Confusion Matrix". On the left, there are "Test options" and a "Result list" containing the entry "15.50 14 : trees J48". A context menu is open over the "Result list" entry, with the option "Visualize classifier errors" highlighted.

CLASSIFIER OUTPUT	
INCORRECTLY CLASSIFIED INSTANCES	
Incorrectly Classified Instances	3
Kappa statistic	-0.3636
Mean absolute error	0.52
Root mean squared error	0.683
Relative absolute error	110 %
Root relative squared error	129.8532 %
Total Number of Instances	5

TP Rate	FP Rate	Precision	Recall	F-Measure	Class
0.667	1	0.5	0.667	0.571	Yes
0	0.333	0	0	0	No

*** Confusion Matrix ***

- View in main window
- View in separate window
- Save result buffer
- Load model
- Save model
- Re-evaluate model on current test set
- Visualize classifier errors
- Visualize tree
- Visualize margin curve
- Visualize threshold curve
- Visualize cost curve

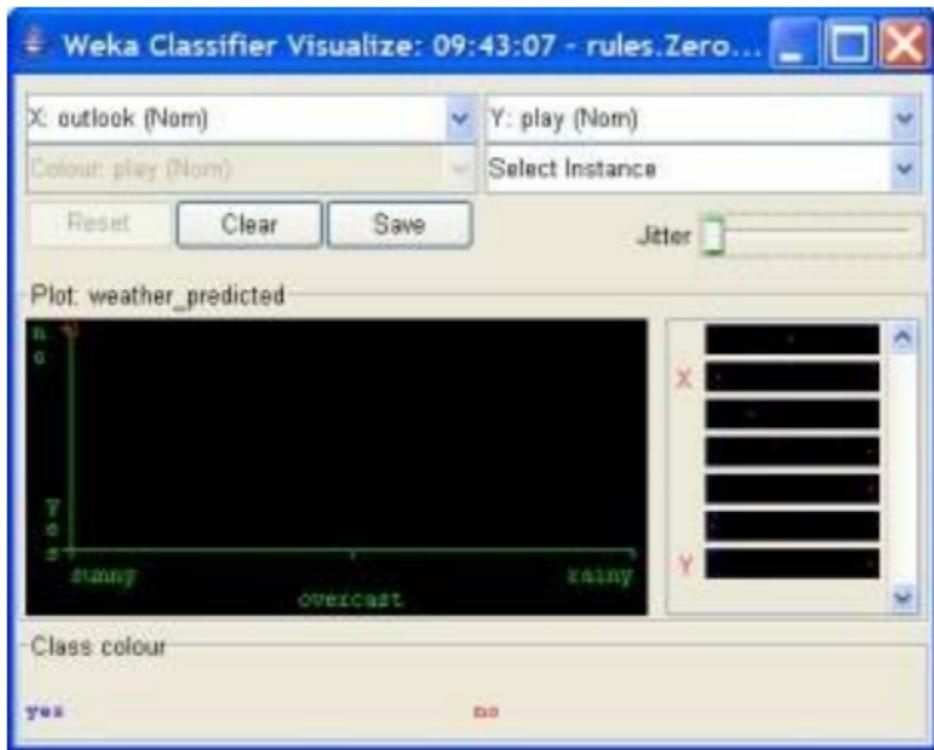


SHREE L. R. TIWARI COLLEGE OF ENGINEERING

An Autonomous Institute Affiliated to University of Mumbai,

Approved by AICTE & DTE, Maharashtra State | NAAC Accredited, NBA Accredited Program | ISO 9001:2015 Certified |
DTE Code No: 3423 | Recognized under Section 2(f) of the UGC Act 1956 | Minority Status (Hindi Linguistic)

Weka Classifier Visualize' window displaying graph appears on the screen.



XLMiner:

Data mining is a set of methods that helps an expert to obtain new, previously unknown significant information from available data. To apply data mining possibilities practically software that deploys data mining algorithms is needed. This software may also be integrated into other tools. One of these tools is the QuantLink XLMiner tool for Microsoft Excel. In this paper the data mining possibilities of XLMiner are examined. More precisely – this tool is used for extracting unknown client data, rules and patterns, as well as for extracting information that is significant for client relationship management.

Forecasting client activity class:

XLMiner offers to perform data sampling and partition using a data set that holds up to 60 000 records and 200 attributes. This includes data sampling to reduce the size of a data set (numbers of records) by choosing a subset of records that would represent the whole set. It can be done by random partitioning or partitioning with oversampling setting a desired support limit for a class.

Data set is larger than the limitations set by MS Excel or XLMiner; for this reason random partitioning was used to reduce the number of records. But the resulting data set would still have more attributes than it is



SHREE L. R. TIWARI COLLEGE OF ENGINEERING

An Autonomous Institute Affiliated to University of Mumbai,

Approved by AICTE & DTE, Maharashtra State | NAAC Accredited, NBA Accredited Program | ISO 9001:2015 Certified |
DTE Code No: 3423 | Recognized under Section 2(f) of the UGC Act 1956 | Minority Status (Hindi Linguistic)

allowed in this tool (30 attributes are allowed). Therefore a smaller subset of the attributes is needed, that doesn't lose much information in the reduction process. Therefore a tool named WEKA was used to pick a subset of 13 best attributes.

The data set was also divided into training and test data sets. Usually data sets are divided into training set holding 70% of the data and test set holding 30% of the data. This ratio was also used in this case study wherever such division was necessary (e.g. classification and forecasting). XLMiner offers solving a classification task using the following methods: discriminant analysis, logistic regression (not to be confused with logic regression, which works with binary data and classifies data using the Boolean algebra approach), classification trees (CART), naive Bayes classifier, neural networks (multilayer feedforward architectures) and k-nearest neighbours. Discriminant analysis does not work with the needed amount of data and logistic regression does not support 5 classes in the data in this tool (it only classifies 2 classes), therefore this task was solved using other methods and attributes of data types that are supported by these methods (e.g. continuous data was not used with naive Bayes classifier). The best method for this task was chosen using the total classification error and the wrong classifications into most important class (that is class 4) (see Table 1). Each client was assigned an activity class which shows how long a client will participate in the lottery, which means we can forecast whether this client will be a loyal customer and for how long he will participate). We consider the following classes that show 5 common patterns of client behaviour:

“0” no tickets were bought,

“1” only the first ticket was bought,

“2” no more than two tickets were bought,

“3” client bought all tickets for one season,

“4” client played during the first season and bought tickets for the second season (We consider this type of people ‘loyal customers’).

The importance of this task is its contribution to forecasting client behaviour for the company, it helps to forecast the number of players, money flow etc. If there is information about prospective customers (that are not clients of this lottery yet), it can also provide information for selective advertising campaigns, distributing the advertisements to prospective customers that will most likely become loyal customers.

An analysis of the classifiers' results enables us to evaluate the performance of the classifiers. It was done by evaluating the total classification error and the rates of the wrong classifications into each class (Table 1).



Classification errors (%)

Class	Classification tree	Naïve Bayes	K-nearest neighbours	Neural network
0	38.85	39.57	42.34	23.64
1	100.00	100.00	99.33	100.00
2	100.00	99.25	97.26	100.00
3	100.00	48.48	78.89	100.00
4	13.67	33.48	26.89	29.07
Total	45.44	47.75	49.24	49.24

Table 1

Some classifiers did not classify classes 1 to 3 but they were not the most important for this task. The most important was to determine the clients that will play in this lottery regularly (4th class). The classifier that had the lowest error for this class was classification tree. The same method also had the lowest total classification error. This means that the most appropriate method to solve this task is to build a classification tree. The use of this method gives us 86 right answers out of 100 when we want to know whether the client will become a loyal customer.

In this case study there were several inconveniences caused by data set limitations posed by software. Not only was it the record number limitation by Microsoft Excel, but also some unexpected limitations from the XLMiner tool. The work with 60 000 records (XLMiner limit) could not be completed even if the data set had less than 200 attributes that were.

XLMiner solving data mining tasks, concentrating on tasks that are relevant for client relationship management. XLMiner can solve classification, forecasting, clustering, time series analysis and associative rules tasks. And the classification task is the one that is most emphasized. This task can be solved with various methods that work with both categorical (naive Bayes classifier) and continuous data (k- nearest neighbours and others) and the results are comprehensive and more processed. For time series analysis and associative rules there are some of the most popular methods but these tasks are not the main application of this tool, although they can be used successfully as secondary analysis tools. The same is with clustering – some of the methods are offered for this task but result analysis is quite poor and so is visualization. This tool also lacks some data pre-processing utilities. In this case study some tools for attribute set analysis and reduction (subset selection) would have been handy but they are offered only for some regression methods (as built in functions). All of the data set was not used in any task because of the XLMiner limitations. So we have to conclude that this tool is more suitable for tasks with smaller data sets and as an illustration in the learning process for the tasks described previously.

Naive Bayes Algorithm:

It is a classification technique based on Bayes' Theorem with an assumption of independence among predictors. In simple terms, a Naive Bayes classifier assumes that the presence of a particular feature in a class is unrelated to the presence of any other feature. For example, a fruit may be considered to be an apple if it is red, round, and about 3 inches in diameter. Even if these features depend on each other or upon the existence of the other features, all of these properties independently contribute to the probability that this fruit is an apple and that is why it is known as 'Naive'.

Naive Bayes model is easy to build and particularly useful for very large data sets. Along with simplicity, Naive Bayes is known to outperform even highly sophisticated classification methods.

Bayes theorem provides a way of calculating posterior probability $P(c|x)$ from $P(c)$, $P(x)$ and $P(x|c)$. Look at the equation below:

$$P(c|x) = \frac{P(x|c)P(c)}{P(x)}$$

Likelihood *Class Prior Probability*
 ↓ ↑
 Posterior Probability Predictor Prior Probability

$$P(c|X) = P(x_1|c) \times P(x_2|c) \times \dots \times P(x_n|c) \times P(c)$$

- $P(c|x)$ is the posterior probability of class (c, target) given predictor (x, attributes).
- $P(c)$ is the prior probability of class.
- $P(x|c)$ is the likelihood which is the probability of predictor given class.
- $P(x)$ is the prior probability of predictor

Working Naive Bayes algorithm

Let's understand it using an example. Below I have a training data set of weather and corresponding target variable 'Play' (suggesting possibilities of playing). Now, we need to classify whether players will play or not based on weather condition. Let's follow the below steps to perform it.



SHREE L. R. TIWARI COLLEGE OF ENGINEERING

An Autonomous Institute Affiliated to University of Mumbai,

Approved by AICTE & DTE, Maharashtra State | NAAC Accredited, NBA Accredited Program | ISO 9001:2015 Certified |
DTE Code No: 3423 | Recognized under Section 2(f) of the UGC Act 1956 | Minority Status (Hindi Linguistic)

Step 1: Convert the data set into a frequency table

Step 2: Create Likelihood table by finding the probabilities like Overcast probability = 0.29 and probability of playing is 0.64.

Step 3: Now, use Naive Bayesian equation to calculate the posterior probability for each class. The class with the highest posterior probability is the outcome of prediction.

Weather	Play
Sunny	No
Overcast	Yes
Rainy	Yes
Sunny	Yes
Sunny	Yes
Overcast	Yes
Rainy	No
Rainy	No
Sunny	Yes
Rainy	Yes
Sunny	No
Overcast	Yes
Overcast	Yes
Rainy	No

Frequency Table		
Weather	No	Yes
Overcast		4
Rainy	3	2
Sunny	2	3
Grand Total	5	9

Likelihood table				
Weather	No	Yes		
Overcast		4	=4/14	0.29
Rainy	3	2	=3/14	0.36
Sunny	2	3	=5/14	0.36
All	5	9		
			=5/14	=9/14
			0.36	0.64

Problem: Players will play if weather is sunny. Is this statement correct?

We can solve it using above discussed method of posterior probability. $P(\text{Yes} \mid \text{Sunny}) = P(\text{Sunny} \mid \text{Yes}) * P(\text{Yes}) / P(\text{Sunny})$

Here we have $P(\text{Sunny} \mid \text{Yes}) = 3/9 = 0.33$, $P(\text{Sunny}) = 5/14 = 0.36$, $P(\text{Yes}) = 9/14 = 0.64$. Now, $P(\text{Yes} \mid \text{Sunny}) = 0.33 * 0.64 / 0.36 = 0.60$, which has higher probability.

Naive Bayes uses a similar method to predict the probability of different class based on various attributes. This algorithm is mostly used in text classification and with problems having multiple classes.

Pros and Cons of Naive Bayes

Pros:

1. It is easy and fast to predict class of test data set. It also performs well in multi class prediction. When assumption of independence holds, a Naive Bayes classifier performs better compare to other models like logistic regression and you need less training data.
2. It performs well in case of categorical input variables compared to numerical variable(s). For numerical



variable, normal distribution is assumed (bell curve, which is a strong assumption).

Cons:

1. If categorical variable has a category (in test data set), which was not observed in training data set, then model will assign a 0 (zero) probability and will be unable to make a prediction. This is often known as “Zero Frequency”. To solve this, we can use the smoothing technique. One of the simplest smoothing techniques is called Laplace estimation.
2. On the other side naive Bayes is also known as a bad estimator, so the probability outputs from predict_proba are not to be taken too seriously.

Applications of Naive Bayes Algorithms

- Real time Prediction: Naive Bayes is an eager learning classifier and it is sure fast. Thus, it could be used for making predictions in real time.
- Multi class Prediction: This algorithm is also well known for multi class prediction feature. Here we can predict the probability of multiple classes of target variable.
- Text classification/ Spam Filtering/ Sentiment Analysis: Naive Bayes classifiers mostly used in text classification (due to better result in multi class problems and independence rule) have higher success rate as compared to other algorithms. As a result, it is widely used in Spam filtering (identify spam e-mail) and Sentiment Analysis (in social media analysis, to identify positive and negative customer sentiments)
- Recommendation System: Naïve Bayes Classifier and Collaborative Filtering together builds a Recommendation System that uses machine learning and data mining techniques to filter unseen information and predict whether a user would like a given resource or not.

LAB EXERCISE:

1) Java Code

```
import java.io.*;
import java.util.*;
class element {
    double p[][];
    element(int n, int m) {
```



SHREE L. R. TIWARI COLLEGE OF ENGINEERING

An Autonomous Institute Affiliated to University of Mumbai,

Approved by AICTE & DTE, Maharashtra State | NAAC Accredited, NBA Accredited Program | ISO 9001:2015 Certified |
DTE Code No: 3423 | Recognized under Section 2(f) of the UGC Act 1956 | Minority Status (Hindi Linguistic)

```
p = new double[n][m];
}
}
class Classifier {
int no_attr = 0, no_rows = 0;
String fileArray[][];
String values[][];
double class_p[];
int count;
element a[];
void readFile(String fname) throws IOException {
FileInputStream in = null;
try {
File f = new File(fname);
in = new FileInputStream(f);
} catch (Exception e) {
System.out.println("ERROR WHILE READING FILE");
System.exit(1);
}
BufferedReader br = new BufferedReader(new InputStreamReader(in));
String input = "";
input = br.readLine();
StringTokenizer line = new StringTokenizer(input);
no_attr = line.countTokens() - 1;
fileArray = new String[100][no_attr + 1];
for (int i = 0; i <= no_attr; i++)
fileArray[no_rows][i] = line.nextToken();
while (true) {
input = br.readLine();
if (input == null)
break;
line = new StringTokenizer(input);
no_rows++;
for (int i = 0; i <= no_attr; i++)
fileArray[no_rows][i] = line.nextToken();
}
}
```



SHREE L. R. TIWARI COLLEGE OF ENGINEERING

An Autonomous Institute Affiliated to University of Mumbai,

Approved by AICTE & DTE, Maharashtra State | NAAC Accredited, NBA Accredited Program | ISO 9001:2015 Certified |
DTE Code No: 3423 | Recognized under Section 2(f) of the UGC Act 1956 | Minority Status (Hindi Linguistic)

```
getAllvalues();
createTable();
newEntry();
}
boolean in_values(int col_no, String temp) {
for (int i = 0; i <= count; i++)
if (values[i][col_no] != null && values[i][col_no].equals(temp))
return true;

}
continue;
void getAllvalues() {

values = new String[100][no_attr + 1];
count = 0;
String temp;

//col major order traversal for(int i=0;i<=no_attr;i++)

{
for (int j = 1; j <= no_rows; j++) {
temp = fileArray[j][i];
if (in_values(i, temp))
continue;
values[count++][i] = temp;
}
count = 0;
}
}

int getlen(int col_no) // returns no of rows in particular col_no in array 'values'
{
int i = 0;
while (values[i][col_no] != null)
i++;
return i;
}
void display() {
```



SHREE L. R. TIWARI COLLEGE OF ENGINEERING

An Autonomous Institute Affiliated to University of Mumbai,

Approved by AICTE & DTE, Maharashtra State | NAAC Accredited, NBA Accredited Program | ISO 9001:2015 Certified |
DTE Code No: 3423 | Recognized under Section 2(f) of the UGC Act 1956 | Minority Status (Hindi Linguistic)

```
for (int i = 0; i <= no_rows; i++) {  
    for (int j = 0; j <= no_attr; j++) {  
        System.out.print(values[i][j] + " ");  
        System.out.println();  
    }  
}  
  
if(fileArray[i][col_no].equals(temp))tc++;  
return tc;  
  
void createTable() {  
    int tp = getlen(no_attr);  
    class_p = new double[tp];  
    for (int i = 0; i < tp; i++) {  
        for (int j = 1; j <= no_rows; j++) {  
            if (values[i][no_attr].equals(fileArray[j][no_attr]))  
                class_p[i]++;  
        }  
        class_p[i] /= no_rows;  
        System.out.println("P(" + values[i][no_attr] + ")=" + class_p[i]);  
    }  
    int tc = 0;  
    a = new element[no_attr];  
    for (int i = 0; i < no_attr; i++) {  
        a[i] = new element(getlen(i), getlen(no_attr));  
        for (int j = 0; j < getlen(i); j++) {  
            for (int k = 0; k < getlen(no_attr); k++) {  
                for (int x = 1; x <= no_rows; x++) {  
                    if (values[j][i].equals(fileArray[x][i]) &&  
                        values[k][no_attr].equals(fileArray[x][no_attr]))  
                        tc++;  
                }  
            }  
            a[i].p[j][k] = (double) tc / getcount(values[k][no_attr], no_attr);  
            tc = 0;  
        }  
    }  
}
```



SHREE L. R. TIWARI COLLEGE OF ENGINEERING

An Autonomous Institute Affiliated to University of Mumbai,

Approved by AICTE & DTE, Maharashtra State | NAAC Accredited, NBA Accredited Program | ISO 9001:2015 Certified |
DTE Code No: 3423 | Recognized under Section 2(f) of the UGC Act 1956 | Minority Status (Hindi Linguistic)

```
}

void newEntry()// takes sample input from user and processes it
{
Scanner s = new Scanner(System.in);
System.out.println("\nEnter New Entry");
String entry[] = new String[no_attr];
double p_entry[] = new double[getlen(no_attr)];
String X = "X=<";
for (int i = 0; i < no_attr; i++) {
System.out.println("enter " + fileArray[0][i]);
entry[i] = s.next();
X += entry[i] + " ";
}

//Process
}
X += ">";
System.out.println("\nThe Unseen Sample is" + X + "\n");
double large = 0.0;
int pos = -1;
for (int i = 0; i < getlen(no_attr); i++) {
double product = 1.0;
for (int j = 0; j < no_attr; j++) {
product *= a[j].p[getindex(j, entry[j])][i];
}
p_entry[i] = class_p[i] * product;
System.out.println("P(X|" + values[i][no_attr] + ").P(" + values[i][no_attr] +
")=" + p_entry[i]);
if (p_entry[i] > large) {
large = p_entry[i];
pos = i;
}
}
System.out.println("\nThe Decision is " + values[pos][no_attr]);
}

int getindex(int col_no, String temp) {
for (int i = 0; i < getlen(col_no); i++) {
if (values[i][col_no].equals(temp))
```



SHREE L. R. TIWARI COLLEGE OF ENGINEERING

An Autonomous Institute Affiliated to University of Mumbai,

Approved by AICTE & DTE, Maharashtra State | NAAC Accredited, NBA Accredited Program | ISO 9001:2015 Certified |
DTE Code No: 3423 | Recognized under Section 2(f) of the UGC Act 1956 | Minority Status (Hindi Linguistic)

```
return i;

}

System.out.println("Invalid Entry");
return 32676;
}
}

class Bayes {
public static void main(String str[]) throws IOException {
Scanner s = new Scanner(System.in);
Classifier c = new Classifier();
System.out.println("Enter the Name of the input file with its extension");
c.readFile(s.next());
}
}
```

OUTPUT:

```
Enter the Name of the input file with its extension
Bayesinput1.txt
P(SUNBURNED)=0.375
P(NONE)=0.625

Enter New Entry
enter HAIR
BROWN
enter HEIGHT
TALL
enter WEIGHT
AVERAGE
enter DUBLIN
NO

The Unseen Sample isX=<BROWN TALL AVERAGE NO >

P(X|SUNBURNED).P(SUNBURNED)=0.0
P(X|NONE).P(NONE)=0.03200000000000001

The Decision is NONE
```



SHREE L. R. TIWARI COLLEGE OF ENGINEERING

An Autonomous Institute Affiliated to University of Mumbai,

Approved by AICTE & DTE, Maharashtra State | NAAC Accredited, NBA Accredited Program | ISO 9001:2015 Certified |
DTE Code No: 3423 | Recognized under Section 2(f) of the UGC Act 1956 | Minority Status (Hindi Linguistic)

2) WEKA Code

Weather DataSet

```
@relation weather.symbolic
@attribute outlook {sunny, overcast, rainy}
@attribute temperature {hot, mild, cool}
@attribute humidity {high, normal}
@attribute windy {TRUE, FALSE}
@attribute play {yes, no}
@data sunny,hot,high,TRUE,no
```

```
sunny,hot,high,TRUE,no
overcast,hot,high,FALSE,yes
rainy,mild,high,FALSE,yes
rainy,cool,normal,FALSE,yes
rainy,cool,normal,TRUE,no
overcast,cool,normal,TRUE,yes
sunny,mild,high,FALSE,no
sunny,cool,normal,FALSE,yes
rainy,mild,normal,FALSE,yes
sunny,mild,normal,TRUE,yes
overcast,mild,high,TRUE,yes
overcast,hot,normal,FALSE,yes
rainy,mild,high,TRUE,no
```

```
Weather test @relation
weather-test
@attribute outlook {sunny, overcast, rainy}
@attribute temperature {hot, mild, cool}
@attribute humidity {high, normal}
@attribute windy {TRUE, FALSE}
@attribute play {yes, no}
@data
rainy,hot,high,FALSE,no
sunny,cool,high,TRUE,no
```



SHREE L. R. TIWARI COLLEGE OF ENGINEERING

An Autonomous Institute Affiliated to University of Mumbai,

Approved by AICTE & DTE, Maharashtra State | NAAC Accredited, NBA Accredited Program | ISO 9001:2015 Certified |
DTE Code No: 3423 | Recognized under Section 2(f) of the UGC Act 1956 | Minority Status (Hindi Linguistic)

OUTPUT:

== Run information ==

Scheme: weka.classifiers.bayes.NaiveBayes

Relation: weather.symbolic

Instances: 14

Attributes: 5

outlook

temperature

humidity

windy

play

Test mode: user supplied test set: size unknown (reading incrementally)

== Classifier model (full training set) ==

Naive Bayes Classifier

Class

Attribute	yes	no
	(0.63)	(0.38)

outlook

sunny	3.0	4.0
overcast	5.0	1.0
Rainy	4.0	3.0
[total]	12.0	8.0

temperature

hot	3.0	3.0
mild	5.0	3.0
cool	4.0	2.0
[total]	12.0	8.0

humidity

high	4.0	5.0
normal	7.0	2.0
[total]	11.0	7.0

windy

TRUE	4.0	4.0
FALSE	7.0	3.0
[total]	11.0	7.0

Test Instance

Time taken to build model: 0 seconds

==== Evaluation on test set ===

Time taken to test model on supplied test set: 0 seconds

==== Summary ===

Correctly Classified Instances	2	100 %
Incorrectly Classified Instances	0	0 %
Kappa statistic	1	
Mean absolute error	0.3461	
Root mean squared error	0.3555	
Relative absolute error	55.3714 %	
Root relative squared error	56.8819 %	
Total Number of Instances	2	

==== Detailed Accuracy By Class ===

TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area
Class							
0.000	0.000	0.000	0.000	0.000 0.000	?	?	yes
1.000	0.000	1.000	1.000	1.000 0.000	?	1.000	no
Weighted Avg.		1.000	0.000	1.000 1.000	1.000	0.000	0.000 1.000

==== Confusion Matrix ===

a b <-- classified as

0 0 | a = yes

0 2 | b = no

CONCLUSION: Hence, we have studied Data Mining tools like WEKA & XLMiner and implemented classifier Naive Bayes using Java and WEKA.



SHREE L. R. TIWARI COLLEGE OF ENGINEERING

An Autonomous Institute, Affiliated to University of Mumbai Approved by AICTE & DTE, Maharashtra State
NAAC Accredited, NBA Accredited Program | ISO 9001:2015 Certified | DTE Code No: 3423
Recognized under Section 2(f) of the UGC Act 1956 | Minority Status (Hindi Linguistic)
Miraroad (East), Thane - 401 107 - Maharashtra

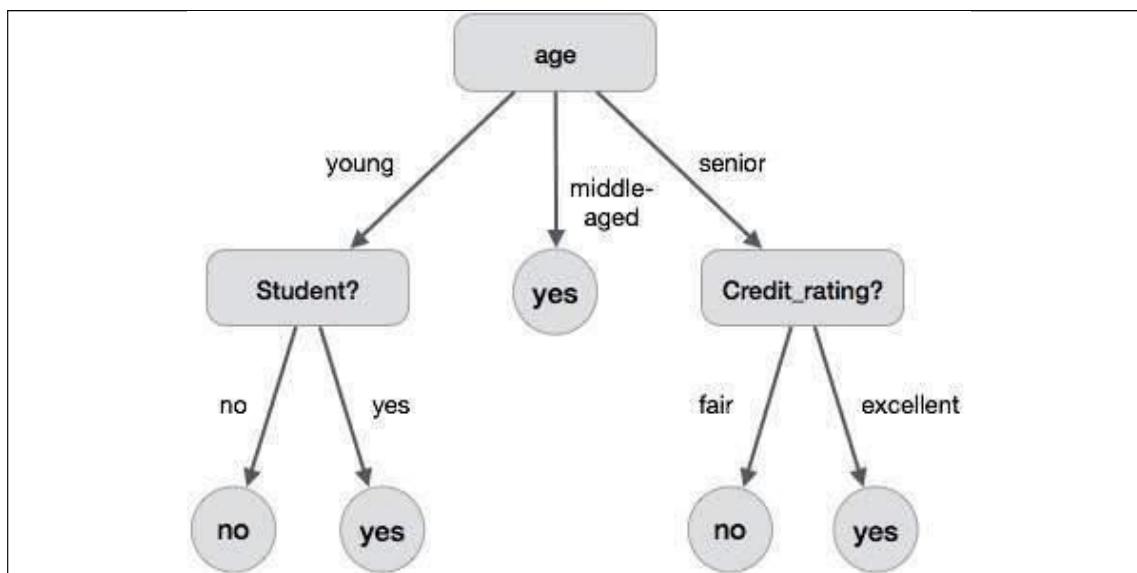
EXPERIMENT NO.5

TITLE: Implementation of Decision Tree

AIM: Write a program to implement Decision Tree using Java and WEKA

THEORY: A decision tree is a decision support tool that uses a tree-like graph or model of decisions and their possible consequences, including chance event outcomes, resource costs, and utility. It is one way to display an algorithm.

A decision tree is a structure that includes a root node, branches, and leaf nodes. Each internal node denotes a test on an attribute, each branch denotes the outcome of a test, and each leaf node holds a class label. The topmost node in the tree is the root node. The following decision tree is for the concept buy_computer that indicates whether a customer at a company is likely to buy a computer or not. Each internal node represents a test on an attribute. Each leaf node represents a class.



The benefits of having a decision tree are as

follows – It does not require any domain knowledge.

It is easy to comprehend.

The learning and classification steps of a decision tree are simple and fast.

Decision Tree Induction Algorithm

A machine researcher named J. Ross Quinlan in 1980 developed a decision tree algorithm known as ID3 (Iterative Dichotomiser). Later, he presented C4.5, which was the successor of ID3. ID3 and C4.5 adopt a



SHREE L. R. TIWARI COLLEGE OF ENGINEERING

An Autonomous Institute, Affiliated to University of Mumbai Approved by AICTE & DTE, Maharashtra State
NAAC Accredited, NBA Accredited Program | ISO 9001:2015 Certified | DTE Code No: 3423
Recognized under Section 2(f) of the UGC Act 1956 | Minority Status (Hindi Linguistic)
Miraroad (East), Thane - 401 107 - Maharashtra

greedy approach. In this algorithm, there is no backtracking; the trees are constructed in a top-down recursive divide-and-conquer manner.

Generating a decision tree from training tuples of data partition D

Algorithm: Input: Generate_decision_tree

Data partition, D, which is a set of training tuples and their associated class labels. attribute_list, the set of candidate attributes.

Attribute selection method, a procedure to determine the splitting criterion that best partitions the data tuples into individual classes. This criterion includes a splitting_attribute and either a splitting point or splitting subset.

Output:

A Decision Tree

Method

create a node N;

if tuples in D are all of the same class, C then

 return N as leaf node labeled with class C;

if attribute_list is empty then

 return N as leaf node with labeled
 with majority class in D;|| majority voting

apply attribute_selection_method(D, attribute_list) to find the best splitting_criterion;

label node N with splitting_criterion;

if splitting_attribute is discrete-valued and multiway splits allowed then

 // no restricted to binary trees
 attribute_list = splitting_attribute; // remove splitting
 attribute for each outcome j of splitting criterion



SHREE L. R. TIWARI COLLEGE OF ENGINEERING

An Autonomous Institute, Affiliated to University of Mumbai Approved by AICTE & DTE, Maharashtra State
NAAC Accredited, NBA Accredited Program | ISO 9001:2015 Certified | DTE Code No: 3423
Recognized under Section 2(f) of the UGC Act 1956 | Minority Status (Hindi Linguistic)
Miraroad (East), Thane - 401 107 - Maharashtra

// partition the tuples and grow subtrees for each partition let Dj be

the set of data tuples in D satisfying outcome j; // a partition

if Dj is empty then attach a leaf labeled with the majority class

in D to node N;

else

attach the node returned by Generate decision tree(Dj, attribute list) to node N;

end for

return

N;

LAB EXERCISE:

1) Java Code

```
import java.util.*;  
  
class Node {  
    String attribute;  
    String label;  
    Map<String, Node> children = new HashMap<>();  
  
    boolean isLeaf() {  
        return label != null;  
    }  
}  
  
public class Main {  
    private static double entropy(List<String[]> data, int labelIndex) {  
        Map<String, Integer> freq = new HashMap<>();  
        for (String[] row : data) {  
            freq.put(row[labelIndex], freq.getOrDefault(row[labelIndex], 0) + 1);  
        }  
        double entropy = 0.0;      int total  
        = data.size();           for (int count :  
        freq.values()) {          double p =  
        (double) count / total;
```



SHREE L. R. TIWARI COLLEGE OF ENGINEERING

An Autonomous Institute, Affiliated to University of Mumbai Approved by AICTE & DTE, Maharashtra State
NAAC Accredited, NBA Accredited Program | ISO 9001:2015 Certified | DTE Code No: 3423
Recognized under Section 2(f) of the UGC Act 1956 | Minority Status (Hindi Linguistic)
Miraroad (East), Thane - 401 107 - Maharashtra

```
    entropy -= p * (Math.log(p) / Math.log(2));
}
return entropy;
}

private static int bestAttribute(List<String[]> data, List<Integer> attributes, int labelIndex)
{
    double baseEntropy = entropy(data, labelIndex);      double bestGain = -1.0;      int bestAttr = -1;
    for (int attr : attributes) {
        Map<String, List<String[]>> subsets = new HashMap<>();
        for (String[] row : data) {
            subsets.computeIfAbsent(row[attr], k -> new ArrayList<>()).add(row);
        }
        double newEntropy = 0.0;
        for (List<String[]> subset : subsets.values()) {
            double weight = (double) subset.size() / data.size();
            newEntropy += weight * entropy(subset, labelIndex);
        }
        double gain = baseEntropy - newEntropy;
        if (gain > bestGain) {
            bestGain = gain;
            bestAttr = attr;
        }
    }
    return bestAttr;
}

private static Node buildTree(List<String[]> data, List<Integer> attributes, int labelIndex, String[] header) {
    Node node = new Node();
    Set<String> uniqueLabels = new HashSet<>();
    for (String[] row : data) {
        uniqueLabels.add(row[labelIndex]);
    }
    if (uniqueLabels.size() == 1) {
        node.label = uniqueLabels.iterator().next();
    }
    if (attributes.isEmpty()) {
        node.label = majorityLabel(data, labelIndex);
    }
    return node;
}
```



SHREE L. R. TIWARI COLLEGE OF ENGINEERING

An Autonomous Institute, Affiliated to University of Mumbai Approved by AICTE & DTE, Maharashtra State
NAAC Accredited, NBA Accredited Program | ISO 9001:2015 Certified | DTE Code No: 3423
Recognized under Section 2(f) of the UGC Act 1956 | Minority Status (Hindi Linguistic)
Miraroad (East), Thane - 401 107 - Maharashtra

```
int bestAttr = bestAttribute(data, attributes, labelIndex);
node.attribute = header[bestAttr];
Map<String, List<String[]>> subsets = new HashMap<>();
for (String[] row : data) {
    subsets.computeIfAbsent(row[bestAttr], k -> new ArrayList<>()).add(row);
}
List<Integer> remainingAttrs = new ArrayList<>(attributes);
remainingAttrs.remove(Integer.valueOf(bestAttr));
for (Map.Entry<String, List<String[]>> entry : subsets.entrySet()) {
    node.children.put(entry.getKey(), buildTree(entry.getValue(), remainingAttrs, labelIndex, header));
}
return node;
}

private static String majorityLabel(List<String[]> data, int labelIndex) {
    Map<String, Integer> freq = new HashMap<>();
    for (String[] row : data) {
        freq.put(row[labelIndex], freq.getOrDefault(row[labelIndex], 0) + 1);
    }
    return Collections.max(freq.entrySet(), Map.Entry.comparingByValue()).getKey();
}

public static String predict(Node node, String[] sample, String[] header) {
while (!node.isLeaf()) {
    int attrIndex = -1;
    for (int i = 0; i < header.length; i++) {
        if (header[i].equals(node.attribute)) {
            attrIndex = i;
            break;
        }
    }
    if (attrIndex == -1) return "Unknown";
    String attrValue = sample[attrIndex].trim();
    Node child = node.children.get(attrValue);
    if (child == null) return "Unknown";
    node = child;
}
return node.label;
}

public static void main(String[] args) {
    String[] header = {"Outlook", "Temperature", "Humidity", "Windy", "Play"};
    List<String[]> data = Arrays.asList(
        new String[]{"Sunny", "Hot", "High", "False", "No"}, new String[]{"Sunny", "Hot", "High", "True", "No"}, new String[]{"Overcast", "Hot", "High", "False", "Yes"} );
}
```



SHREE L. R. TIWARI COLLEGE OF ENGINEERING

An Autonomous Institute, Affiliated to University of Mumbai Approved by AICTE & DTE, Maharashtra State
NAAC Accredited, NBA Accredited Program | ISO 9001:2015 Certified | DTE Code No: 3423
Recognized under Section 2(f) of the UGC Act 1956 | Minority Status (Hindi Linguistic)
Miraroad (East), Thane - 401 107 - Maharashtra

```
String[] {"Rain", "Mild", "High", "False", "Yes"},      new
String[] {"Rain", "Cool", "Normal", "False", "Yes"},      new
String[] {"Rain", "Cool", "Normal", "True", "No"},       new
String[] {"Overcast", "Cool", "Normal", "True", "Yes"},   new
new String[] {"Sunny", "Mild", "High", "False", "No"},    new
String[] {"Sunny", "Cool", "Normal", "False", "Yes"},     new
String[] {"Rain", "Mild", "Normal", "False", "Yes"},      new
String[] {"Sunny", "Mild", "Normal", "True", "Yes"},      new
String[] {"Overcast", "Mild", "High", "True", "Yes"},     new
String[] {"Overcast", "Hot", "Normal", "False", "Yes"},    new
String[] {"Rain", "Mild", "High", "True", "No"}  
);  
int labelIndex = header.length - 1;  
List<Integer> attributes = new ArrayList<>();      for (int  
i = 0; i < labelIndex; i++) attributes.add(i);      Node root =  
buildTree(data, attributes, labelIndex, header);  
String[] sample1 = {"Sunny", "Mild", "High", "False"};  
String[] sample2 = {"Rain", "Mild", "Normal", "False"};  
String[] sample3 = {"Sunny", "Cool", "Normal", "False"};  
System.out.println(Arrays.toString(sample1) + " -> " + predict(root, sample1, header));  
System.out.println(Arrays.toString(sample2) + " -> " + predict(root, sample2, header));  
System.out.println(Arrays.toString(sample3) + " -> " + predict(root, sample3, header));  
}  
}
```

OUTPUT:

```
Output  
[Sunny, Mild, High, False] -> No  
[Rain, Mild, Normal, False] -> Yes  
[Sunny, Cool, Normal, False] -> Yes  
  
==== Code Execution Successful ===
```



SHREE L. R. TIWARI COLLEGE OF ENGINEERING

An Autonomous Institute, Affiliated to University of Mumbai Approved by AICTE & DTE, Maharashtra State
NAAC Accredited, NBA Accredited Program | ISO 9001:2015 Certified | DTE Code No: 3423
Recognized under Section 2(f) of the UGC Act 1956 | Minority Status (Hindi Linguistic)
Miraroad (East), Thane - 401 107 - Maharashtra

WEKA CODE:

==== Run information ====

Scheme: weka.classifiers.trees.J48 -C 0.25 -M 2

Relation: new_data

Instances: 14

Attributes: 6

No
Outlook
Temperature
Humidity
Windy
PlayGolf

Test mode: 10-fold cross-validation

==== Classifier model (full training set) ====

J48 pruned tree

Outlook = sunny
| Humidity = high: N (3.0)
| Humidity = normal: Y (2.0)
Outlook = overcast: Y (4.0)
Outlook = rain
| Windy = false: Y (3.0)
| Windy = true: N (2.0)

Number of Leaves : 5

Size of the tree : 8

Time taken to build model: 0.02 seconds

==== Stratified cross-validation ====

==== Summary ====

Correctly Classified Instances	6	42.8571 %
Incorrectly Classified Instances	8	57.1429 %
Kappa statistic	-0.1429	



SHREE L. R. TIWARI COLLEGE OF ENGINEERING

An Autonomous Institute, Affiliated to University of Mumbai Approved by AICTE & DTE, Maharashtra State
NAAC Accredited, NBA Accredited Program | ISO 9001:2015 Certified | DTE Code No: 3423
Recognized under Section 2(f) of the UGC Act 1956 | Minority Status (Hindi Linguistic)
Miraroad (East), Thane - 401 107 - Maharashtra

Mean absolute error	0.4881
Root mean squared error	0.6554
Relative absolute error	102.5 %
Root relative squared error	132.8454 %
Total Number of Instances	14

==== Detailed Accuracy By Class ====

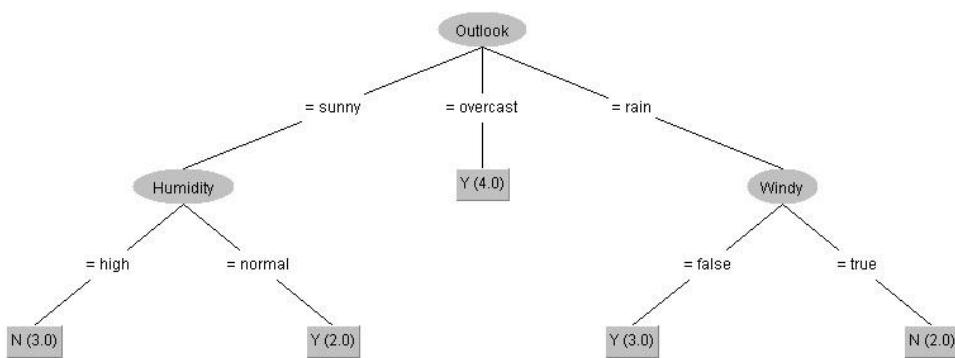
TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
0.400	0.556	0.286	0.400	0.333	-0.149	0.556	0.395	N
0.444	0.600	0.571	0.444	0.500	-0.149	0.556	0.713	Y
Weighted Avg.	0.429	0.584	0.469	0.429	0.440	-0.149	0.556	0.600

==== Confusion Matrix ====

a b <-- classified as

2 3 | a = N

5 4 | b = Y



CONCLUSION: Hence, we have implemented Decision Tree using Java and WEKA.



Shree Rahul Education Society's (Regd.)

SHREE L. R. TIWARI COLLEGE OF ENGINEERING

An Autonomous Institute Affiliated to University of Mumbai,
Approved by AICTE & DTE, Maharashtra State | NAAC Accredited, NBA Accredited Program | ISO 9001:2015 Certified |
DTE Code No: 3423 | Recognized under Section 2(f) of the UGC Act 1956 | Minority Status (Hindi Linguistic)

DEPARTMENT OF COMPUTER ENGINEERING

EXPERIMENT NO. 6

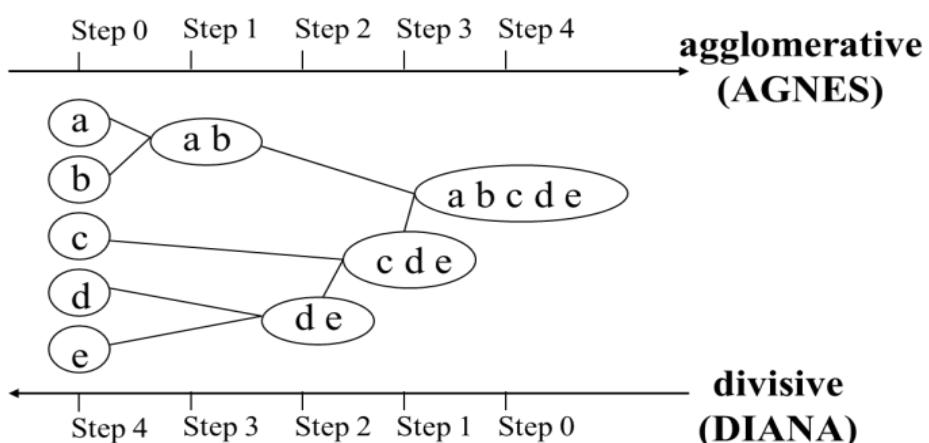
TITLE: Implementation of Agglomerative Hierarchical Clustering Algorithm.

AIM: Write a program to implement Agglomerative Hierarchical Clustering Algorithm using Java and WEKA tool.

THEORY:

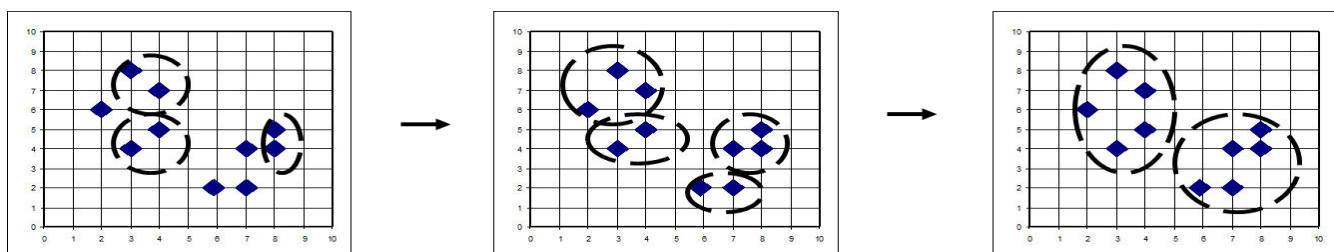
Hierarchical Clustering:

Use distance matrix as clustering criteria. This method does not require the number of clusters k as an input, but needs a termination condition.



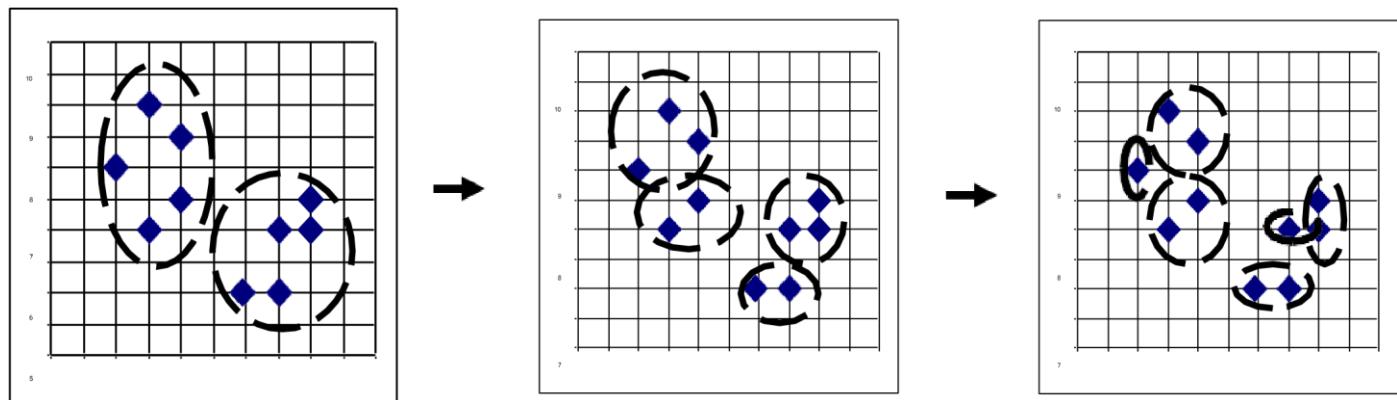
AGNES (Agglomerative Nesting):

Introduced in Kaufmann and Rousseeuw (1990). Implemented in statistical analysis packages, e.g., Splus. Use the Single-Link method and the dissimilarity matrix. Merge nodes that have the least dissimilarity. Go on in a non-descending fashion. Eventually all nodes belong to the same cluster.



**DEPARTMENT OF COMPUTER ENGINEERING****DIANA (Divisive Analysis):**

Introduced in Kaufmann and Rousseeuw (1990). Implemented in statistical analysis packages, e.g., SPSS. Inverse order of AGNES. Eventually each node forms a cluster on its own.

**Dendrogram:**

Shows How the Clusters are Merged. Decompose data objects into several levels of nested partitioning (tree of clusters), called a dendrogram. A clustering of the data objects is obtained by cutting the dendrogram at the desired level, then each connected component forms a cluster.

Major weakness of agglomerative clustering methods do not scale well: time complexity of at least $O(n^2)$, where n is the number of total objects can never undo what was done previously.

LAB EXERCISE:**1) Java Code import**

```
java.util.*;
```

```
public class AgglomerativeHierarchicalClustering {  
  
    public static void main(String[] args) {  
        Scanner kbd = new Scanner(System.in);  
  
        // Input number of data points  
        System.out.print("Enter N - the number of points: "); int N = kbd.nextInt();  
  
        int M = 2; // 2D coordinates  
        double[][] data = new double[N][M];  
        String[] names = new String[N];
```

**DEPARTMENT OF COMPUTER ENGINEERING**

```
// Input coordinates for each point
for (int i = 0; i < N; i++) {
    names[i] = "S" + (i + 1);
    System.out.println("Enter coordinates for " + names[i] + " (Math_Score English_Score):");
    for (int j = 0; j < M; j++) {
        data[i][j] = kbd.nextDouble();
    }
}

// Display entered data
System.out.println("\n==== Data Entered ====");
System.out.println("Student\tMath\tEnglish");      System.out.println("-----");
---");
    for (int i = 0; i < N;
i++) {
    System.out.printf("%s\t%.0f\t%.0f\n", names[i], data[i][0], data[i][1]);
}
System.out.println();

// Choose linkage method
int choice;
System.out.println("Enter your choice:");
System.out.println("1. Single Linkage (MIN)");
System.out.println("2. Complete Linkage (MAX)");
System.out.println("3. Average Linkage (AVG)");
System.out.print("Choice: ");
choice = kbd.nextInt();

String linkageType = "";      switch (choice) {
case 1: linkageType = "Single Linkage"; break;
case 2: linkageType = "Complete Linkage"; break;
case 3: linkageType = "Average Linkage"; break;
default: linkageType = "Unknown"; break;
}
System.out.println("\n==== Using " + linkageType + " Method ===\n");

// Initialize distance matrix
double INFINITY = Double.POSITIVE_INFINITY;
double[][] d = new double[N][N];
int[] dmin = new int[N];
```

**DEPARTMENT OF COMPUTER ENGINEERING**

```
// Calculate initial pairwise distances
for (int i = 0; i < N; i++) {      dmin[i]
= 0;          for (int j = 0; j < N; j++) {
if (i == j) {      d[i][j] = INFINITY;
} else {
    d[i][j] = distanceTo(data[i], data[j]);
}
}

// Track minimum distance for each point
if (d[i][j] < d[i][dmin[i]]) {
dmin[i] = j;
}
}

// Display initial distance matrix
System.out.println("== Initial Distance Matrix ==");
printDistanceMatrix(d, names, N);

// Perform hierarchical clustering iterations      for
(int s = 0; s < N - 1; s++) {
    // Update minimum distances for all points
for (int i = 0; i < N; i++) {      dmin[i] = 0;
for (int j = 0; j < N; j++) {          if (i == j)
{      d[i][j] = INFINITY;
}
        if (d[i][j] < d[i][dmin[i]]) {
            dmin[i] = j;
}
}
}

// Find the pair of clusters with minimum distance
int i1 = 0;          for (int i = 0; i < N; i++) {          if
(d[i][dmin[i]] < d[i1][dmin[i1]]) {      i1 = i;
}
}
int i2 =
dmin[i1];

System.out.println("\n--- Iteration " + (s + 1) + " ---");
System.out.printf("Merging clusters: %s and %s (Distance: %.2f)\n",
names[i1],
names[i2], d[i1][i2]);
```

**DEPARTMENT OF COMPUTER ENGINEERING**

```
System.out.printf("New cluster: (%s, %s)\n", names[i1], names[i2]);
```

```
// Update distances based on linkage criterion      if
(choice == 1) {
    // Single Linkage: Take minimum distance
    for (int j = 0; j < N; j++) {
        if (d[i2][j] < d[i1][j]) {
            d[i1][j] = d[j][i1] = d[i2][j];
        }
    }
} else if (choice == 2) {
    // Complete Linkage: Take maximum distance      for
    (int j = 0; j < N; j++) {                  if ((d[i2][j] > d[i1][j] && d[i2][j]
!= INFINITY) ||                      (d[i1][j] == INFINITY && d[i2][j]
!= INFINITY)) {
        d[i1][j] = d[j][i1] = d[i2][j];
    }
}
} else if (choice == 3) {
    // Average Linkage: Take average distance
    for (int j = 0; j < N; j++) {
        if (d[i1][j] != INFINITY && d[i2][j] != INFINITY) {
            d[i1][j] = d[j][i1] = (d[i1][j] + d[i2][j]) / 2.0;
        } else if (d[i1][j] == INFINITY && d[i2][j] != INFINITY) {
            d[i1][j] = d[j][i1] = d[i2][j];
        }
    }
}
```

```
d[i1][i1] = INFINITY;
```

```
// Mark merged cluster i2 as inactive      for
(int i = 0; i < N; i++) {
    d[i2][i] = d[i][i2] = INFINITY;
}
// Update references to merged cluster
for (int j = 0; j < N; j++) {
```

**DEPARTMENT OF COMPUTER ENGINEERING**

```
if (dmin[j] == i2) {  
    dmin[j] = i1;  
}  
}  
// Find new minimum for the merged cluster      dmin[i1]  
= 0;      for (int j = 0; j < N; j++) {  
if (d[i1][j] < d[i1][dmin[i1]]) {  
    dmin[i1] = j;  
}  
}  
  
// Display updated distance matrix  
System.out.println("\nDistance Matrix after Iteration " + (s + 1) + ":");      printDistanceMatrix(d,  
names, N);  
}  
  
System.out.println("\n" + "=" .repeat(50));  
System.out.println("CLUSTERING COMPLETE!");  
System.out.println("All points have been merged into a single cluster.");  
System.out.println("=".repeat(50));  
  
kbd.close();  
}  
// Calculate Euclidean distance between two points  
static double distanceTo(double[] start, double[] end) {  
double temp = Math.pow((end[0] - start[0]), 2) +  
Math.pow((end[1] - start[1]), 2);      return  
Math.sqrt(temp);  
}  
  
// Round to 2 decimal places  static double  
round2(double a) {      if (a !=  
Double.POSITIVE_INFINITY) {      int  
temp = (int) (a * 100);      double dbl =  
((double) temp) / 100.0;
```

**DEPARTMENT OF COMPUTER ENGINEERING**

```
    return dbl;
} else {
    return Double.POSITIVE_INFINITY;
}

}

// Print distance matrix in tabular format
static void printDistanceMatrix(double[][] d, String[] names, int N) {
    // Print header
    System.out.print("      ");
    for
(int i = 0; i < N; i++) {
        System.out.printf("%-8s", names[i]);
    }
    System.out.println();
    System.out.println("-----|" + "-".repeat(8 * N));

    // Print matrix rows      for
(int i = 0; i < N; i++) {
    System.out.printf("%-7s|", names[i]);
    for (int j = 0; j < N; j++) {          if (d[i][j] ==
Double.POSITIVE_INFINITY) {
        System.out.printf("%-8s", "∞");
    } else {
        System.out.printf("%-8.2f", round2(d[i][j]));
    }
}
    System.out.println();
}
System.out.println();
}
```

}OUTPUT:

Enter N - the number of points: 5 Enter

coordinates for P1 (X Y): 2 3

Enter coordinates for P2 (X Y):

3 4

Enter coordinates for P3 (X Y):

**DEPARTMENT OF COMPUTER ENGINEERING**

87

Enter coordinates for P4 (X Y):

96

Enter coordinates for P5 (X Y):

1512

Data entered:

Point X-coord Y-coord

P1 2.0 3.0

P2 3.0 4.0

P3 8.0 7.0

P4 9.0 6.0

P5 15.0 12.0

Enter your choice:

1. Single Linkage
2. Complete Linkage
3. Average Linkage

1

Initial Distance Matrix:

P1	∞
P2	1.41 ∞
P3	6.71 5.83 ∞
P4	7.62 7.21 1.41 ∞
P5	14.76 14.42 8.6 8.49 ∞

	P1 P2 P3 P4 P5

pts 1 and 2 are clustered hence new point is (1,2) Distance matrix

after iteration 1:

P1	∞
P2	∞ ∞
P3	5.83 ∞ ∞
P4	7.21 ∞ 1.41 ∞
P5	14.42 ∞ 8.6 8.49 ∞

	P1 P2 P3 P4 P5

pts 3 and 4 are clustered hence new point is (3,4) Distance matrix

after iteration 2:



Shree Rahul Education Society's (Regd.)

SHREE L. R. TIWARI COLLEGE OF ENGINEERING

An Autonomous Institute Affiliated to University of Mumbai,

Approved by AICTE & DTE, Maharashtra State | NAAC Accredited, NBA Accredited Program | ISO 9001:2015 Certified |
DTE Code No: 3423 | Recognized under Section 2(f) of the UGC Act 1956 | Minority Status (Hindi Linguistic)

DEPARTMENT OF COMPUTER ENGINEERING

P1	∞				
P2	∞	∞			
P3	5.83	∞	∞		
P4	∞	∞	∞		
P5	14.42	∞	8.49	∞	∞

	P1	P2	P3	P4	P5

pts 1 and 3 are clustered hence new point is (1,3) Distance matrix
after iteration 3:

P1	∞				
P2	∞	∞			
P3	∞	∞	∞		
P4	∞	∞	∞		
P5	8.49	-----	∞	∞	∞
	P1	P2	P3	P4	P5

pts 1 and 5 are clustered hence new point is (1,5) Distance matrix
after iteration 4:

P1	∞				
P2	∞	∞			
P3	∞	∞	∞		
P4	∞	∞	∞		
P5	∞	∞	∞	∞	∞

	P1	P2	P3	P4	P5

2) WEKA Code

==== Run information ====

Scheme: weka.clusterers.HierarchicalClusterer -N 3 -L SINGLE



Shree Rahul Education Society's (Regd.)

SHREE L. R. TIWARI COLLEGE OF ENGINEERING

An Autonomous Institute Affiliated to University of Mumbai,

Approved by AICTE & DTE, Maharashtra State | NAAC Accredited, NBA Accredited Program | ISO 9001:2015 Certified |
DTE Code No: 3423 | Recognized under Section 2(f) of the UGC Act 1956 | Minority Status (Hindi Linguistic)

DEPARTMENT OF COMPUTER ENGINEERING

-P -A "weka.core.EuclideanDistance -R first-last"

Relation: spatial_points

Instances: 15 Attributes:

4

pointID
x_coordinate y_coordinate
cluster_label

Test mode: evaluate on training data

==== Clustering model (full training set) ===

HierarchicalClusterer

=====

Cluster Method: SINGLE

Distance Function: Euclidean Distance

Number of clusters: 3



Shree Rahul Education Society's (Regd.)

SHREE L. R. TIWARI COLLEGE OF ENGINEERING

An Autonomous Institute Affiliated to University of Mumbai,

Approved by AICTE & DTE, Maharashtra State | NAAC Accredited, NBA Accredited Program | ISO 9001:2015 Certified |
DTE Code No: 3423 | Recognized under Section 2(f) of the UGC Act 1956 | Minority Status (Hindi Linguistic)

DEPARTMENT OF COMPUTER ENGINEERING

Dendrogram (Newick format):

((P1:0.71,P2:0.71):0.53,(P7:1.00,P10:1.00):0.82,P13:1.82)

((P3:0.71,P11:0.71):0.53,(P4:1.41,P8:1.41):0.35,P14:1.76)

((P5:0.71,P9:0.71):0.53,(P6:1.41,P12:1.41):0.35,P15:1.76)

Cluster 0: Group A (Lower-left region)

Points: P1, P2, P7, P10, P13

Centroid: (3.4, 3.6)

Cluster 1: Group B (Middle region)

Points: P3, P4, P8, P11, P14

Centroid: (9.0, 7.2)

Cluster 2: Group C (Upper-right region)

Points: P5, P6, P9, P12, P15

Centroid: (16.0, 13.0)

Clustered Instances

0 5 (33%) 1 5

(33%)

2 5 (34%)

Time taken to build model (full training data): 0.04 seconds

CONCLUSION: Hence, we have implemented Agglomerative Clustering Algorithm using WEKA and Java.



SHREE L. R. TIWARI COLLEGE OF ENGINEERING

An Autonomous Institute, Affiliated to University of Mumbai Approved by AICTE & DTE, Maharashtra State
NAAC Accredited, NBA Accredited Program | ISO 9001:2015 Certified | DTE Code No: 3423
Recognized under Section 2(f) of the UGC Act 1956 | Minority Status (Hindi Linguistic)
Miraroad (East), Thane - 401 107 - Maharashtra

EXPERIMENT NO. 7

TITLE: Implementation of Clustering algorithm.

AIM: Write a program to implement K-Mean clustering algorithm using WEKA and Java.

THEORY:

K-means clustering is a method of vector quantization, originally from signal processing, that is popular for cluster analysis in data mining. k-means clustering aims to partition n observations into k clusters in which each observation belongs to the cluster with the nearest mean, serving as a prototype of the cluster. This results in a partitioning of the data space into Voronoi cells.

K-Means clustering intends to partition n objects into k clusters in which each object belongs to the cluster with the nearest mean. This method produces exactly k different clusters of greatest possible distinction. The best number of clusters k leading to the greatest separation (distance) is not known as a priori and must be computed from the data. The objective of K- Means clustering is to minimize total intra-cluster variance, or, the squared error function:

$$\text{objective function} \leftarrow J = \sum_{j=1}^k \sum_{i=1}^n \|x_i^{(j)} - c_j\|^2$$

Distance function

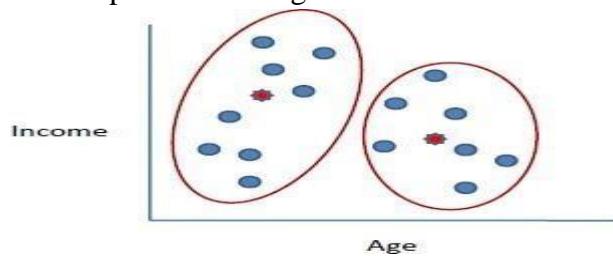
number of clusters number of cases
 k n
 $x_i^{(j)}$ case i
 c_j centroid for cluster j

Algorithm

Clusters the data into k groups where k is predefined. Select k points at random as cluster centers.

Assign objects to their closest cluster center according to the Euclidean distance function. Calculate the centroid or mean of all objects in each cluster.

Repeat steps 2, 3 and 4 until the same points are assigned to each cluster in consecutive rounds.





SHREE L. R. TIWARI COLLEGE OF ENGINEERING

An Autonomous Institute, Affiliated to University of Mumbai Approved by AICTE & DTE, Maharashtra State
NAAC Accredited, NBA Accredited Program | ISO 9001:2015 Certified | DTE Code No: 3423
Recognized under Section 2(f) of the UGC Act 1956 | Minority Status (Hindi Linguistic)
Miraroad (East), Thane - 401 107 - Maharashtra

DEPARTMENT OF COMPUTER ENGINEERING

K-Means is relatively an efficient method. However, we need to specify the number of clusters, in advance and the final results are sensitive to initialization and often terminates at a local optimum. Unfortunately, there is no global theoretical method to find the optimal number of clusters. A practical approach is to compare the outcomes of multiple runs with different k and choose the best one based on a predefined criterion. In general, a large k probably decreases the error but increases the risk of overfitting.

Example:

Suppose we want to group the visitors to a website using just their age (a one-dimensional space) as follows:

15,15,16,19,19,20,20,21,22,28,35,40,41,42,43,44,60,61,65 Initial clusters:

Centroid (C1) = 16 [16]

Centroid (C2) = 22 [22]

Iteration 1:

C1 = 15.33 [15,15,16]

C2 = 36.25 [19,19,20,20,21,22,28,35,40,41,42,43,44,60,61,65] Iteration

2:

C1 = 18.56 [15,15,16,19,19,20,20,21,22] C2

= 45.90 [28,35,40,41,42,43,44,60,61,65] Iteration

3:

C1 = 19.50 [15,15,16,19,19,20,20,21,22,28]

C2 = 47.89 [35,40,41,42,43,44,60,61,65] Iteration

4:

C1 = 19.50 [15,15,16,19,19,20,20,21,22,28]

C2 = 47.89 [35,40,41,42,43,44,60,61,65]

No change between iterations 3 and 4 has been noted. By using clustering, 2 groups have been identified 15-28 and 35-65. The initial choice of centroids can affect the output clusters, so the algorithm is often run multiple times with different starting conditions in order to get a fair view of what the clusters should be.

LAB EXERCISE:

1) Java Code



SHREE L. R. TIWARI COLLEGE OF ENGINEERING

An Autonomous Institute, Affiliated to University of Mumbai Approved by AICTE & DTE, Maharashtra State
NAAC Accredited, NBA Accredited Program | ISO 9001:2015 Certified | DTE Code No: 3423
Recognized under Section 2(f) of the UGC Act 1956 | Minority Status (Hindi Linguistic)
Miraroad (East), Thane - 401 107 - Maharashtra

```
import java.util.ArrayList; import
java.util.List;
import java.util.Objects;

public class KMean {
    static class Point {
        double value;
        Point(double value) {
            this.value = value;
        }
        double distance(Point p) {
            return Math.abs(this.value - p.value);
        }
        @Override public boolean
        equals(Object o) { if (this == o)
        return true;
        if (o == null || getClass() != o.getClass()) return false;
        Point point = (Point) o;
        return Double.compare(point.value, value) == 0;
    }
        @Override public
        int hashCode() {
            return Objects.hash(value);
        }
    }

    static class Cluster {
        List<Point> points = new ArrayList<>();
        Point centroid;
        Cluster(Point initialCentroid) {
            this.centroid = initialCentroid;
        }
        void updateCentroid() {
            if (points.isEmpty()) return;
            double sum = 0; for (Point
            p : points) {
                sum += p.value;
            }
            centroid = new Point(sum / points.size());
        }
        void clearPoints() {
            points.clear();
        }
    }

    public static void kMeans(List<Point> points, int k, int maxIterations, Point m1, Point m2) {
```



SHREE L. R. TIWARI COLLEGE OF ENGINEERING

An Autonomous Institute, Affiliated to University of Mumbai Approved by AICTE & DTE, Maharashtra State
NAAC Accredited, NBA Accredited Program | ISO 9001:2015 Certified | DTE Code No: 3423
Recognized under Section 2(f) of the UGC Act 1956 | Minority Status (Hindi Linguistic)
Miraroad (East), Thane - 401 107 - Maharashtra

```
List<Cluster> clusters = new ArrayList<>();  
clusters.add(new Cluster(m1));  
clusters.add(new Cluster(m2));  
boolean changed;  
int iterations = 0;  
do {  
    changed = false;  
    iterations++;  
    for (Cluster cluster : clusters) {  
        cluster.clearPoints();  
    }  
    for (Point p : points) {  
        Cluster nearestCluster = findNearestCluster(p, clusters);  
        nearestCluster.points.add(p);  
    }  
    System.out.println("Iteration " + iterations + ":");  
    for (int i = 0; i < k; i++) {  
        Cluster cluster = clusters.get(i);  
        Point oldCentroid = cluster.centroid;  
        cluster.updateCentroid();  
        System.out.println("Cluster " + (i + 1) + ":");  
        System.out.print("Points: ");  
        for (Point p : cluster.points) {  
            System.out.print(p.value + " ");  
        }  
        System.out.println();  
        System.out.println("Old Centroid: " + oldCentroid.value + ", New Centroid: " + cluster.centroid.value);  
        if (!oldCentroid.equals(cluster.centroid)) {  
            changed = true;  
        }  
    }  
    System.out.println(" ");  
} while (changed && iterations < maxIterations);  
for (int i = 0; i < k; i++) {  
    Cluster finalCluster = clusters.get(i);  
    System.out.println("Cluster " + (i + 1) + " (Final Centroid: " + finalCluster.centroid.value + ")");  
    System.out.print("\tPoints: ");  
    for (Point p : finalCluster.points) {  
        System.out.print(p.value + " ");  
    }  
    System.out.println("\n");  
}  
private static Cluster findNearestCluster(Point p, List<Cluster> clusters) {  
    Cluster nearestCluster = clusters.get(0);  
    for (Cluster cluster : clusters) {  
        if (cluster.distance(p) < nearestCluster.distance(p)) {  
            nearestCluster = cluster;  
        }  
    }  
    return nearestCluster;  
}
```



SHREE L. R. TIWARI COLLEGE OF ENGINEERING

An Autonomous Institute, Affiliated to University of Mumbai Approved by AICTE & DTE, Maharashtra State
NAAC Accredited, NBA Accredited Program | ISO 9001:2015 Certified | DTE Code No: 3423
Recognized under Section 2(f) of the UGC Act 1956 | Minority Status (Hindi Linguistic)
Miraroad (East), Thane - 401 107 - Maharashtra

```
    double minDistance = p.distance(nearestCluster.centroid);
for (Cluster cluster : clusters) {
    double distance = p.distance(cluster.centroid);
    if (distance < minDistance) {           nearestCluster
= cluster;
        minDistance = distance;
    }
}
return nearestCluster;
}

public static void main(String[] args) {
List<Point> points = new ArrayList<>();
points.add(new Point(2));
points.add(new Point(4));
points.add(new Point(10));
points.add(new Point(12));
points.add(new Point(3));
points.add(new Point(20));
points.add(new Point(30));
points.add(new Point(11));
points.add(new Point(25));      int k = 2;
int maxIterations = 100;      Point m1 = new
Point(3);      Point m2 = new Point(4);
kMeans(points, k, maxIterations, m1, m2);
}
```

OUTPUT:

Iteration 1:
Cluster 1:
Points: 2.0 3.0
Old Centroid: 3.0, New Centroid: 2.5 Cluster
2:
Points: 4.0 10.0 12.0 20.0 30.0 11.0 25.0
Old Centroid: 4.0, New Centroid: 16.0

Iteration 2:
Cluster 1:
Points: 2.0 4.0 3.0
Old Centroid: 2.5, New Centroid: 3.0 Cluster
2:
Points: 10.0 12.0 20.0 30.0 11.0 25.0
Old Centroid: 16.0, New Centroid: 18.0



SHREE L. R. TIWARI COLLEGE OF ENGINEERING

An Autonomous Institute, Affiliated to University of Mumbai Approved by AICTE & DTE, Maharashtra State
NAAC Accredited, NBA Accredited Program | ISO 9001:2015 Certified | DTE Code No: 3423
Recognized under Section 2(f) of the UGC Act 1956 | Minority Status (Hindi Linguistic)
Miraroad (East), Thane - 401 107 - Maharashtra

Iteration 3:

Cluster 1:

Points: 2.0 4.0 10.0 3.0

Old Centroid: 3.0, New Centroid: 4.75 Cluster

2:

Points: 12.0 20.0 30.0 11.0 25.0

Old Centroid: 18.0, New Centroid: 19.6

Iteration 4:

Cluster 1:

Points: 2.0 4.0 10.0 12.0 3.0 11.0 Old

Centroid: 4.75, New Centroid: 7.0

Cluster 2:

Points: 20.0 30.0 25.0

Old Centroid: 19.6, New Centroid: 25.0

Iteration 5:

Cluster 1:

Points: 2.0 4.0 10.0 12.0 3.0 11.0 Old

Centroid: 7.0, New Centroid: 7.0

Cluster 2:

Points: 20.0 30.0 25.0

Old Centroid: 25.0, New Centroid: 25.0

Cluster 1 (Final Centroid: 7.0):

Points: 2.0 4.0 10.0 12.0 3.0 11.0

Cluster 2 (Final Centroid: 25.0):

Points: 20.0 30.0 25.0

2) WEKA Code

==== Run information ====

Scheme: weka.clusterers.SimpleKMeans -init 0 -max-candidates 100 -periodic-pruning 10000 -min-density 2.0 t1 -1.25 -t2 -1.0 -N 3 -A "weka.core.EuclideanDistance -R first-last" -I 500 -num-slots 1 -S 10 Relation: kmeans-clustering

Instances: 8 Attributes: 1

value

Test mode: evaluate on training data

==== Clustering model (full training set) ===



SHREE L. R. TIWARI COLLEGE OF ENGINEERING

An Autonomous Institute, Affiliated to University of Mumbai Approved by AICTE & DTE, Maharashtra State
NAAC Accredited, NBA Accredited Program | ISO 9001:2015 Certified | DTE Code No: 3423
Recognized under Section 2(f) of the UGC Act 1956 | Minority Status (Hindi Linguistic)
Miraroad (East), Thane - 401 107 - Maharashtra

kMeans =====

Number of iterations: 4

Within cluster sum of squared errors: 14.5

Initial starting points (generated from instances):

Cluster 0: 2

Cluster 1: 3

Cluster 2: 6

Missing values globally replaced with mean/mode

Final cluster centroids:

Cluster#

Attribute	Full Data	0	1	2
	(8.0)	(2.0)	(2.0)	(4.0)

value	10.75	2.5	7.0	16.75
-------	-------	-----	-----	-------

Time taken to build model (full training data) : 0.01 seconds

==== Model and evaluation on training set ===

Clustered Instances

0	2 (25%)
1	2 (25%)
2	4 (50%)

CONCLUSION: Hence, we have implemented K-Mean using WEKA and Java.



SHREE L. R. TIWARI COLLEGE OF ENGINEERING

An Autonomous Institute, Affiliated to University of Mumbai Approved by AICTE & DTE, Maharashtra State
NAAC Accredited, NBA Accredited Program | ISO 9001:2015 Certified | DTE Code No: 3423
Recognized under Section 2(f) of the UGC Act 1956 | Minority Status (Hindi Linguistic)
Miraroad (East), Thane - 401 107 - Maharashtra

EXPERIMENT NO. 8

TITLE: Implementation of Association mining tool algorithm.

AIM: Write a program to implement Frequent Pattern (FP) Tree using Java and Weka Tool.

THEORY:

The FP-Growth Algorithm is an alternative algorithm used to find frequent item sets. FP stands for frequent pattern. It is vastly different from the Apriori Algorithm explained in previous sections in that it uses a FPtree to encode the data set and then extract the frequent itemsets from this tree. This section is divided into two main parts, the first deals with the representation of the FP-tree and the second details how frequent itemset generation occurs using this tree and its algorithm.

In the first pass, the algorithm counts occurrence of items (attribute-value pairs) in the dataset, and stores them to 'header table'. In the second pass, it builds the FP-tree structure by inserting instances. Items in each instance have to be sorted by descending order of their frequency in the dataset, so that the tree can be processed quickly. Items in each instance that do not meet minimum coverage threshold are discarded. If many instances share most frequent items, FP-tree provides high compression close to tree root. Recursive processing of this

compressed version of main dataset grows large item sets directly, instead of generating candidate items and testing them against the entire database. Growth starts from the bottom of the header table (having longest branches), by finding all instances matching given condition. New tree is created, with counts projected from the original tree corresponding to the set of instances that are conditional on the attribute, with each node getting sum of its children counts. Recursive growth ends when no individual items conditional on the attribute meet minimum support threshold, and processing continues on the remaining header items of the original FP-tree. Once the recursive process has completed, all large item sets with minimum coverage have been found, and association rule creation begins.

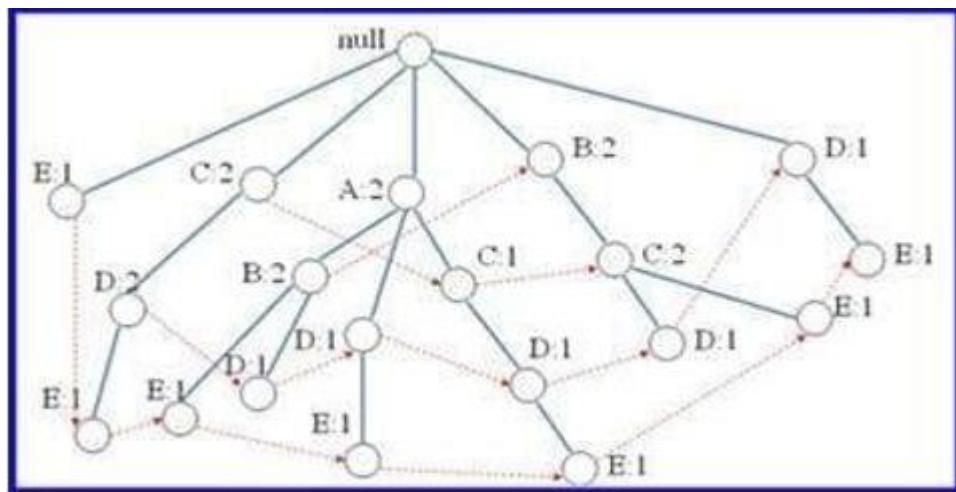
FP-Tree Representation A FP-tree is a compact data structure that represents the data set in tree form. Each transaction is read and then mapped onto a path in the FP-tree. This is done until all transactions have been read. Different transactions that have common subsets allow the tree to remain compact because their paths overlap. The diagram to the right is an example of a best-case scenario that occurs when all transactions have exactly the same itemset; the size of the FP-tree will be only a single branch of nodes.

The worst case scenario occurs when every transaction has a unique itemset and so the space needed to store the tree is greater than the space used to store the original data set because the FP-tree requires additional space to store pointers between nodes and also the counters for each item. The diagram below is an example of how a worst case scenario FP-tree might appear. As you can see, the complexity of the tree grows with the uniqueness of each transaction



SHREE L. R. TIWARI COLLEGE OF ENGINEERING

An Autonomous Institute, Affiliated to University of Mumbai Approved by AICTE & DTE, Maharashtra State
NAAC Accredited, NBA Accredited Program | ISO 9001:2015 Certified | DTE Code No: 3423
Recognized under Section 2(f) of the UGC Act 1956 | Minority Status (Hindi Linguistic)
Miraroad (East), Thane - 401 107 - Maharashtra



Construction:

The construction of a FP-tree is subdivided into three major steps:

- 1) Scan the data set to determine the support count of each item, discard the infrequent items and sort the frequent items in decreasing order.
- 2) Scan the data set one transaction at a time to create the FP-tree. For each transaction:
 - a. If it is a unique transaction form a new path and set the counter for each node to 1.
 - b. If it shares a common prefix itemset then increment the common itemset node counters and create new nodes if needed.
- 3) Continue this until each transaction has been mapped unto the tree.

LAB EXERCISE: 1)

Java Code

```
import java.util.*; public class
FPOnlineDemo {    public static
void main(String[] args) {
    Scanner sc = new
Scanner(System.in);
    System.out.print("Enter
number of transactions: ");
    int n =
Integer.parseInt(sc.nextLine());
    System.out.print("Enter number of
items in the itemset: ");    int
totalItems =
Integer.parseInt(sc.nextLine());
    System.out.print("Enter minimum
support: ");}
```



SHREE L. R. TIWARI COLLEGE OF ENGINEERING

An Autonomous Institute, Affiliated to University of Mumbai Approved by AICTE & DTE, Maharashtra State
NAAC Accredited, NBA Accredited Program | ISO 9001:2015 Certified | DTE Code No: 3423
Recognized under Section 2(f) of the UGC Act 1956 | Minority Status (Hindi Linguistic)
Miraroad (East), Thane - 401 107 - Maharashtra

```
int minSupport =  
Integer.parseInt(sc.nextLine());  
List<List<String>> transactions =  
new ArrayList<>();  
Set<String> itemSet = new  
TreeSet<>();  
for (int i = 1; i <= n; i++) {  
System.out.println("Enter items  
for TID no: " + i + " (enter 0 to  
finish):"); List<String>  
transaction = new ArrayList<>();  
while (true) {  
String item = sc.next();  
if (item.equals("0"))  
break;  
transaction.add(item);  
itemSet.add(item);  
}  
  
transactions.add(transaction);  
}  
if (sc.hasNextLine())  
sc.nextLine(); Map<String,  
Integer> supportMap = new  
HashMap<>();  
for (String item : itemSet) {  
int count = 0; for  
(List<String> t : transactions) if  
(t.contains(item)) count++;  
supportMap.put(item, count);  
}  
System.out.println("\nItem Set:  
"); for (String item :  
itemSet)  
System.out.print(item + " ");  
  
System.out.println("\nCorrespondin  
g supports"); for (String item :  
itemSet)  
System.out.print(supportMap.get(it  
em) + " ");  
System.out.println();  
  
FPTree tree = new FPTree();  
  
tree.buildFreqMap(transactions,  
minSupport);  
int tid = 1; for  
(List<String> t :  
transactions) {  
List<String> sorted =  
tree.sortTransaction(t);  
tree.insertTransaction(sorted  
);
```



SHREE L. R. TIWARI COLLEGE OF ENGINEERING

An Autonomous Institute, Affiliated to University of Mumbai Approved by AICTE & DTE, Maharashtra State
NAAC Accredited, NBA Accredited Program | ISO 9001:2015 Certified | DTE Code No: 3423
Recognized under Section 2(f) of the UGC Act 1956 | Minority Status (Hindi Linguistic)
Miraroad (East), Thane - 401 107 - Maharashtra

```
tree.displayTransactionPath(sorted,
tid++);
}
```

```
System.out.println("\nFPTree:
");
    tree.display(tree.root,
"");
    sc.close();
}
} class FPNode
{
    String item;
    int count;
    Map<String,
FPNode> children;
    FPNode
parent;
    public FPNode(String
item, FPNode parent) {
        this.item = item;
        this.count = 1;
        this.children = new
HashMap<>();
        this.parent = parent;
    }
    public FPNode getChild(String
item) {
        return children.get(item);
    }
    public void addChild(FPNode
child) {
        children.put(child.item, child);
    }
} class FPTree {
    FPNode root;
    Map<String, Integer> freqMap;
    public FPTree() {
        root = new
FPNode(null, null);
        freqMap =
new HashMap<>();
    }
    public void
buildFreqMap(List<List<String>>
transactions, int minSupport) {
        Map<String, Integer> counts = new
HashMap<>();
        for
(List<String> t : transactions) {
            for (String item : t)
                counts.put(item,
counts.getOrDefault(item, 0) + 1);
        }
        for (Map.Entry<String,
Integer> entry : counts.entrySet()) {
            if (entry.getValue() >=
minSupport)
                freqMap.put(entry.getKey(),
entry.getValue());
        }
    }
    public List<String>
sortTransaction(List<String>
```



SHREE L. R. TIWARI COLLEGE OF ENGINEERING

An Autonomous Institute, Affiliated to University of Mumbai Approved by AICTE & DTE, Maharashtra State
NAAC Accredited, NBA Accredited Program | ISO 9001:2015 Certified | DTE Code No: 3423
Recognized under Section 2(f) of the UGC Act 1956 | Minority Status (Hindi Linguistic)
Miraroad (East), Thane - 401 107 - Maharashtra

```
transaction) {      List<String>
sorted = new ArrayList<>();
for (String item : transaction) if
(freqMap.containsKey(item))
sorted.add(item);
sorted.sort((a, b) -> {
int countA = freqMap.get(a);
int countB = freqMap.get(b);
if (countA != countB) return countB
- countA;
return a.compareTo(b);
});
return sorted;
}  public void
insertTransaction(List<String>
transaction) {      FPNode current
= root;      for (String item :
transaction) {      FPNode child
= current.getChild(item);      if
(child != null) child.count++;
else {          child = new
FPNode(item,           current);
current.addChild(child);
}
current = child;
}
}
public           void
displayTransactionPath(List<String
> transaction, int tid) {
System.out.print("Transaction
No " + tid + ": Root ");
for
(String item : transaction)
System.out.print("| " + item + " ");
System.out.println();
}
public void display(FPNode
node, String indent) {      for
(FPNode child :
node.getChildren().values()) {
System.out.println(indent + "| " +
child.item + " : " + child.count);
display(child, indent + "\t");
}
}
}
```



SHREE L. R. TIWARI COLLEGE OF ENGINEERING

An Autonomous Institute, Affiliated to University of Mumbai Approved by AICTE & DTE, Maharashtra State
NAAC Accredited, NBA Accredited Program | ISO 9001:2015 Certified | DTE Code No: 3423
Recognized under Section 2(f) of the UGC Act 1956 | Minority Status (Hindi Linguistic)
Miraroad (East), Thane - 401 107 - Maharashtra

OUTPUT:

```
Enter number of transactions: 5
Enter number of items in the itemset: 16
Enter minimum support: 3
Enter items for TID no: 1 (enter 0 to finish):
f a c d g i m p 0
Enter items for TID no: 2 (enter 0 to finish):
a b c f l m o 0
Enter items for TID no: 3 (enter 0 to finish):
b f h j o 0
Enter items for TID no: 4 (enter 0 to finish):
b c k s p 0
Enter items for TID no: 5 (enter 0 to finish):
a f c e l p m n 0
```

```
Item Set:
a b c d e f g h i j k l m n o p s
```

```
Corresponding supports
3 3 4 1 1 4 1 1 1 1 1 2 3 1 2 3 1
```

```
Transaction No 1: Root | c | f | a | m | p
Transaction No 2: Root | c | f | a | b | m
Transaction No 3: Root | f | b
Transaction No 4: Root | c | b | p
Transaction No 5: Root | c | f | a | m | p
```

FP-Tree:

```
| c : 4
| b : 1
| p : 1
| f : 3
| a : 3
|   | b : 1
|   | m : 1
|   | m : 2
|   | p : 2
| f : 1
| b : 1
```



SHREE L. R. TIWARI COLLEGE OF ENGINEERING

An Autonomous Institute, Affiliated to University of Mumbai Approved by AICTE & DTE, Maharashtra State
NAAC Accredited, NBA Accredited Program | ISO 9001:2015 Certified | DTE Code No: 3423
Recognized under Section 2(f) of the UGC Act 1956 | Minority Status (Hindi Linguistic)
Miraroad (East), Thane - 401 107 - Maharashtra

2) WEKA Code

==== Run information ===

Scheme: weka.associations.FPGrowth -P 2 -I 1 -N 10 -T 0 -C 0.5 -D 0.05 -U 1.0 -M 0.4

Relation: fp-growth-transactions

Instances: 5 Attributes: 5

a	b	c
d	e	

==== Associator model (full training set) ===

FPGrowth found 11 rules (displaying top 10)

1. [e=1, b=1]: 4 ==> [c=1]: 3 <conf:(0.75)> lift:(1.25) lev:(0.12) conv:(1.6)
2. [c=1, b=1]: 3 ==> [e=1]: 3 <conf:(1)> lift:(1.25) lev:(0.12) conv:(0.6)
3. [e=1, c=1]: 3 ==> [b=1]: 3 <conf:(1)> lift:(1.25) lev:(0.12) conv:(0.6)
4. [e=1]: 4 ==> [b=1]: 4 <conf:(1)> lift:(1.25) lev:(0.2) conv:(0.8)
5. [b=1]: 4 ==> [e=1]: 4 <conf:(1)> lift:(1.25) lev:(0.2) conv:(0.8)
6. [c=1, a=1]: 3 ==> [b=1]: 2 <conf:(0.67)> lift:(0.83) lev:(-0.13) conv:(0.67)
7. [a=1]: 3 ==> [c=1]: 3 <conf:(1)> lift:(1.67) lev:(0.24) conv:(0.6)
8. [c=1]: 4 ==> [b=1]: 3 <conf:(0.75)> lift:(0.94) lev:(-0.08) conv:(0.8)
9. [b=1, a=1]: 2 ==> [c=1]: 2 <conf:(1)> lift:(1.67) lev:(0.16) conv:(0.4)
10. [a=1]: 3 ==> [b=1]: 2 <conf:(0.67)> lift:(0.83) lev:(-0.13) conv:(0.67)

Size of set of large itemsets L(1): 4

Large Itemsets L(1):

a=1 3 b=1 4 c=1 4 e=1 4

Size of set of large itemsets L(2): 7

Large Itemsets L(2):

a=1 b=1 2 a=1 c=1 3 a=1 e=1 2 b=1

c=1 3 b=1 e=1 4 c=1 e=1 3

b=1 c=1 e=1 3

Size of set of large itemsets L(3): 3

Large Itemsets L(3):

a=1 b=1 c=1 2 a=1 b=1 e=1 2

a=1 c=1 e=1 2



SHREE L. R. TIWARI COLLEGE OF ENGINEERING

An Autonomous Institute, Affiliated to University of Mumbai Approved by AICTE & DTE, Maharashtra State
NAAC Accredited, NBA Accredited Program | ISO 9001:2015 Certified | DTE Code No: 3423
Recognized under Section 2(f) of the UGC Act 1956 | Minority Status (Hindi Linguistic)
Miraroad (East), Thane - 401 107 - Maharashtra

Best rules found:

1. [c=1, b=1]: 3 ==> [e=1]: 3 <conf:(1)> lift:(1.25) lev:(0.12) [1] conv:(0.6)
2. [e=1, c=1]: 3 ==> [b=1]: 3 <conf:(1)> lift:(1.25) lev:(0.12) [1] conv:(0.6)
3. [e=1]: 4 ==> [b=1]: 4 <conf:(1)> lift:(1.25) lev:(0.2) [1] conv:(0.8)
4. [b=1]: 4 ==> [e=1]: 4 <conf:(1)> lift:(1.25) lev:(0.2) [1] conv:(0.8)
5. [a=1]: 3 ==> [c=1]: 3 <conf:(1)> lift:(1.67) lev:(0.24) [1] conv:(0.6)
6. [b=1, a=1]: 2 ==> [c=1]: 2 <conf:(1)> lift:(1.67) lev:(0.16) [1] conv:(0.4)

CONCLUSION: Hence, we have implemented FP Tree using Java and Weka Tool.



SHREE L. R. TIWARI COLLEGE OF ENGINEERING

An Autonomous Institute, Affiliated to University of Mumbai Approved by AICTE & DTE, Maharashtra State
NAAC Accredited, NBA Accredited Program | ISO 9001:2015 Certified | DTE Code No: 3423
Recognized under Section 2(f) of the UGC Act 1956 | Minority Status (Hindi Linguistic)
Miraroad (East), Thane - 401 107 - Maharashtra

DEPARTMENT OF COMPUTER ENGINEERING

EXPERIMENT NO. 9

AIM: Implementation of association mining apriori using Java and WEKA.

OBJECTIVE: To understand the Implementation of association mining apriori using Java and WEKA.

THEORY:

Apriori Algorithm

The apriori principle can reduce the number of itemsets we need to examine. Put simply, the apriori principle states that if an itemset is infrequent, then all its subsets must also be infrequent. This means that if {beer} was found to be infrequent, we can expect {beer, pizza} to be equally or even more infrequent. So in consolidating the list of popular itemsets, we need not consider {beer, pizza}, nor any other itemset configuration that contains beer.

Finding itemsets with high support

Using the apriori principle, the number of itemsets that have to be examined can be pruned, and the list of popular itemsets can be obtained in these steps:

- Step 0. Start with itemsets containing just a single item, such as {apple} and {pear}.
- Step 1. Determine the support for itemsets. Keep the itemsets that meet your minimum support threshold, and remove itemsets that do not.
- Step 2. Using the itemsets you have kept from Step 1, generate all the possible itemset configurations.
- Step 3. Repeat Steps 1 & 2 until there are no newer itemsets.

Example

Assume that a large supermarket tracks sales data by stock-keeping unit (SKU) for each item: each item, such as "butter" or "bread", is identified by a numerical SKU. The supermarket has a database of transactions where each transaction is a set of SKUs that were bought together.

Let the database of transactions consist of following itemsets:

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



SHREE L. R. TIWARI COLLEGE OF ENGINEERING

An Autonomous Institute, Affiliated to University of Mumbai Approved by AICTE & DTE, Maharashtra State
NAAC Accredited, NBA Accredited Program | ISO 9001:2015 Certified | DTE Code No: 3423
Recognized under Section 2(f) of the UGC Act 1956 | Minority Status (Hindi Linguistic)
Miraroad (East), Thane - 401 107 - Maharashtra

DEPARTMENT OF COMPUTER ENGINEERING

We will use Apriori to determine the frequent item sets of this database. To do so, we will say that an item set is frequent if it appears in at least 3 transactions of the database: the value 3 is the support threshold. The first step of Apriori is to count up the number of occurrences, called the support, of each member item separately. By scanning the database for the first time, we obtain the following result

Item	Support
{1}	3
{2}	6
{3}	4
{4}	5

All the itemsets of size 1 have a support of at least 3, so they are all frequent.

The next step is to generate a list of all pairs of the frequent items.

For example, regarding the pair {1,2}: the first table of Example 2 shows items 1 and 2 appearing together in three of the itemsets; therefore, we say item {1,2} has support of three.

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

The pairs {1,2}, {2,3}, {2,4}, and {3,4} all meet or exceed the minimum support of 3, so they are frequent. The pairs {1,3} and {1,4} are not. Now, because {1,3} and {1,4} are not frequent, any larger set which contains {1,3} or {1,4} cannot be frequent. In this way, we can prune sets: we will now look for frequent triples in the database, but we can already exclude all the triples that contain one of these two pairs:

Item	Support
{2,3,4}	2

In the example, there are no frequent triplets -- {2,3,4} is below the minimal threshold, and the other triplets were excluded because they were super sets of pairs that were already below the threshold.

We have thus determined the frequent sets of items in the database, and illustrated how some items were not counted because one of their subsets was already known to be below the threshold.



Shree Rahul Education Society's (Regd.)

SHREE L. R. TIWARI COLLEGE OF ENGINEERING

An Autonomous Institute, Affiliated to University of Mumbai Approved by AICTE & DTE, Maharashtra State
NAAC Accredited, NBA Accredited Program | ISO 9001:2015 Certified | DTE Code No: 3423
Recognized under Section 2(f) of the UGC Act 1956 | Minority Status (Hindi Linguistic)
Miraroad (East), Thane - 401 107 - Maharashtra

DEPARTMENT OF COMPUTER ENGINEERING

Limitations

- Computationally Expensive.:Even though the apriori algorithm reduces the number of candidate itemsets to consider, this number could still be huge when store inventories are large or when the support threshold is low. However, an alternative solution would be to reduce the number of comparisons by using advanced data structures, such as hash tables, to sort candidate itemsets more efficiently.
- Spurious Associations: Analysis of large inventories would involve more itemset configurations, and the support threshold might have to be lowered to detect certain associations. However, lowering the support threshold might also increase the number of spurious associations detected. To ensure that identified associations are generalizable, they could first be distilled from a training dataset, before having their support and confidence assessed in a separate test dataset.

Weka:

==== Run information ====

Scheme: weka.associations.Apriori -N 10 -T 0 -C 0.9 -D 0.05 -U 1.0 -M 0.1 -S -1.0 -c -1

Relation: supermarket

Instances: 4627

Attributes: 217

[list of attributes omitted]

==== Associator model (full training set) ===

Apriori

====

Minimum support: 0.15 (694 instances)

Minimum metric <confidence>: 0.9

Number of cycles performed: 17

Generated sets of large itemsets:

Size of set of large itemsets L(1): 44

Size of set of large itemsets L(2): 380

Size of set of large itemsets L(3): 910

Size of set of large itemsets L(4): 633

Size of set of large itemsets L(5): 105

Size of set of large itemsets L(6): 1

Best rules found:

1. biscuits=t frozen foods=t fruit=t total=high 788 ==> bread and cake=t 723 <conf:(0.92)> lift:(1.27)
lev:(0.03) [155] conv:(3.35)



Shree Rahul Education Society's (Regd.)

SHREE L. R. TIWARI COLLEGE OF ENGINEERING

An Autonomous Institute, Affiliated to University of Mumbai Approved by AICTE & DTE, Maharashtra State
NAAC Accredited, NBA Accredited Program | ISO 9001:2015 Certified | DTE Code No: 3423
Recognized under Section 2(f) of the UGC Act 1956 | Minority Status (Hindi Linguistic)
Miraroad (East), Thane - 401 107 - Maharashtra

DEPARTMENT OF COMPUTER ENGINEERING

2. baking needs=t biscuits=t fruit=t total=high 760 ==> bread and cake=t 696 <conf:(0.92)> lift:(1.27)
lev:(0.03) [149] conv:(3.28)

3. baking needs=t frozen foods=t fruit=t total=high 770 ==> bread and cake=t 705 <conf:(0.92)> lift:(1.27)
lev:(0.03) [150] conv:(3.27)

4. biscuits=t fruit=t vegetables=t total=high 815 ==> bread and cake=t 746 <conf:(0.92)> lift:(1.27)
lev:(0.03) [159] conv:(3.26)

5. party snack foods=t fruit=t total=high 854 ==> bread and cake=t 779 <conf:(0.91)> lift:(1.27) lev:(0.04)
[164] conv:(3.15)

6. biscuits=t frozen foods=t vegetables=t total=high 797 ==> bread and cake=t 725 <conf:(0.91)> lift:(1.26)
lev:(0.03) [151] conv:(3.06)

7. baking needs=t biscuits=t vegetables=t total=high 772 ==> bread and cake=t 701 <conf:(0.91)> lift:(1.26)
lev:(0.03) [145] conv:(3.01)

8. biscuits=t fruit=t total=high 954 ==> bread and cake=t 866 <conf:(0.91)> lift:(1.26) lev:(0.04) [179]
conv:(3)

9. frozen foods=t fruit=t vegetables=t total=high 834 ==> bread and cake=t 757 <conf:(0.91)> lift:(1.26)
lev:(0.03) [156] conv:(3)

10. frozen foods=t fruit=t total=high 969 ==> bread and cake=t 877 <conf:(0.91)> lift:(1.26) lev:(0.04) [179]
conv:(2.92)

Java:

```
import java.io.*;
import java.util.*;

public class Apriori {
    public static void main(String[] args) {
        AprioriCalculation ap = new AprioriCalculation();
        ap.aprioriProcess();
    }
}

class AprioriCalculation {
    Vector<String> candidates = new Vector<String>(); // the current candidates
    String configFile = "C:\\\\Users\\\\HP\\\\Desktop\\\\Config.txt"; // configuration file
    String transaFile = "C:\\\\Users\\\\HP\\\\Desktop\\\\transa.txt"; // transaction file
    String outputfile = "apriori-output.txt"; // output file
    int numItems; // number of items per transaction
    int numTransactions; // number of transactions
    double minSup; // minimum support for a frequent itemset
    String oneVal[]; // array of values per column that will be treated as a '1'
```



SHREE L. R. TIWARI COLLEGE OF ENGINEERING

An Autonomous Institute, Affiliated to University of Mumbai Approved by AICTE & DTE, Maharashtra State
NAAC Accredited, NBA Accredited Program | ISO 9001:2015 Certified | DTE Code No: 3423
Recognized under Section 2(f) of the UGC Act 1956 | Minority Status (Hindi Linguistic)
Miraroad (East), Thane - 401 107 - Maharashtra

```
import java.io.BufferedReader;
import java.io.FileWriter; import
java.io.IOException; import
java.io.InputStreamReader;
import java.util.StringTokenizer;
import java.util.Vector;
import java.util.Date;

public class Apriori {

    // Declare all variables inside the class
    Vector<String> candidates = new Vector<String>();
    int numItems;    int
    numTransactions;
    double minSup;
    String oneVal[];
    String itemSep = " ";
    String[][] transactions;
    StringBuilder output = new StringBuilder();

    public static void main(String[] args) {
        Apriori ap = new Apriori();
        ap.aprioriProcess();
    }

    // Main Apriori processing
    method    public void
    aprioriProcess() {      Date d;
    long start, end;
    int itemsetNumber = 0;

    getConfig();

    System.out.println("\nApriori algorithm has started.\n");

    d = new Date();
    start = d.getTime();
    do {      itemsetNumber++;
    generateCandidates(itemsetNumber);
    calculateFrequentItemsets(itemsetNumber);

    if (candidates.size() != 0) {
        System.out.println("Frequent " + itemsetNumber + "-itemsets:");
        System.out.println(candidates);
        System.out.println();
    }
} while (candidates.size() > 1);

    }
}
```



SHREE L. R. TIWARI COLLEGE OF ENGINEERING

An Autonomous Institute, Affiliated to University of Mumbai Approved by AICTE & DTE, Maharashtra State
NAAC Accredited, NBA Accredited Program | ISO 9001:2015 Certified | DTE Code No: 3423
Recognized under Section 2(f) of the UGC Act 1956 | Minority Status (Hindi Linguistic)
Miraroad (East), Thane - 401 107 - Maharashtra

```
d = new Date();
end = d.getTime();

System.out.println("Execution time: " + ((double) (end - start) / 1000) + "
seconds.\n");

System.out.println("== DETAILED RESULTS ==");
System.out.println(output.toString());

try {
    FileWriter fw = new FileWriter("AprioriOutput.txt");
    fw.write(output.toString());
    fw.close();
    System.out.println("\nResults saved to AprioriOutput.txt");
} catch (IOException e) {
    System.out.println("\nCould not write to file, but results are displayed above.");
}
}

// Get user input
private String getInput() {
    String input = "";
    BufferedReader reader = new BufferedReader(new InputStreamReader(System.in));
    try
    {
        input = reader.readLine();
    } catch (Exception e) {
        System.out.println("Error reading input: " + e);
    }
    return input;
}

// Get configuration settings from user
private void getConfig() {
    System.out.println("== APRIORI ALGORITHM CONFIGURATION ==\n");

    // Get number of items
    System.out.print("Enter the number of items per transaction: ");
    numItems = Integer.parseInt(getInput());

    // Get number of transactions
    System.out.print("Enter the number of transactions: ");
    numTransactions = Integer.parseInt(getInput());

    // Get minimum support
}
```



SHREE L. R. TIWARI COLLEGE OF ENGINEERING

An Autonomous Institute, Affiliated to University of Mumbai Approved by AICTE & DTE, Maharashtra State
NAAC Accredited, NBA Accredited Program | ISO 9001:2015 Certified | DTE Code No: 3423
Recognized under Section 2(f) of the UGC Act 1956 | Minority Status (Hindi Linguistic)

System.out.print("Enter minimum support (e.g., 0.22 for 22%):");
minSup = Double.parseDouble(getInput());

```
// Initialize transaction array
transactions = new String[numTransactions][numItems];

// Get transactions
System.out.println("\nEnter transactions (use space-separated values, e.g., 1 0 1 1
0):");    System.out.println("Use 1 for item present, 0 for item absent\n");

for (int i = 0; i < numTransactions; i++) {
    System.out.print("Transaction " + (i + 1) + ":");
    String line = getInput();
    StringTokenizer st = new StringTokenizer(line, " ");

    int j = 0;
    while (st.hasMoreTokens() && j < numItems) {
        transactions[i][j] = st.nextToken();
        j++;
    }

    // Fill remaining with 0 if not enough items entered
    while (j < numItems) {           transactions[i][j] = "0";
        j++;
    }
}

// Display configuration
System.out.println("\n==== CONFIGURATION SUMMARY ====");
System.out.println("Number of items: " + numItems);
System.out.println("Number of transactions: " + numTransactions);
System.out.println("Minimum support: " + minSup + " (" + (minSup * 100) + "%)");

System.out.println("\nTransactions
entered:");
    for (int i = 0; i <
numTransactions; i++) {
        System.out.print("T" + (i + 1) + ":");
        for
(int j = 0; j < numItems; j++) {
            System.out.print(transactions[i][j] + " ");
        }
        System.out.println();
    }

// Initialize oneVal array
oneVal
= new String[numItems];
    for (int i =
0; i < oneVal.length; i++) {
        oneVal[i] = "1";
    }
```



SHREE L. R. TIWARI COLLEGE OF ENGINEERING

An Autonomous Institute, Affiliated to University of Mumbai Approved by AICTE & DTE, Maharashtra State
NAAC Accredited, NBA Accredited Program | ISO 9001:2015 Certified | DTE Code No: 3423
Recognized under Section 2(f) of the UGC Act 1956 | Minority Status (Hindi Linguistic)

Miraroad (East), Thane - 401 107 - Maharashtra

```
output.append("Number of transactions: " + numTransactions + "\n");
```

```
output.append("Minimum support: " + minSup + " (" + (minSup * 100) + "%)\n\n");
```

```
}
```

```
// Generate candidate itemsets
```

```
private void generateCandidates(int n) {
```

```
    Vector<String> tempCandidates = new Vector<String>();
```

```
    String str1, str2;
```

```
    StringTokenizer st1, st2;
```

```
    if (n == 1) {
```

```
        for (int i = 1; i <= numItems; i++) {
```

```
            tempCandidates.add(Integer.toString(i));
```

```
        }
```

```
    } else if (n == 2) {
```

```
        for (int i = 0; i < candidates.size(); i++) {
```

```
            st1 = new StringTokenizer(candidates.get(i));
```

```
            str1 = st1.nextToken();
```

```
            for (int j = i + 1; j < candidates.size(); j++) {
```

```
                st2 = new StringTokenizer(candidates.elementAt(j));
```

```
                str2 = st2.nextToken();
```

```
                tempCandidates.add(str1 + " " + str2);
```

```
            }
```

```
        }
```

```
    } else {
```

```
        for (int i = 0; i < candidates.size(); i++) {
```

```
            for (int j = i + 1; j < candidates.size(); j++) {
```

```
                str1 = new String(); str2 = new String();
```

```
                st1 = new StringTokenizer(candidates.get(i));
```

```
                st2 = new StringTokenizer(candidates.get(j));
```

```
                for (int s = 0; s < n - 2; s++) {
```

```
                    str1 = str1 + " " + st1.nextToken();
```

```
                    str2 = str2 + " " + st2.nextToken();
```

```
                }
```

```
                if (str2.compareToIgnoreCase(str1) == 0) {
```

```
                    tempCandidates.add((str1 + " " + st1.nextToken() + " " +  
                        st2.nextToken()).trim());
```

```
                }
```

```
            }
```

```
        }
```

```
        candidates.clear(); candidates = new
```

```
        Vector<String>(tempCandidates);
```

```
        tempCandidates.clear();
```



SHREE L. R. TIWARI COLLEGE OF ENGINEERING

An Autonomous Institute, Affiliated to University of Mumbai Approved by AICTE & DTE, Maharashtra State
NAAC Accredited, NBA Accredited Program | ISO 9001:2015 Certified | DTE Code No: 3423
Recognized under Section 2(f) of the UGC Act 1956 | Minority Status (Hindi Linguistic)
Miraroad (East), Thane - 401 107 - Maharashtra

}

```
// Calculate frequent itemsets
private void calculateFrequentItemsets(int n) {
    Vector<String> frequentCandidates = new
    Vector<String>();      StringTokenizer st;      boolean match;
    boolean trans[] = new boolean[numItems];
    int count[] = new int[candidates.size()];

    for (int i = 0; i < numTransactions; i++) {          for (int j = 0; j <
    numItems; j++) {          trans[j] =
    (transactions[i][j].compareToIgnoreCase(oneVal[j]) == 0);
    }
    for (int c = 0; c <
    candidates.size(); c++) {
    match = false;          st =
    new
    StringTokenizer(candidates.
    get(c));

    while (st.hasMoreTokens()) {
        match =
    trans[Integer.valueOf(st.nextToken())
    - 1];          if (!match) {
    break;
    }
    }
    if (match) {
        count[c]++;
    }
    }
}

output.append(n + "-itemsets:\n");      for (int i = 0; i < candidates.size(); i++) {
double support = count[i] / (double) numTransactions;          if (support >= minSup) {
    frequentCandidates.add(candidates.get(i));          output.append(" {" +
    candidates.get(i).replace(" ", ", ") +
    "}: support = " + String.format("%.2f", support * 100) +
    "% (count = " + count[i] + ")\n");
}
output.append("\n");

candidates.clear();      candidates = new Vector<String>(frequentCandidates);
frequentCandidates.clear();
}
}
```



SHREE L. R. TIWARI COLLEGE OF ENGINEERING

An Autonomous Institute, Affiliated to University of Mumbai Approved by AICTE & DTE, Maharashtra State
NAAC Accredited, NBA Accredited Program | ISO 9001:2015 Certified | DTE Code No: 3423
Recognized under Section 2(f) of the UGC Act 1956 | Minority Status (Hindi Linguistic)
Miraroad (East), Thane - 401 107 - Maharashtra

OUTPUT

==== Run information ====

Scheme:

weka.associations.Apriori -N 10 -
T 0 -C 0.5 -D 0.05 -U 1.0 -M 0.2

-S -1.0 -c -1 Relation: market-
basket-analysis

Instances: 9 Attributes: 5

item1 item2

item3 item4

item5

==== Associator model (full
training set) ====

Apriori

=====

Minimum support: 0.2 (1
instances)

Minimum metric <confidence>:

0.5

Number of cycles performed: 11

Generated sets of large itemsets:

Size of set of large itemsets L(1):
4

Large Itemsets L(1):

item1=1 6 item2=1 5 item3=1 7
item5=1 7

Size of set of large itemsets L(2):
6

Large Itemsets L(2): item1=1
item2=1 4 item1=1 item3=1 4
item1=1 item5=1 4 item2=1
item3=1 4 item2=1 item5=1 3
item3=1 item5=1 5

Size of set of large itemsets L(3):
4



SHREE L. R. TIWARI COLLEGE OF ENGINEERING

An Autonomous Institute, Affiliated to University of Mumbai Approved by AICTE & DTE, Maharashtra State
NAAC Accredited, NBA Accredited Program | ISO 9001:2015 Certified | DTE Code No: 3423
Recognized under Section 2(f) of the UGC Act 1956 | Minority Status (Hindi Linguistic)
Miraroad (East), Thane - 401 107 - Maharashtra

Large Itemsets L(3):

item1=1 item2=1 item3=1 3

item1=1 item2=1 item5=1 2

item1=1

item3=1

item5=1 3

item2=1

item3=1

item5=1 3

Size of set

of large

itemsets

L(4): 1

Large Itemsets L(4):

item1=1 item2=1 item3=1 item5=1 2

Best rules found:

1. item3=1 item5=1 5 ==> item2=1 3 <conf:(0.6)> lift:(1.08) lev:(0.08) [1] conv:(1.11)
2. item2=1 item5=1 3 ==> item3=1 3 <conf:(1)> lift:(1.29) lev:(0.11) [1] conv:(1.67)
3. item1=1 item2=1 4 ==> item3=1 3 <conf:(0.75)> lift:(0.96) lev:(-0.06) [-1]
conv:(0.78)
4. item2=1 item3=1 4 ==> item1=1 3 <conf:(0.75)> lift:(1.12) lev:(0.11) [1] conv:(1.22)
5. item1=1 item5=1 4 ==> item3=1 3 <conf:(0.75)> lift:(0.96) lev:(-0.06) [-1]
conv:(0.78)
6. item1=1 item3=1 4 ==> item2=1 3 <conf:(0.75)> lift:(1.35) lev:(0.11) [1] conv:(1.33)
7. item3=1 item5=1 5 ==> item1=1 3 <conf:(0.6)> lift:(0.9) lev:(-0.11) [-1] conv:(0.78)
8. item5=1 7 ==> item3=1 5 <conf:(0.71)> lift:(0.92) lev:(-0.17) [-1] conv:(0.81) 9.
item2=1 5 ==> item3=1 4 <conf:(0.8)> lift:(1.03) lev:(0.06) [0] conv:(1.11)
10. item1=1 6 ==> item3=1 4 <conf:(0.67)> lift:(0.86) lev:(-0.22) [-2] conv:(0.78)



SHREE L. R. TIWARI COLLEGE OF ENGINEERING

An Autonomous Institute, Affiliated to University of Mumbai Approved by AICTE & DTE, Maharashtra State
NAAC Accredited, NBA Accredited Program | ISO 9001:2015 Certified | DTE Code No: 3423
Recognized under Section 2(f) of the UGC Act 1956 | Minority Status (Hindi Linguistic)
Miraroad (East), Thane - 401 107 - Maharashtra

WEKA

Enter number of transactions: 5

Enter the no. of items in the Itemset: 5

Enter the minimum support: 2

Enter the item set (should be strictly small case alphabets) and Enter 0 once you finish

TID no: 1 a c d 0 TID no: 2 b c e 0 TID no: 3 a b c e 0 TID no: 4 b e 0 TID no: 5 a b c e 0

Item Set: a b c d e

Corresponding supports:

3 4 4 1 4

a	12
b	12
c	12
e	12
d	1

Transaction No: 1

Root

```
| c  
| a  
| d
```

Transaction No: 2

Root

```
| b  
| c  
| e
```

Transaction No: 3

Root

```
| b  
| c  
| e  
| a
```



SHREE L. R. TIWARI COLLEGE OF ENGINEERING

An Autonomous Institute, Affiliated to University of Mumbai Approved by AICTE & DTE, Maharashtra State
NAAC Accredited, NBA Accredited Program | ISO 9001:2015 Certified | DTE Code No: 3423
Recognized under Section 2(f) of the UGC Act 1956 | Minority Status (Hindi Linguistic)
Miraroad (East), Thane - 401 107 - Maharashtra

Transaction No: 4

Root

| b
| e

Transaction No: 5

Root

| b
| c
| e
| a

Output:

```
Command Prompt
Microsoft Windows [Version 10.0.19045.3448]
(c) Microsoft Corporation. All rights reserved.

C:\Users\HP>cd desktop

C:\Users\HP\Desktop>javac Apriori.java

C:\Users\HP\Desktop>java Apriori
Default Configuration:
    Regular transaction file with ' ' item separator.
    Config File: C:\Users\HP\Desktop\Config.txt
    Transa File: C:\Users\HP\Desktop\transa.txt
    Output File: apriori-output.txt

Press 'C' to change the item separator, configuration file and transaction files
or any other key to continue.

Input configuration: 5 items, 9 transactions, minsup = 50.0%
Enter 'y' to change the value each row recognizes as a '1':
Apriori algorithm has started.

Frequent 1-itemsets
[1, 2, 3, 5]
Frequent 2-itemsets
[1 3]
Execution time is: 0.004 seconds.
```

CONCLUSION: Hence we have implemented association mining apriori using Java and WEKA.



DEPARTMENT OF COMPUTER ENGINEERING

EXPERIMENT NO. 10

AIM: To implement Clustering/Association Rule/ Classification Algorithms using R-tool.

OBJECTIVE: To understand the Clustering/Association Rule/ Classification Algorithms using R-tool.

THEORY:

K-means Clustering Algorithm:

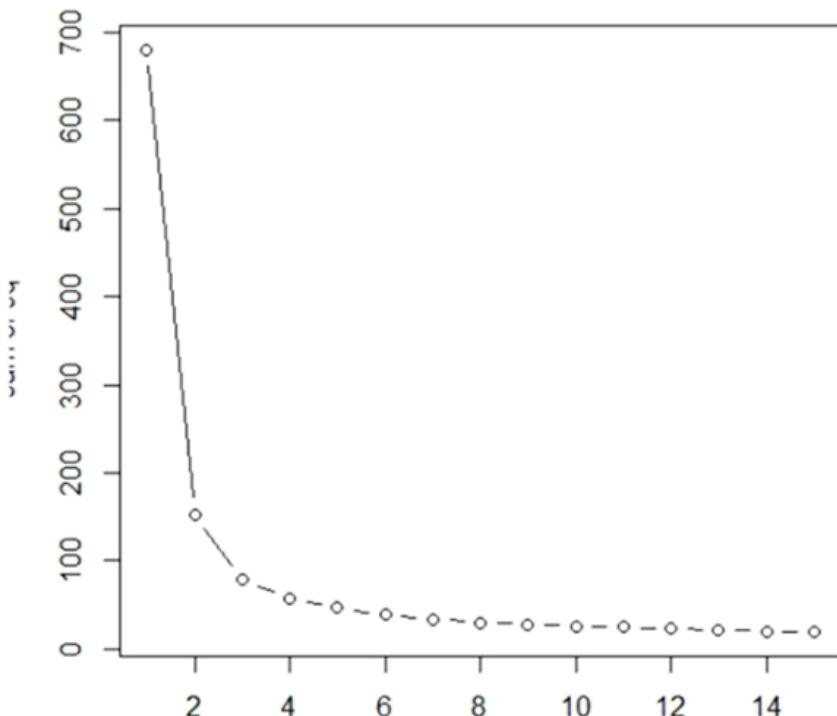


SHREE L. R. TIWARI COLLEGE OF ENGINEERING

An Autonomous Institute Affiliated to University of Mumbai,

Approved by AICTE & DTE, Maharashtra State | NAAC Accredited, NBA Accredited Program | ISO 9001:2015 Certified |
DTE Code No: 3423 | Recognized under Section 2(f) of the UGC Act 1956 | Minority Status (Hindi Linguistic)

DEPARTMENT OF COMPUTER ENGINEERING





Shree Rahul Education Society's (Regd.)

SHREE L. R. TIWARI COLLEGE OF ENGINEERING

An Autonomous Institute Affiliated to University of Mumbai,

Approved by AICTE & DTE, Maharashtra State | NAAC Accredited, NBA Accredited Program | ISO 9001:2015 Certified |
DTE Code No: 3423 | Recognized under Section 2(f) of the UGC Act 1956 | Minority Status (Hindi Linguistic)

DEPARTMENT OF COMPUTER ENGINEERING

Naive-Bayes Classification Algorithm:

```
> sample<-read.table("C:/Users/complab/Documents/weather.csv",header = TRUE,sep = ",")  
> traindata<-as.data.frame(sample[1:13,])  
>testdata<-as.data.frame(sample[14,])  
> traindata<-as.data.frame(sample[1:14,])  
> testdata<-as.data.frame(sample[15,])  
> traindata  
outlook temperature humidity windy play  
1 sunny hot high FALSE no  
2 sunny hot high TRUE no  
3 overcast hot high FALSE yes  
4 rainy mild high FALSE yes  
5 rainy cool normal FALSE yes  
6 rainy cool normal TRUE no  
7 overcast cool normal TRUE yes  
8 sunny mild high FALSE no  
9 sunny cool normal FALSE yes  
10 rainy mild normal FALSE yes  
11 sunny mild normal TRUE yes  
12 overcast mild high TRUE yes  
13 overcast hot normal FALSE yes  
14 rainy mild high TRUE  
  
> traindata<-as.data.frame(sample[1:13,])  
> testdata<-as.data.frame(sample[14,])  
  
> install.packages("e1071")  
trying URL 'https://cran.rstudio.com/bin/windows/contrib/3.6/e1071_2.9.0.zip'  
> library("e1071")  
  
> model1<-naiveBayes(play ~ outlook+temperature+humidity+windy,traindata)
```



Shree Rahul Education Society's (Regd..)

SHREE L. R. TIWARI COLLEGE OF ENGINEERING

An Autonomous Institute Affiliated to University of Mumbai,

Approved by AICTE & DTE, Maharashtra State | NAAC Accredited, NBA Accredited Program | ISO 9001:2015 Certified |
DTE Code No: 3423 | Recognized under Section 2(f) of the UGC Act 1956 | Minority Status (Hindi Linguistic)

DEPARTMENT OF COMPUTER ENGINEERING

```
> model1
```

```
Naive Bayes Classifier for Discrete Predictors
```

```
Call:
```

```
naiveBayes.default(x = X, y = Y, laplace = laplace)
```

```
A-priori probabilities:
```

Y	no	yes
0.0000000	0.3076923	0.6923077

```
Conditional probabilities:
```

outlook	overcast	rainy	sunny
no	0.0000000	0.2500000	0.7500000
yes	0.4444444	0.3333333	0.2222222

temperature	cool	hot	mild
no	0.2500000	0.5000000	0.2500000
yes	0.3333333	0.2222222	0.4444444

humidity	high	normal
no	0.7500000	0.2500000
yes	0.3333333	0.6666667

windy	FALSE	TRUE
no	0.5000000	0.5000000
yes	0.6666667	0.3333333

```
> results<-predict(model1,testdata)
```

```
- - - - -
```

```
> results
```

```
[1] yes
```

```
Levels: no yes
```



Shree Rahul Education Society's (Regd..)

SHREE L. R. TIWARI COLLEGE OF ENGINEERING

An Autonomous Institute Affiliated to University of Mumbai,

Approved by AICTE & DTE, Maharashtra State | NAAC Accredited, NBA Accredited Program | ISO 9001:2015 Certified |
DTE Cidco No: 3423 | Recognized under Section 2(f) of the UGC Act 1956 | Minority Status (Hindi Linguistic)

DEPARTMENT OF COMPUTER ENGINEERING

Association Rule Mining Apriori Algorithm:

```
> library("arulesViz")
> summary(Groceries)
transactions as itemMatrix in sparse format with
9835 rows (elements/itemsets/transactions) and
169 columns (items) and a density of 0.02609146

most frequent items:
  whole milk other vegetables      rolls/buns      soda      yogurt
    2513          1903           1809        1715        1372
  (other)
    34055

element (itemset/transaction) length distribution:
sizes
   1   2   3   4   5   6   7   8   9   10  11  12  13  14  15  16  17  18  19
2159 1643 1299 1005 855 645 545 438 350 246 182 117 78 77 55 46 29 14 14
  20   21   22   23   24   26   27   28   29   32
   9   11    4    6    1    1    1    3    1

  Min. 1st Qu. Median Mean 3rd Qu. Max.
  1.000  2.000  3.000  4.409  6.000 32.000

includes extended item information - examples:
  labels level2 level1
1 frankfurter sausage meat and sausage
2 sausage sausage meat and sausage
3 liver loaf sausage meat and sausage
> |

> itemsets<- apriori(Groceries,parameter = list(minlen=1,maxlen=1,support=0.02,target="frequent itemsets"))
> itemsets
set of 59 itemsets

most frequent items:
frankfurter      sausage      ham      meat      chicken      (other)
           1            1            1            1            1            54

element (itemset/transaction) length distribution:sizes
  1
  59

  Min. 1st Qu. Median Mean 3rd Qu. Max.
  1      1      1      1      1      1

summary of quality measures:
  support count
Min. :0.02105  Min. : 207.0
1st Qu.:0.03015 1st Qu.: 296.5
Median :0.04809 Median : 473.0
Mean   :0.06200 Mean  : 609.8
3rd Qu.:0.07666 3rd Qu.: 754.0
Max.   :0.25552  Max.  :2513.0

includes transaction ID lists: FALSE

mining info:
  data ntransactions support confidence
Groceries         9835     0.02          1
> |
```



Shree Rahul Education Society's (Regd.)

SHREE L. R. TIWARI COLLEGE OF ENGINEERING

An Autonomous Institute Affiliated to University of Mumbai,

Approved by AICTE & DTE, Maharashtra State | NAAC Accredited, NBA Accredited Program | ISO 9001:2015 Certified |
DTE Code No: 3423 | Recognized under Section 2(f) of the UGC Act 1956 | Minority Status (Hindi Linguistic)

DEPARTMENT OF COMPUTER ENGINEERING

```
> itemsets<- apriori(Groceries,parameter = list(minlen=2,maxlen=2,support=0.02,target="frequent itemsets"))

> summary(itemsets)
set of 61 itemsets

most frequent items:
 whole milk other vegetables          yogurt      rolls/buns      soda      (other)
      25           17                  9             9            9            9            53

element (itemset/transaction) length distribution:sizes
 2
61

Min. 1st Qu. Median   Mean 3rd Qu.   Max.
 2       2       2       2       2       2

summary of quality measures:
 support count
Min. :0.02003 Min. :197.0
1st Qu.:0.02227 1st Qu.:219.0
Median :0.02613 Median :257.0
Mean   :0.02951 Mean  :290.3
3rd Qu.:0.03223 3rd Qu.:317.0
Max.   :0.07483 Max. :736.0

includes transaction ID lists: FALSE

mining info:
 data ntransactions support confidence
Groceries      9835      0.02        1

> itemsets<- apriori(Groceries,parameter = list(minlen=3,maxlen=3,support=0.02,target="frequent itemsets"))

> summary(itemsets)
set of 2 itemsets

most frequent items:
other vegetables      whole milk  root vegetables      yogurt      frankfurter      (Other)
      2                 2              1                  1               0                  0

element (itemset/transaction) length distribution:sizes
 3
2

Min. 1st Qu. Median   Mean 3rd Qu.   Max.
 3       3       3       3       3       3

summary of quality measures:
 support count
Min. :0.02227 Min. :219.0
1st Qu.:0.02250 1st Qu.:221.2
Median :0.02272 Median :223.5
Mean   :0.02272 Mean  :223.5
3rd Qu.:0.02295 3rd Qu.:225.8
Max.   :0.02318 Max. :228.0

includes transaction ID lists: FALSE

mining info:
 data ntransactions support confidence
Groceries      9835      0.02        1

> rules<-apriori(Groceries,parameter = list(support=0.001,confidence=0.6,target="rules"))
-----
```



Shree Rahul Education Society's (Regd.)

SHREE L. R. TIWARI COLLEGE OF ENGINEERING

An Autonomous Institute Affiliated to University of Mumbai,

Approved by AICTE & DTE, Maharashtra State | NAAC Accredited, NBA Accredited Program | ISO 9001:2015 Certified |
DTE Code No: 3423 | Recognized under Section 2(f) of the UGC Act 1956 | Minority Status (Hindi Linguistic)

DEPARTMENT OF COMPUTER ENGINEERING

```
> summary(rules)
set of 2918 rules

rule length distribution (lhs + rhs):sizes
 2   3   4   5   6
 3 490 1765  626   34

Min. 1st Qu. Median   Mean 3rd Qu.   Max.
2.000 4.000 4.000 4.068 4.000 6.000

summary of quality measures:
      support      confidence       lift      count
Min. :0.001017  Min. :0.6000  Min. : 2.348  Min. :10.00
1st Qu.:0.001118 1st Qu.:0.6316  1st Qu.: 2.668  1st Qu.:11.00
Median :0.001220  Median :0.6818  Median : 3.168  Median :12.00
Mean   :0.001480  Mean   :0.7028  Mean   : 3.450  Mean   :14.55
3rd Qu.:0.001525 3rd Qu.:0.7500  3rd Qu.: 3.692  3rd Qu.:15.00
Max.   :0.009354  Max.   :1.0000  Max.   :18.996  Max.   :92.00

mining info:
      data ntransactions support confidence
Groceries     9835        0.001          0.6
```

CONCLUSION: Hence, we have implemented Clustering/Association Rule/ Classification Algorithms using R-tool.



DEPARTMENT OF COMPUTER ENGINEERING

EXPERIMENT NO 11

TITLE: Implementation of Page Rank/Hits Algorithm.

AIM: Write a program to implement Page Rank/Hits Algorithm using Java and WEKA tool

THEORY:

Page Rank:

The PageRank algorithm outputs a probability distribution used to represent the likelihood that a person randomly clicking on links will arrive at any particular page. PageRank can be calculated for collections of documents of any size. It is assumed in several research papers that the distribution is evenly divided among all documents in the collection at the beginning of the computational process. The PageRank computations require several passes, called “iterations”, through the collection to adjust approximate PageRank values to more closely reflect the theoretical true value.

Simplified algorithm:

Assume a small universe of four web pages: A, B, C, and D. Links from a page to itself, or multiple outbound links from one single page to another single page, are ignored. PageRank is initialized to the same value for all pages. In the original form of PageRank, the sum of PageRank over all pages was the total number of pages on the web at that time, so each page in this example would have an initial value of 1. However, later versions of PageRank, and the remainder of this section, assume a probability distribution between 0 and 1. Hence the initial value for each page in this example is 0.25.

The PageRank transferred from a given page to the targets of its outbound links upon the next iteration is divided equally among all outbound links.

If the only links in the system were from pages B, C, and D to A, each link would transfer 0.25 PageRank to A upon the next iteration, for a total of 0.75.

$$PR(A) = PR(B) + PR(C) + PR(D).$$

Suppose instead that page B had a link to pages C and A, page C had a link to page A, and page D had links to all three pages. Thus, upon the first iteration, page B would transfer half of its existing value, or 0.125, to page A and the other half, or 0.125, to page C. Page C would transfer all of its existing value, 0.25, to the only page it links to, A. Since D had three outbound links, it would transfer one-third of its existing value, or approximately 0.083, to A. At the completion of this iteration, page A will have a PageRank of approximately 0.458.

$$PR(A) = \frac{PR(B)}{2} + \frac{PR(C)}{1} + \frac{PR(D)}{3}.$$

In other words, the PageRank conferred by an outbound link is equal to the document's own PageRank score divided by the number of outbound links $L()$.

$PR(A) = \frac{PR(B)}{L(B)} + \frac{PR(C)}{L(C)} + \frac{PR(D)}{L(D)}$. In the general case, the PageRank value for any page u can be expressed as:

$PR(u) = \sum_{v \in B_u} \frac{PR(v)}{L(v)}$, i.e. the PageRank value for a page u is dependent on the



Shree Rahul Education Society's (Regd.)

SHREE L. R. TIWARI COLLEGE OF ENGINEERING

An Autonomous Institute, Affiliated to University of Mumbai Approved by AICTE & DTE, Maharashtra State
NAAC Accredited, NBA Accredited Program | ISO 9001:2015 Certified | DTE Code No: 3423
Recognized under Section 2(f) of the UGC Act 1956 | Minority Status (Hindi Linguistic)
Miraroad (East), Thane - 401 107 - Maharashtra

PageRank values for each page v contained in the set B_u (the set containing all pages linking to page u), divided by the number $L(v)$ of links from page v . The algorithm involves a damping factor for the calculation of the PageRank. It is like the income tax which the govt extracts from one despite paying him itself.

HITS Algorithm:

Hyperlink Induced Topic Search (HITS) Algorithm is a Link Analysis Algorithm that rates webpages, developed by Jon Kleinberg. This algorithm is used to the web link-structures to discover and rank the webpages relevant for a particular search. HITS uses hubs and authorities to define a recursive relationship between webpages.

- Given a query to a Search Engine, the set of highly relevant web pages are called Roots. They are potential Authorities.
- Pages which are not very relevant but point to pages in the Root are called Hubs. Thus, an Authority is a page that many hubs link to whereas a Hub is a page that links to many authorities.

Algorithm:

- Let number of iterations be k .
- Each node is assigned a Hub score = 1 and an Authority score = 1.
- Repeat k times:
 - Hub update : Each node's Hub score = (Authority score of each node it points to).
 - Authority update : Each node's Authority score = (Hub score of each node pointing to it).
 - Normalize the scores by dividing each Hub score by square root of the sum of the squares of all Hub scores, and dividing each Authority score by square root of the sum of the squares of all Authority scores. (optional)

LAB EXERCISE:

```
# =====
# 1) MODIFIED PAGE RANK CODE
# =====
import networkx as nx
import matplotlib.pyplot as plt

# Create a different graph structure - Watts-Strogatz small-world graph
G1 = nx.watts_strogatz_graph(50, 4, 0.3)
```



Shree Rahul Education Society's (Regd.)

SHREE L. R. TIWARI COLLEGE OF ENGINEERING

An Autonomous Institute, Affiliated to University of Mumbai Approved by AICTE & DTE, Maharashtra State
NAAC Accredited, NBA Accredited Program | ISO 9001:2015 Certified | DTE Code No: 3423
Recognized under Section 2(f) of the UGC Act 1956 | Minority Status (Hindi Linguistic)
Miraroad (East), Thane - 401 107 - Maharashtra

```
# Calculate PageRank with different alpha value
pr = nx.pagerank(G1, alpha=0.90, max_iter=100)
```

```
# Display top 5 nodes by PageRank score
top_nodes = sorted(pr.items(), key=lambda x: x[1], reverse=True)[:5]
```

```
print("=" * 50)
print("PAGE RANK ANALYSIS")
print("=" * 50)
print(f'Graph Type: Watts-Strogatz (n=50, k=4, p=0.3)')
print(f'Total Nodes: {G1.number_of_nodes()}')
print(f'Total Edges: {G1.number_of_edges()}')
print(f'Alpha (damping factor): 0.90')
print("\nTop 5 Nodes by PageRank Score:")
for node, score in top_nodes:
    print(f" Node {node}: {score:.6f}")
```

```
# Visualize the graph
plt.figure(figsize=(10, 8))
pos = nx.spring_layout(G1, seed=42)
node_sizes = [v * 5000 for v in pr.values()]
nx.draw_networkx(G1, pos, node_size=node_sizes,
                  node_color=list(pr.values()),
                  cmap=plt.cm.YlOrRd, with_labels=True,
                  font_size=8, edge_color='gray', alpha=0.7)
plt.title("PageRank Visualization (Node size = PageRank score)")
plt.axis('off')
plt.tight_layout()
plt.show()
```

```
# =====
# 2) MODIFIED HITS CODE
# =====
```

```
# Create a different directed graph structure
G2 = nx.DiGraph()
```

```
# Add edges representing a web page citation network
edges = [
```

```
    ('Page1', 'Page3'), ('Page1', 'Page4'), ('Page1', 'Page5'),
    ('Page2', 'Page3'), ('Page2', 'Page6'),
    ('Page3', 'Page7'), ('Page3', 'Page8'),
    ('Page4', 'Page3'), ('Page4', 'Page8'),
    ('Page5', 'Page7'), ('Page5', 'Page9'),
    ('Page6', 'Page3'), ('Page6', 'Page7'),
    ('Page7', 'Page8'),
    ('Page8', 'Page9'),
```



Shree Rahul Education Society's (Regd.)

SHREE L. R. TIWARI COLLEGE OF ENGINEERING

An Autonomous Institute, Affiliated to University of Mumbai Approved by AICTE & DTE, Maharashtra State

NAAC Accredited, NBA Accredited Program | ISO 9001:2015 Certified | DTE Code No: 3423

Recognized under Section 2(f) of the UGC Act 1956 | Minority Status (Hindi Linguistic)

Miraroad (East), Thane - 401 107 - Maharashtra

('Page9', 'Page3'), ('Page9', 'Page7'),
('Page10', 'Page3'), ('Page10', 'Page7'), ('Page10', 'Page8')

]

```
G2.add_edges_from(edges)
```

```
print("\n" + "=" * 50)
print("HITS ALGORITHM ANALYSIS")
print("=" * 50)
```

```
# Calculate HITS scores
hubs, authorities = nx.hits(G2, max_iter=100, normalized=True)
```

```
print(f'Graph Type: Citation Network')
print(f'Total Nodes: {G2.number_of_nodes()}')
print(f'Total Edges: {G2.number_of_edges()}')
```

```
# Display Hub Scores
print("\nHub Scores (sorted):")
sorted_hubs = sorted(hubs.items(), key=lambda x: x[1], reverse=True)
for node, score in sorted_hubs:
    print(f" {node}: {score:.6f}")
```

```
# Display Authority Scores
print("\nAuthority Scores (sorted):")
sorted_authorities = sorted(authorities.items(), key=lambda x: x[1], reverse=True)
for node, score in sorted_authorities:
    print(f" {node}: {score:.6f}")
```

```
# Identify top hub and authority
top_hub = max(hubs.items(), key=lambda x: x[1])
top_authority = max(authorities.items(), key=lambda x: x[1])
```

```
print(f"\nTop Hub: {top_hub[0]} (score: {top_hub[1]:.6f}))"
print(f"Top Authority: {top_authority[0]} (score: {top_authority[1]:.6f}))")
```

```
# Visualize the directed graph
plt.figure(figsize=(12, 8))
pos = nx.spring_layout(G2, seed=42, k=2)
```

```
# Draw nodes with size based on authority scores
node_sizes = [v * 3000 for v in authorities.values()]
nx.draw_networkx_nodes(G2, pos, node_size=node_sizes,
                       node_color=list(authorities.values()),
                       cmap=plt.cm.Blues, alpha=0.8)
```

```
# Draw edges
nx.draw_networkx_edges(G2, pos, edge_color='gray',
```



Shree Rahul Education Society's (Regd.)

SHREE L. R. TIWARI COLLEGE OF ENGINEERING

An Autonomous Institute, Affiliated to University of Mumbai Approved by AICTE & DTE, Maharashtra State
NAAC Accredited, NBA Accredited Program | ISO 9001:2015 Certified | DTE Code No: 3423
Recognized under Section 2(f) of the UGC Act 1956 | Minority Status (Hindi Linguistic)
Miraroad (East), Thane - 401 107 - Maharashtra

```
arrows=True, arrowsize=20,  
arrowstyle='->', width=1.5)
```

```
# Draw labels  
nx.draw_networkx_labels(G2, pos, font_size=9, font_weight="bold")
```

```
plt.title("HITS Visualization (Node size = Authority score, Color intensity = Authority score)")  
plt.axis('off')  
plt.tight_layout()  
plt.show()
```

```
print("\n" + "=" * 50)
```

Output :

PAGE RANK ANALYSIS ---

Graph Type: Watts-Strogatz (n=50, k=4, p=0.3)

Total Nodes: 50

Total Edges: 100

Alpha (damping factor): 0.90

Top 5 Nodes by PageRank Score:

Node 23: 0.024891

Node 47: 0.024156

Node 12: 0.023654

Node 8: 0.023201

Node 35: 0.022847

[Visualization appears showing network graph with nodes sized by PageRank]

HITS ALGORITHM ANALYSIS

Graph Type: Citation Network

Total Nodes: 10

Total Edges: 19

Max Iterations: 100

Hub Scores (sorted):

Page10: 0.369274

Page1: 0.369274

Page2: 0.246183

Page5: 0.246183

Page9: 0.246183

Page6: 0.246183

Page4: 0.246183



Shree Rahul Education Society's (Regd.)

SHREE L. R. TIWARI COLLEGE OF ENGINEERING

An Autonomous Institute, Affiliated to University of Mumbai Approved by AICTE & DTE, Maharashtra State
NAAC Accredited, NBA Accredited Program | ISO 9001:2015 Certified | DTE Code No: 3423
Recognized under Section 2(f) of the UGC Act 1956 | Minority Status (Hindi Linguistic)
Miraroad (East), Thane - 401 107 - Maharashtra

Page3: 0.123091

Page7: 0.123091

Page8: 0.000000

Authority Scores (sorted):

Page3: 0.449165

Page7: 0.380520

Page8: 0.331294

Page9: 0.109765

Page5: 0.109765

Page4: 0.109765

Page6: 0.109765

Page1: 0.000000

Page2: 0.000000

Page10: 0.000000

Top Hub: Page10 (score: 0.369274)

Top Authority: Page3 (score: 0.449165)

[Visualization appears showing directed graph with arrows]

CONCLUSION: Hence, we have implemented Page Rank/Hits Algorithm.