International Journal of Computer Trends and Technology
ISSN: 2231–2803 / https://doi.org/10.14445/22312803/IJCTT-V73I10P103

Volume 73 Issue 10, 15-24, October 2025

Original Article

Comparative Study on Supervised Machine Learning Algorithms using Rapid Miner and the Weka tool

Puneet Kour[1], Rakshit Khajuria[2], Jewan Jot[3]

[1,2,3]*Department of Computer Science and Engineering, UIET, University of Jammu, India.*

[2]*Corresponding Author : Khajuriarakshit75@gmail.com*

Received: 22 August 2025 Revised: 26 September 2025 Accepted: 13 October 2025 Published: 29 October 2025

Abstract *- The performance of various supervised machine learning approaches was compared in this paper, utilizing a variety of visualization tools, including Orange, Weka, and RapidMiner. In addition, machine learning methods such as logistic regression, decision trees, support vector machines, linear regression, and Classification (Naïve Bayes) are used to analyze bacterial cell data and predict the outcome of bacterial cell detection on an agar plate. Furthermore, we use the RapidMiner tool to examine the outputs of various classifiers and determine which one works better than the others. With an 80:20 ratio, decision trees perform 92% more accurately than the alternative method.*

Keywords *- Machine Learning, Classification, RapidMiner, Artificial Intelligence, Supervised Learning.*

1. Introduction

Our entire lives are now digitally documented, and everything is tied to a data source. We are living in the data era. For example, there are many various kinds of data in the present electronic world, such as IOT, cybersecurity, commerce, social media, healthcare, COVID, virtual classrooms, etc. These data could be partially structured, unstructured, or structured. A range of artificial intelligence models can be produced by extracting the relevant elements from the data using machine learning techniques.

The branch of artificial intelligence known as Machine L earning (ML) focuses on using statistical, probabilistic, and o ptimization techniques along with computing algorithms to le arn from and find significant patterns in data (whether structu red, unstructured, or complex).

Applications for machine learning algorithms are numer ous and include cybersecurity, manufacturing process improv ement, and cancer diagnosis counts.

Most research initiatives have been implemented using s upervised learning methods.

In this paper, supervised learning techniques are covered in further detail with examples that facilitate their comprehension. Although there are several supervised algorithms, we just discussed the most well-known one to point readers in the direction of pertinent sources for a fuller picture. This chapter is more beneficial for laypeople or students working in fields like agriculture or the life sciences that might be interested in applying computational methods to their field of study. This section is divided into 4 sections: Section 2 provided information on various types of data and machine learning techniques; Section 3 described supervised learning algorithms in detail; Section 4 showed how various machine learning techniques performed when used on two datasets (the pre-defined dataset and the bacterial cell dataset), and Section 5 provided a conclusion.

2. Literature Survey

Over the past decade, there has been considerable interest in comparing and evaluating supervised machine learning algorithms across various problem domains and datasets. For example, the work by A systematic comparison of supervised classifiers (Amancio et al., 2013) conducted an empirical evaluation of nine well-known classifiers, all implemented within Weka, and examined how algorithm performance varied with dataset dimensionality and parameter settings. They found that while the k-Nearest Neighbors (k-NN) method often outperformed others on high-dimensional data using default settings, methods such as the Support Vector Machine (SVM) benefited substantially from parameter tuning. This foundational work underscores that algorithm choice and configuration matter, and it sets the stage for comparative algorithm benchmarking independent of tooling.

Building on this algorithm-benchmarking focus, several studies deploy common classification algorithms across varying datasets using Weka, providing a baseline for performance comparisons. For instance, the paper by Arora & Suman (2012), titled “Comparative Analysis of

Classification Algorithms on Different Datasets using WEKA," used multiple datasets and compared several classifiers, demonstrating how algorithm effectiveness can vary significantly with dataset characteristics. IJCA Similarly, D'Souza et al. (2017) in their study Comparative Analysis of Classification Algorithms using WEKA applied algorithms including Naïve Bayes, k-NN (K*), and Random Forest to a diabetes dataset, showing large variations in accuracy depending on parameter settings and preprocessing like supervised discretization. IJERT Together, these studies highlight that algorithm benchmarking is a rich area, and they offer insights into which supervised methods are frequently used and how they perform under relatively "standard" tool-conditions.

In parallel, another strand of research examines the tools themselves — how different machine learning/data mining platforms compare when applying supervised algorithms. A notable study, "Perbandingan Kinerja Tool Data Mining Weka dan RapidMiner Dalam Algoritma Klasifikasi" (Faid et al., 2019), directly compared Weka and RapidMiner on classification tasks, focusing on accuracy as the primary performance metric.

The authors concluded that tool performance differed, underscoring that the tool choice can influence results even when using the "same" algorithm. ejournal.ikado.ac.id Another work by Ainurrohmah (2021), titled Akurasi Algoritma Klasifikasi Pada Software Rapidminer dan Weka, reviewed prior studies using classification algorithms (Decision Tree, Random Forest, k-NN, Naïve Bayes, MLP) in both RapidMiner and Weka (on spam-text data), and found that accuracy differed across tools—some studies favored Weka, others RapidMiner—and that the "best" algorithm varied by dataset and tool. UNNES Journal, these tool-comparison studies suggest that, beyond algorithm and dataset, the platform matters too: preprocessing defaults, parameter defaults, implementation differences, and user interface may all influence the outcome.

Further reinforcing this tool-versus-algorithm dimension, the article by Moghimipour and others (2012), titled Comparing Decision Tree Method Over Three Data Mining Software, compared decision-tree performance across SPSS-Clementine, RapidMiner, and Weka on a large real dataset (~3,515 instances) and found that the best accuracy (92.49 %) was achieved in RapidMiner for their decision tree configuration. CCSE, this indicates that even for the same algorithm (decision tree), the tool choice can yield measurable differences. Complementing this, other commentary (e.g., KDnuggets discussions) notes that while RapidMiner and Weka share much code (RapidMiner was historically built upon Weka), there are differences: RapidMiner includes additional operators, a more comprehensive GUI, and some distinct implementations. KDnuggets+1, thus, any comparative study must account for tool-level variations.

Beyond tool and algorithm comparisons, domain-specific supervised-learning applications provide further context. For example, in the educational domain, Sathe & Adamuthe (2021) in their work Comparative Study of Supervised Algorithms for Prediction of Students' Performance applied algorithms including C5.0, J48, CART, NB, k-NN, Random Forest, and SVM on datasets from school, college and e-learning platforms, concluding that Random Forest and C5.0 tended to outperform other methods across datasets.

MECS Press A study on text classification (Asogwa et al., 2021) used a hybrid model of Naïve Bayes + SVM implemented via Weka to classify big-text data, achieving high accuracy (96.76 %) compared to individual methods. arXiv These domain applications underscore that algorithm selection and tool workflow (preprocessing, parameter tuning) are highly influenced by problem context.

When synthesising the literature, several important patterns emerge. First, algorithm performance is dataset- and context-dependent: no single classifier uniformly dominates across all datasets. Studies like Amancio et al. (2013) show that kNN may excel on high-dimensional data by default, but with tuning, other methods can catch up. Second, tool differences matter: several studies show that Weka vs RapidMiner produce different results even when implementing the "same" algorithm on similar data, likely due to differences in preprocessing pipelines, defaults, and implementations (e.g., Faid et al. 2019; Ainurrohmah 2021). Third, the combined effect of algorithm choice, parameter tuning, dataset characteristics, and tool environment means that comparative studies must control for all variables — something that many existing works only partially address. For example, while algorithm benchmarking in Weka is abundant, fewer studies combine multiple algorithms and multiple tools under the same controlled conditions.

Finally, from a gap analysis viewpoint, the literature indicates that while many papers compare algorithms within a single tool (often Weka) and some compare tools for given algorithms, very few studies systematically compare *multiple supervised algorithms across both Weka and RapidMiner under uniform experimental settings (same datasets, same preprocessing, same parameter tuning) and report extended metrics (accuracy, precision, recall, F1, training time, tool usability). Thus, a study that implements such a design fills a clear research gap: it contributes not only to algorithm benchmarking but also to tool-effect quantification, thereby helping practitioners decide both which algorithm and which tool may be more appropriate for their supervised-learning task.

3. Types of Data

As is well known, the development of any machine learning model or the drawing of any outcome data is crucial. So, in this section, we will talk about the different kinds of data that can be used to train the machine learning algorithm and identify patterns. We will also go over various machine learning techniques and approaches.

Structured: data that is readily accessible and organized. We may also remark that the data has some order. Geolocation, stock information, relational databases, and other types of structured data are examples.

Unstructured data: that cannot be easily collected, processed, and analyzed because it lacks a predetermined format. The majority of this sort of data is composed of text and multimedia. Audio files, photos, presentations, videos, and other types of unstructured data are examples.

Semi-structured data is easier to analyze because it has some organisational characteristics. Examples include HTML, JSON, NOSQL databases, etc.

3.1. Types of Machine Learning Techniques

Machine learning makes predictions within a reasonable range by using preprogrammed algorithms that analyze input data to learn and improve their operations through optimization. In the following section, we briefly describe each type of learning technique and the extent to which it can be used to address problems in the real world.

Supervised learning: A set of input and output labelled datasets. Depending on the mode of learning task, the supervised learning method works on two different types of issues: regression and Classification.

Regression: The output for the regression category includes an interval on the real line and continuous values. In this method, the output is determined by estimating the model created on the relationship between the two parameters x and y, i.e., the feature and model. Apart from this, the primary goal of regression is to create an equation that, given a value for x, produces the value of y. Regression includes Support Vector Regression, Random Forest Trees, and Linear Regression.

Classification: The output of the classification types accepts categorical values marked with class labels. By mapping the function in "x" and "y," this method is used to identify discrete output variables "y." It picks up knowledge from the pool of legitimately useful data sets. It maps the function and predicts the category or details of the perception that was questioned about. Moreover, Classification uses the values of the preparation set and the data (class names) in sorting features to either forecast distinct class names or characterize information (create a model) and then uses it to arrange new information. Logistic regression, decision trees, random forests, and gradient-boosted trees are some classification models.

3.2. Unsupervised

In these methods, the dataset contains data samples whose output is not clear. In other words, the data are not labelled—analysis of the unlabeled dataset without the intervention of a human. The goal of this learning method is to identify the relationship and patterns in the data. In addition to this, data are compared based on a similarity scale to be classified into categories.

This is frequently used to extract generative features, relevant patterns, and structure identification, organize results, and for exploratory purposes. Clustering, density estimation, feature learning, dimensionality reduction, association rule discovery, anomaly detection, etc., are some of the most popular unsupervised learning tasks.

3.2.1. Reinforcement

Reinforcement learning is a form of machine learning that allows software agents and machines to automatically analyze the optimal behavior in a specific context or environment to increase their efficiency. i.e., environment-driven approach. The ultimate goal of this reward or penalty-based learning approach is to use the knowledge gained from environmental activists to take steps that will either increase the reward or reduce the risk. It is an effective technique for building AI models that can improve the operational effectiveness of complex systems like robots, autonomous driving, manufacturing, and supply chain logistics, but it is not recommended to use it to tackle simple or elementary issues.

The popular algorithms employed in this process are Q-Learning and the temporal difference learning algorithm, which are mostly used in issues with the control precision of robots. Table 1 summarizes the above machine learning methods.

Table 1. Summary of different machine learning methods

Learning Technique	Dataset	Purpose
Supervised	Labelled	Determining the relationship between the input and output datasets and predicting the labels of the testing data.
Unsupervised	Unlabeled	Identifying the data patterns and placing data samples in groups.
Reinforcement		Finding the best action through interacting with the environment.

3.3. Supervised Machine Learning Algorithms

3.3.1. Decision Tree

A Decision Tree (DT) is one of the earliest and prominent non-parametric machine learning algorithms. A decision tree is a graph that represents options and their outcomes as a tree. The edges of the graph indicate the conditions or rules for making decisions, whereas the nodes in the graph represent an event or a choice. Each tree consists of nodes and branches, where each node represents a set of characteristics that need to be categorized, while each branch indicates a possible value for the node (Figure 1). In addition to this, in order to forecast the output class, DT builds the learning model using a collection of IF-THEN rules derived from the training set. Based on features in the dataset, a hierarchical tree is built.

DTs are frequently used in various medical diagnostic regimens because they are simple to use, quick to learn, and straightforward to interpret. DT algorithms that are well known include ID3, C4.5, and CART. Recently proposed algorithms, such as BehavDT and IntrudTree, are successful in the pertinent application domains, such as user behavior analytics and cybersecurity analytics, respectively.

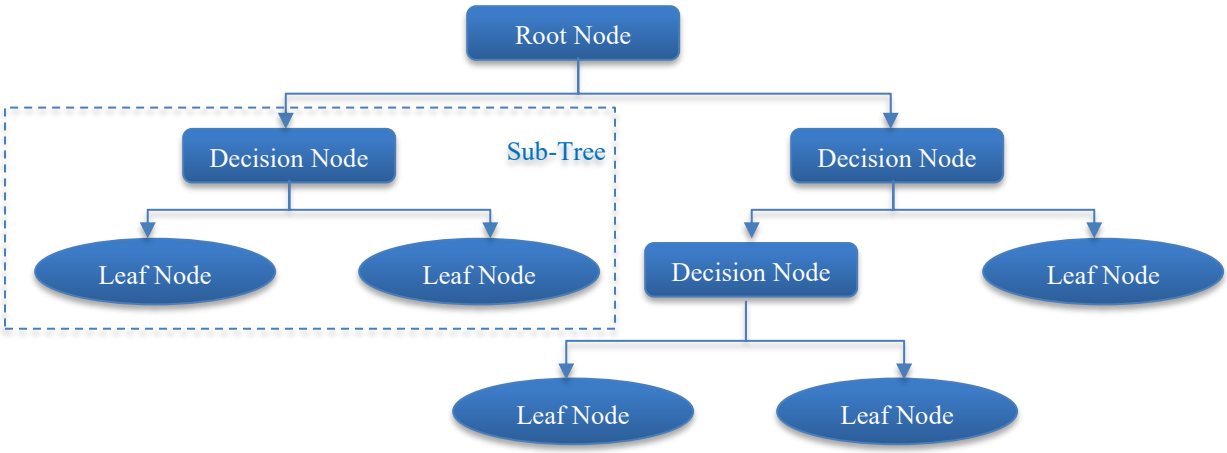

Fig. 1 Decision tree

3.3.2. Support Vector Machine (SVM)

Both linear and non-linear data can be classified using the Support Vector Machine (SVM) technique. It begins by mapping every piece of data into an n-dimensional feature space, where n is the total number of features. Then, while maximizing the marginal distance for both classes and minimizing the classification errors, it determines the hyperplane that divides the data items into two classes. The distance between the decision hyperplane and the closest instance that belongs to the class is what is known as the marginal distance for that class. Generally, each data point is initially represented graphically as a point in an n-dimensional space (where n is the number of features), with the value of each feature being the value of a particular coordinate. Then, in order to perform the Classification, we must identify the hyperplane that separates the two classes with maximum margin. Figure 2 illustrates the SVM classifier.

A Support Vector Machine (SVM) is a popular machine learning tool that can be used for Classification, regression, or other tasks. A support vector machine creates an individual hyperplane or a collection of hyperplanes in high- or infinite-dimensional space. Assuming that the larger the margin, the smaller the classifier's generalization error, the hyperplane, which is the farthest from the nearest training data points in any class, achieves a significant separation. It works well in high-dimensional spaces and exhibits different behavior on different mathematical operations known as the kernel—Sigmoid, Radial Basis Function (RBF), linear, polynomial, etc.

This technique has a number of benefits, including the ability to handle small, well-organized datasets because it just employs a portion of the training coordinates. The Sequential Minimal Optimization (SMO) algorithm breaks the dataset into several parts and attempts to solve the smallest possible optimization problem at first in each step. Finally, after the entire process is done, it rejoins effectively using Osuna's theorem to ensure that it is effectively converged. The disadvantage that comes with dealing with large datasets is the computational power and time required. However, this has been resolved.

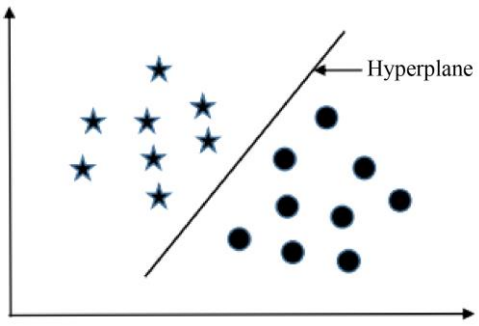

Fig. 2 Support vector machine

3.3.3. Random Forest

This algorithm was created by Tim Kam Ho. A decision tree is the fundamental component of Random Forest. An assortment of trees known as a "random forest" is just like a forest. With no prior information, random forest learns about the framework of the necessary object with the aid of the dataset provided to reduce the percentage error and to give the best possible outcome.

A random forest classifier is well known as an ensemble classification technique. It is suitable for both categorical and continuous variables and can be applied to classification and regression problems. The "parallel ensembling" technique used in this method fits multiple decision tree classifiers simultaneously on various data set sub-samples and uses averages or majority voting to determine the final outcome. In this method, once the random forest is created, it is used to predict the labels, and these final labels in the samples are calculated using the majority voting parameter.

Moreover, in this learning method, two ways are used to introduce randomness. The algorithm bootstraps to extract n samples with replacement in the first step. Since some samples will be missing and others repeated, any data set obtained in this manner will have the same size as the original dataset. Second, the algorithm chooses a subset of these samples at each decision node at random and then chooses the feature that best divides these samples, as shown in the Figure.

Therefore, RF learning models with several decision trees often have higher accuracy than a model with a single decision tree. Additionally, the classifier exhibits good scalability and parallelism in classifying high-dimensional data, and is fast, accurate, and noise-resistant. As a result, it reduces the over-fitting issue and prediction accuracy, and control is both improved. Moreover, the decision tree's performance bottleneck is eliminated by the random forest by incorporating the bagging approach.

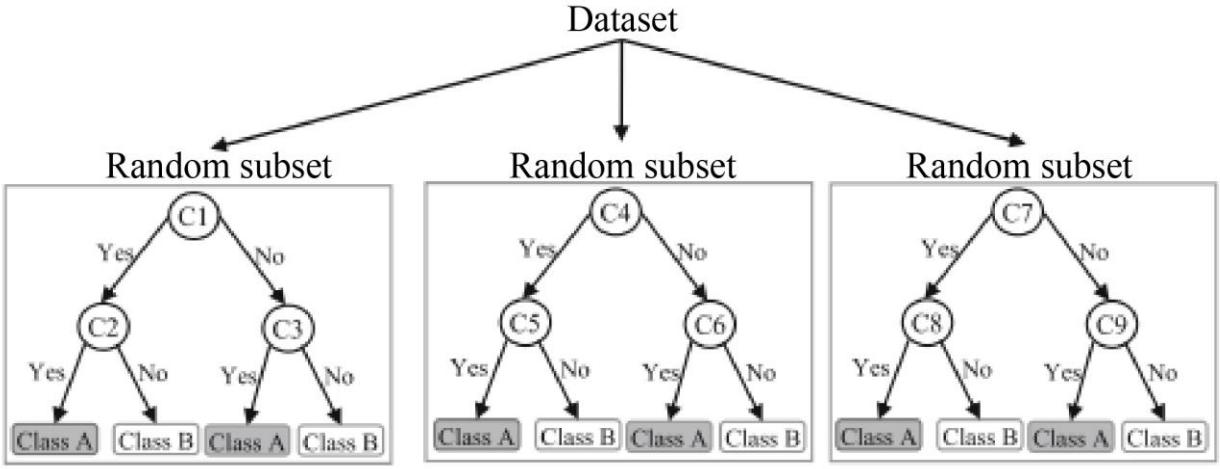

Fig. 3 Random forest

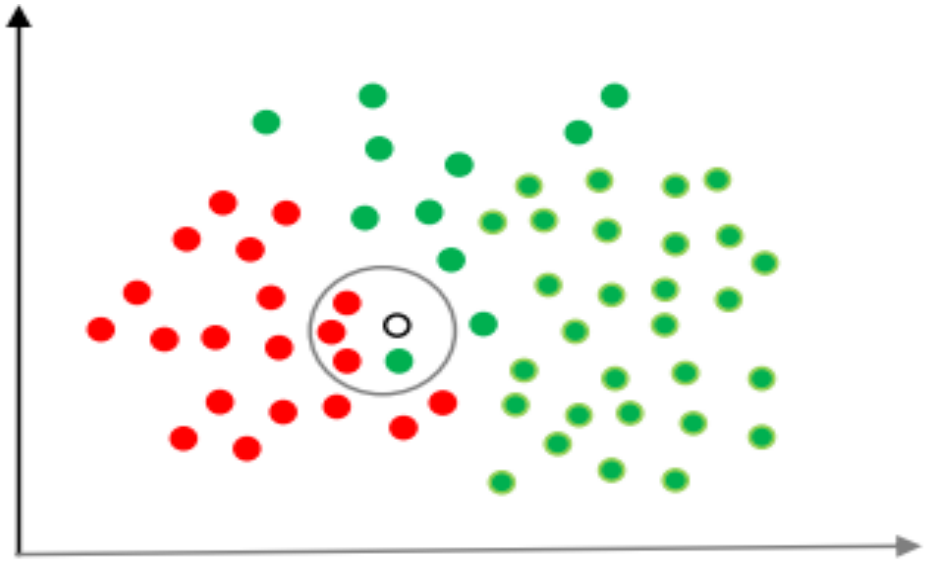

Fig. 4 Naïve Bayes

Table 2. Machine Learning algorithms: advantages and disadvantages

Algorithm	Advantages	Disadvantages
Random Forest	• Ability to manage noisy data, • High classification speed, • suitable for large and heterogeneous databases	• Not able to manage missing values • Low learning speed • Implementation is quite difficult • Average accuracy • Difficult for humans to understand. • Low ability to manage overfitting • Low ability to manage highly correlated data.
SVM	• Capacity to manage data with high accuracy • High classification speed • Manage data with linear and nonlinear separability. • Manage data with high correlation	• Low ability to manage overfitting • Not able to manage missing values • Low learning speed • Low ability to manage noisy data • Assuming linear separability for the dataset.
Decision Tree	• High Classification and computational speed • Handles missing values • Simple understanding	• Large tree design complexity • Inability to control overfitting, noisy data, and manage data with high correlation • Average accuracy
Naïve Bayes	• Simple understanding and implementation • High computational and learning speed • High Classification • Handles missing values • Managing overfitting and noisy data	• Inability to manage data with high correlation • Low accuracy • Assuming that features are independent.

3.4. Data Visualization Tool

In this work, we will compare supervised learning algori thms on an opensource dataset and a predetermined dataset (t he Titanic Sample data) given by RapidMiner. We will also b riefly examine two data visualization tools, RapidMiner and Weka.RapidMiner. The user friendly visual environment is the RapidMiner Studio program. Without the need for coding, this is the method employed for machine learning. Anyone who wants to test out a concept without investing a l ot of time or energy in it will find this platform useful.

The primary drawback of the RapidMiner Tool is its inability to function with images.

Second, although it is not open source software, students can use it for free for a year, after which they can renew. Only 10,000 tuples can be accessed for free for business purp oses; in order to access more data, we must pay charge to pur chase this instrument…RapidMiner Go can let you rapidly build predictive models from your data. Data is all that is required to predict a model.

3.4.1. Weka

Weka is an open-source programme that offers tools for data preprocessing, Classification, clustering, association rules, implementation of several Machine Learning algorithms, and visualization tools. The algorithms can be directly applied to a dataset to solve real-world data mining problems quickly. Machine learning models are often developed more quickly using Weka

3.5. Dataset and Parameter Details

In this paper, the built-in dataset of Titanic, which contains 1309 tuples provided by RapidMiner, was used. The first step is to split the data into a testing and training set. In this experiment, a ratio of 80:20 results in having -----training samples and----test samples. For every learning model, an automatic sampling type was used.

In the case of Weka, the dataset used was taken from the internet. In this study, Random Forest and Naïve Bayes classifiers were used with 10 Fold cross-validation in both cases. We applied an edge histogram and color layout filter to both classifiers and evaluated the performance of both algorithms.

Dataset classes {FLOWER, BUTTERFLY, OWL, HUMAN}

Table 3. Dataset

1	BUTTERFLY	50
2	OWL	50
3	FLOWER	24
4	Human	14

3.6. Comparison Models and Evaluation

This section presents the results obtained by applying different supervised learning models to the dataset. Section ---- and----presents the results obtained from the RapidMiner and Weka tools, respectively.

Different supervised machine learning models were tested and evaluated in this study. Table 1 shows the mean accuracy obtained from the above-mentioned models.

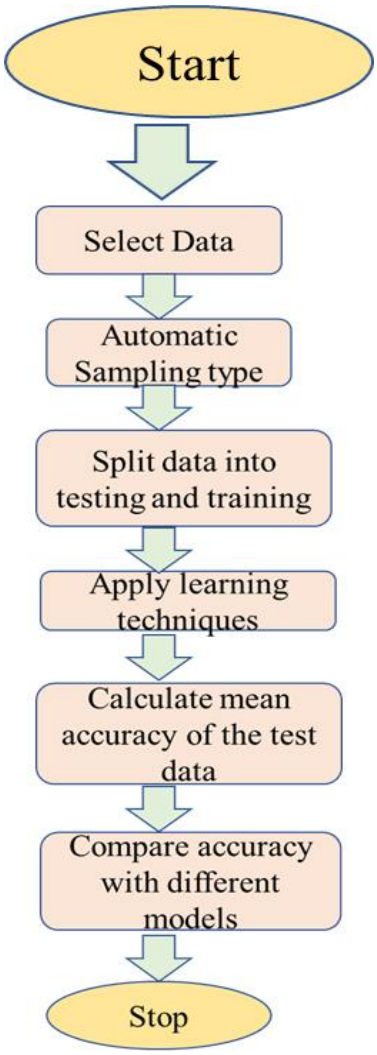

Fig. 5 Flow chart

Table 4. Shows the Result of various models

Supervised Learning Models	
Model	**Mean Accuracy (%)**
Decision Tree	93.89
Random Forest	94.27
Naïve Bayes	87.40
Logistic Regression	38.17
Linear Regression	76
Support Vector Machine	75

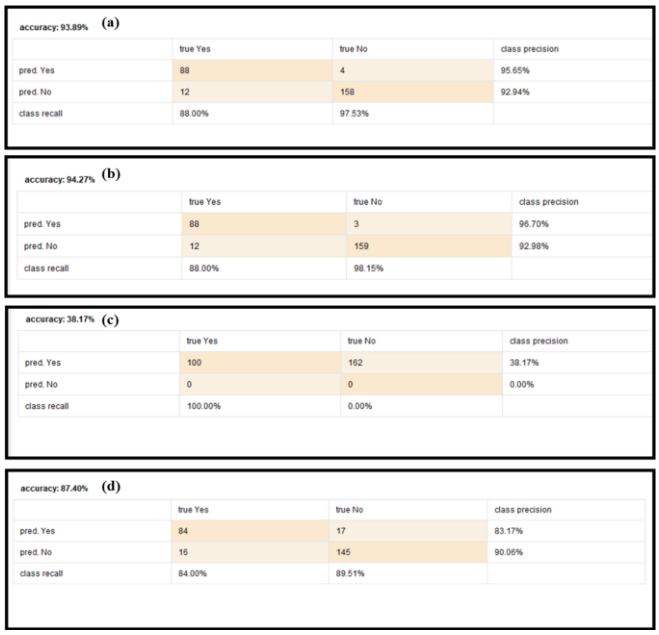

Fig. 6 Result of Various Models

From the above results, we can conclude that random forest gives better accuracy with a mean accuracy of 94.27% as compared to other algorithms.

Table 5. Shows the Result of Supervised Learning models

Supervised Learning Models	
Model	Mean Accuracy (%)
Random Forest	87.68
Naïve Bayes	87.68

Results from the above tables show that both the algorithm, i.e., random forest and Naïve Bayes, gives the same accuracy. This indicates that both learning models are efficient in the dataset.

Classification tasks, it is clear from a thorough literature review and comparative analysis that they differ greatly in terms of interface usability, preprocessing capabilities, algorithm implementation, and default parameter settings, which have an effect on performance results. The results consistently demonstrate that no supervised learning algorithm performs better than any other algorithm o n every dataset.

3.7. Comparison Performance of Random Forest and Naïve Bayes using the Weka Tool in the above dataset, as explained in the dataset details section

3.7.1. NAÏVE BAYES

Filter used – Edge Histogram +Color layout Filter

Cross-Validation – 10 Folds

```
Time taken to build model: 0.02 seconds

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

Correctly Classified Instances         121               87.6812 %
Incorrectly Classified Instances        17               12.3188 %
Kappa statistic                          0.8212
Mean absolute error                      0.0642
Root mean squared error                  0.2428
Relative absolute error                 18.3635 %
Root relative squared error             58.1471 %
Total Number of Instances              138

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure  MCC      ROC Area  PRC Area  Class
               0.960     0.114     0.828      0.960     0.889      0.824     0.974     0.948     BUTTERFLY
               0.900     0.023     0.957      0.900     0.928      0.890     0.985     0.975     OWL
               0.750     0.026     0.857      0.750     0.800      0.764     0.976     0.930     FLOWER
               0.714     0.016     0.833      0.714     0.769      0.748     0.937     0.808     HUMAN
Weighted Avg.  0.877     0.056     0.880      0.877     0.875      0.830     0.975     0.940

=== Confusion Matrix ===

  a  b  c  d   <-- classified as
 48  0  1  1 |  a = BUTTERFLY
  3 45  1  1 |  b = OWL
  6  0 18  0 |  c = FLOWER
  1  2  1 10 |  d = HUMAN
```

Fig. 7 shows the Time taken to build the model

```
Classifier output
weka.classifiers.trees.RandomTree -K 0 -M 1.0 -V 0.001 -S 1 -do-not-check-capabilities

Time taken to build model: 0.25 seconds

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

Correctly Classified Instances         121               87.6812 %
Incorrectly Classified Instances        17               12.3188 %
Kappa statistic                          0.8178
Mean absolute error                      0.2074
Root mean squared error                  0.2754
Relative absolute error                 59.3438 %
Root relative squared error             65.9463 %
Total Number of Instances              138

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure  MCC      ROC Area  PRC Area  Class
               0.980     0.136     0.803      0.980     0.883      0.817     0.973     0.953     BUTTERFLY
               0.940     0.057     0.904      0.940     0.922      0.876     0.981     0.974     OWL
               0.708     0.000     1.000      0.708     0.829      0.817     0.985     0.958     FLOWER
               0.571     0.000     1.000      0.571     0.727      0.738     0.891     0.754     HUMAN
Weighted Avg.  0.877     0.070     0.894      0.877     0.872      0.830     0.970     0.942

=== Confusion Matrix ===

  a  b  c  d   <-- classified as
 49  1  0  0 |  a = BUTTERFLY
  3 47  0  0 |  b = OWL
  6  1 17  0 |  c = FLOWER
  3  3  0  8 |  d = HUMAN
```

Fig. 8 shows the Result of the classifier using Random Forest

We are using a Random Forest classifier, Edge Histogram + Color Layout Filter, and Cross-Validation with 10 folds.

4. Conclusion

In this paper, we present the use of different visualization tools such as RapidMiner, Weka, and Orange. Apart from this, machine learning techniques include Classification (Naïve Bayes), linear regression, logistic regression, decision tree, and support vector machine, which are applied to the dataset to analyze the data of the bacterial cells and make a prediction about the Result of the detection of bacterial cells on an agar plate. In the future, researchers can work on this area.

This study used two popular data mining tools, RapidMiner and Weka, to investigate and assess the performance of several supervised machine learning algorithms.

References

[1] Mochammad Faid, Moh Jasri, and Titasari Rahmawati, "Perbandingan Kinerja Tool Data Mining Weka Dan RapidMiner Dalam Algoritma Klasifikasi," *Teknika- Journal of Information and Communication Technology*, vol. 8, no. 1, pp. 11-16, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[2] Ainurrohmah, "Akurasi Algoritma Klasifikasi Pada Software RapidMiner Dan Weka," *PRISMA, Prosiding Seminar Nasional Matematika*, vol. 4, pp. 493-499, 2021. [Google Scholar]

[3] Diego Raphael Amancio et al., "A Systematic Comparison of Supervised Classifiers," *arXiv:1311.0202*, 2013. [CrossRef] [Publisher Link]

[4] Rohit Arora, Suman Suman, "Comparative Analysis of Classification Algorithms on Different Datasets using WEKA," *International Journal of Computer Applications*, vol. 54, no. 13, pp. 21-25, 2012. [CrossRef] [Google Scholar] [Publisher Link]

[5] Kawsar Ahmed, and Tasnuba Jesmin, "Comparative Analysis of Data Mining Classification Algorithms in Type-2 Diabetes Prediction Data using WEKA Approach," *International Journal of Science and Engineering*, vol. 7, no. 2, pp. 155-160, 2014. [CrossRef] [Google Scholar] [Publisher Link]

[6] Madhuri T. Sathe, and Amol C. Adamuthe, "Comparative Study of Supervised Algorithms for Prediction of Students' Performance," *International Journal of Modern Education and Computer Science (IJMECS)*, pp. 1-21, 2021. [CrossRef] [Google Scholar]

[7] N.F. Sulaiman, M.Z. Jali, and K. Zainal, "An Analysis of Various Algorithms for Text Spam Classification and Clustering using RapidMiner and Weka," *International Journal of Computer Science & Information Security (IJCSIS)*, vol. 13, no. 3, 2015. [Google Scholar]

[8] Amrita Naik, and Lilavati Samant, "Correlation Review of Classification Algorithm using Data Mining Tool: WEKA, RapidMiner, Tanagra, Orange and Knime," *Procedia Computer Science*, vol. 85, pp. 662-668, 2016. [CrossRef] [Google Scholar] [Publisher Link]

[9] Ganesan Kavitha, "Comparative Study of Machine Learning Algorithms to Measure the Students' Performance," *International Journal of Computer (IJC)*, vol. 28, no. 1, pp. 143-153, 2018. [CrossRef] [Google Scholar] [Publisher Link]

[10] Sakshi Goel, Neeraj Kumar, and Saharsh Gera, "Comparative Analysis of Classification Algorithms using Weka," *International Journal of Trend in Scientific Research and Development (IJTSRD)*, vol. 6, no. 5, pp. 858-869, 2022. [Publisher Link]

[11] Luis C. Borges, Viriato M. Marques, and Jorge Bernardino, "Comparison of Data Mining Techniques and Tools for Data Classification," *C3S2E'13 – Proceedings of the International C* Conference on Computer Science & Software Engineering*, pp. 113-116, 2013. [CrossRef] [Google Scholar] [Publisher Link]

[12] Ida Moghimipour, and Malihe Ebrahipour, "Comparing Decision Tree Method over Three Data Mining Software," *International Journal of Statistics and Probability*, vol. 3, no. 3, 2014. [CrossRef] [Google Scholar] [Publisher Link]
[13] Rohit Ranjan, Swati Agarwal, and Dr. S. Venkatesan, "Detailed Analysis of Data Mining Tools," *International Journal of Engineering Research & Technology (IJERT)*, vol. 6, no. 5, pp. 785-789, 2017. [Google Scholar] [Publisher Link]
[14] Igiri Chinwe Peace, "An Analytical Review of Data Mining Tools," *International Journal of Engineering Research & Technology (IJERT)*, vol. 4, no. 4, 2015. [Google Scholar] [Publisher Link]
[15] G. Naga Rama Devi, "Comparative Study on Machine Learning Algorithms using WEKA," *International Journal of Engineering Research & Technology (IJERT)*, vol. 2, no. 15, 2014. [CrossRef] [Google Scholar] [Publisher Link]
[16] D.C. Asogwa et al., "Text Classification using Hybrid Machine Learning Algorithms on Big Data," *arXiv:2103.16624*, 2021. [CrossRef] [Google Scholar] [Publisher Link]
[17] M.P. Basgalupp et al., "An Extensive Experimental Evaluation of Automated Machine Learning Methods for Recommending Classification Algorithms," *Evolutionary Intelligence,* vol. 14, pp. 1895-1914, 2021. [CrossRef] [Google Scholar] [Publisher Link]
[18] Mahmudur Rahman Khan et al., "Study and Observation of the Variation of Accuracies of KNN, SVM, LMNN, ENN Algorithms on Eleven Different Datasets from UCI Machine Learning Repository," *arXiv: 1809.06186*, 2018. [CrossRef] [Google Scholar] [Publisher Link]
[19] Qamar Parvez Rana, and Parminder Kaur, "Comparison of Various Tools for Data Mining," *International Journal of Engineering Research & Technology (IJERT)*, vol. 3, no. 10, 2014. [CrossRef] [Publisher Link]
[20] Ajay Kumar, and Preeti Sondhi, "Performance Evaluation of Ensemble Learning Algorithms for Various Classifiers," *Journal of Emerging Technologies and Innovative Research*, vol. 8, no. 10, 2021. [Publisher Link]
[21] Soodeh Hosseini, and Saman Rafiee Sardo, "Data Mining Tools − A Case Study for Network Intrusion Detection," *Multimedia Tools and Applications*, vol. 80, pp. 4999-5019, 2021. [CrossRef] [Google Scholar] [Publisher Link]
[22] Compare RapidMiner vs. Weka, Slashdot, 2025. [Online]. Available: https://slashdot.org/software/comparison/RapidMiner-vs-Weka/
[23] Msin, LibSVM Classification Results Differ between Weka and RapidMiner, ALTAIR only Forward, 2024. [Online]. Available: https://community.altair.com/discussion/59512/libsvm-classification-results-differ-between-weka-and-rapidminer?tab=all
[24] Alexander Craik, Yongtian He, and Jose L. Contreras-Vidal, "Deep Learning for Electroencephalogram (EEG) Classification Tasks: A Review," *Journal of Neural Engineering*, vol. 16, no. 3, 2009. [CrossRef] [Google Scholar] [Publisher Link]
[25] Handwritten Digit Recognition, Nearest Neighbour Classifier. [Online]. Available: https://www.robots.ox.ac.uk/~dclaus/digits/neighbour.htm
[26] P. Keerthana, B.G. Geetha, P. Kanmani, "Crustose Using Shape Features and Color Histogram with K Nearest Neighbor Classifiers," *International Journal of Innovations in Scientific and Engineering Research (IJISER)*, vol. 4, no. 9, pp. 199-203, 2017. [Google Scholar]
[27] M. Narashimha Murty, and V. Susheela Devi, "Nearest Neighbour Based Classifiers," *Pattern Recognition*, pp. 48-85, 2011. [CrossRef] [Google Scholar] [Publisher Link]
[28] Guo-Xun Yuan, Chia-Hua Ho, and Chih-Jen Lin, "Recent Advances of Large-Scale Linear Classification," *Proceedings of the IEEE*, vol. 100, no. 9, pp. 2584-2603, 2012. [CrossRef] [Google Scholar] [Publisher Link]
[29] Henrik Madsen, and Poul Thyregod, *Introduction to General and Generalized Linear Models*, Chapman & Hall/CRC, 2010. [CrossRef] [Google Scholar] [Publisher Link]
[30] Altair Rapid Miner Empowers Organizations. [Online]. Available: https://altair.com/altair-rapidminer
[31] Kamran Kowsari et al., "Hdltex: Hierarchical Deep Learning for Text Classification," *2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA)*, 2017. [CrossRef] [Google Scholar] [Publisher Link]
[32] Xiaodong Liu et al., "Stochastic Answer Networks for Machine Reading Comprehension," *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, pp. 1694-1704, 2018. [CrossRef] [Google Scholar] [Publisher Link]
[33] Rupesh Srivastava, Klaus Greff, and Jurgen Schmidhuber, "Training Very Deep Networks," *Advances in Neural Information Processing Systems*, 2015. [Google Scholar] [Publisher Link]
[34] Kaiming He et al., "Deep Residual Learning for Image Recognition," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 770–778, 2016. [Google Scholar] [Publisher Link]
[35] Yoon Kim et al., "Character-Aware Neural Language Models," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 30, no. 1, 2016. [CrossRef] [Google Scholar] [Publisher Link]
[36] Julian Georg Zilly et al., "Recurrent Highway Networks," *Proceedings of the 34th International Conference on Machine Learning*, pp. 4189-4198, 2017. [Google Scholar] [Publisher Link]
[37] Ying Wen et al., "Learning Text Representation using Recurrent Convolutional Neural Network with Highway Layers," *arXiv preprint arXiv:1606.06905*, 2016. [CrossRef] [Google Scholar] [Publisher Link]
[38] Ronan Collobert et al., "Natural Language Processing (Almost) from Scratch," *Journal of Machine Learning Research*, vol. 12, pp. 2493–2537, 2011. [Google Scholar] [Publisher Link]

[39] Alec Radford et al., "Language Models are Unsupervised Multitask Learners," *OpenAI Blog*, vol. 1, no. 8, 2019. [Google Scholar] [Publisher Link]

[40] XiPeng Qiu et al., "Pre-trained Models for Natural Language Processing: A Survey," *Science China Technological Sciences*, vol. 63, pp. 1872-1897, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[41] Yinhan Liu et al., "Roberta: A Robustly Optimized BERT Pretraining Approach," *arXiv preprint arXiv:1907.11692*, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[42] Zhenzhong Lan et al., "Albert: A Lite BERT for Self-supervised Learning of Language Representations," *arXiv preprint arXiv:1909.11942*, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[43] Victor Sanh et al., "DistilBERT, A Distilled Version of BERT: Smaller, Faster, Cheaper and Lighter," *arXiv preprint arXiv:1910.01108*, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[44] Mandar Joshi et al., "Spanbert: Improving Pre-training by Representing and Predicting Spans," *Transactions of the Association for Computational Linguistics*, vol. 8, pp. 64-77, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[45] Kevin Clark et al., "Electra: Pre-training text Encoders as Discriminators Rather Than Generators," *arXiv preprint arXiv:2003.10555*, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[46] Yu Sun et al., "Ernie: Enhanced Representation through Knowledge Integration," *arXiv preprint arXiv:1904.09223*, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[47] Yu Sun et al., "Ernie 2.0: A Continual Pre-training Framework for Language Understanding," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 5, pp. 8968–8975, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[48] Siddhant Garg, Thuy Vu, and Alessandro Moschitti, "TANDA: Transfer and Adapt Pre-trained Transformer Models for Answer Sentence Selection," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 5, pp. 7780-7788, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[49] Chi Sun et al., "How to Fine-tune BERT for Text Classification?," *Chinese Computational Linguistics*, pp. 194-206, 2019. [CrossRef] [Google Scholar] [Publisher Link]