ARUNAI ENGINEERING COLLEGE
(An Autonomous Institution)
Velu Nagar, Tiruvannamalai – 606603
www.arunai.org

DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND DATA SCIENCE

BACHELOR OF TECHNOLOGY
2024–2025

SIXTH SEMESTER

CCS341-DATA WAREHOUSING LABORATORY

DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND DATA SCIENCE

CERTIFICATE

Certified that this is a Bonafide record of work done by

Name :

UniversityReg.No :

Semester :

Branch :

Year :

Staff-in-Charge **Head of the Department**

Submitted for the_____

Practical Examination held on_____

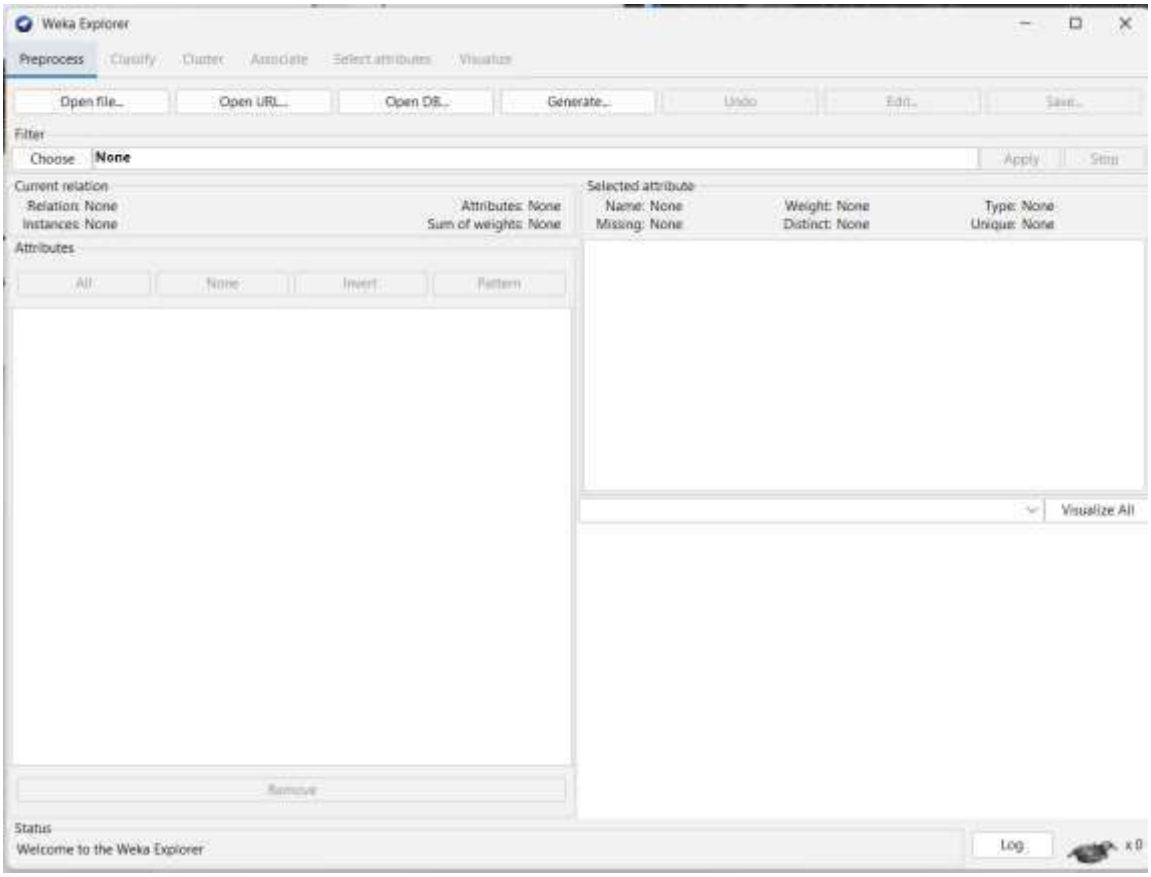
Internal Examiner **External Examiner**

TABLE OF CONTENTS

Ex No: 1&2 **Date:**	**DATA EXPLORATION AND INTEGRATION, DATA VALIDATION WITH WEKA**

INTRODUCTION:

Invoke Weka from the Windows Start menu (on Linux or the Mac, double-click weka.jar or weka.app, respectively). This starts up the Weka GUI Chooser.Click the Explorer button to enter the Weka Explorer. The Preprocess panel opens up when the Explorer interface is started. Click the open file option and starts perform the respective operations, this can be shown below the figure.



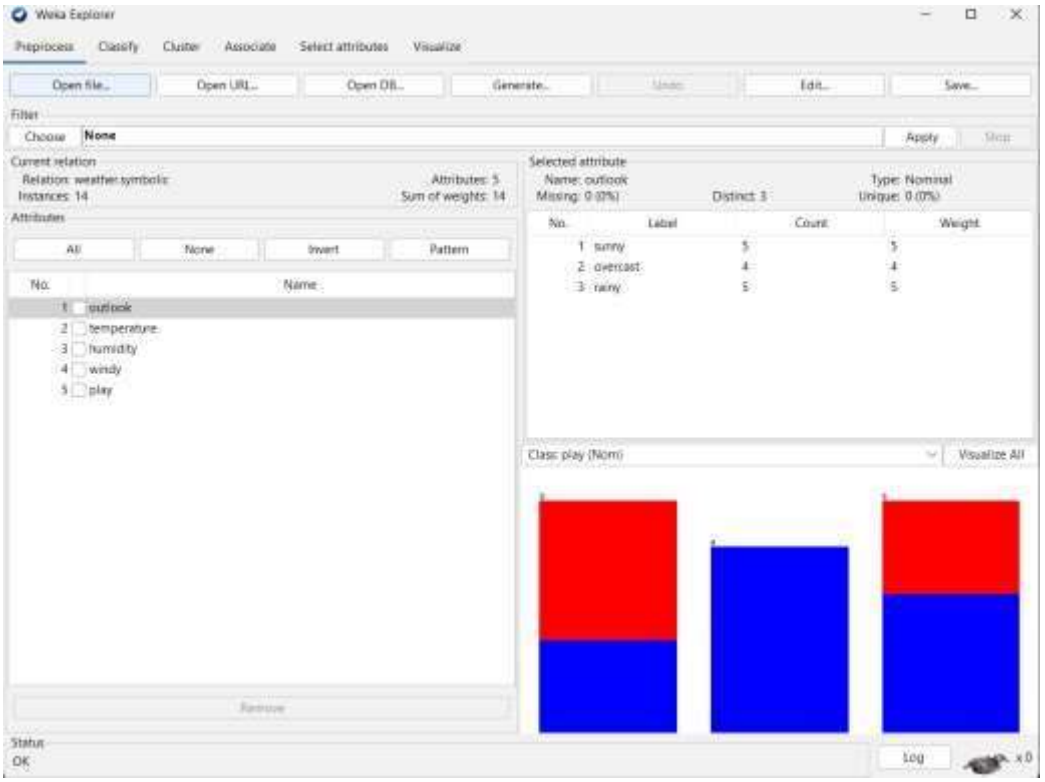
THE PANELS:

1. PREPROCESS.
2. CLASSIFY.
3. CLUSTER.
4. ASSOCIATE.
5. SELECT ATTRIBUTE
6. VISUALIZE

PREPROCESS PANEL

LOADING THE DATA-SET:

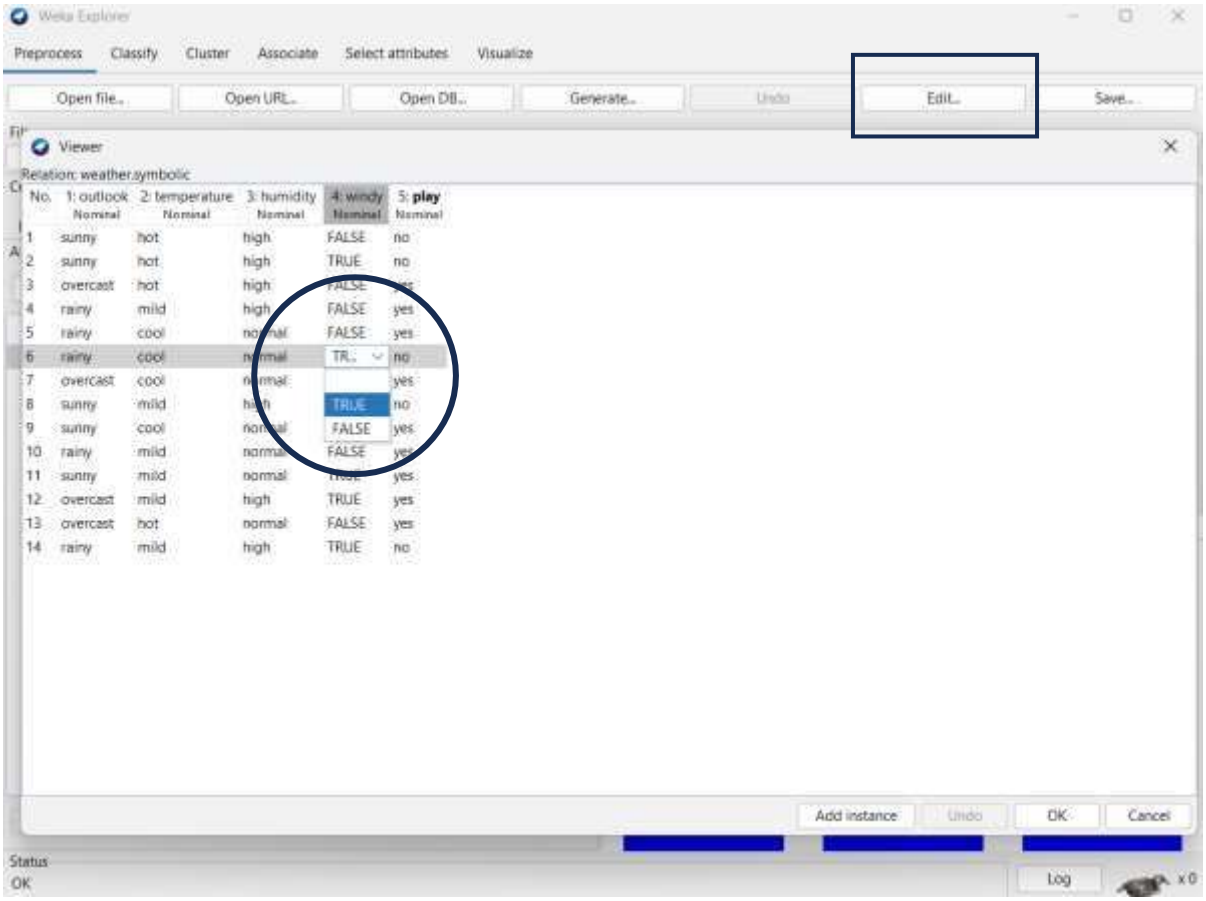
Load the dataset from the data folder at the open file option, choose the required dataset from the list of datasets. Here, for the experiment we chose *"Weather.nominal.arff"* dataset and analyse the attributes from the dataset.



As the result shows, the weather data has 14 instances, and 5 attributes called *outlook*, *temperature*, *humidity*, *windy*, and *play*. Click on the name of an attribute in the left subpanel to see information about the selected attribute on the right, such as its values and how many times an instance in the dataset has a particular value. This information is also shown in the form of a histogram. All attributes in this dataset are "nominal"—that is, they have a predefined finite set of values. The last attribute, *play*, is the "class" attribute; its value can be *yes* or *no*.

DATA SET EDITOR:

It is possible to view and edit an entire dataset from within Weka. To do this, load the *weather.nominal.arff* file again. Click the *Edit* button from the row of buttons at the top of the Preprocess panel. This opens a new window called Viewer, which lists all instances of the weather data.



APPLYING FILTER:

As you know, Weka "filters" can be used to modify datasets in a systematic fashion--that is, they are data Preprocessing tools. Reload the *weather.nominal* dataset, and let's remove an attribute from it. The appropriate filter is called *Remove*; its full name is:

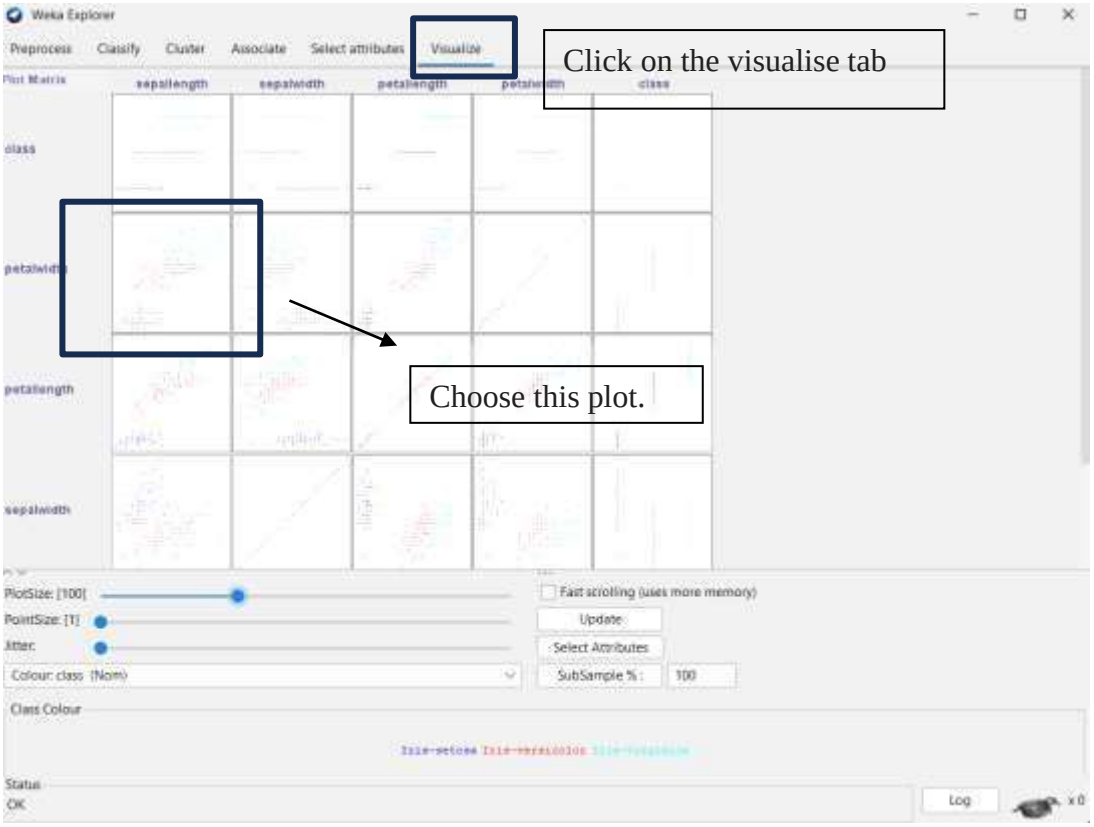
```
weka.filters.unsupervised.attribute.Rem
```

THE VISUALISE PANEL:

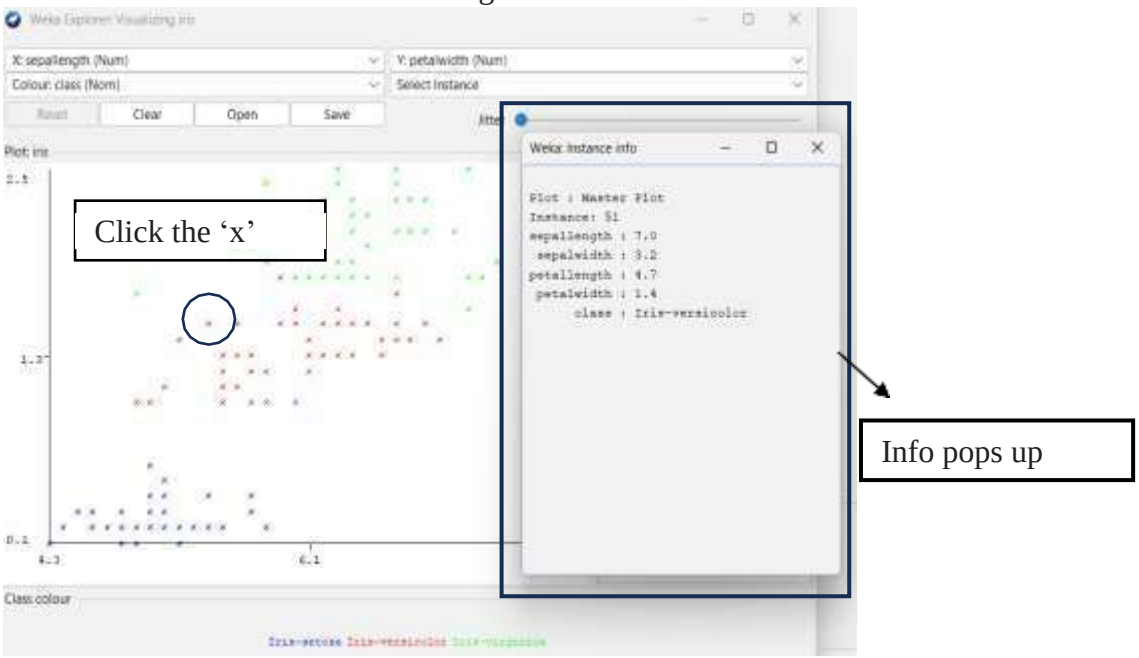
Now take a look at Weka's data visualization facilities. These work best with numeric data, so we use the iris data. Load iris.arff, which contains the iris dataset containing 50 examples of three types of Iris: Iris setosa, Iris versicolor, and Iris virginica.

1. Click the Visualize tab to bring up the Visualize panel.

2. Click the first plot in the second row to open a window showing an enlarged plot using the selected axes. Instances are shown as little crosses, the colour of which depends on the instance's class. The x-axis shows the sepal length attribute, and the y-axis shows petal width.



3. Clicking on one of the crosses opens up an Instance Info window, which lists the values of all attributes for the selected instance. Close the Instance Info window again.



The selection fields at the top of the window containing the scatter plot determine which attributes are used for the *x*- and *y*-axes. Change the *x*-axis to *petalwidth* and the *y*-axis to *petallength*. The field showing *Color: class (Num)* can be used to change the color coding.

Each of the barlike plots to the right of the scatter plot window represents a single attribute. In each bar, instances are placed at the appropriate horizontal position and scattered randomly in the vertical direction. Clicking a bar uses that attribute for the *x*-axis of the scatter plot. Right-clicking a bar does the same for the *y*-axis. Use these bars to change the *x*- and *y*-axes back to *sepallength* and *petalwidth*.

The ***Jitter*** slider displaces the cross for each instance randomly from its true position, and can reveal situations where instances lie on top of one another.

Experiment a little by moving the slider.

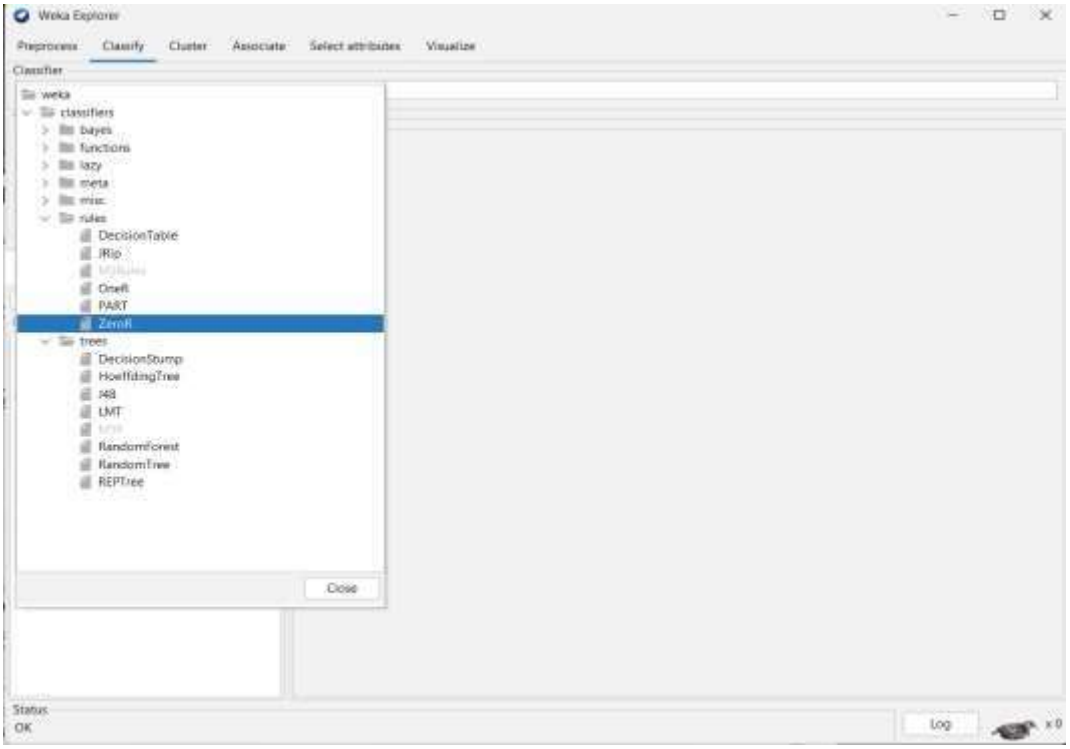
The ***Select Instance*** button and the *Reset*, *Clear*, and *Save* buttons let you modify the dataset. Certain instances can be selected and the others removed. Try the Rectangle option: Select an area by left-clicking and dragging the mouse. The *Reset* button changes into a *Submit* button. Click it, and all instances outside the rectangle are deleted. You could use *Save* to save the modified dataset to a file. *Reset* restores the original dataset.T

CLASSIFY PANEL:

Now we apply a classifier to the weather data. Load the weather data again. Go to the Preprocess panel, click the *Open file* button, and select **"*weather.nominal.arff*"** from the data directory. Then switch to the Classify panel by clicking the *Classify* tab at the top of the window.

USING THE C4.5 CLASSIFIER:

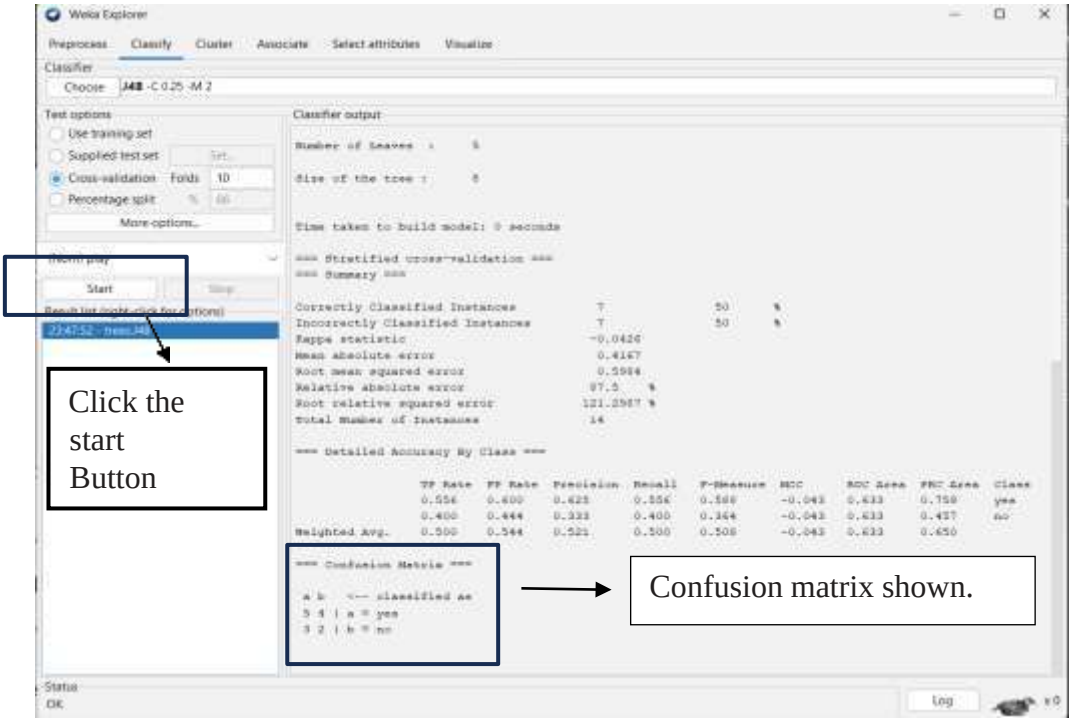
The C4.5 algorithm for building decision trees is implemented in Weka as a classifier called *J48*. Select it by clicking the *Choose* button near the top of the *Classify* tab. A dialog window appears showing various types of classifier. Click the *trees* entry to reveal its subentries, and click *J48* to choose that classifier. Classifiers, like filters, are organized in a hierarchy: *J48* has the full name *weka.classifiers.trees.J48*.



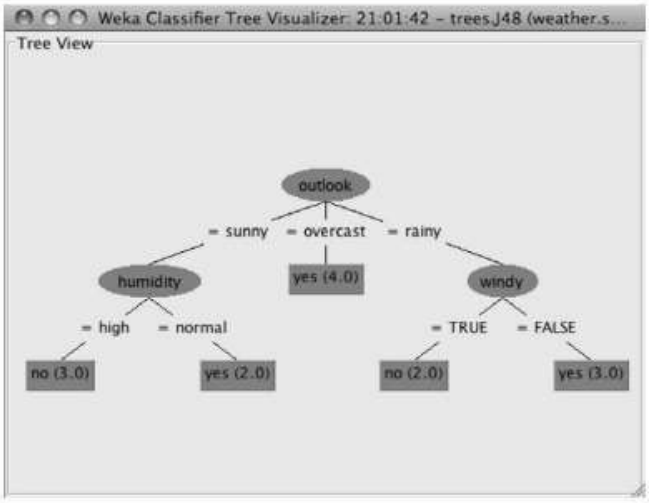
OUTPUT:

The outcome of training and testing appears in the Classifier Output box on the right. Scroll through the text and examine it. First, look at the part that describes the decision tree, reproduce in image below. This represents the decision tree that was built, including the number of instances that fall under each leaf. The textual representation is clumsy to interpret, but Weka can generate an equivalent graphical version.

Here's how to get the graphical tree. Each time the *Start* button is pressed and a new classifier is built and evaluated, a new entry appears in the Result List panel in the lower left corner.



BUILTING THE DECISION TREE:



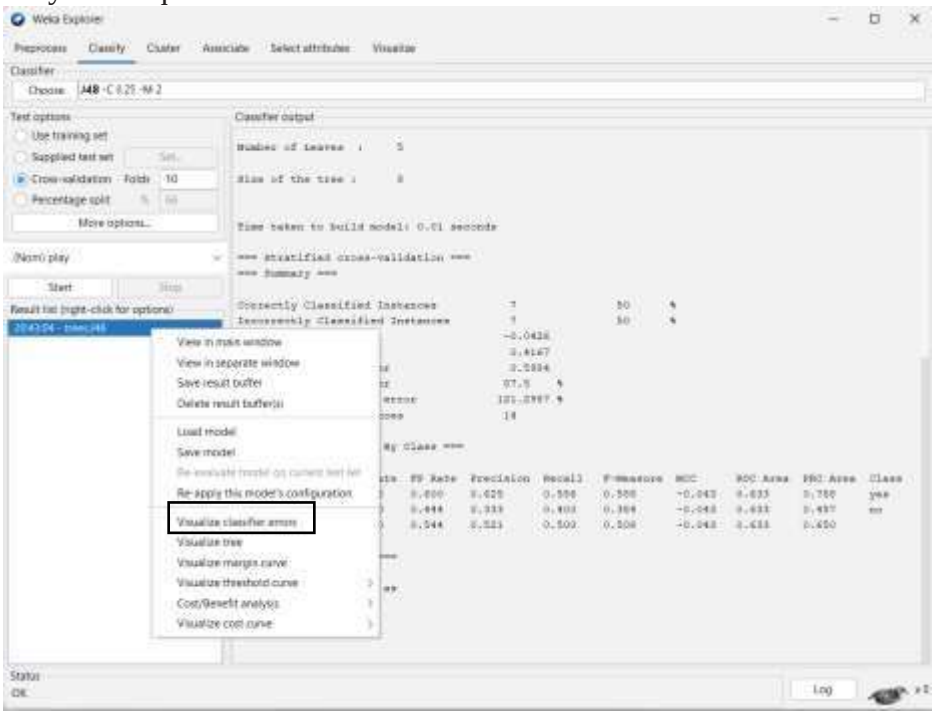
Setting the Test Method:

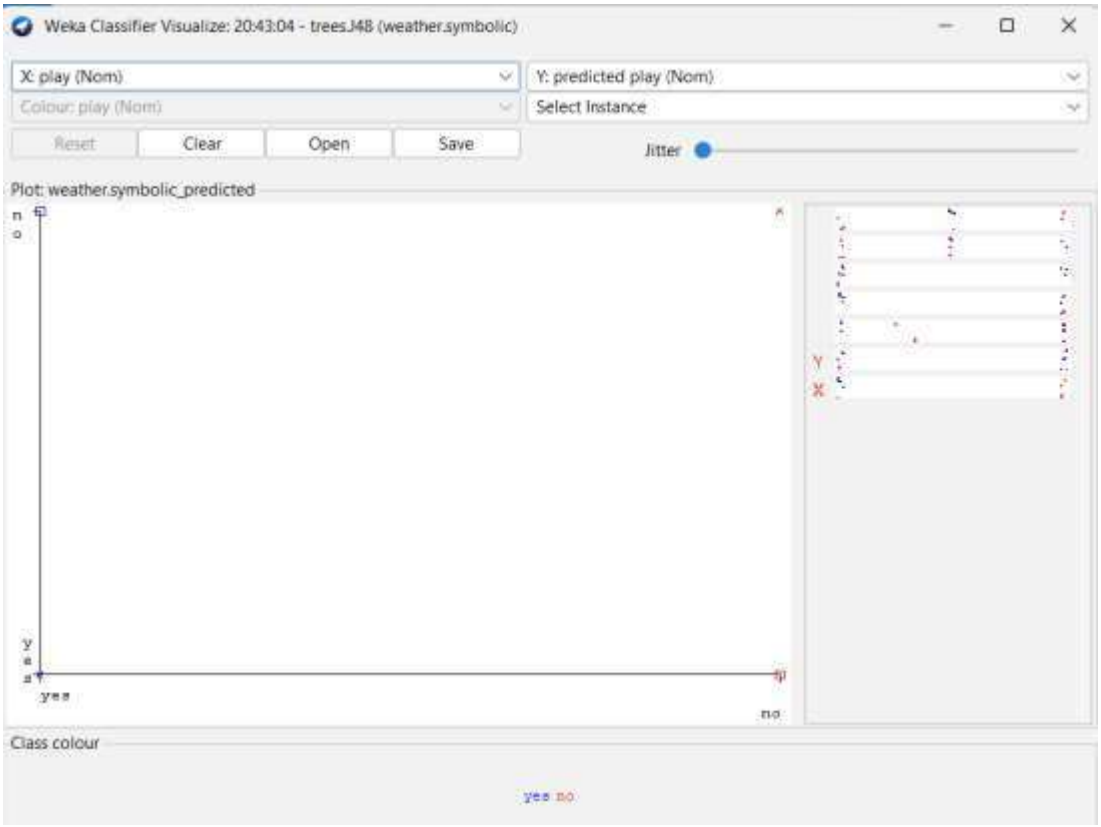
When the *Start* button is pressed, the selected learning algorithm is run and the dataset that was loaded in the Preprocess panel is used with the selected test protocol.

For example, in the case of tenfold cross-validation this involves running the learning algorithm 10 times to build and evaluate 10 classifiers. A model built from the *full* training set is then printed into the Classifier **Output area:** This may involve running the learning algorithm one final time. The remainder of the output depends on the test protocol that was chosen using test options.

VISUALISE THE ERRORS:

Right-click the *trees.J48* entry in the result list and choose *Visualize classifier errors*. A scatter plot window pops up. Instances that have been classified correctly are marked by little crosses; ones that are incorrect are marked by little squares.





CLUSTER PANEL :

Clustering Data :

WEKA contains "clusterers" for finding groups of similar instances in a dataset. The clustering schemes available in WEKA are,

- ✓ *k*-Means,
- ✓ EM,
- ✓ Cobweb,
- ✓ *X*-means,
- ✓ Farthest First.

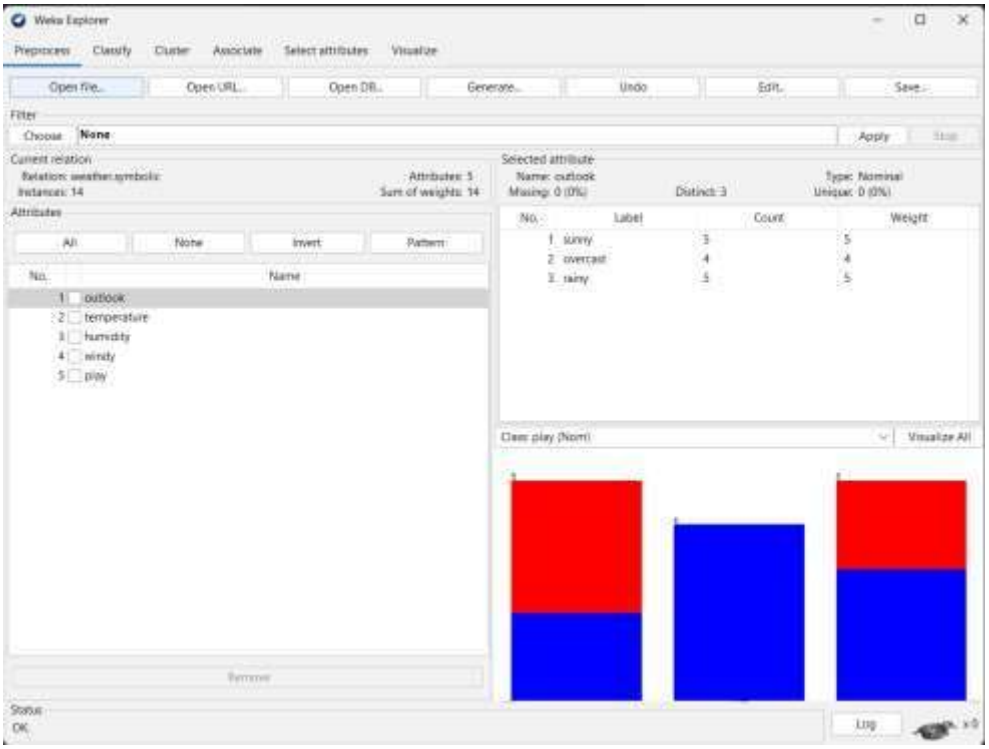
Clusters can be visualized and compared to "true" clusters (if given). Evaluation is based on log likelihood if clustering scheme produces a probability distribution.

For this exercise we will use customer data that is contained in "customers.arff" file and analyze it with "***k-means"*** clustering scheme.

Steps:

(i) Select the file from WEKA

In 'Preprocess' window click on 'Open file…' button and select ***"weather.arff"*** file. Click 'Cluster' tab at the top of WEKA Explorer window.

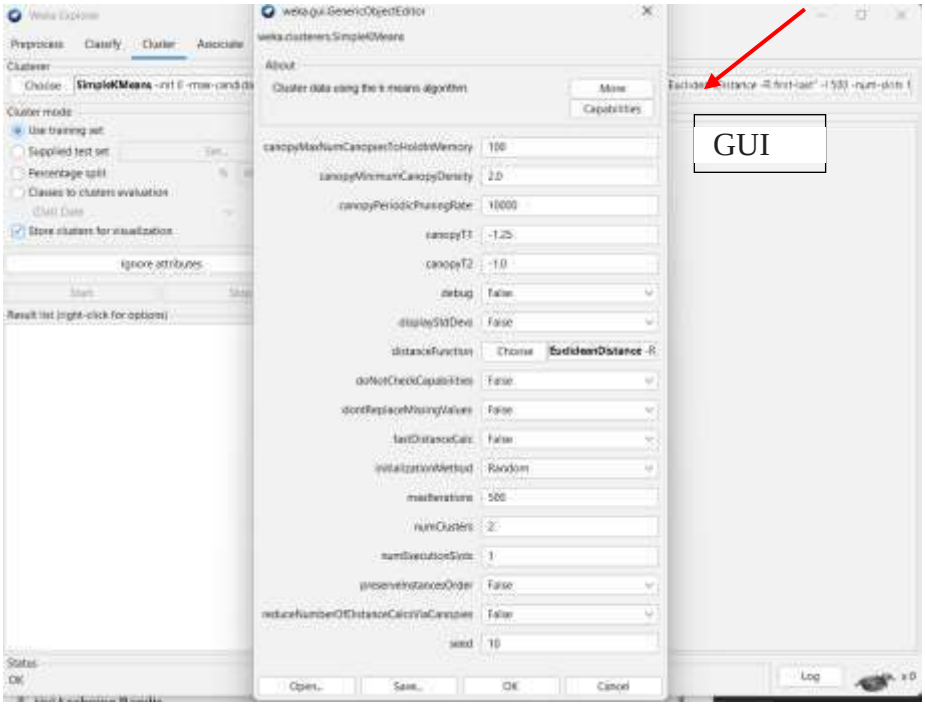


(ii) Choose the Cluster Scheme.

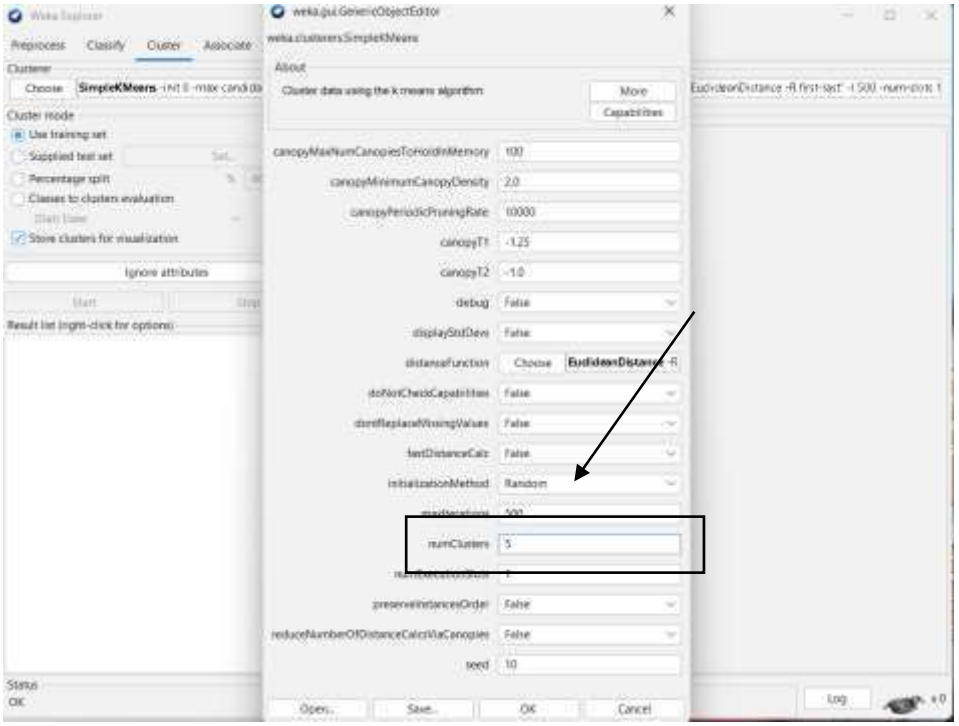
1. In the 'Clusterer' box click on 'Choose' button. In pull-down menu select WEKA Clusterers, and select the cluster scheme ***'SimpleKMeans'***. Some implementations of K-means only allow numerical values for attributes; therefore, we do not need to use a filter.

2. Once the clustering algorithm is chosen, right-click on the algorithm, ***"weak.gui.GenericObjectEditor"*** comes up to the screen.

.

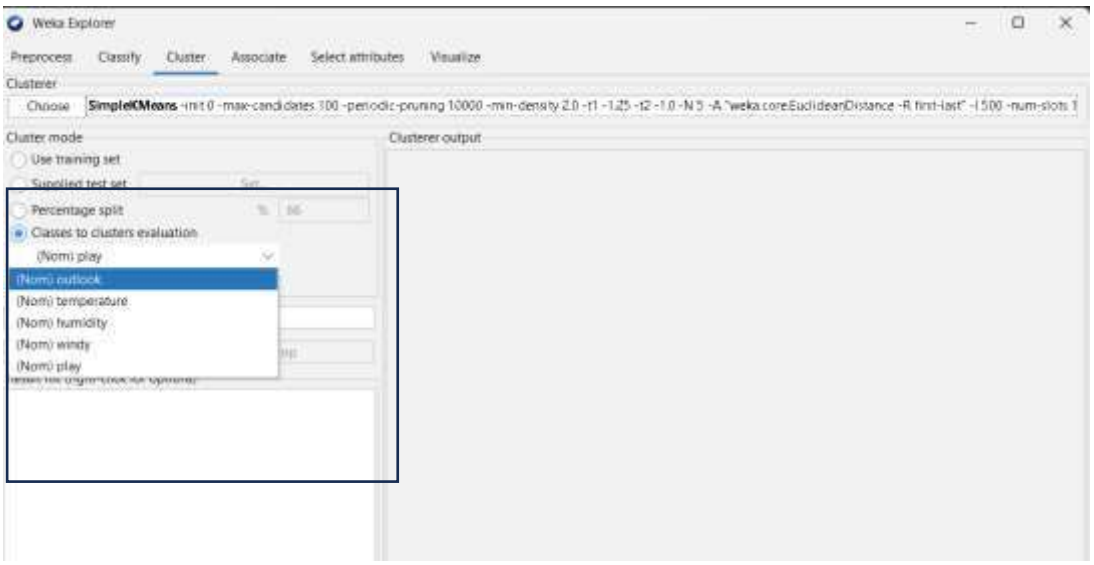


3. Set the value in ***"numClusters"*** box to 5 (Instead of default 2) because you have five clusters in your .arff file. Leave the value of 'seed' as is. The seed value is used in generating a random number, which is used for making the initial assignment of instances to clusters. Note that, in general, K-means is quite sensitive to how clusters are initially assigned. Thus, it is often necessary to try different values and evaluate the results.



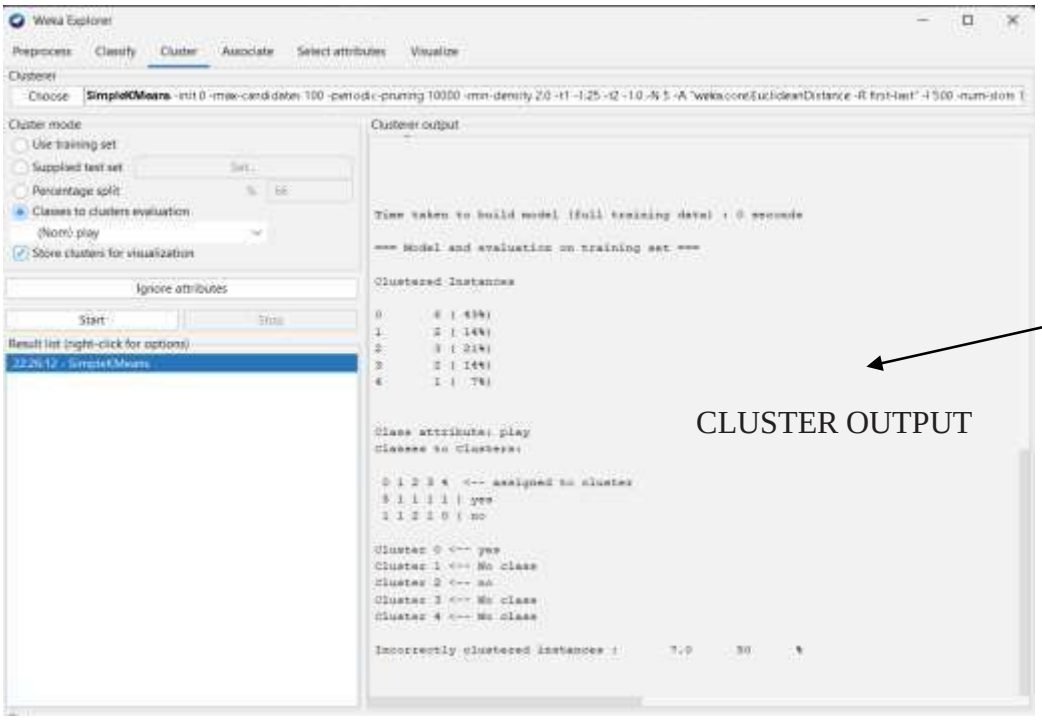
(iii) Setting the test options.

1. Before you run the clustering algorithm, you need to choose 'Cluster mode'.
2. Click on 'Classes to cluster evaluation' radio-button in 'Cluster mode' box and select 'Play' in the pull-down box below. It means that you will compare how well the chosen clusters match up with a pre-assigned class (''Play'') in the data.
3. Once the options have been specified, you can run the clustering algorithm. Click on the 'Start' button to execute the algorithm.



4. When training set is complete, the 'Cluster' output area on the right panel of 'Cluster'

window is filled with text describing the results of training and testing. A new entry appears in the ‘Result list’ box on the left of the result. These behave just like their classification counterparts.



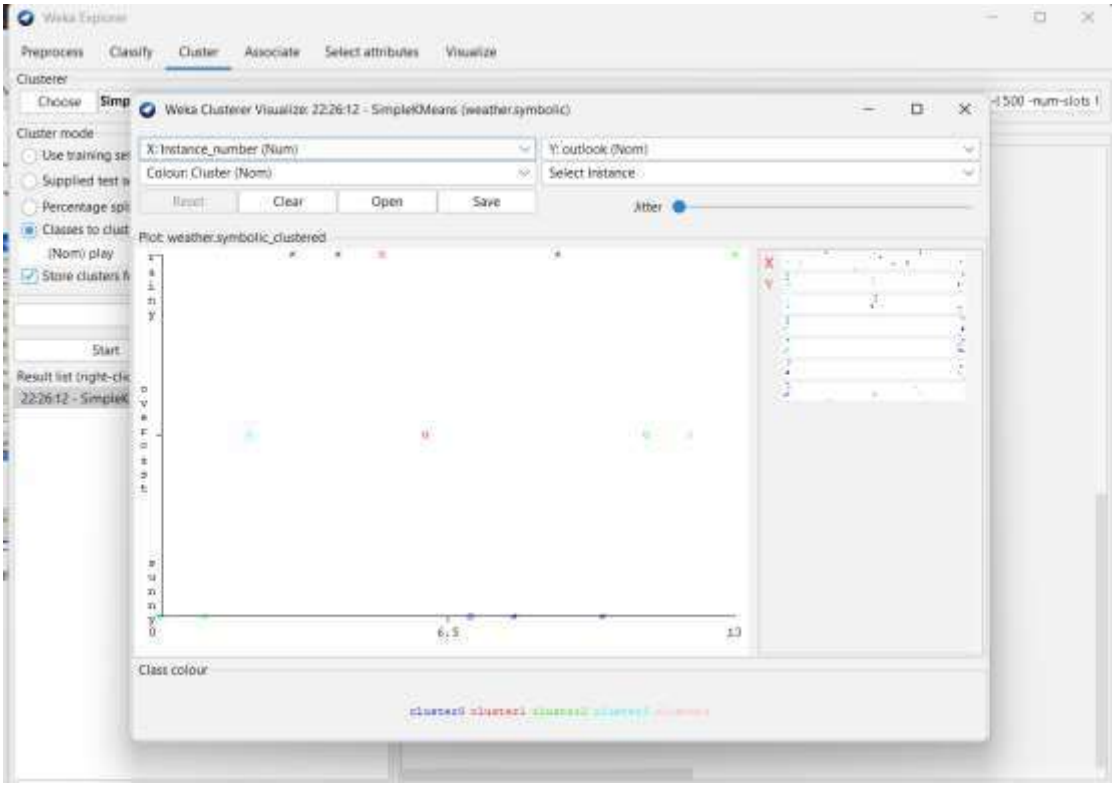
CLUSTER OUTPUT

(iv) Analysing Results.

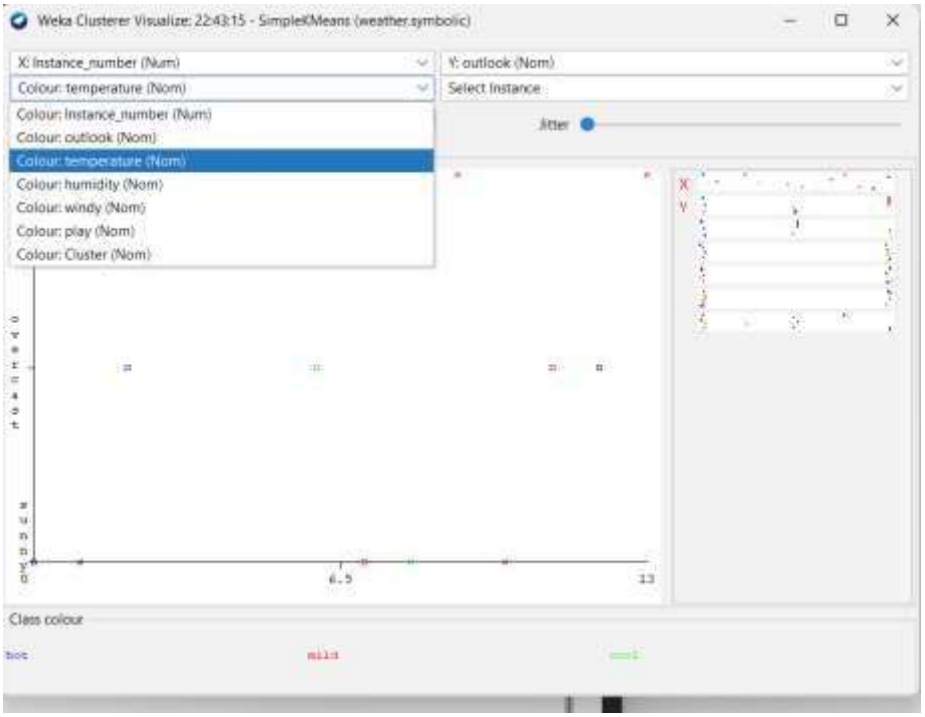
The clustering model shows the centroid of each cluster and statistics on the number and percentage of instances assigned to different clusters. Cluster centroids are the mean vectors foreach cluster; so, each dimension value and the centroid represent the mean value for that dimension in the cluster.

(v) Visualisation of Results.

1. Another way of representation of results of clustering is through visualization.
2. Right-click on the entry in the ‘Result list’ and select ‘Visualize cluster assignments’ in the pull-down window. This brings up the ‘Weka Clusterer Visualize’ window.

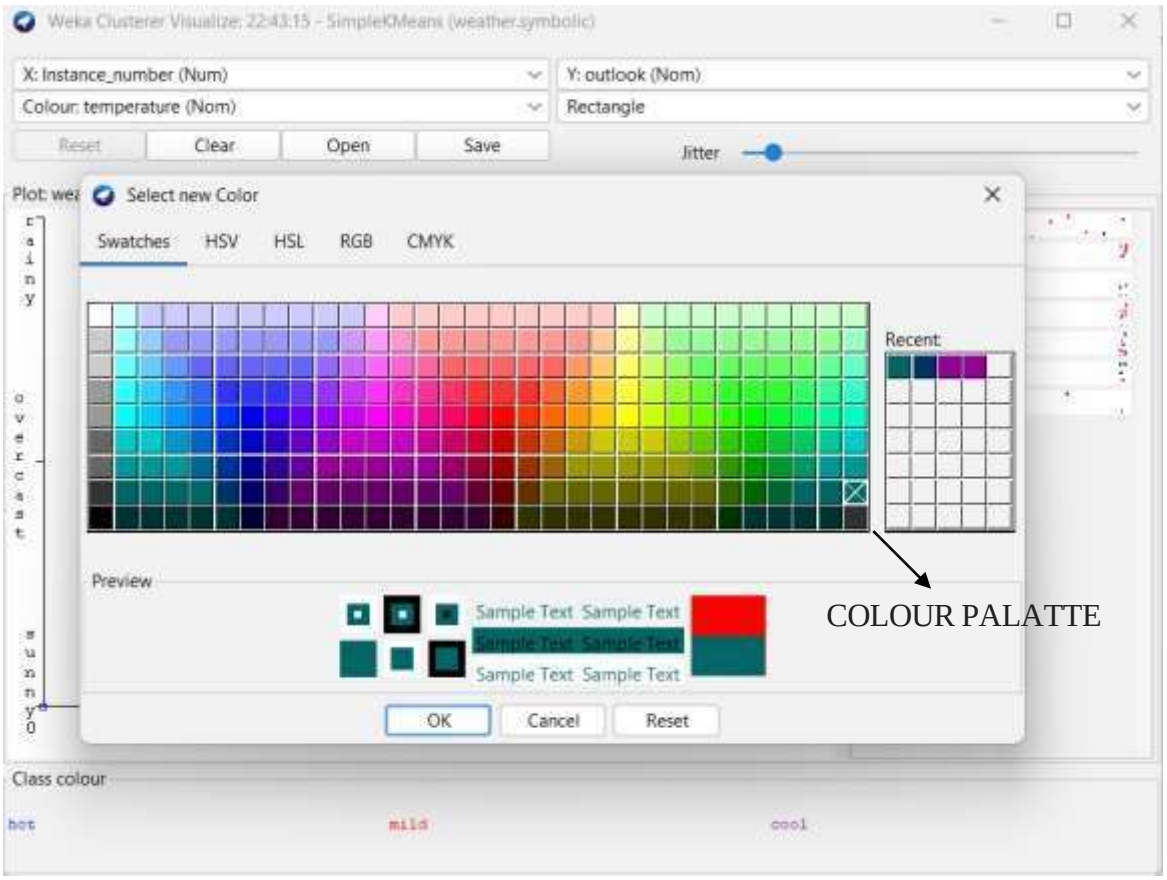


3. On the 'Weka Clusterer Visualize' window, beneath the X-axis selector there is a drop down list, ***'Colour'***, for choosing the color scheme. This allows you to choose the colour of points based on the attribute selected.

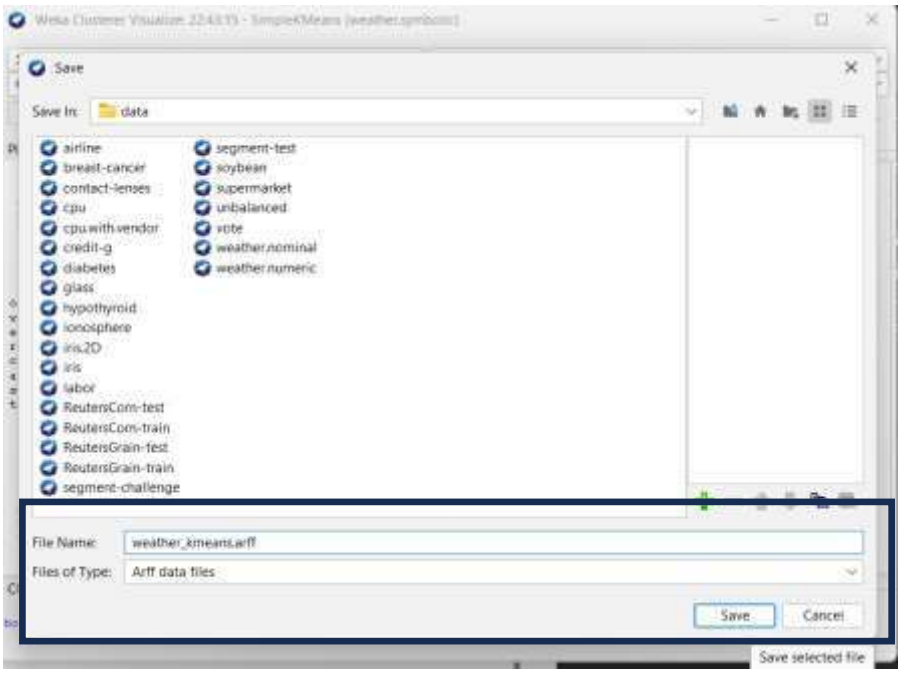


4.Below the plot area, there is a legend that describes what values the colours correspond to. In your example, Seven different colours represent Seven numbers (number of children). For better visibility you should change the colour of label '3'.

5. Left click on ‘3’ in the ‘Class colour’ box and select lighter color from the color palette.



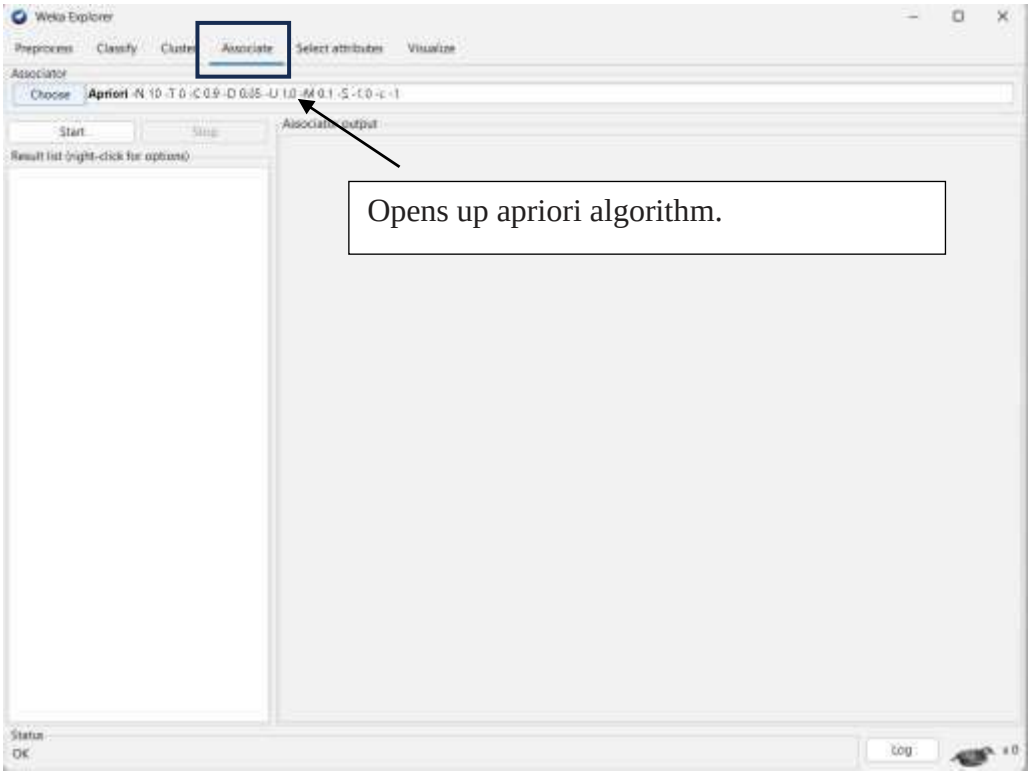
6. You may want to save the resulting data set, which included each instance along with its assigned cluster. To do so, click ‘Save’ button in the visualization window and save the result as the file ***“weather_kmeans.arff”.***



ASSOCIATION PANEL

(i) opening the file

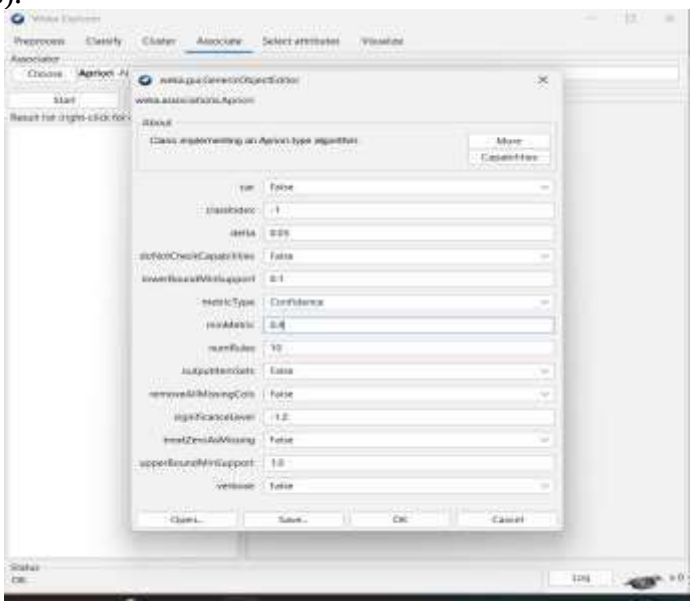
1. Click ‘Associate’ tab at the top of ‘WEKA Explorer’ window. It brings up interface for the Apriori algorithm.



2. The association rule scheme cannot handle numeric values; therefore, for this exercise you will use grocery store data from the ***“weather.arff”*** file where all values are nominal.Open ***“weather.arff”*** file.

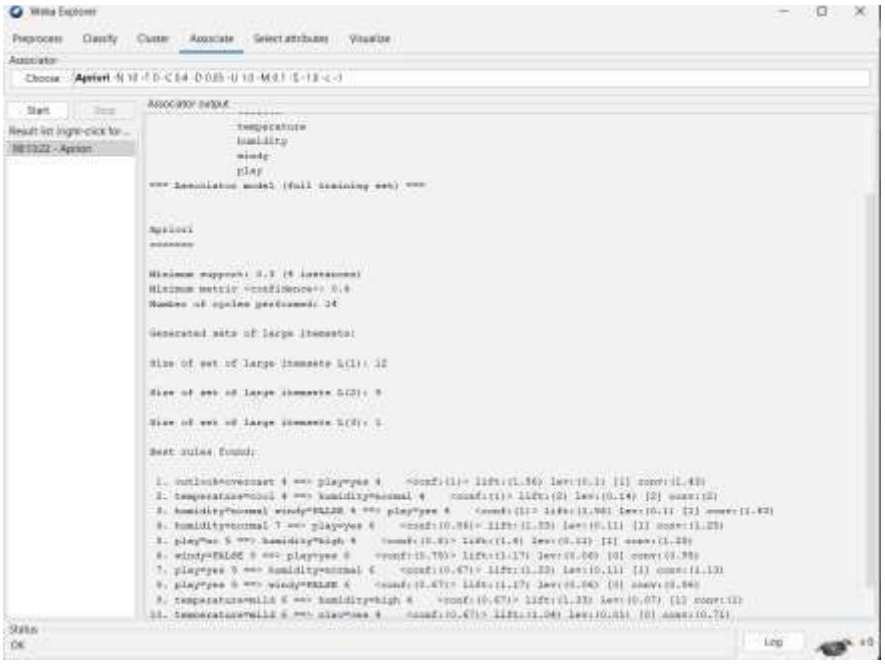
(ii) setting the test-options

1. Right-click on the ‘Associator’ box, ‘GenericObjectEditor’ appears on your screen. In the dialog box, change the value in ‘minMetric’ to 0.4 for confidence = 40%. Make sure that the default value of rules is set to 100. The upper bound for minimum support ‘upperBoundMinSupport’ should be set to 1.0 (100%) and ‘lowerBoundMinSupport’ to 0.1. Apriori in WEKA starts with the upper bound support and incrementally decreases support (by delta increments, which by default is set to 0.05 or 5%).



4.The algorithm halts when either the specified number of rules is generated, or the lower bound for minimum support is reached. The 'significanceLevel' testing option is only applicable in the case of confidence and is (-1.0) by default (not used).

5. Once the options have been specified, you can run Apriori algorithm. Click on the 'Start' button to execute the algorithm.



(iii) Analysing the results

```
=== Run information ===

Scheme:       weka.associations.Apriori -N 10 -T 0 -C 0.4 -D 0.05 -U 1.0 -M 0.1 -S -1.0 -c -1
Relation:     weather.symbolic
Instances:    14
Attributes:   5
              outlook
              temperature
              humidity
              windy
              play
```

The results for Apriori algorithm are the following:

-> First, the program generated the sets of large itemsets found for each support size considered. In this case five item sets of three items were found to have the required minimum support.

->By default, Apriori tries to generate ten rules. It begins with a minimum support of 100% of the data items and decreases this in steps of 5% until there are at least ten rules with the required minimum confidence, or until the support has reached a lower bound of 10% whichever occurs first. The minimum confidence is set 0.4 (40%).

->As you can see, the minimum support decreased to 0.3 (30%), before the required number of rules can be generated. Generation of the required number of rules involved a total of 14 iterations.

->The last part gives the association rules that are found. The number preceding = => symbol indicates the rule's support, that is, the number of items covered by its premise. Following the rule is the number of those items for which the rule's consequent holds as well. In the parentheses there is a confidence of the rule.

Ex No: 3 Date:	PLAN THE ARCHITECTURE FOR REAL TIME APPLICATION

AIM:

To study the architecture for real time application.

EXPERIMENT:

1. Data Ingestion:

-Design a component responsible for continuously ingesting real-time data from various sources. This data could be generated by sensors, user interactions, or other sources.

2. Data Preprocessing:

- Preprocess the incoming data to clean, transform, and prepare it for analysis. This may involve handling missing values, scaling features, and encoding categorical data.

3. Model Management:

- Implement a component for managing machine learning models built using WEKA. This component should be able to load and update models as needed.

4. Real-time Prediction:

- Utilize the models to make real-time predictions on incoming data. The predictions can be used for various purposes, such as anomaly detection, classification, or recommendation.

5. Feedback Loop:

- Implement a feedback loop to continuously update and retrain machine learning models as new data becomes available. This ensures that the models remain accurate and up-to-date.

6. Visualization and Monitoring:

- Create a dashboard or monitoring system to visualize the real-time data, model predictions, and other relevant metrics. This helps in tracking the performance of the application and identifying potential issues.

7. Deployment and Scalability:

- Deploy the application on a scalable infrastructure to handle varying data loads. Consider containerization (e.g., Docker) and orchestration (e.g., Kubernetes) for scalability and easy deployment.

8. Integration with WEKA:

- Develop connectors or APIs to interact with WEKA from your application. This allows you to utilize WEKA's data mining and machine learning capabilities.

9. Security and Privacy:

- Implement security measures to protect the data, models, and application from unauthorized access and ensure compliance with privacy regulations.

10. Error Handling and Logging:

- Implement robust error handling and logging mechanisms to capture and log errors, exceptions, and issues that may arise during real-time data processing and model predictions.

11. Continuous Testing and Validation:

- Implement automated testing and validation processes to ensure the correctness and reliability of the real-time application, including the accuracy of machine learning models.

12. Load Balancing and High Availability:

- Ensure high availability and load balancing to handle high volumes of incoming data and ensure the application remains responsive even during peak loads.

13. Documentation and Maintenance:

- Maintain thorough documentation of the application architecture, data sources, preprocessing steps, model details, and deployment procedures. Regularly update and maintain the application to address changes in data or requirements.

14. Compliance and Governance:

- Address compliance and governance requirements relevant to your application, especially if it deals with sensitive or regulated data.

15. Scalable Data Storage:

- Implement a scalable data storage solution to store historical data for training and retraining models.

16. User Interface (Optional):

- If the application has user-facing components, design a user interface that allows users to interact with the system, visualize results, and provide feedback.

Creating a real-time application with WEKA involves a blend of data engineering, machine learning, and software engineering. Be prepared for ongoing monitoring, maintenance, and updates to ensure the application's effectiveness and accuracy in real-time scenarios.

RESULT :

Thus the planning for architecture was executed successfully.

Ex No:4 Date:	WRITE A QUERY FOR SCHEMA DEFINITION

AIM:

To write a query for schema definition.

Schema Definition

Multidimensional schema is defined using Data Mining Query Language (DMQL). The two primitives, cube definition and dimension definition, can be used for defining the data warehouses and data marts.

Syntax for Cube Definition

define cube < cube_name > [< dimension-list > }: < measure_list >

Syntax for Dimension Definition

define dimension < dimension_name > as (< attribute_or_dimension_list >)

Star Schema Definition

The star schema that we have discussed can be defined using Data Mining Query Language (DMQL) as follows −

define cube sales star [time, item, branch, location]:

dollars sold = sum(sales in dollars), units sold = count(*)

define dimension time as (time key, day, day of week, month, quarter, year)
define dimension item as (item key, item name, brand, type, supplier type)
define dimension branch as (branch key, branch name, branch type)
define dimension location as (location key, street, city, province or state, country)

Snowflake Schema Definition

Snowflake schema can be defined using DMQL as follows −

define cube sales snowflake [time, item, branch, location]:

dollars sold = sum(sales in dollars), units sold = count(*)

define dimension time as (time key, day, day of week, month, quarter, year)
define dimension item as (item key, item name, brand, type, supplier (supplier key, supplier type))
define dimension branch as (branch key, branch name, branch type)
define dimension location as (location key, street, city (city key, city, province or state, country))

Fact Constellation Schema Definition

Fact constellation schema can be defined using DMQL as follows −

define cube sales [time, item, branch, location]:

dollars sold = sum(sales in dollars), units sold = count(*)

define dimension time as (time key, day, day of week, month, quarter, year)

define dimension item as (item key, item name, brand, type, supplier type)
define dimension branch as (branch key, branch name, branch type)
define dimension location as (location key, street, city, province or state,country)
define cube shipping [time, item, shipper, from location, to location]:

dollars cost = sum(cost in dollars), units shipped = count(*)

define dimension time as time in cube sales
define dimension item as item in cube sales
define dimension shipper as (shipper key, shipper name, location as location in cube sales, shipper type)
define dimension from location as location in cube sales
define dimension to location as location in cube sales

RESULT:

Thus the query for schema definition was executed successfully.

Ex No:5 Date:	DESIGN DATA WAREHOUSE FOR REAL TIME APPLICATION

AIM:

To design data warehouse for real time application.

STUDY EXPERIMENT:

Step 1: Needs Assessment and Requirements Gathering

Identify the specific requirements of the financial institutions, including consultancies, finance departments, banks, investment funds, government agencies, and ministries.
Conduct interviews with key stakeholders to understand their data and analytical needs.

Step 2: Data Source Identification

Identify the various sources of data, including transaction databases, credit bureaus, market data providers, external data sources, and more.
Assess the quality and reliability of data from each source.

Step 3: Data Integration

Set up Extract, Transform, Load (ETL) processes to extract data from various sources and transform it into a common format.
Ensure data consistency, accuracy, and timeliness during the ETL process.

Step 4: Data Modeling

Design a data model, which could be a star schema or snowflake schema, to structure the data effectively.
Create fact tables for transaction records and financial metrics and dimension tables for customer information, products, time, and other attributes.
Implement Slowly Changing Dimensions (SCD) for historical data and define hierarchies for reporting and analysis.

Step 5: Data Security

Implement robust security measures, including role-based access control, encryption, and data masking.
Ensure compliance with data protection regulations and audit trails for data access.

Step 6: Data Quality and Governance

Define data quality standards and governance policies to maintain data accuracy and compliance.
Establish data lineage to track data sources and transformations.

Step 7: Data Storage

Select a high-performance and scalable data storage solution, which can be a distributed data warehouse or a data lake.
Ensure data redundancy and fault tolerance for data availability.

Step 8: Data Processing

Utilize powerful analytical processing tools and technologies for complex analytics, such as SQL-based query engines and in-memory databases.
Implement distributed processing frameworks for handling large volumes of data.

Step 9: Metadata Management

Implement metadata management solutions to catalog and document the data warehouse, making it easy for users to discover and understand the data.

Step 10: Data Access and Reporting

Provide multiple methods for users to access and analyze the data, including SQL-based querying, business intelligence (BI) dashboards, and data visualization tools.
Implement data analytics and machine learning platforms for advanced analysis.

Step 11: Performance Optimization

Implement performance tuning and optimization techniques to ensure fast query responses, including indexing, caching, and query optimization.
Regularly monitor and fine-tune the system for performance improvements.

Step 12: Disaster Recovery and Backup

Develop a comprehensive disaster recovery plan to ensure data availability in case of unexpected events.
Regularly back up the data warehouse and test recovery procedures.

Step 13: Compliance and Regulation

Ensure that the data warehouse complies with relevant financial regulations, such as GDPR, Dodd-Frank, or Basel III, depending on the jurisdiction and type of institution.

Step 14: Scalability

Plan for future growth and ensure the data warehouse can scale horizontally and vertically to accommodate increasing data volumes.

Step 15: Monitoring and Alerts

Implement monitoring tools to track system performance, data quality, and security.
Configure alerts for anomalies or issues and establish incident response procedures.

Step 16: User Training

Provide training to end-users and administrators to ensure they can effectively use the data warehouse for analysis and reporting.

Step 17: Documentation

Maintain comprehensive documentation on the data warehouse's structure, ETL processes, and data governance policies.
Document data definitions, lineage, and metadata.

Step 18: Regular Maintenance

Regularly maintain and optimize the data warehouse to ensure it meets the evolving needs of the financial institutions.
Perform routine maintenance, updates, and performance monitoring.

Step 19: Continuous Improvement

Continuously improve the data warehouse based on user feedback and changing business requirements.
Stay updated with technological advancements and best practices in data warehousing.

Step 20: Collaboration

Encourage collaboration and knowledge sharing among different entities within the financial institutions for a holistic view of the data.
Foster communication and cooperation among departments for better data-driven decision-making.

RESULT:

Thus the designing the data warehouse for real time application was executed successfully.

Ex No : 6 Date:	ANALYSE THE DIMENSIONAL MODELING

AIM:

To analyse the dimensional modelling using weka tool.

STUDY EXPERIMENT:

i. Star Schema and Snowflake Schema:

Dimensional modeling typically involves creating one of two main schema types - the star schema and the snowflake schema. In a star schema, there is a central fact table connected to dimension tables, while in a snowflake schema, dimension tables are further normalized into sub-dimensions. The choice between these two depends on the specific data requirements and the balance between simplicity and data integrity.

ii. Fact Tables and Dimension Tables:

In dimensional modeling, fact tables store quantitative data (facts), such as sales revenue or quantity sold, while dimension tables store descriptive data, like product names, customer names, and time periods. This separation allows for efficient storage and querying of data.

iii. Hierarchies and Levels:

Dimension tables often include hierarchies with multiple levels. For instance, a time dimension might include levels such as year, quarter, month, and day. This hierarchy allows for easier aggregation and drilling down in reports and queries.

iv. Surrogate Keys:

Dimensional modeling often uses surrogate keys, which are auto-generated, meaningless keys used to uniquely identify records in dimension tables. This simplifies the process of updating and maintaining data in the dimension tables.

v. Data Integrity:

Dimensional modeling prioritizes performance but doesn't enforce strict data integrity constraints as heavily as traditional relational modeling. While this can lead to faster query performance, it may require additional attention to data quality and consistency.

vi. Query Performance:

Dimensional models are optimized for query performance, making them well-suited for business intelligence and reporting. Aggregations and joins are typically simpler and faster in this modeling approach.

vii. **Denormalization:**

Dimensional modeling often involves some level of denormalization, which can improve query performance but might lead to redundancy in data. This trade-off is made consciously to enhance reporting and analysis.

viii. **Conformed Dimensions**:

In multi-dimensional models that serve different business areas, it's important to have conformed dimensions. These are dimensions that are consistent across different parts of the data warehouse, ensuring data consistency and accuracy.

ix. **Historical Data**:

Handling historical data is essential in many business intelligence applications. Dimensional models often incorporate techniques to manage historical changes in dimensions over time, such as slowly changing dimensions (SCDs).

x. **Scalability**:

Dimensional models can be highly scalable and are well-suited for large datasets and complex reporting needs.

RESULT :

Thus the analysis of the dimensional modelling was executed successfully.

Ex No: 7 Date:	CASE STUDY USING OLAP

AIM:

Case study using OLAP tool.

Case Study:

Optimizing Retail Sales with OLAP

Company Background:

ABC Retailers is a leading chain of electronics stores with locations across the country. They sell a wide range of electronic products, including smartphones, laptops, cameras, and accessories.

Problem Statement:

ABC Retailers wants to improve its sales performance by analyzing their historical sales data. They aim to identify trends, patterns, and insights that can guide pricing strategies, inventory management, and marketing campaigns.

Solution:

The company decides to implement OLAP for record analysis to gain deeper insights into their sales data.

OLAP Implementation:

Data Collection: ABC Retailers gather detailed sales data, including product information, sales date, location, customer demographics, and transaction details, from their various stores.

Data Warehousing: They store this data in a central data warehouse, organized for OLAP processing. The data is structured in a star or snowflake schema, with a central fact table and dimension tables for products, time, location, and customers.

OLAP Cube Creation: Using OLAP tools, they create a multidimensional cube. This cube allows them to slice and dice the data across various dimensions, enabling more in-depth analysis. Key dimensions include:

Product Dimension: Product categories, brands, etc.
Time Dimension: Year, quarter, month, day, etc.
Location Dimension: Store location, region, city, etc.
Customer Dimension: Age group, gender, loyalty status, etc.
Measures: They define key performance indicators (KPIs) or measures for analysis, such as total sales revenue, profit margin, average transaction value, and customer retention rate.

Analysis:

Sales Trends: Using OLAP, they analyze sales trends over time, identifying seasonality and growth patterns.
Product Performance: They analyze which products are the best-sellers and identify underperforming products that may need adjustments.
Store Analysis: Store performance is assessed to allocate resources more effectively, such as inventory and marketing budgets.

Customer Segmentation: They segment customers based on demographics and purchase behavior, tailoring marketing efforts accordingly.
Pricing Strategies: Pricing strategies are optimized by analyzing price elasticity and customer response to discounts and promotions.
Visualization: Data visualizations, such as charts and graphs, are created to make the insights more accessible to stakeholders.

Decision-Making: The insights gained from OLAP analysis are used to make informed decisions, such as adjusting pricing strategies, optimizing inventory levels, and targeting specific customer segments with marketing campaigns.

Continuous Improvement: ABC Retailers continually updates and refines their OLAP cube as new data becomes available. This allows them to stay agile and adapt to changing market conditions.

BENEFITS:

- ✓ **Improved Sales Performance:** OLAP analysis helps ABC Retailers identify opportunities for revenue growth and cost savings.
- ✓ **Data-Driven Decision-Making:** Decision-makers have access to actionable insights for strategic planning.
- ✓ **Enhanced Customer Experience:** Tailored marketing efforts result in better customer engagement and retention.
- ✓ **Competitive Advantage:** ABC Retailers can respond quickly to market changes and outperform competitors.

RESULT:

Thus the case study using OLAP was executed successfully.

Ex No:8 Date :	CASE STUDY USING OLTP

AIM:

Case study using OLTP tool.

Case Study:

Optimizing Retail Sales with OLAP

Company Background:

ABC Retailers is a leading chain of electronics stores with locations across the country. They sell a wide range of electronic products, including smartphones, laptops, cameras, and accessories.

Problem Statement:

ABC Retailers wants to improve its sales performance by analyzing their historical sales data. They aim to identify trends, patterns, and insights that can guide pricing strategies, inventory management, and marketing campaigns.

Solution:

The company decides to implement OLAP for record analysis to gain deeper insights into their sales data.

OLAP Implementation:

Data Collection: ABC Retailers gather detailed sales data, including product information, sales date, location, customer demographics, and transaction details, from their various stores.

Data Warehousing: They store this data in a central data warehouse, organized for OLAP processing. The data is structured in a star or snowflake schema, with a central fact table and dimension tables for products, time, location, and customers.

OLAP Cube Creation: Using OLAP tools, they create a multidimensional cube. This cube allows them to slice and dice the data across various dimensions, enabling more in-depth analysis. Key dimensions include:

Product Dimension: Product categories, brands, etc.
Time Dimension: Year, quarter, month, day, etc.
Location Dimension: Store location, region, city, etc.
Customer Dimension: Age group, gender, loyalty status, etc.
Measures: They define key performance indicators (KPIs) or measures for analysis, such as total sales revenue, profit margin, average transaction value, and customer retention rate.

Analysis:

Sales Trends: Using OLAP, they analyze sales trends over time, identifying seasonality and growth patterns.
Product Performance: They analyze which products are the best-sellers and identify underperforming products that may need adjustments.
Store Analysis: Store performance is assessed to allocate resources more effectively, such as inventory and marketing budgets.

Customer Segmentation: They segment customers based on demographics and purchase behavior, tailoring marketing efforts accordingly.
Pricing Strategies: Pricing strategies are optimized by analyzing price elasticity and customer response to discounts and promotions.
Visualization: Data visualizations, such as charts and graphs, are created to make the insights more accessible to stakeholders.

Decision-Making: The insights gained from OLAP analysis are used to make informed decisions, such as adjusting pricing strategies, optimizing inventory levels, and targeting specific customer segments with marketing campaigns.

Continuous Improvement: ABC Retailers continually updates and refines their OLAP cube as new data becomes available. This allows them to stay agile and adapt to changing market conditions.

Benefits:

- ✓ **Improved Sales Performance:** OLAP analysis helps ABC Retailers identify opportunities for revenue growth and cost savings.
- ✓ **Data-Driven Decision-Making:** Decision-makers have access to actionable insights for strategic planning.
- ✓ **Enhanced Customer Experience:** Tailored marketing efforts result in better customer engagement and retention.
- ✓ **Competitive Advantage:** ABC Retailers can respond quickly to market changes and outperform competitors.

RESULT:

Thus the case study using OLTP tool was executed Successfully.

Ex No:9 Date:	IMPLEMENTATION OF WAREHOUSE TESTING

AIM :

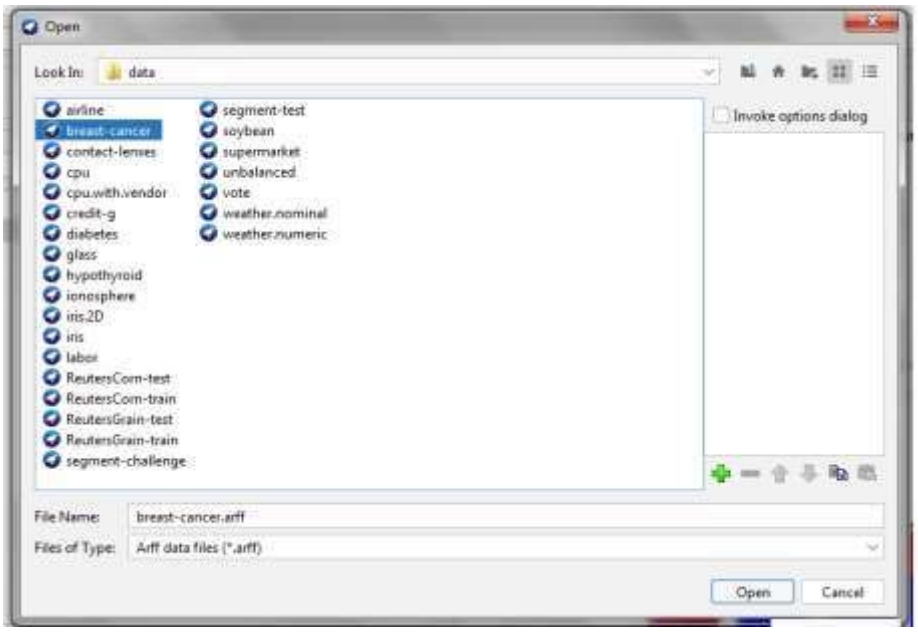
To implement warehouse testing using weka tool.

TESTING STEPS:

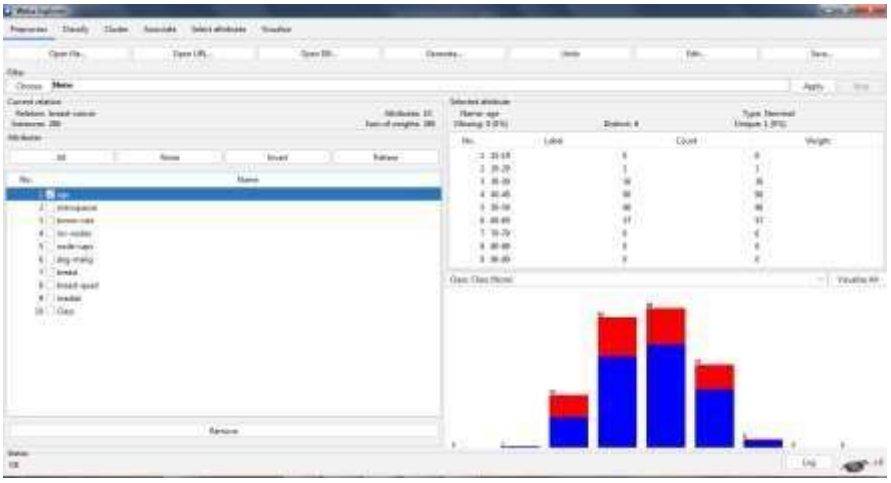
Weka tool is primarily designed for machine learning modeling and analysis. Therefore, it is not directly suitable for implementing warehouse testing. However, we can use Weka as a part of the testing process to run data mining models on the data and verify if the data quality is enough for accurate modeling.

Here are the general steps to implement warehouse testing in Weka tool:

1. Data Sampling: First, we need to select a sample of data that represents the entire warehouse. This sample data would be used for Weka analysis.

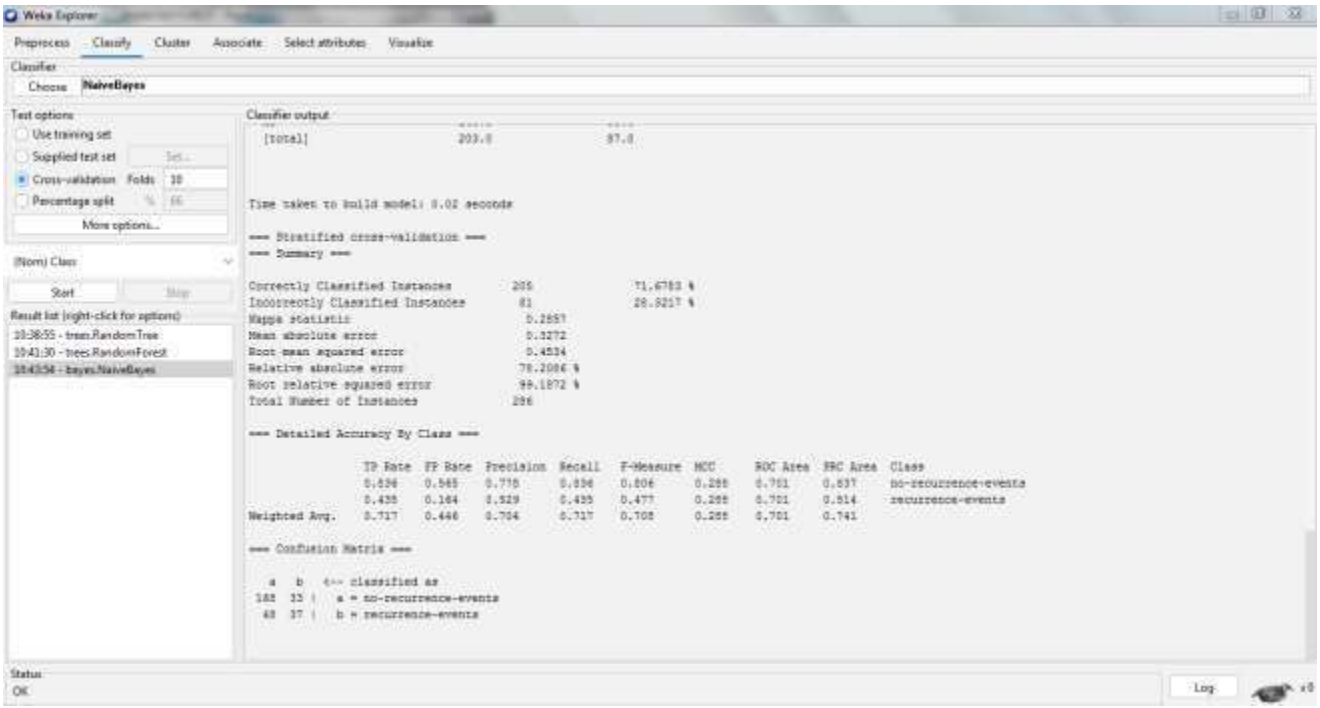


2.Data Preprocessing: Weka provides several tools for data preprocessing, including data normalization, discretization, and attribute selection. We can use these tools to prepare the data before running any machine learning algorithms.

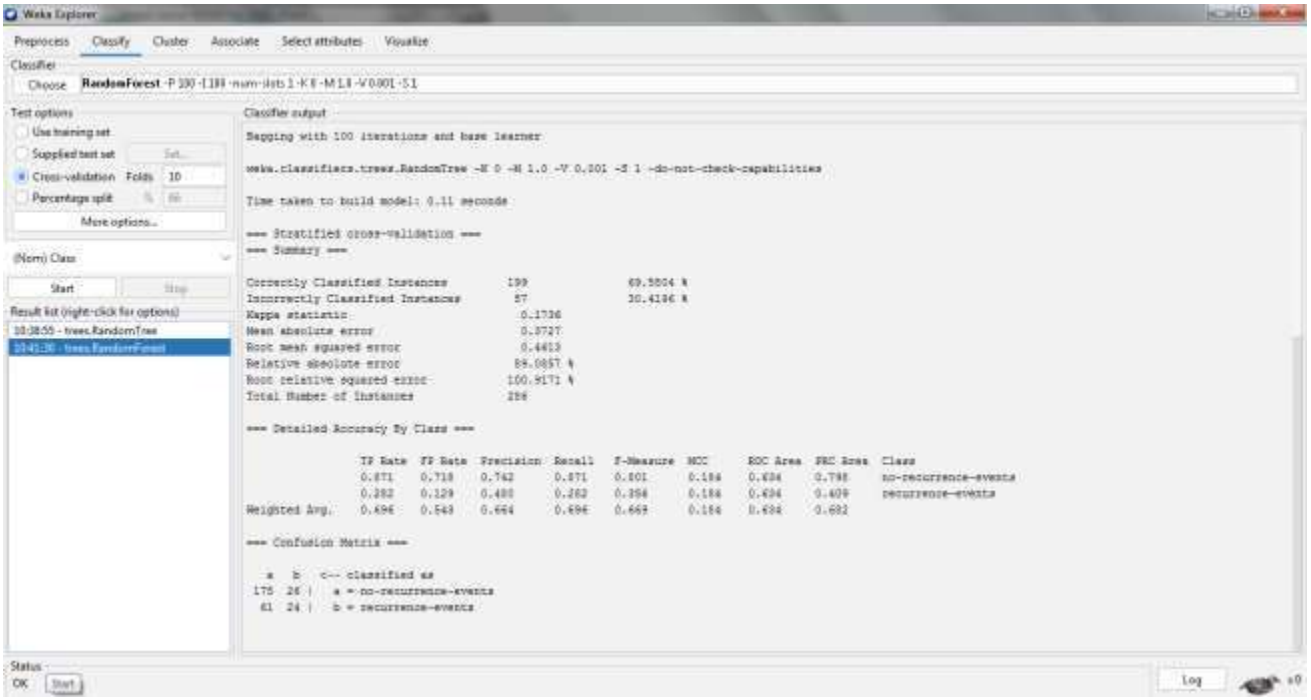


3. **Machine Learning Modelling:** Weka includes several classification, regression, and clustering algorithms that can be applied to the data for testing. We can select an algorithm based on the test requirements and run it on the cleaned and pre-processed sample data.

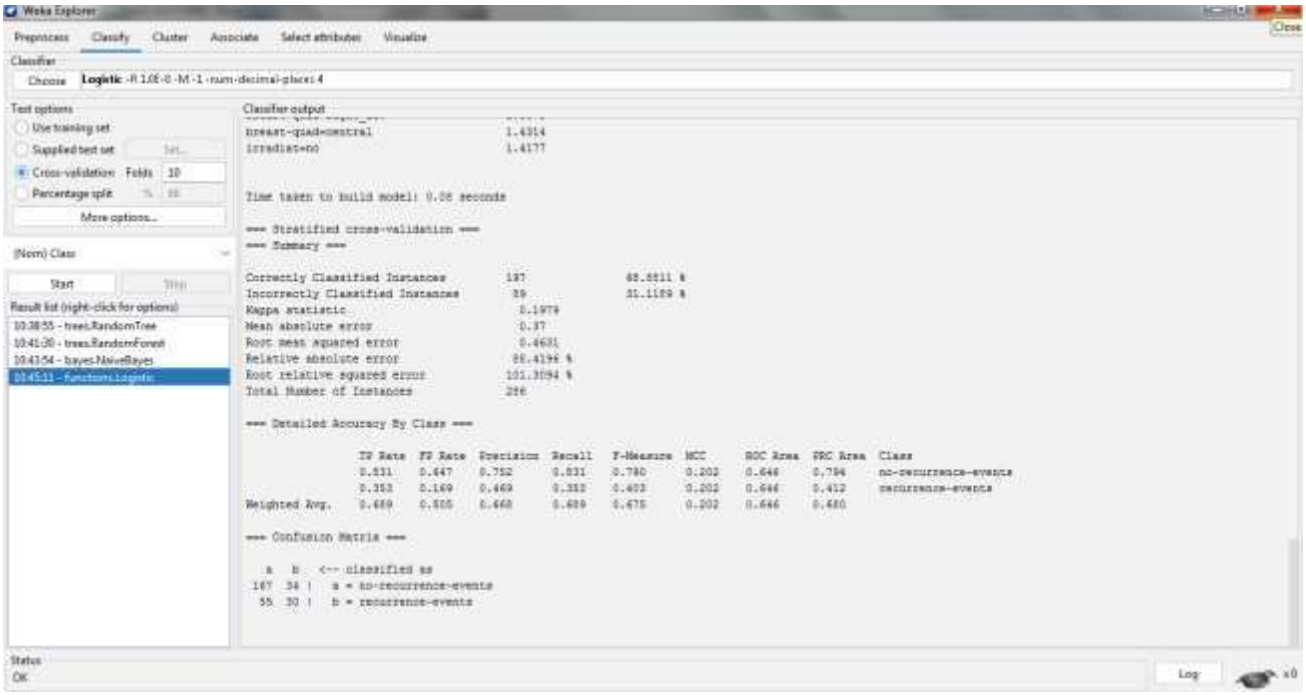
(i) Modeling the Dataset with Naïve bayes algorithm.



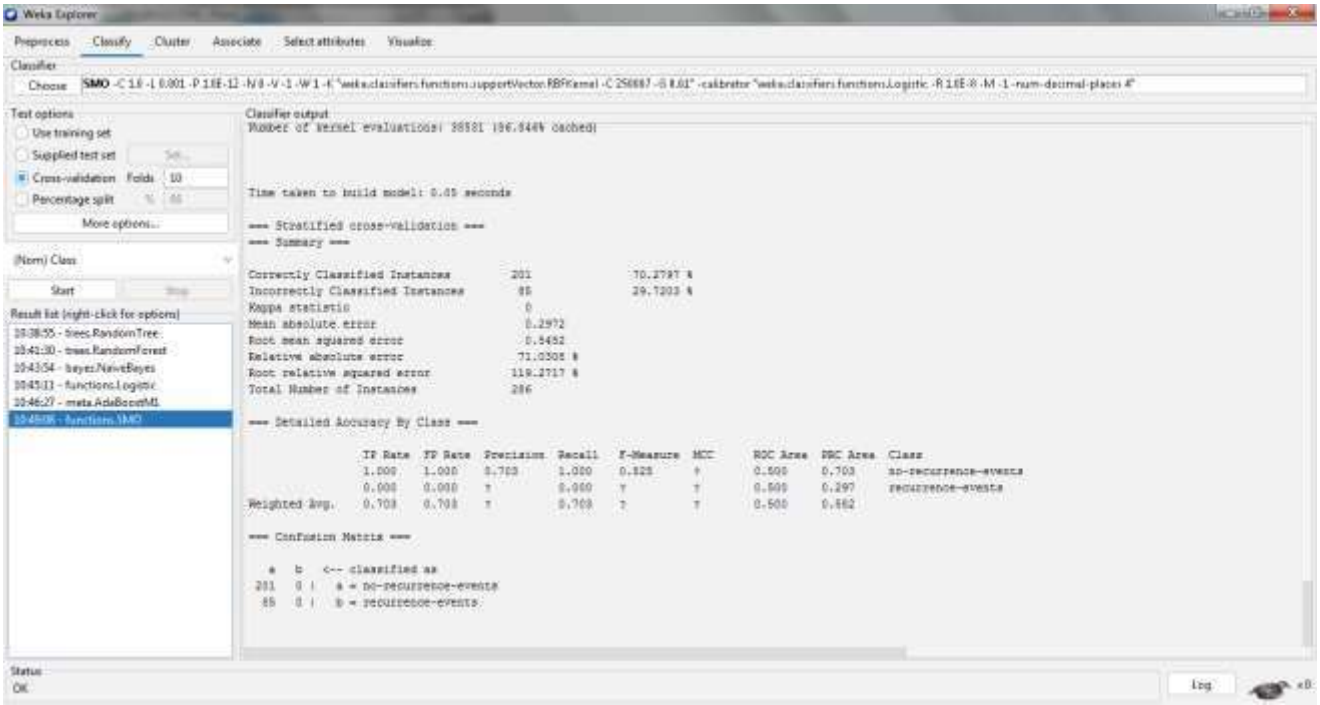
(ii) Modeling the dataset using Random forest algorithm.



(iii) Modeling the Dataset using Logistic Regression.



(iv) Modeling the Dataset using the SVM classification.



4.Model Evaluation: After running the machine learning algorithm, we need to evaluate the model's accuracy, sensitivity, specificity, and other metrics to assess the quality of the data used for testing.

Analysis: Among the algorithm it is predicted that SVM has the highest accuracy.

5. Repeating the Process: If the test results are not satisfactory, we need to repeat the entire process until the test results show that the data quality is good enough for accurate modeling.

Although Weka may not be directly used for testing a warehouse, it can be a valuable tool in the testing process, especially in the data quality assessment step.

RESULT:

Thus the testing the warehouse using the weka tool was executed successfully.