DATA WAREHOUSING AND DATA MINING

IT- 405

2023- 2024

PROJECT REPORT

Performance Evaluation of Classification Algorithms in WEKA using Diabetes Dataset

Submitted to:

Ms. Bhawna
Department of Information Technology

Submitted by:

Neeharika Singh
2K22/MSCMAT/25

Ritika Gupta
2K22/MSCMAT/54

Lakita
2K22/MSCMAT/57

ACKNOWLEDGEMENT

We would like to express our deepest gratitude to our project supervisor, Ms. Bhawna, for her invaluable guidance, continuous support, and insightful feedback throughout the project titled "Performance Evaluation of Classification Algorithms in WEKA using Diabetes Dataset". Her expertise and mentorship were instrumental in shaping the direction and quality of this project.

We extend our thanks to the authors of the Diabetes dataset and the developers of WEKA for making their resources available, enabling us to conduct this study and evaluate various classification algorithms.

This project would not have been possible without the collective contributions, support, and encouragement of all those mentioned above.

Sincerely,

Neeharika Singh (2K22/MSCMAT/25)
Ritika Gupta (2K22/MSCMAT/54)
Lakita (2K22/MSCMAT/57)

INDEX

1. AIM

To evaluate performance of different Classification Algorithms in WEKA using Diabetes Dataset.

2. THEORY

2.1. Dataset

The dataset considered is a diabetes dataset. There are 8 independent attributes (all numeric) and 1 dependent attribute (class) which determines whether an individual is diabetic or not. Total number of instances is 769. The 9 attributes are defined as follows:

1. Number of times pregnant
2. Plasma glucose concentration a 2 hours in an oral glucose tolerance test
3. Diastolic blood pressure (mm Hg)
4. Triceps skin fold thickness (mm)
5. 2-Hour serum insulin (mu U/ml)
6. Body mass index (weight in kg/(height in m)^2)
7. Diabetes pedigree function
8. Age (years)
9. Class variable (0 or 1)

Class Distribution: (class value 1 is interpreted as "tested positive for diabetes")

2.2. Classification Algorithms

2.2.1. Logistic Regression

Logistic regression is a supervised machine learning algorithm. It models the relationship between a binary dependent variable and one or more independent variables by estimating the probability of the dependent variable taking a particular value (usually 0 or 1). It's widely used in machine learning for tasks like spam detection, medical diagnosis, and more. This probability is modeled using the logistic function (also called the sigmoid function).

2.2.2. J48

The C4.5 algorithm/J48 algorithm is used in machine learning and data mining for decision tree classification. J48 constructs a decision tree by recursively splitting the data into subsets based on the most informative attributes, and it is capable of handling both categorical and numerical data. The resulting tree can be used for classification and can help make decisions based on the input features.

2.2.3. Naïve Bayes

Naïve Bayes is a supervised machine learning algorithm used for classification and text categorization tasks. It is based on Bayes' theorem and assumes that the features used to make predictions are conditionally independent, which is a "naive" assumption because it's often not entirely accurate in real-world data.

2.2.4. Stochastic Gradient Descent

SGD is an optimization algorithm commonly used in machine learning for training models, particularly in the context of large datasets. It's a variant of gradient descent that updates the model's parameters iteratively by considering only a small, random subset of the training data at each step, as opposed to the entire dataset in traditional gradient descent.

2.2.5. Random Forest

Random Forest is an ensemble learning algorithm used for both classification and regression tasks. It is an extension of decision trees and combines multiple decision trees to create a more robust and accurate model. It operates on the principles of aggregating the predictions of multiple decision trees to make more accurate predictions.

2.3. Classification Metrics

2.3.1. Kappa Score

Cohen's Kappa/Kappa is a statistical measure used to assess the level of agreement between two raters or evaluators for categorical data, taking into account the possibility of agreement occurring by chance.

$$Kappa\ Score\ =\ \frac{Agree - Chance\_Agree}{1 - Chance\_Agree}$$

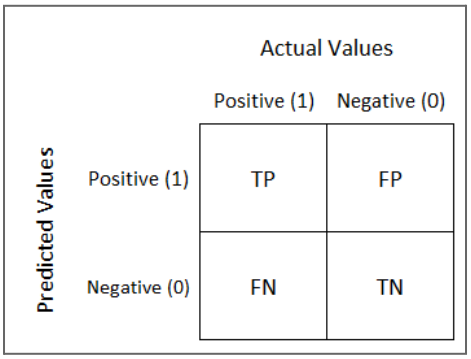
2.3.2. Accuracy

Accuracy is the percentage of correct classifications that a trained model achieves.

$$Accuracy\ =\ 100 \times \frac{Number\ of\ correct\ predictions}{Total\ number\ of\ predictions}$$

2.3.3. Confusion Matrix

A confusion matrix is a tabular representation used to evaluate the performance of a classification model on a dataset, especially in binary classification tasks. It breaks down the model's predictions into four categories based on the actual and predicted class labels. These categories are:
True Positives (TP): The instances correctly predicted as positive.
True Negatives (TN): The instances correctly predicted as negative.
False Positives (FP): The instances incorrectly predicted as positive.
False Negatives (FN): The instances incorrectly predicted as negative.



2.3.4. Precision

Precision measures how good the model is at correctly identifying the positive class. It is a metric that quantifies the number of correct positive predictions made.

$$Precision = TP / (TP + FP)$$

2.3.5. Recall

Recall measures the ability of a model to identify and correctly classify all actual positive instances within a dataset.

$$Recall = TP / (TP + FN)$$

2.3.6. F1 Score

The F1 score is the harmonic mean of precision and recall. Its range is from 0 to 1. If it is 1 this indicates perfect precision and recall. If it is 0 this means that either the precision or the recall is 0.

$$F1\ Score = 2\ \frac{Precision \times Recall}{Precision + Recall}$$

2.3.7. Matthew's Correlation Coefficient

MCC is a metric used to evaluate the performance of a binary classification model, particularly when dealing with imbalanced datasets. It takes into account all four possible outcomes in a binary classification task: TP, TN, FP and FN. Just like any correlation coefficient, its value will lie between -1 and +1. A value of +1 would indicate a perfect model.

$$MCC = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP)\ (TP + FN)\ (TN + FP)\ (TN + FN)}}$$

2.3.8. ROC Area

It represents the area under the Receiver Operating Characteristic (ROC) curve. The ROC curve is a plot of the performance of the model (a plot of the true positive rate and the false positive rate) at all classification thresholds. ROC Area values range from 0 to 1, where a higher value indicates better model performance.

2.3.9. PRC Area

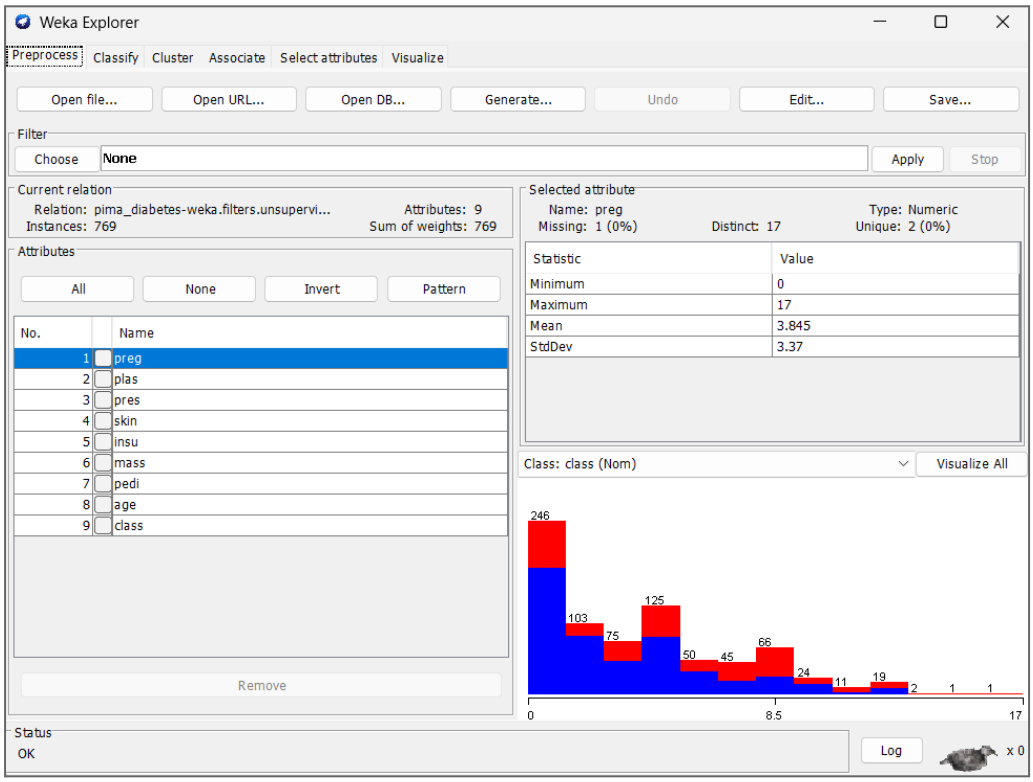
It quantifies the overall performance of a binary classification model by calculating the area under the Precision-Recall curve. This curve visually represents the trade-off between precision and recall as the classification threshold varies.

3. PROCEDURE

3.1. Data Preprocessing

3.1.1. Loading the Dataset in WEKA:

The Diabetes dataset is imported in ARFF format.



Viewer

Relation: pima_diabetes-weka.filters.unsupervised.attribute.RemoveUseless-M99.0

No.	1: preg Numeric	2: plas Numeric	3: pres Numeric	4: skin Numeric	5: insu Numeric	6: mass Numeric	7: pedi Numeric	8: age Numeric	9: class Nominal
1	6.0	148.0	72.0	35.0	0.0	33.6	0.627	50.0	tested...
2	1.0	85.0	66.0	29.0	0.0	26.6	0.351	31.0	tested...
3	8.0	183.0	64.0	0.0	0.0	23.3	0.672	32.0	tested...
4	1.0	89.0	66.0	23.0	94.0	28.1	0.167	21.0	tested...
5	0.0	137.0	40.0	35.0	168.0	43.1	2.288	33.0	tested...
6	5.0	116.0	74.0	0.0	0.0	25.6	0.201	30.0	tested...
7	3.0	78.0	50.0	32.0	88.0	31.0	0.248	26.0	tested...
8	10.0	115.0	0.0	0.0	0.0	35.3	0.134	29.0	tested...
9	2.0	197.0	70.0	45.0	543.0	30.5	0.158	53.0	tested...
10	8.0	125.0	96.0	0.0	0.0	0.0	0.232	54.0	tested...
11	4.0	110.0	92.0	0.0	0.0	37.6	0.191	30.0	tested...
12	10.0	168.0	74.0	0.0	0.0	38.0	0.537	34.0	tested...
13	10.0	139.0	80.0	0.0	0.0	27.1	1.441	57.0	tested...
14	1.0	189.0	60.0	23.0	846.0	30.1	0.398	59.0	tested...
15	5.0	166.0	72.0	19.0	175.0	25.8	0.587	51.0	tested...
16	7.0	100.0	0.0	0.0	0.0	30.0	0.484	32.0	tested...
17	0.0	118.0	84.0	47.0	230.0	45.8	0.551	31.0	tested...
18	7.0	107.0	74.0	0.0	0.0	29.6	0.254	31.0	tested...
19	1.0	103.0	30.0	38.0	83.0	43.3	0.183	33.0	tested...
20	1.0	115.0	70.0	30.0	96.0	34.6	0.529	32.0	tested...
21	3.0	126.0	88.0	41.0	235.0	39.3	0.704	27.0	tested...
22	8.0	99.0	84.0	0.0	0.0	35.4	0.388	50.0	tested...
23	7.0	196.0	90.0	0.0	0.0	39.8	0.451	41.0	tested...
24	9.0	119.0	80.0	35.0	0.0	29.0	0.263	29.0	tested...
25	11.0	143.0	94.0	33.0	146.0	36.6	0.254	51.0	tested...
26	10.0	125.0	70.0	26.0	115.0	31.1	0.205	41.0	tested...
27	7.0	147.0	76.0	0.0	0.0	39.4	0.257	43.0	tested...
28	1.0	97.0	66.0	15.0	140.0	23.2	0.487	22.0	tested...
29	13.0	145.0	82.0	19.0	110.0	22.2	0.245	57.0	tested...
30	5.0	117.0	92.0	0.0	0.0	34.1	0.337	38.0	tested...

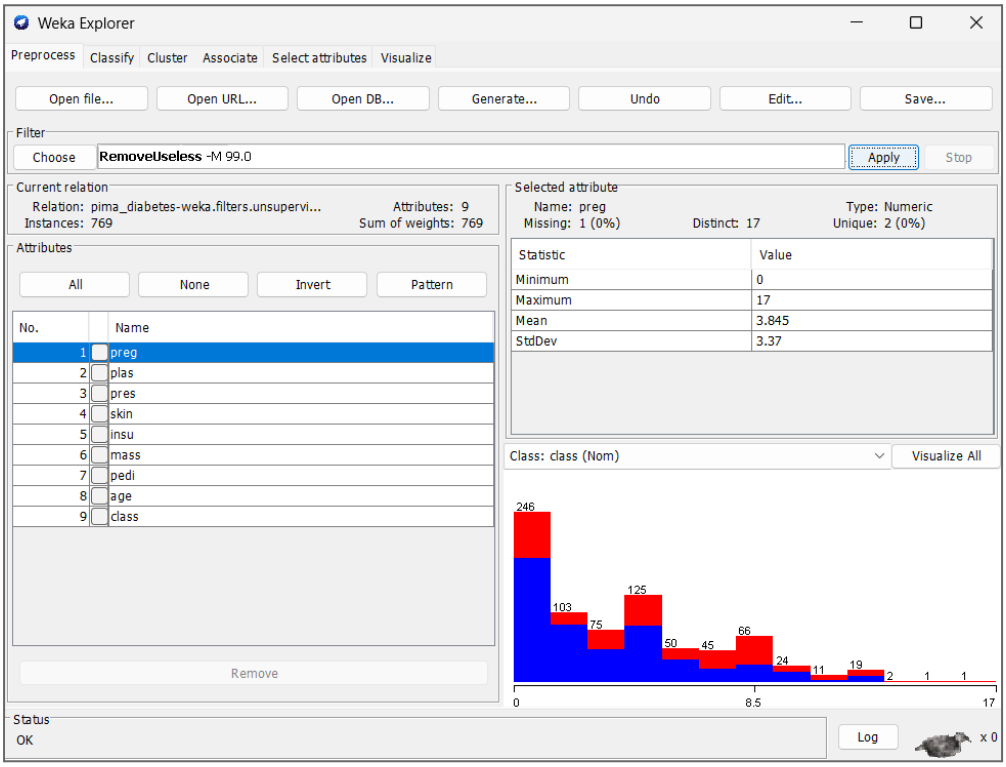
Add instance | Undo | OK | Cancel

3.1.2. Removing useless attributes from the dataset:

Filter used: RemoveUseless

Filter path: Filters > Unsupervised > Attribute > RemoveUseless

This filter removes attributes that do not vary at all or that vary too much.

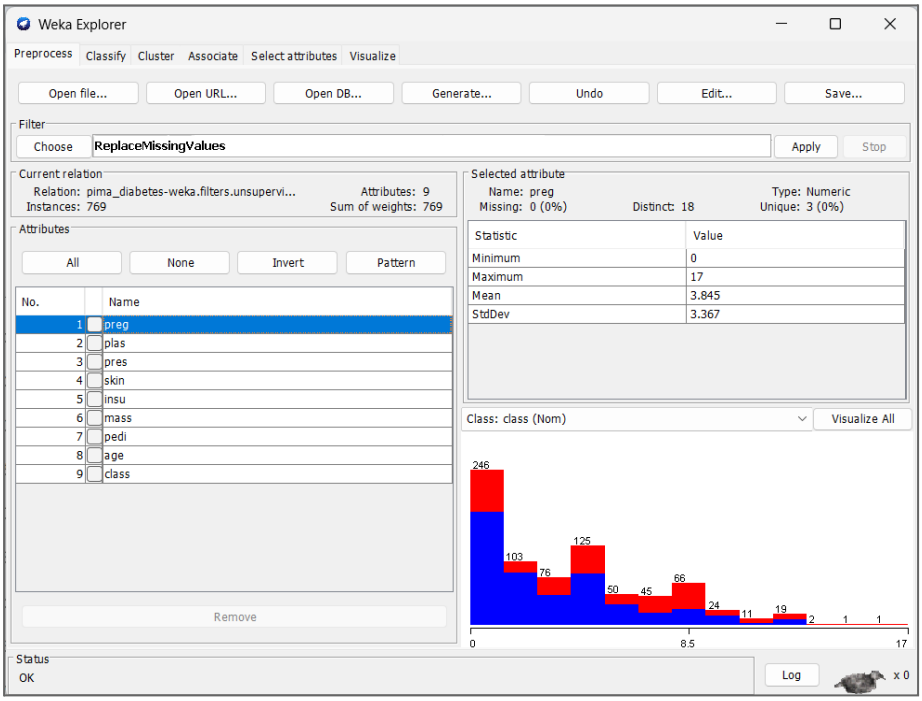


3.1.3. Dealing with missing data:

Filter used: ReplaceMissingValues

Filter path: Filters > Unsupervised > Attribute > RemoveMissingValues

Replaces all missing values for nominal and numeric attributes in a dataset with the modes and means from the training data.

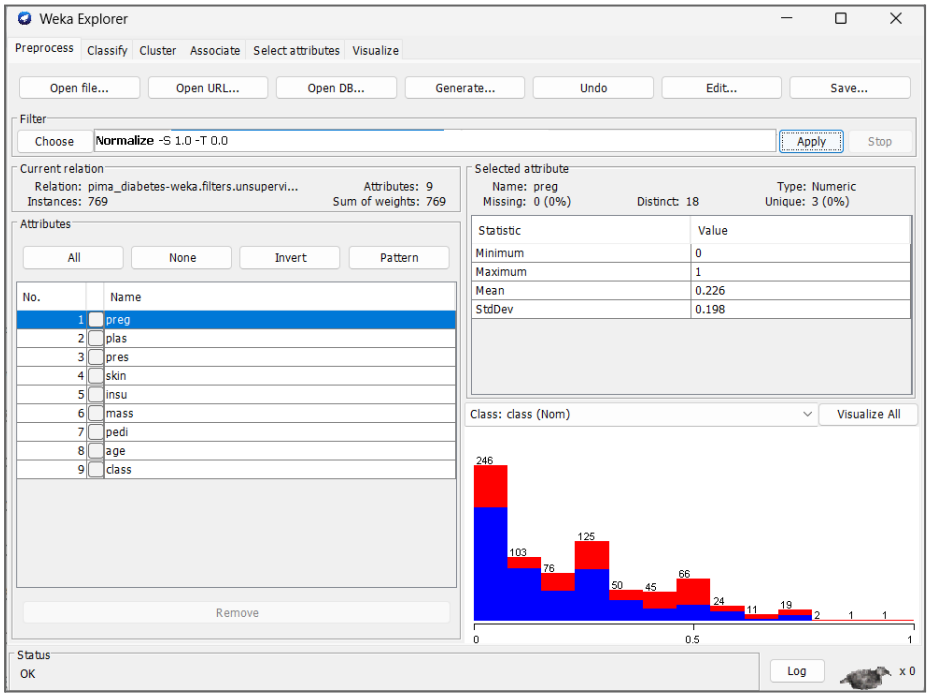


3.1.4. Normalizing the data:

Filter used: Normalize

Filter path: Filters > Unsupervised > Attribute > Normalize

Normalizes all numeric values in the given dataset, i.e., scaling the data so that all values fit in the range of 0 to 1 (apart from the class attribute).

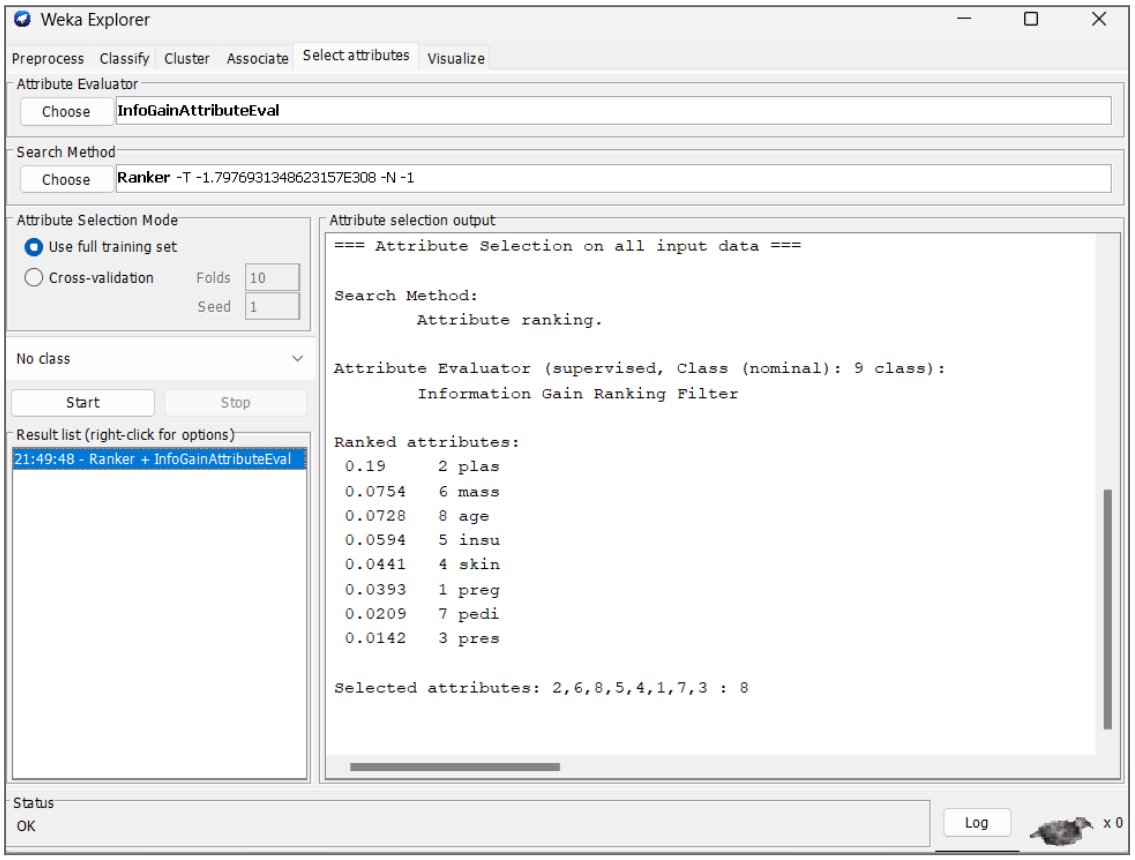


3.1.5. Attribute Selection:

In a dataset, certain attributes are not required and are also responsible for the skewed nature and problems in classification. After obtaining the ranks of the attributes, the ones with very low ranks are removed. Since here all attributes are somehow contributing to the classification, we will not remove any attribute.

Attribute Evaluator used: InfoGainAttributeEval

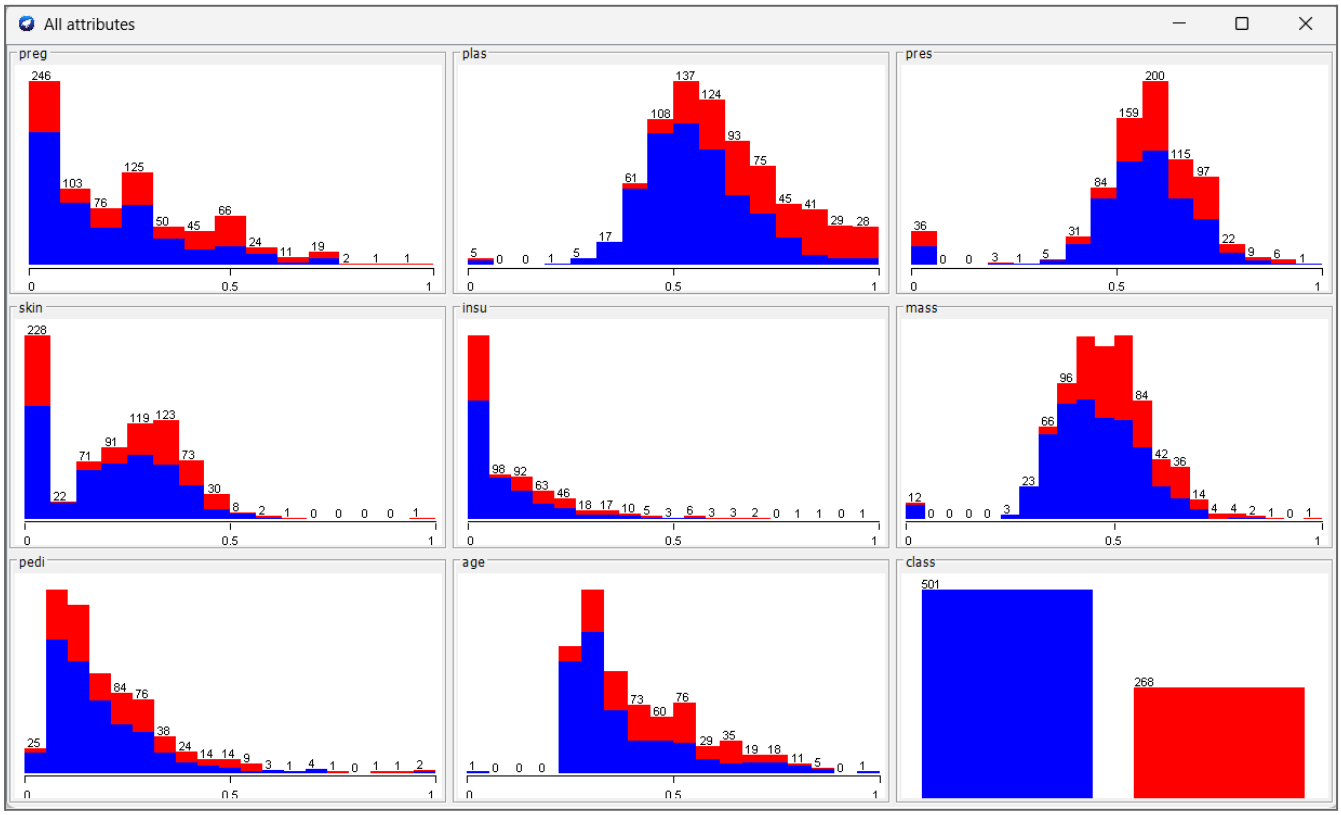
Evaluates the worth of an attribute by measuring the information gain with respect to the class.



3.2. Visualization

3.2.1. Visualization of attributes/features:

All the attributes are represented separately along with the values.



3.2.2. Visualization of the relation among attributes:

The relation among attributes helps in finding the connections among various attributes and how one attribute contributes towards the others.

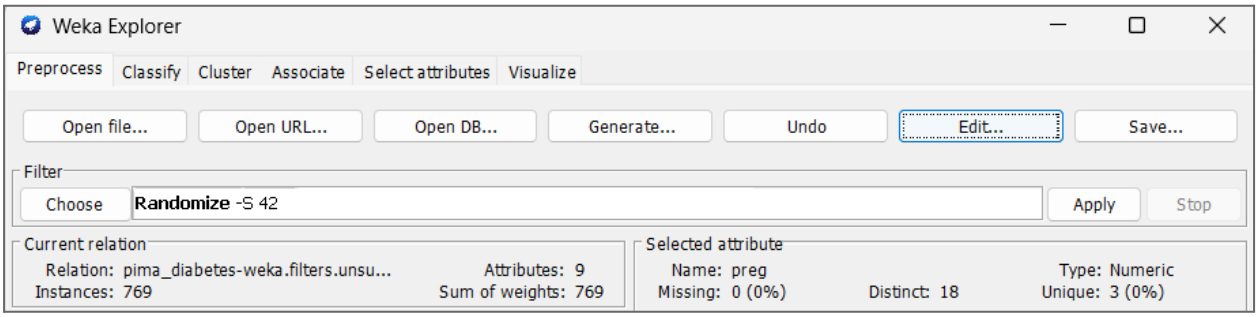
3.3. Splitting the Data

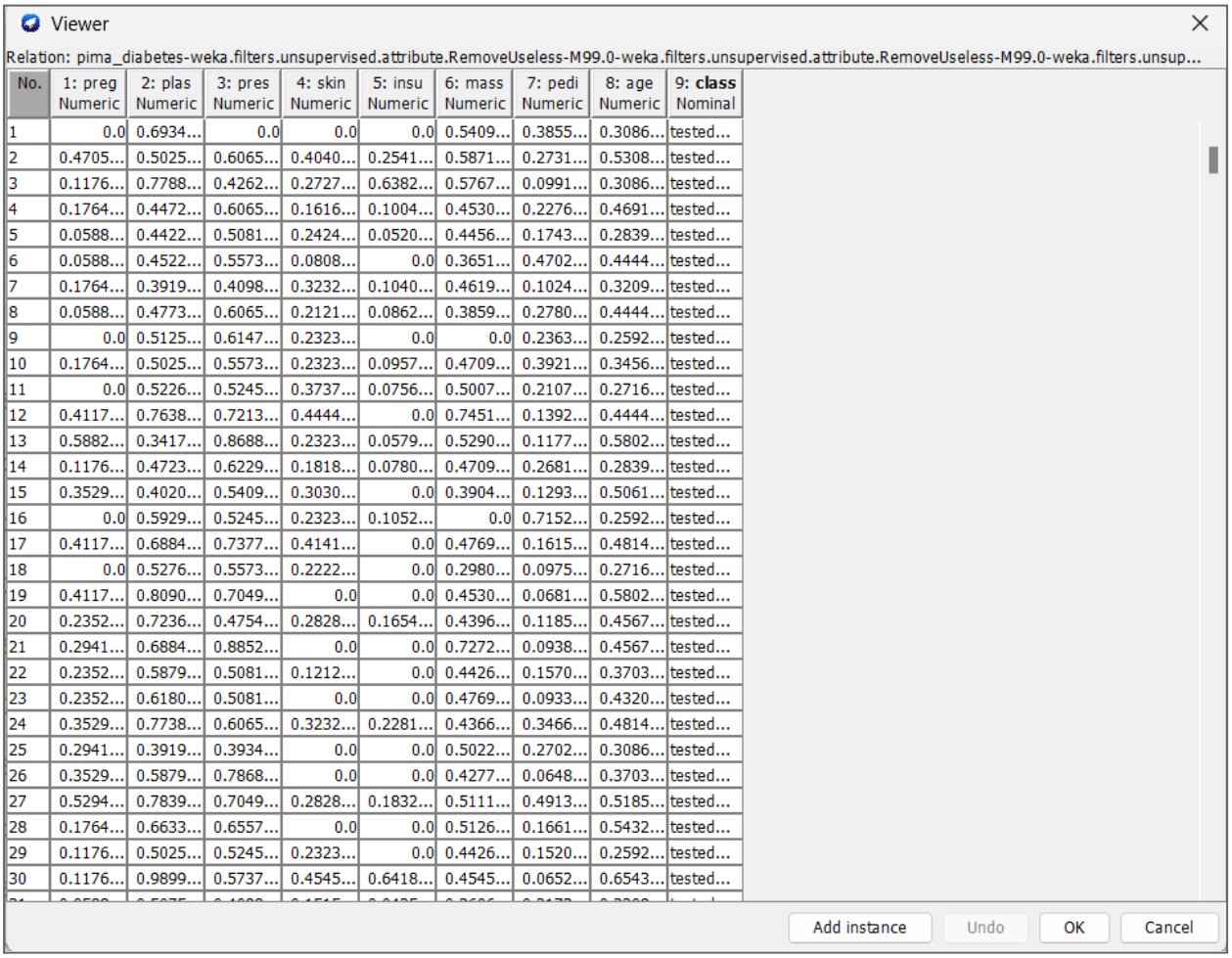
3.3.1. Randomize the dataset:

Randomizing data when splitting prevents bias, overfitting, and ensures generalization for unbiased model evaluation.

Filter used: Randomize

Filter path: Filters > Unsupervised > Instance > Randomize





No.	1: preg Numeric	2: plas Numeric	3: pres Numeric	4: skin Numeric	5: insu Numeric	6: mass Numeric	7: pedi Numeric	8: age Numeric	9: class Nominal
1	0.0	0.6934...	0.0	0.0	0.0	0.5409...	0.3855...	0.3086...	tested...
2	0.4705...	0.5025...	0.6065...	0.4040...	0.2541...	0.5871...	0.2731...	0.5308...	tested...
3	0.1176...	0.7788...	0.4262...	0.2727...	0.6382...	0.5767...	0.0991...	0.3086...	tested...
4	0.1764...	0.4472...	0.6065...	0.1616...	0.1004...	0.4530...	0.2276...	0.4691...	tested...
5	0.0588...	0.4422...	0.5081...	0.2424...	0.0520...	0.4456...	0.1743...	0.2839...	tested...
6	0.0588...	0.4522...	0.5573...	0.0808...	0.0	0.3651...	0.4702...	0.4444...	tested...
7	0.1764...	0.3919...	0.4098...	0.3232...	0.1040...	0.4619...	0.1024...	0.3209...	tested...
8	0.0588...	0.4773...	0.6065...	0.2121...	0.0862...	0.3859...	0.2780...	0.4444...	tested...
9	0.0	0.5125...	0.6147...	0.2323...	0.0	0.0	0.2363...	0.2592...	tested...
10	0.1764...	0.5025...	0.5573...	0.2323...	0.0957...	0.4709...	0.3921...	0.3456...	tested...
11	0.0	0.5226...	0.5245...	0.3737...	0.0756...	0.5007...	0.2107...	0.2716...	tested...
12	0.4117...	0.7638...	0.7213...	0.4444...	0.0	0.7451...	0.1392...	0.4444...	tested...
13	0.5882...	0.3417...	0.8688...	0.2323...	0.0579...	0.5290...	0.1177...	0.5802...	tested...
14	0.1176...	0.4723...	0.6229...	0.1818...	0.0780...	0.4709...	0.2681...	0.2839...	tested...
15	0.3529...	0.4020...	0.5409...	0.3030...	0.0	0.3904...	0.1293...	0.5061...	tested...
16	0.0	0.5929...	0.5245...	0.2323...	0.1052...	0.0	0.7152...	0.2592...	tested...
17	0.4117...	0.6884...	0.7377...	0.4141...	0.0	0.4769...	0.1615...	0.4814...	tested...
18	0.0	0.5276...	0.5573...	0.2222...	0.0	0.2980...	0.0975...	0.2716...	tested...
19	0.4117...	0.8090...	0.7049...	0.0	0.0	0.4530...	0.0681...	0.5802...	tested...
20	0.2352...	0.7236...	0.4754...	0.2828...	0.1654...	0.4396...	0.1185...	0.4567...	tested...
21	0.2941...	0.6884...	0.8852...	0.0	0.0	0.7272...	0.0938...	0.4567...	tested...
22	0.2352...	0.5879...	0.5081...	0.1212...	0.0	0.4426...	0.1570...	0.3703...	tested...
23	0.2352...	0.6180...	0.5081...	0.0	0.0	0.4769...	0.0933...	0.4320...	tested...
24	0.3529...	0.7738...	0.6065...	0.3232...	0.2281...	0.4366...	0.3466...	0.4814...	tested...
25	0.2941...	0.3919...	0.3934...	0.0	0.0	0.5022...	0.2702...	0.3086...	tested...
26	0.3529...	0.5879...	0.7868...	0.0	0.0	0.4277...	0.0648...	0.3703...	tested...
27	0.5294...	0.7839...	0.7049...	0.2828...	0.1832...	0.5111...	0.4913...	0.5185...	tested...
28	0.1764...	0.6633...	0.6557...	0.0	0.0	0.5126...	0.1661...	0.5432...	tested...
29	0.1176...	0.5025...	0.5245...	0.2323...	0.0	0.4426...	0.1520...	0.2592...	tested...
30	0.1176...	0.9899...	0.5737...	0.4545...	0.6418...	0.4545...	0.0652...	0.6543...	tested...

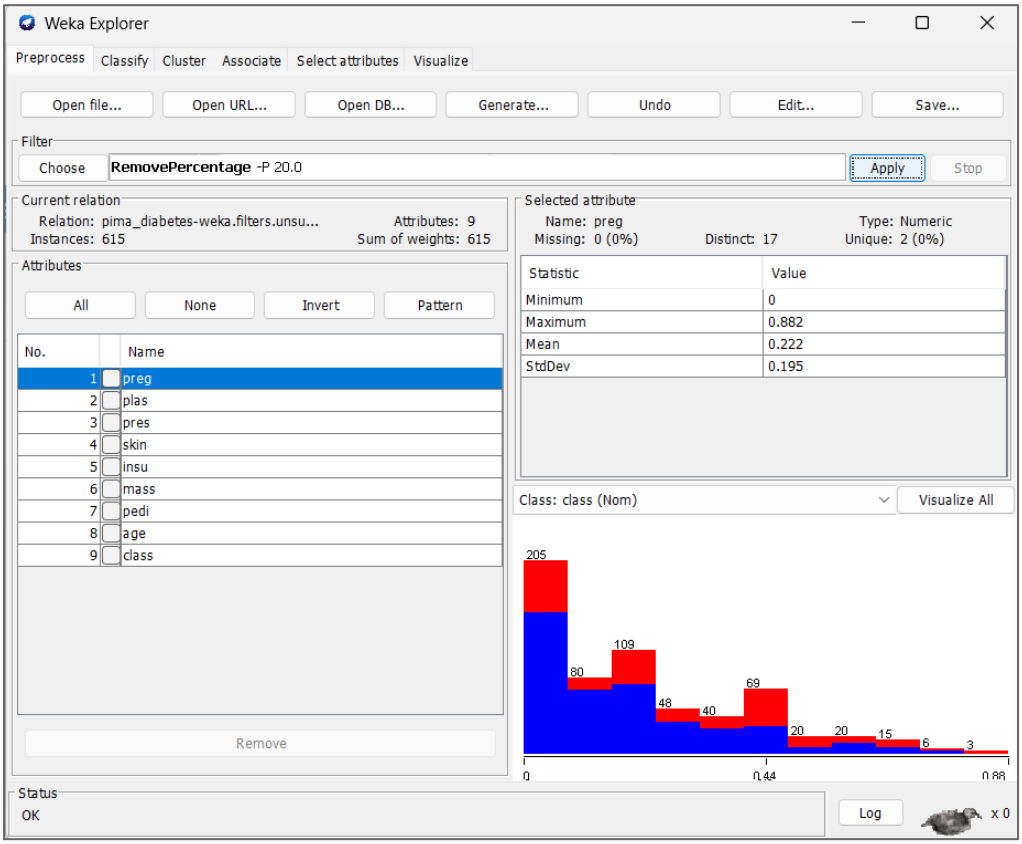
3.3.2. Splitting the dataset into training and testing data

Data splitting separates a dataset for model training and testing, enabling unbiased performance assessment, model selection, and overfitting prevention.

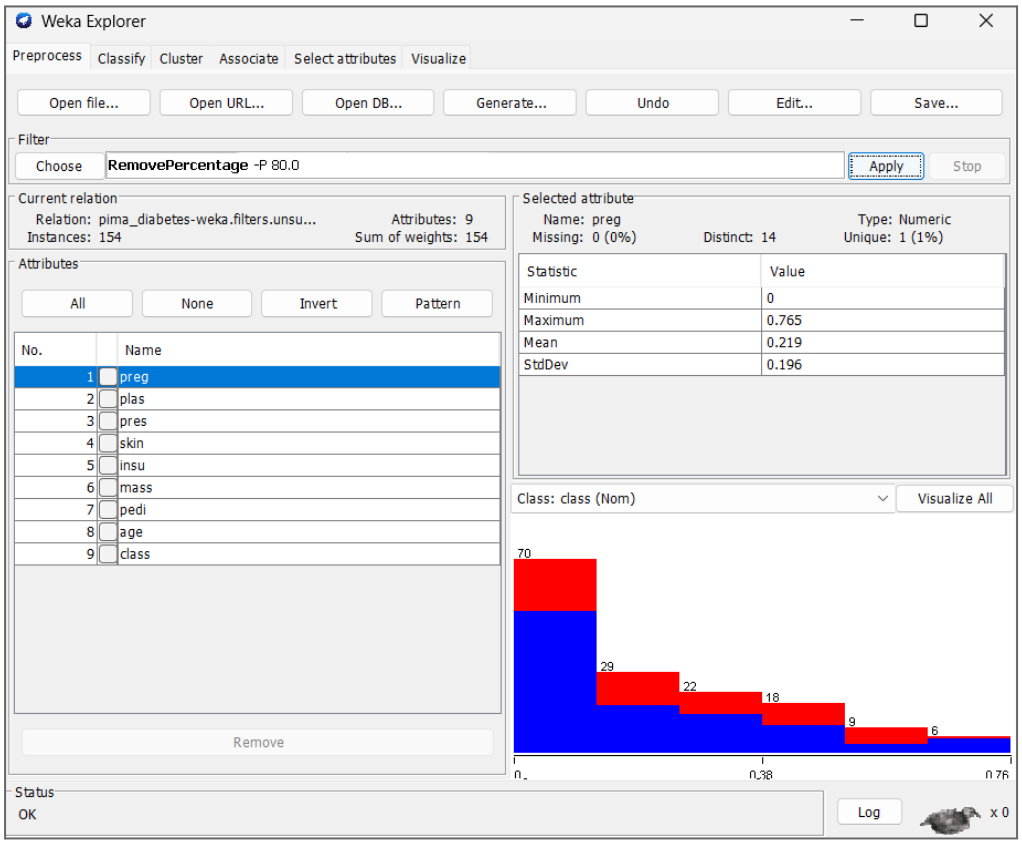
Filter used: RemovePercentage

Filter path: Filters > Unsupervised > Instance > RemovePercentage

20% DATA REMOVED FROM THE ORIGINAL DATASET
SAVING THIS DATASET AS TRAINING DATA



80% DATA REMOVED FROM THE ORIGINAL DATASET
SAVING THIS DATASET AS TESTING DATA

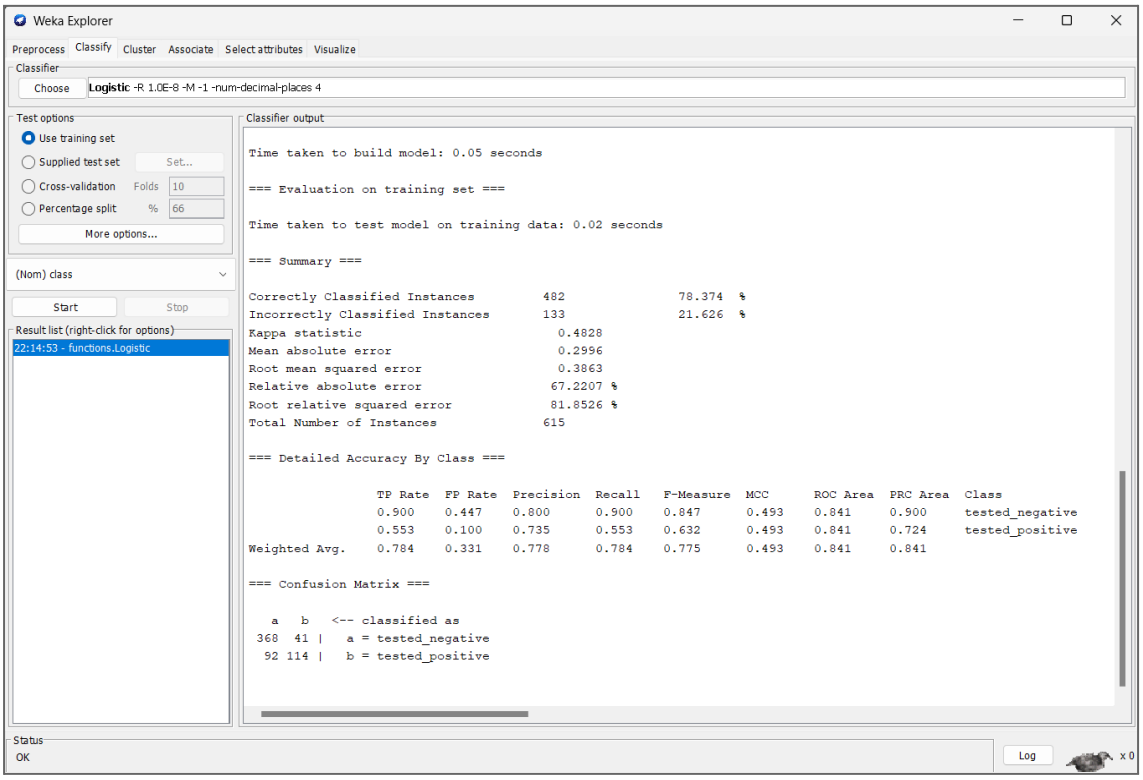


3.4. Classification

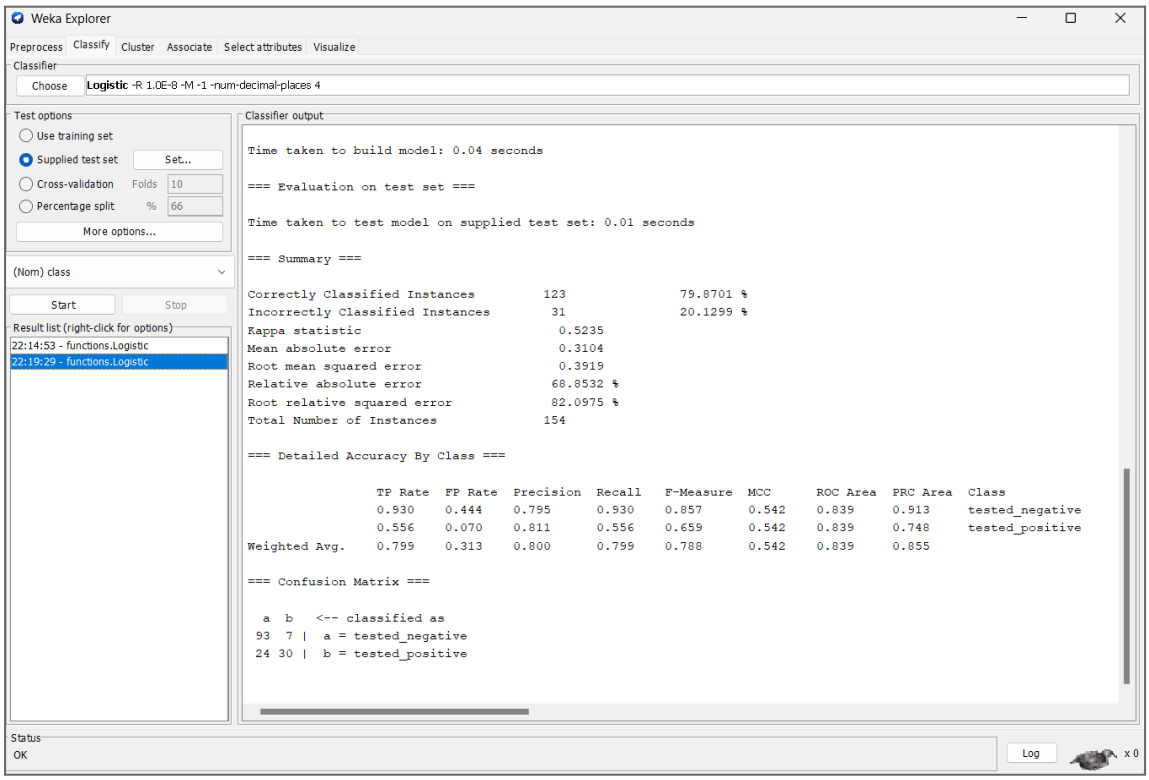
Applying classification algorithms to the training dataset:

3.4.1. Logistic Regression

TRAINING DATA

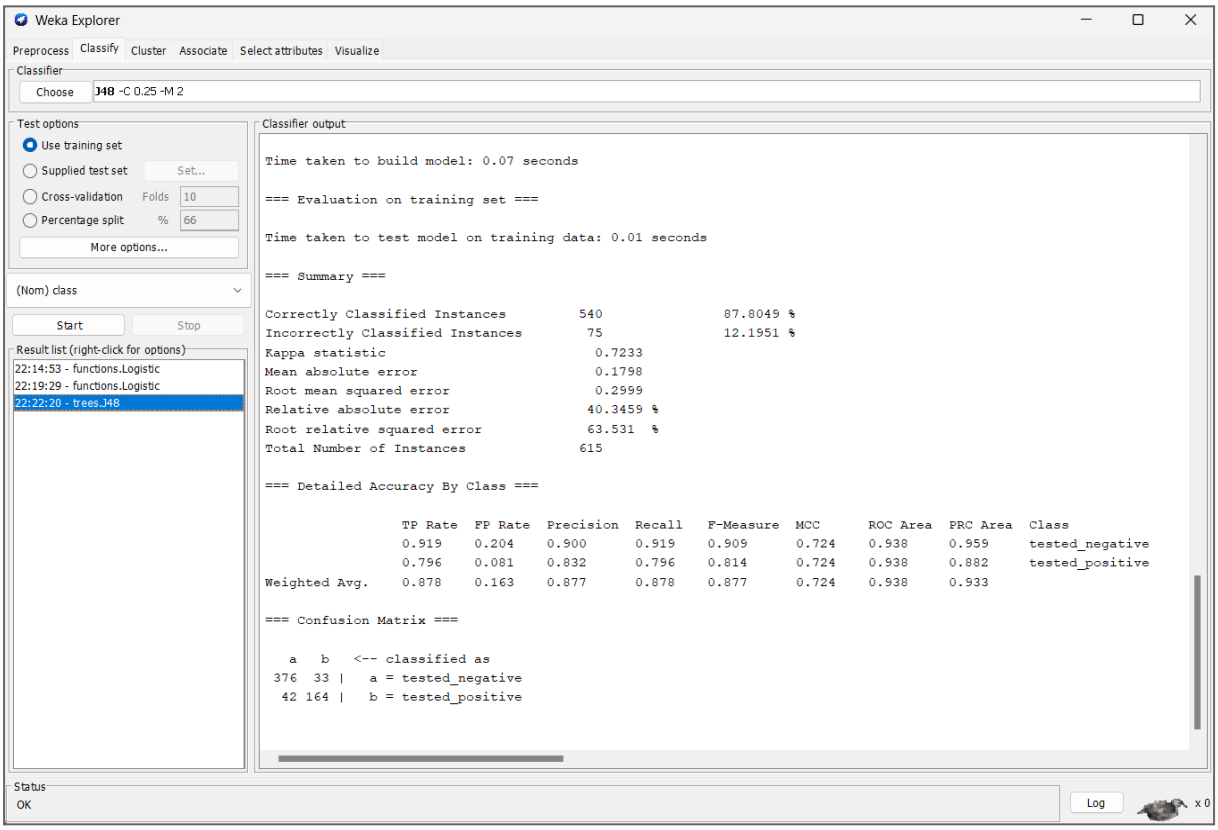


TESTING DATA

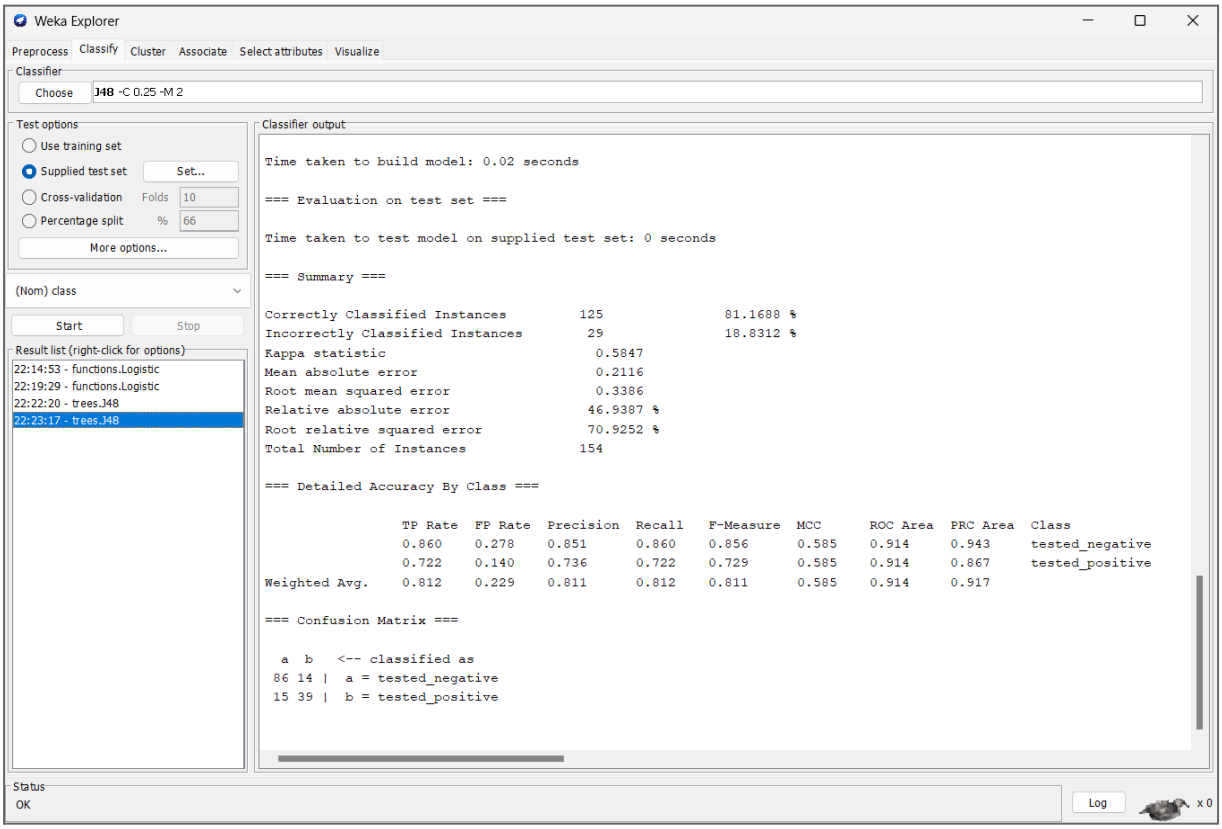


3.4.2. J48

TRAINING DATA

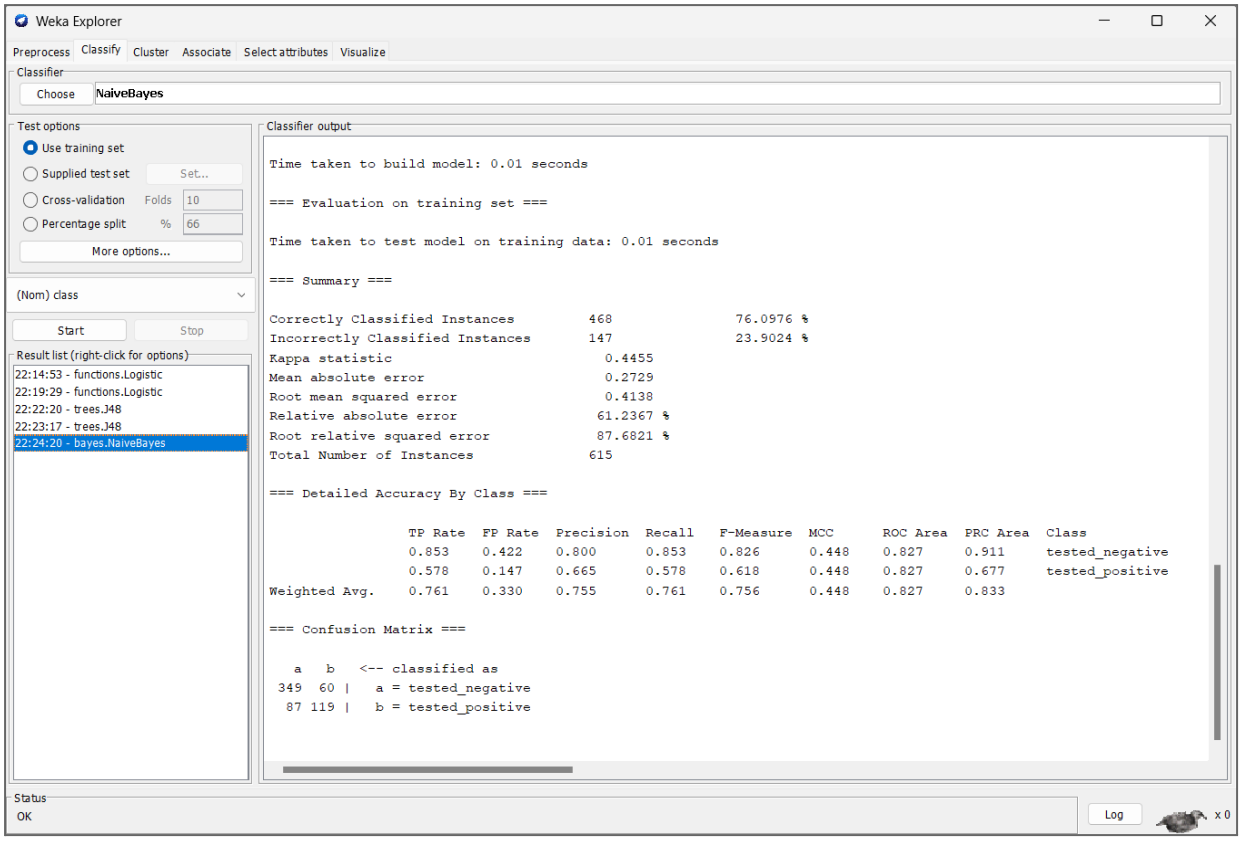


TESTING DATA

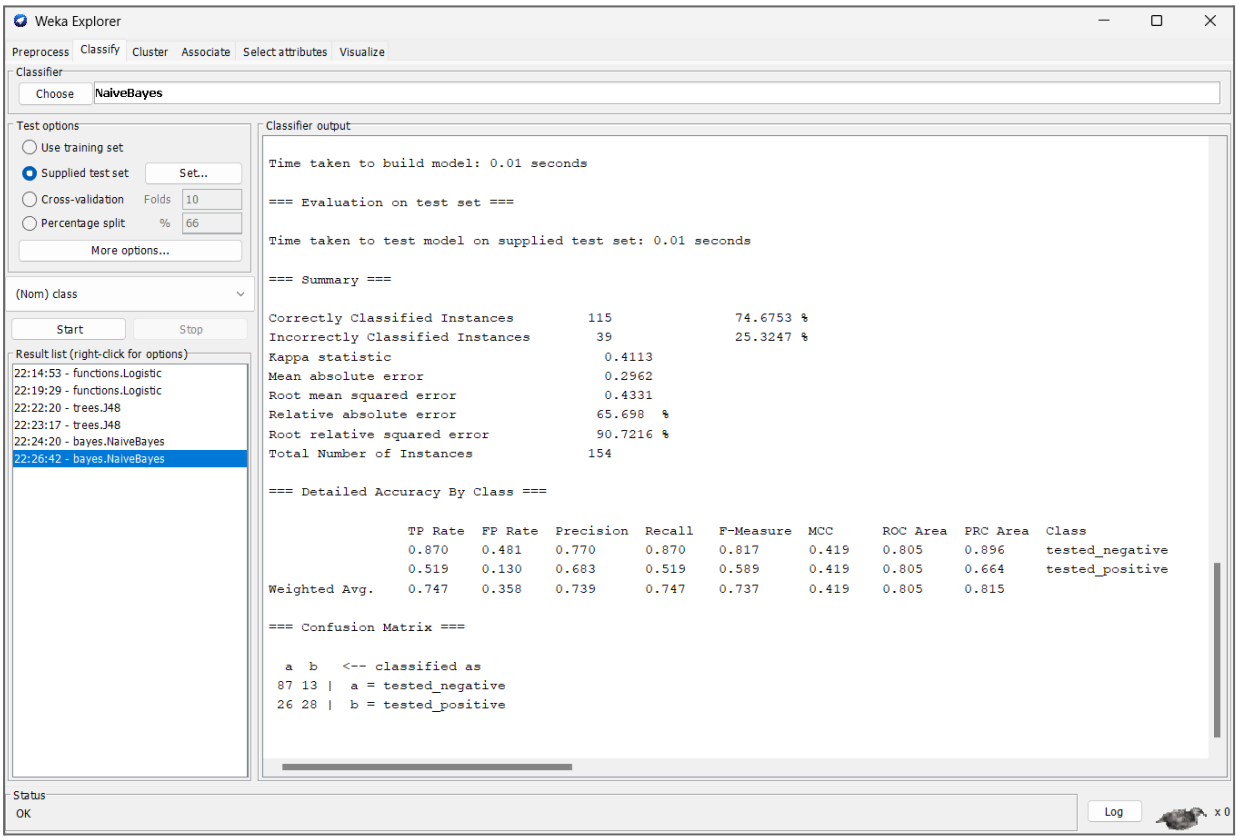


3.4.3. Naïve Bayes

TRAINING DATA

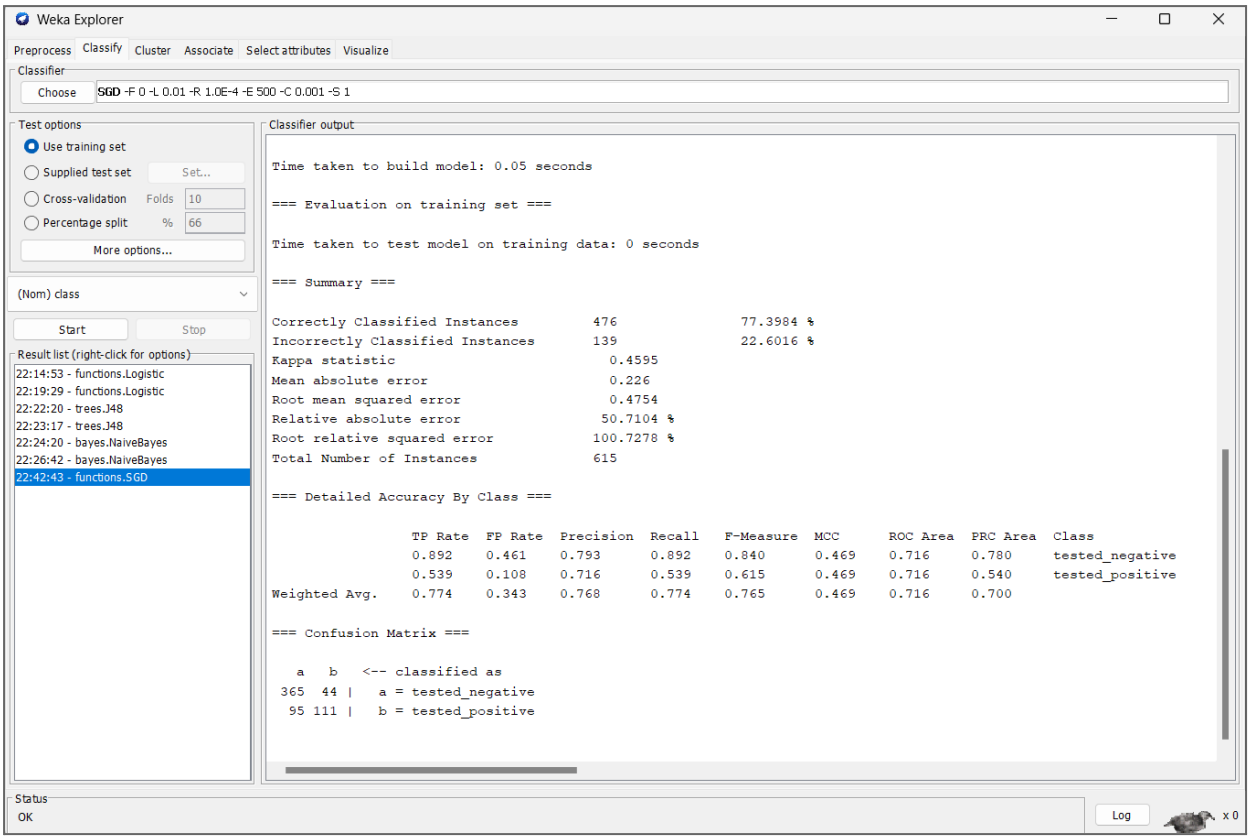


TESTING DATA

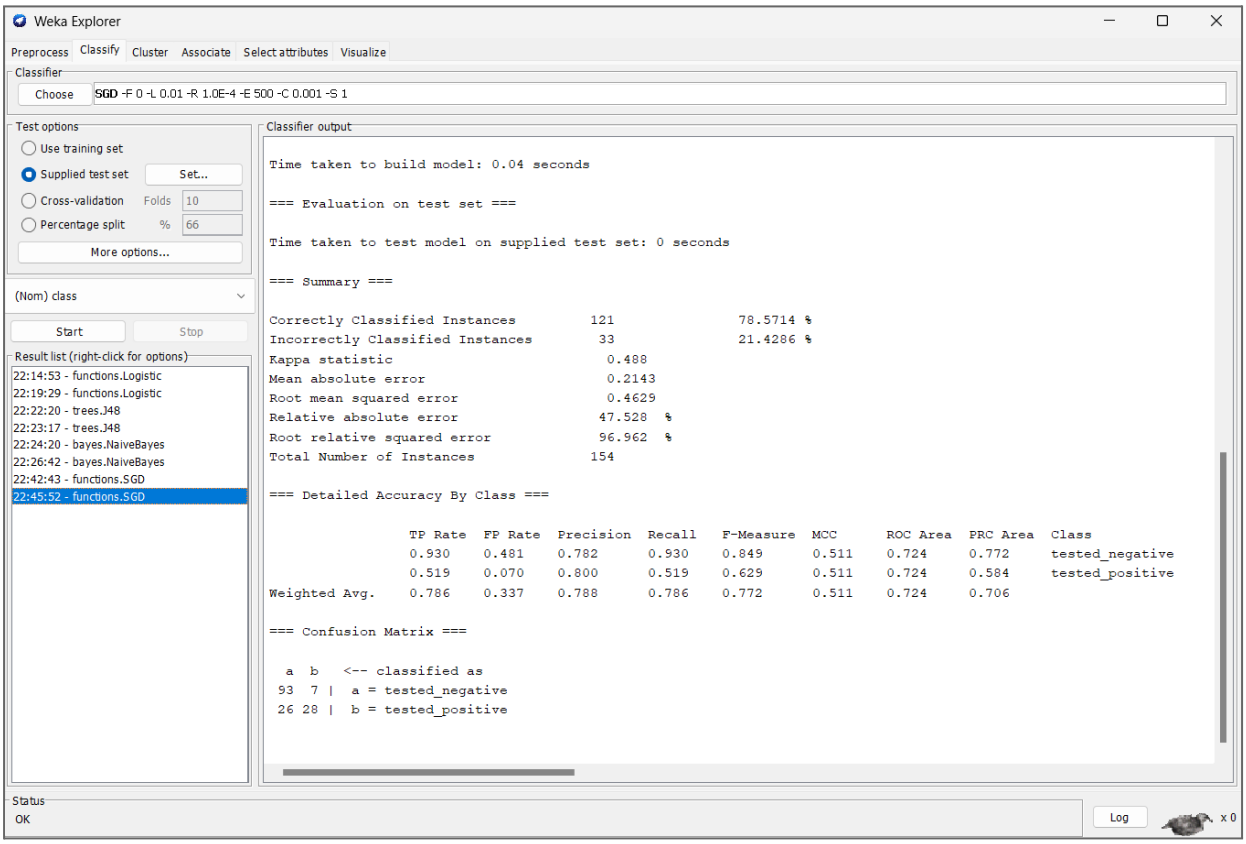


3.4.4. Stochastic Gradient Descent

TRAINING DATA

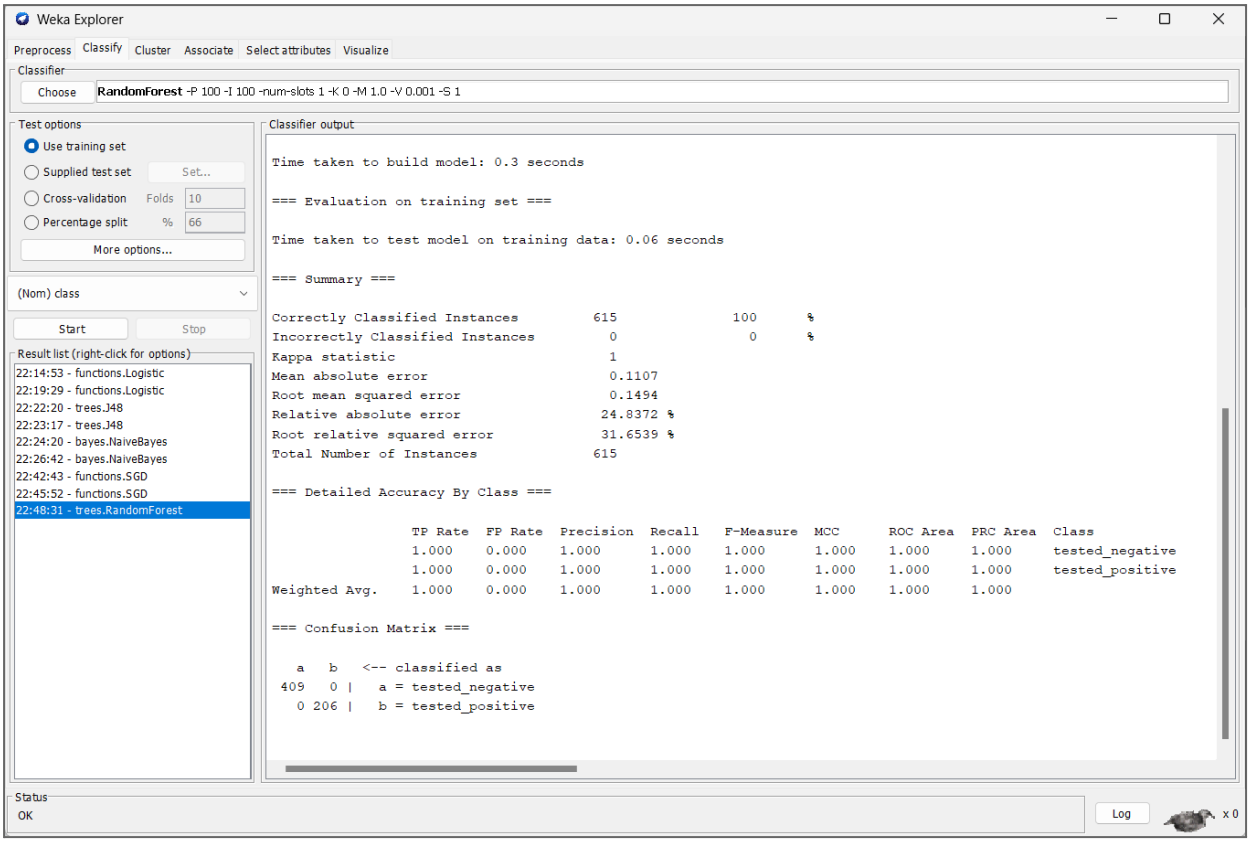


TESTING DATA

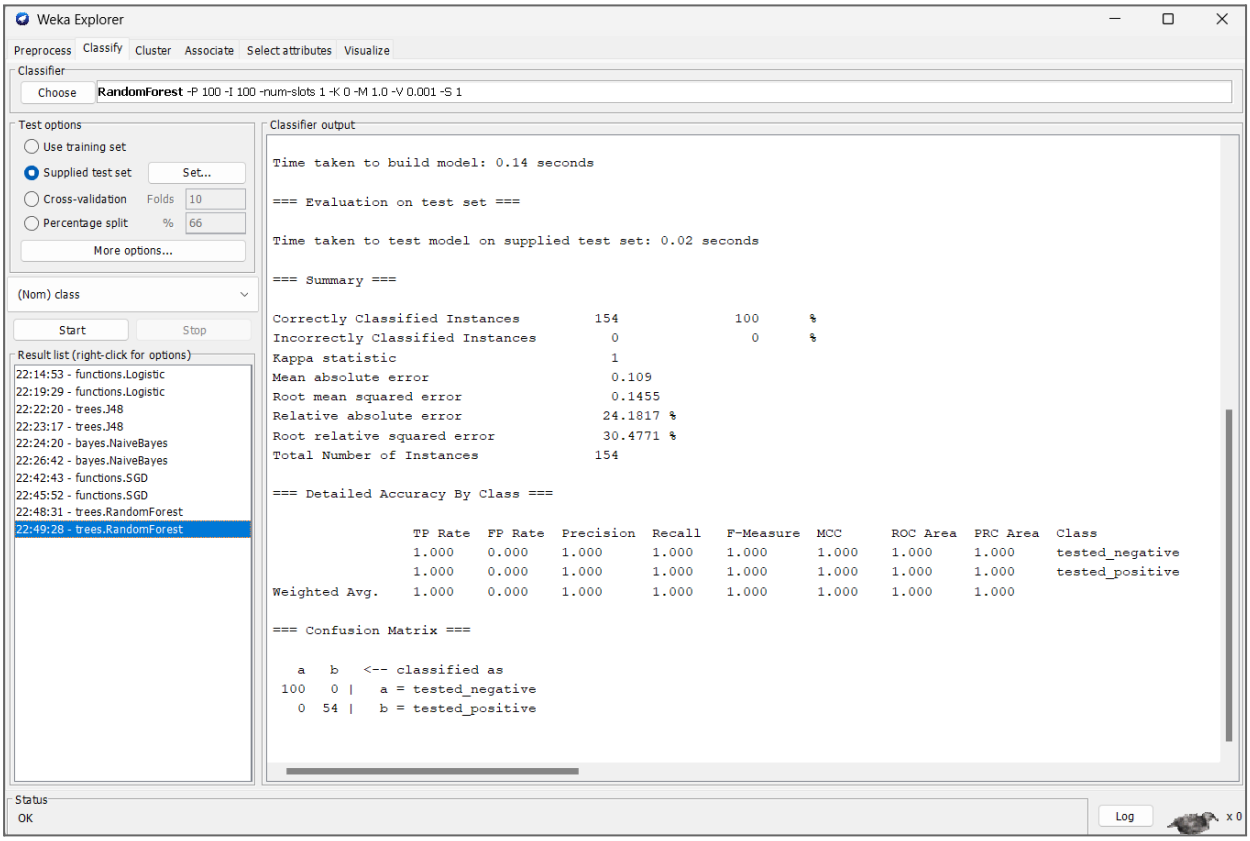


3.4.5. Random Forest

TRAINING DATA



TESTING DATA



4. DISCUSSIONS

Observation Table

Algorithm	Accuracy (training)	F-score (training)	MCC (training)	Accuracy (testing)	F-score (testing)	MCC (testing)
Logistic Regression	78.374	0.847	0.493	79.8701	0.857	0.542
J48	87.8049	0.909	0.724	81.1688	0.856	0.585
Naïve Bayes	76.0976	0.826	0.448	74.6753	0.817	0.419
SGD	77.3984	0.840	0.469	78.5714	0.849	0.511
Random Forest	100	1.00	1.00	100	1.00	1.00

The project's evaluation of classification algorithms on the diabetes dataset in Weka revealed varying performance. Random Forest achieved perfect accuracy on both training and testing dataset. J48 exhibited high training accuracy but lower testing accuracy with a reduced Matthew's correlation coefficient, indicating potential overfitting. Naïve Bayes and SGD offered consistent results and are suitable for a balance between accuracy and simplicity. The choice of the best algorithm depends on specific project requirements and trade-offs between accuracy, interpretability, and generalization. Overall, Random Forest is the most suitable algorithm for this dataset.

5. REFERENCES

Pima Indians Diabetes Database, National Institute of Diabetes and Digestive and Kidney Diseases (https://github.com/renatopp/arff-datasets/blob/master/classification/diabetes.arff)