Experiment 8

Title: Demonstration on "J48", "RandomForest" and "NaiveBayes" classification algorithms using test options.

1. What is "J48" algorithm?
J48 is WEKA's implementation of the C4.5 decision tree algorithm. It builds a tree by selecting attributes that best split the data using information gain or gain ratio, then prunes it to reduce overfitting. It can handle both numerical and categorical data and is easy to interpret.

2. What is "RandomForest" algorithm?
RandomForest is an ensemble method that builds multiple decision trees on random subsets of the data and combines their predictions through majority voting. It reduces overfitting, works well with large datasets, and handles missing values effectively.

3. What is "NaiveBayes" algorithm?
NaiveBayes is a simple probabilistic classifier based on Bayes' Theorem, assuming independence between features. It is fast, works well with large datasets, and is effective for text classification despite the independence assumption often being unrealistic.

8.1 Importance and purpose of Training Dataset/Data for Classification/Prediction.

The training dataset is the foundation for building classification or prediction models. It contains labeled examples (input features with known outputs) that allow the algorithm to learn patterns, relationships, and decision boundaries in the data. The purpose of training data is to teach the model how to make accurate predictions on new, unseen data by minimizing errors during the learning process. High-quality, diverse, and representative training data ensures better model accuracy, generalization, and reliability, while poor or biased data can lead to incorrect predictions and reduced performance.

8.2 Importance and purpose of Training Dataset/Data for Classification/ Prediction.

A training dataset is essential for teaching a classification or prediction model to recognize patterns and relationships in data. It provides labeled examples that the algorithm uses to learn decision rules, enabling accurate predictions on new, unseen data. Good-quality and representative training data improves accuracy, generalization, and reliability, while poor-quality data can lead to biased or incorrect results.

8.3 Importance and purpose of Test Dataset/Data for Classification/ Prediction.

A test dataset is used to evaluate the performance of a classification or prediction model after it has been trained. It contains unseen data (often with known labels) that checks how well the model generalizes beyond the training data. The purpose of the test dataset is to measure accuracy, detect overfitting, and ensure the model can make reliable predictions in real-world scenarios.

8.4 Importance and purpose of Model for Classification/Prediction.

A model for classification or prediction is the core outcome of the training process, representing the learned patterns and decision rules from the data. Its importance lies in enabling automated, consistent, and efficient decision-making without manual intervention. The purpose of the model is to apply this learned knowledge to classify new data points or predict future outcomes accurately, ensuring the results are reliable, scalable, and applicable to real-world problems.

8.5 Process of Creating Classification/Prediction Model & Using the model on Test Dataset/Data.

The process begins with collecting and preparing a labeled dataset, followed by splitting it into training and test sets. The training dataset is used to train a chosen algorithm (e.g., J48, RandomForest, NaiveBayes) so the model can learn patterns and decision rules. Once trained, the model is applied to the test dataset, which contains unseen examples, to evaluate its accuracy and performance. This step checks the model's ability to generalize and make correct predictions in real-world scenarios. Based on test results, the model may be fine-tuned to improve accuracy before deployment.

8.6 Test Options with its importance, way of working for Classification, how to use it?

In classification, test options define how a model's performance is evaluated in tools like WEKA. They are important because they determine the reliability and generalization ability of the model. Common test options include:

1. Use Training Set – Tests the model on the same data it was trained on; useful for quick checks but may cause overfitting.
2. Supplied Test Set – Uses a separate dataset provided by the user; good for independent evaluation.
3. Cross-validation – Splits data into *k* folds, trains on *k-1* folds, and tests on the remaining fold repeatedly; provides a more reliable accuracy estimate.
4. Percentage Split – Splits data into a fixed percentage for training (e.g., 70%) and the rest for testing; simple and quick for performance evaluation.

Way of Working:
Each option decides how the training and testing data are chosen, ensuring the model's performance is measured in different ways. Cross-validation gives a balanced accuracy estimate, percentage split offers simplicity, supplied test set checks on truly unseen data, and training set evaluation is fast but prone to bias.
How to Use:
In WEKA, after selecting the classification algorithm, choose a test option in the "Test Options" panel, set its parameters (like *k* for cross-validation or percentage for split), then run the model to view results such as accuracy, confusion matrix, and error rates.

8.7 Details about J48, Random Forest, Naïve Bayes (with algorithm, way of working etc.)

1. J48

- Overview:
  J48 is WEKA's implementation of the C4.5 decision tree algorithm. It creates a model in the form of a tree structure where each internal node tests an attribute, each branch represents an outcome of the test, and each leaf node represents a class label.
- Algorithm Steps:
  1. Select the attribute with the highest information gain or gain ratio.
  2. Split the dataset into subsets based on this attribute.
  3. Repeat the process recursively for each subset until all records belong to the same class or stopping criteria is met.
  4. Apply pruning to remove branches that do not improve classification accuracy.
- Way of Working:
  J48 learns rules from the training data to classify new instances. When classifying, the model follows the decision tree from the root to a leaf node based on attribute values in the input record.
- Key Features:
  Handles both categorical and numeric attributes, can manage missing values, and provides human-readable rules.

2. RandomForest

- Overview:
  RandomForest is an ensemble learning method that builds multiple decision trees and combines their predictions.

- Algorithm Steps:
    1. Generate multiple bootstrap samples (random subsets) from the training dataset.
    2. For each subset, grow a decision tree, but at each split, consider only a random subset of attributes.
    3. For classification, each tree votes for a class; the majority vote is the final prediction.
- Way of Working:
  By combining many decision trees trained on different data subsets and feature sets, RandomForest reduces overfitting and increases prediction accuracy.
- Key Features:
  Robust to noise, works well with large datasets, can handle missing data, and gives feature importance scores.

3. NaïveBayes

- Overview:
  NaïveBayes is a probabilistic classifier based on Bayes' Theorem with a strong assumption that all features are independent given the class label.
- Algorithm Steps:
    1. Calculate the prior probability for each class from the training data.
    2. Calculate the likelihood probability of each feature value given each class.
    3. Apply Bayes' Theorem to compute the posterior probability for each class given the input features.
    4. Choose the class with the highest posterior probability.
- Way of Working:
  The model uses probabilities learned from the training set to predict the most probable class for new instances.
- Key Features:
  Simple, fast, requires small training data, works well with text and categorical data, and performs surprisingly well even when independence assumption is not perfectly true.

8.8 Algorithm wise Details about all the Parameters of each algorithm.

1. J48 Parameters (WEKA)

- confidenceFactor – Controls pruning amount (default 0.25; lower = more pruning).
- minNumObj – Minimum number of instances per leaf node.
- unpruned – If true, creates an unpruned tree.
- reducedErrorPruning – Enables reduced-error pruning.
- numFolds – Folds for reduced-error pruning.
- binarySplits – Forces binary splits on nominal attributes.
- saveInstanceData – Stores instance data at each node.

2. RandomForest Parameters (WEKA)

- numTrees – Number of trees to build (default 100).
- maxDepth – Maximum depth of each tree (0 = unlimited).
- numFeatures – Number of features considered at each split.
- seed – Random seed for reproducibility.
- maxDepth – Limit depth to avoid overfitting.
- breakTiesRandomly – Breaks ties randomly between equally good splits.

3. NaïveBayes Parameters (WEKA)

- useKernelEstimator – Uses kernel density estimation for numeric attributes.
- useSupervisedDiscretization – Discretizes numeric attributes before building model.
- displayModelInOldFormat – Displays the model in old WEKA output format.

8.9 Dataset selection.

Dataset selection is the process of choosing an appropriate and relevant dataset for building and evaluating a classification or prediction model. The dataset should contain sufficient instances, relevant attributes, and accurate labels to represent the real-world problem being solved. A good dataset is balanced (equal representation of classes), clean (minimal missing or noisy data), and representative of the domain to ensure the model can generalize well. In tools like WEKA, dataset selection is done by importing data files (e.g., .arff, .csv) through the "Open file" option, ensuring that the chosen dataset matches the intended classification or prediction task.

8.10 Applying J48 on dataset using all 4 test options with result analysis (along with "visualize tree", confusion matrix etc. – all aspects of results).

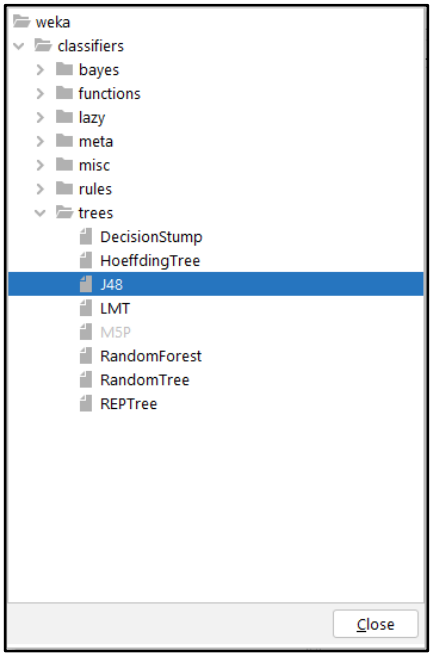

Fig 8.1[Choose algorithm]

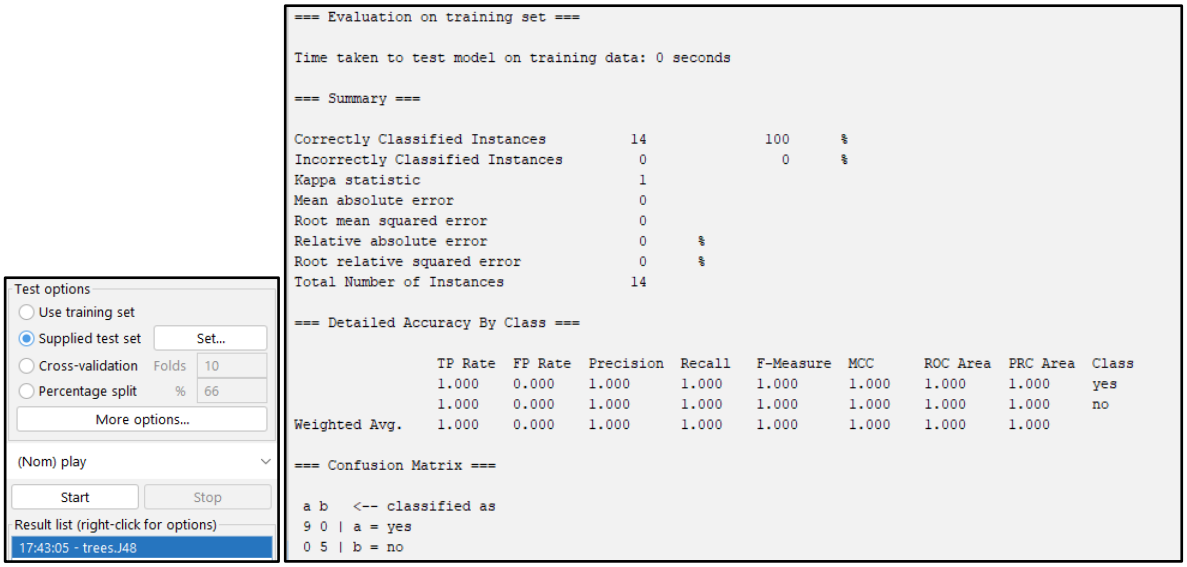

Fig 8.2[Test option 1: Use training set]

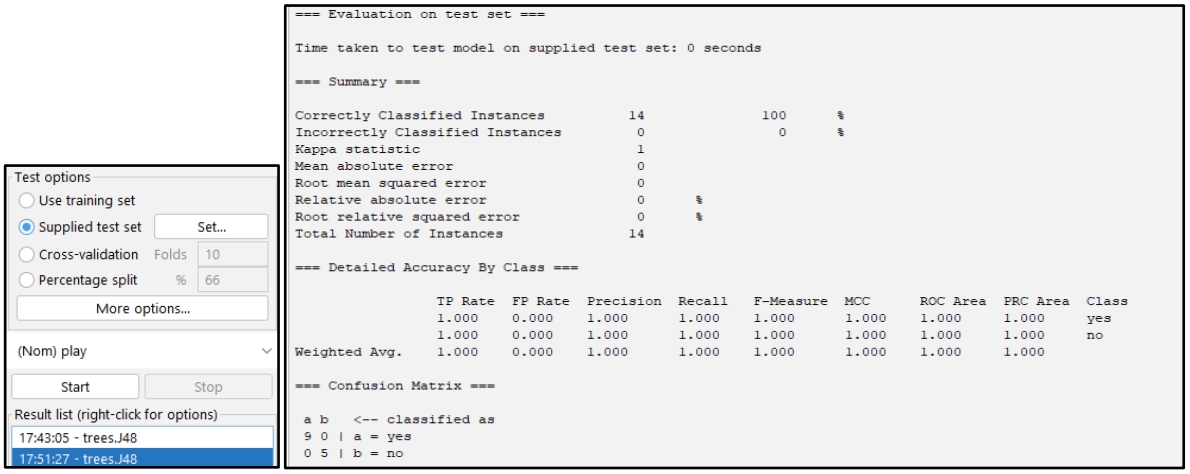

Fig 8.3[Test option 2: Supplied test set]

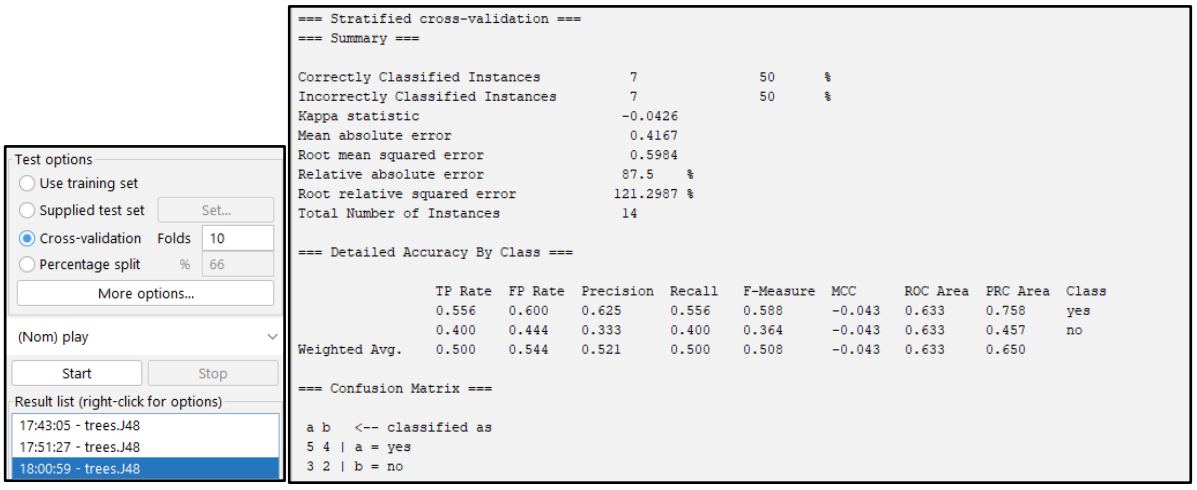

Fig 8.4[Test option 3: Cross-validation]

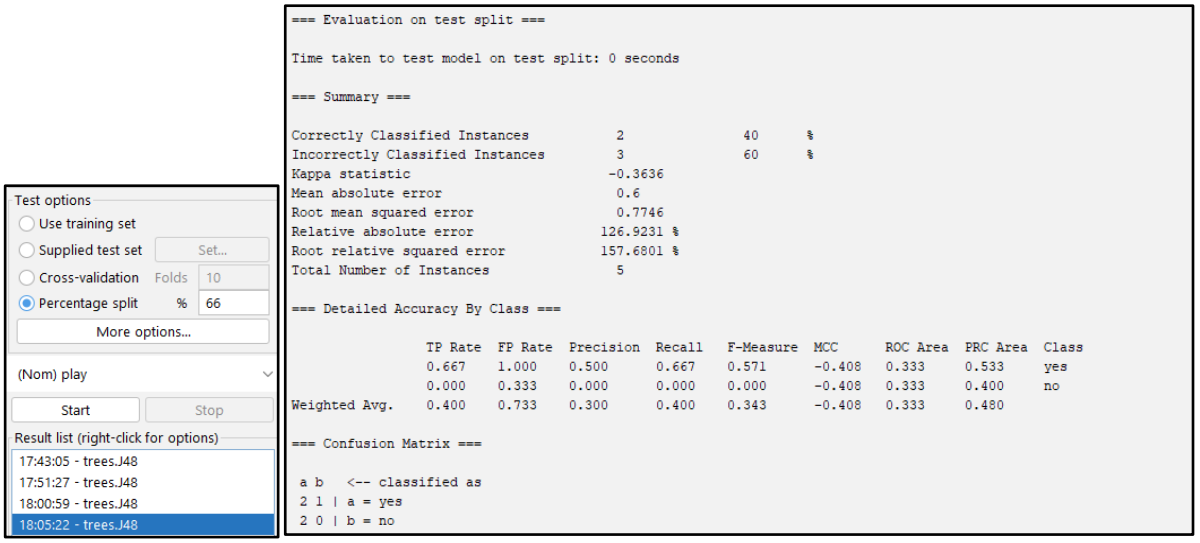

Fig 8.5[Test option 4: Percentage split]

8.11 Applying Random Forest on dataset using all 4 test options with result analysis (along with "visualize tree", confusion matrix etc. – all aspects of results).

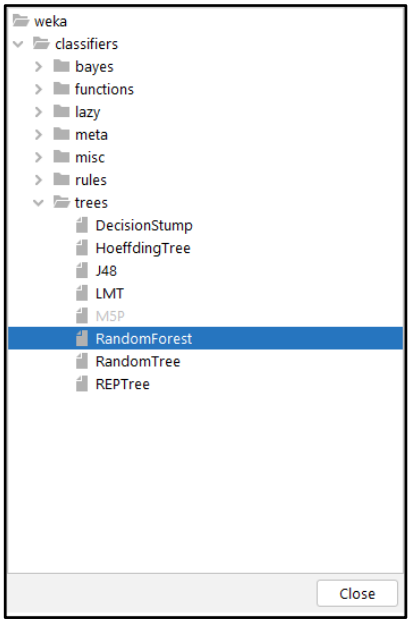

Fig 8.6[Choose algorithm]

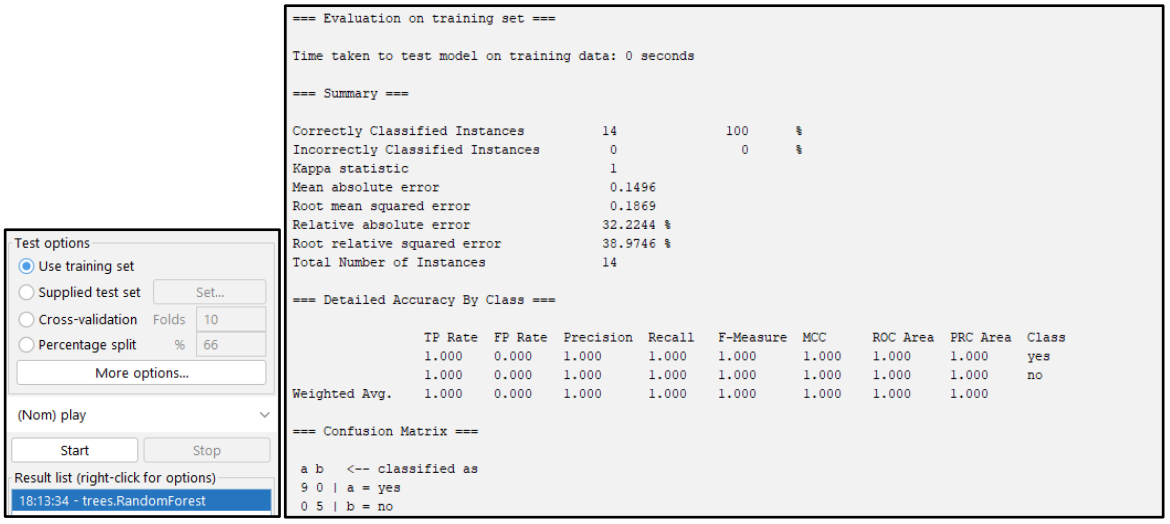

Fig 8.7[Test option 1: Use training set]

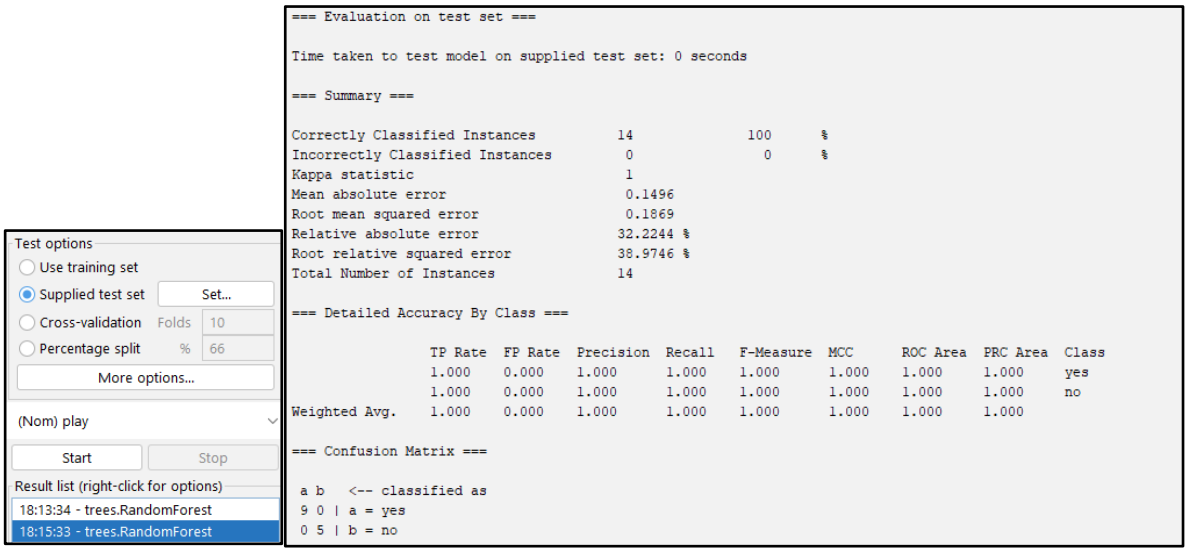

Fig 8.8[Test option 2: Supplied test set]

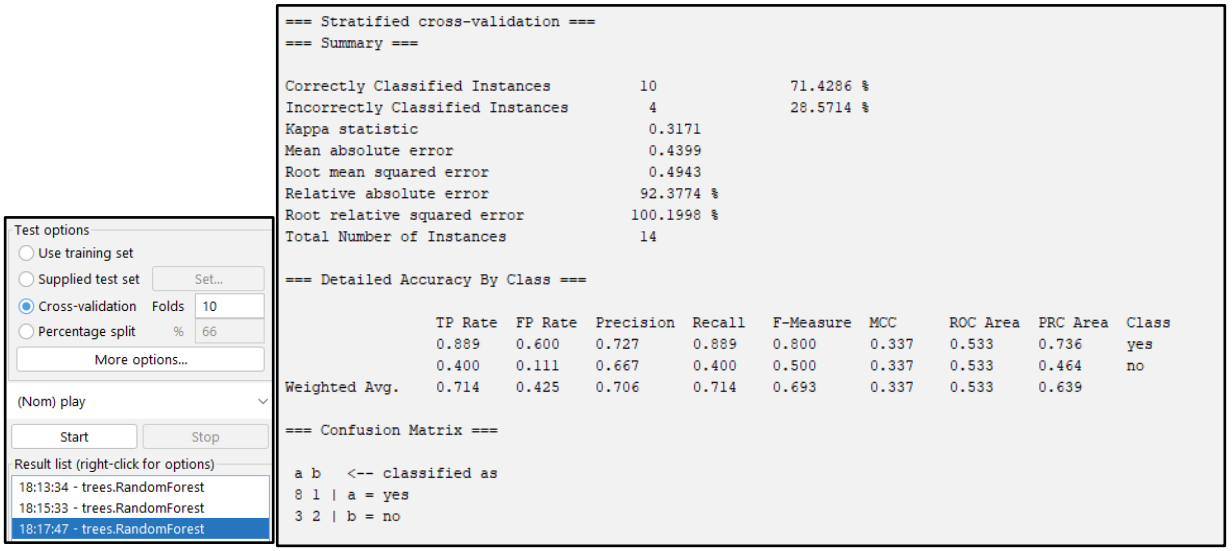

Fig 8.9[Test option 3: Cross-validation]

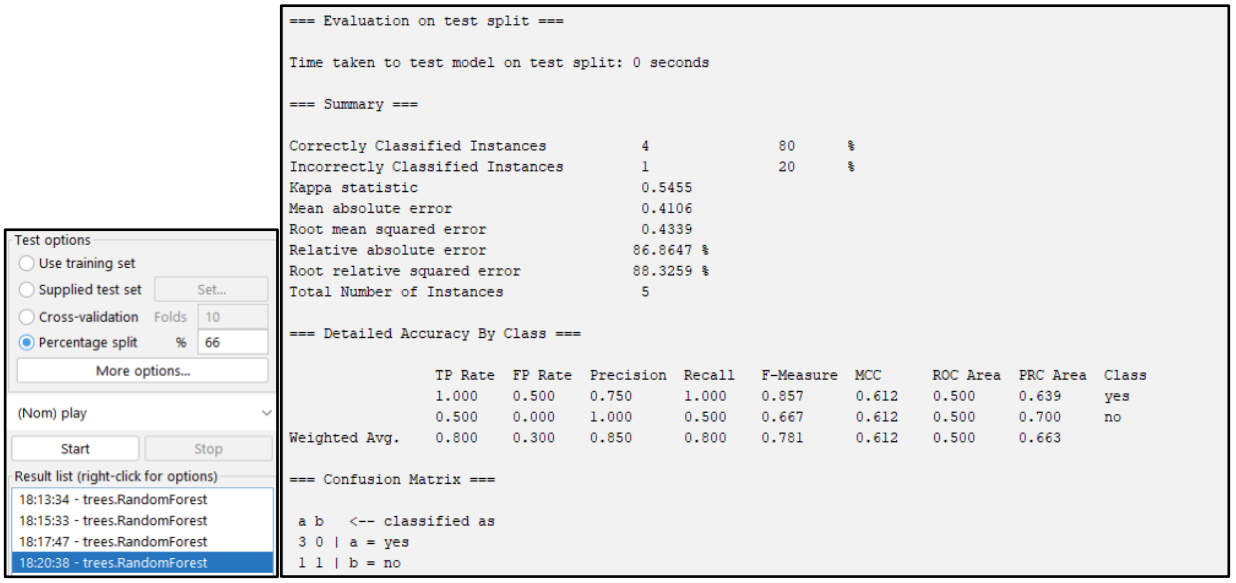

Fig 8.10[Test option 4: Percentage split]

8.12 Applying Naive Bayes on dataset using all 4 test options with result analysis (along with confusion matrix etc. – all aspects of results).

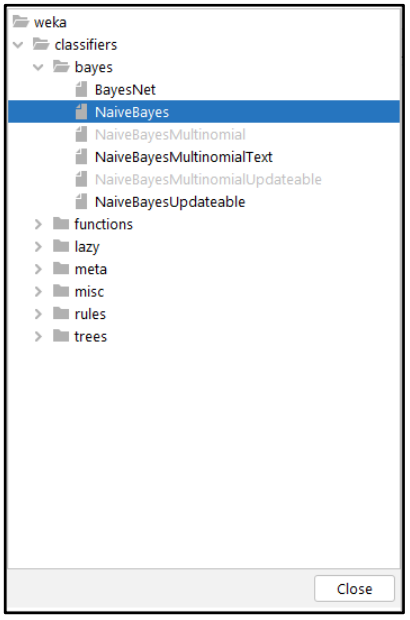

Fig 8.11[Choose algorithm]

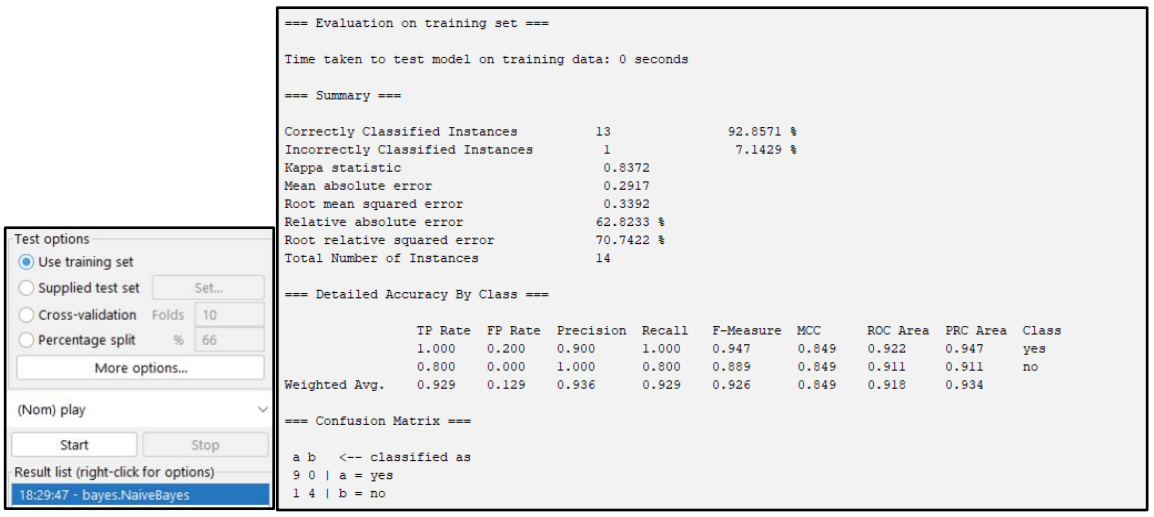

Fig 8.12[Test option 1: Use training set]

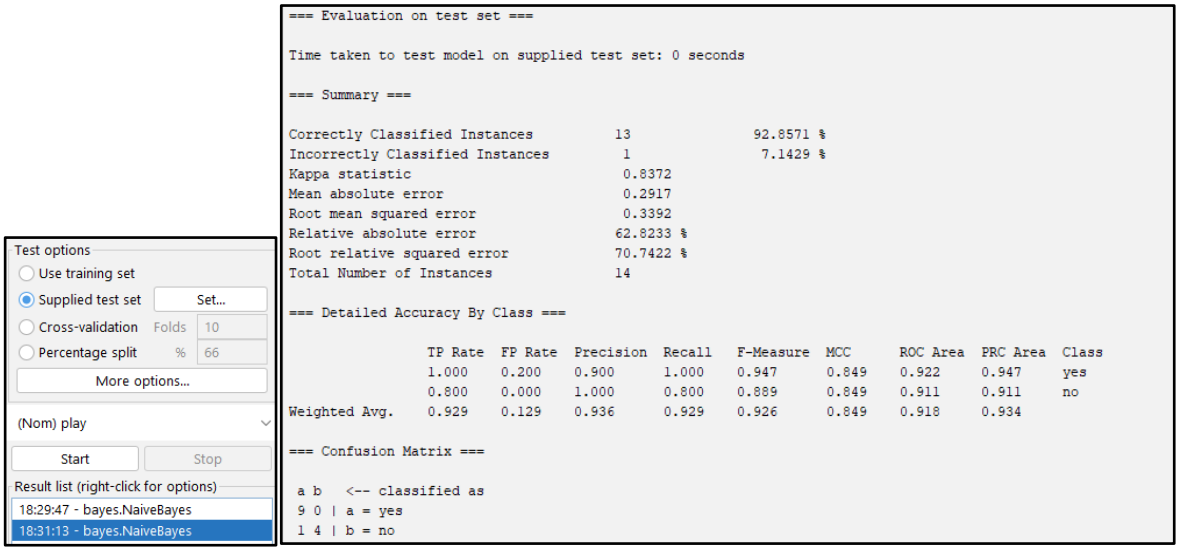

Fig 8.13[Test option 2: Supplied test set]

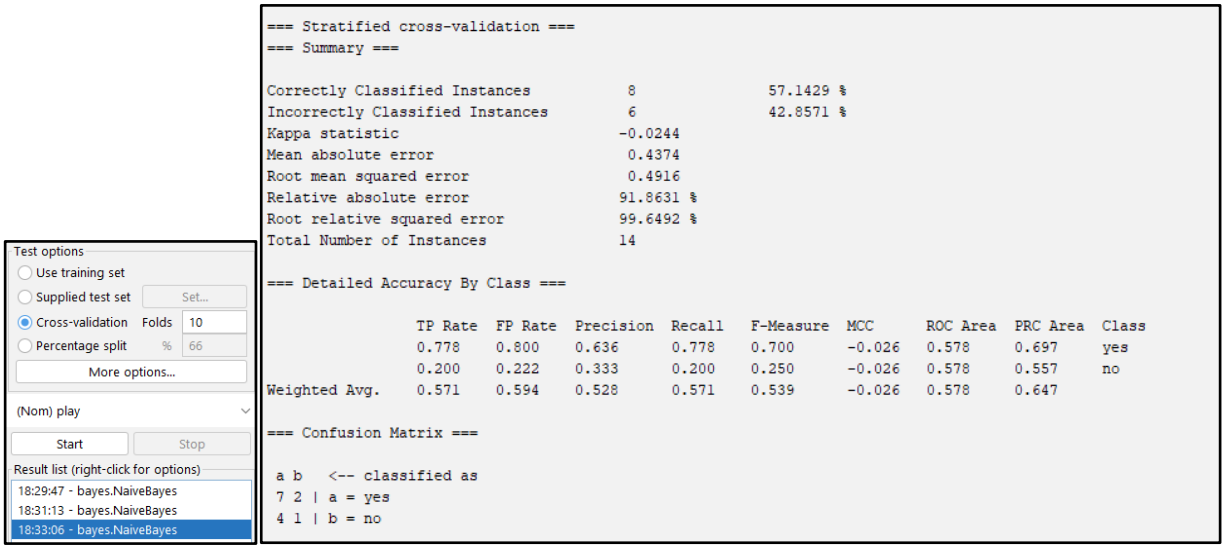

Fig 8.14[Test option 3: Cross-validation]

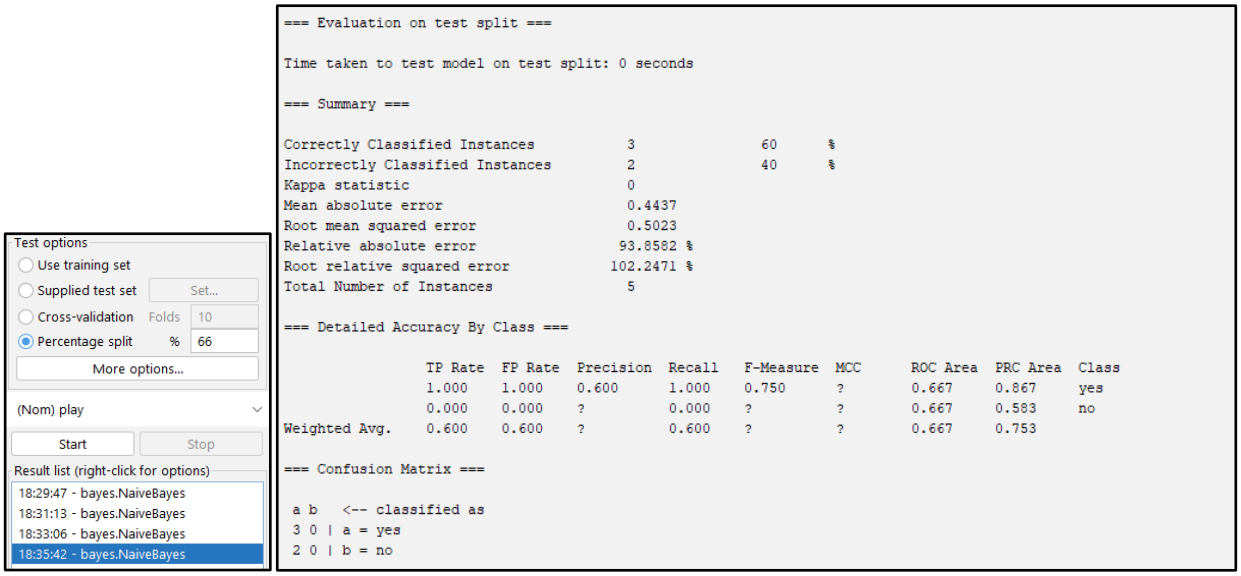

Fig 8.15[Test option 4: Percentage split]

- **Experiment Outcome**

  The experiments using J48, RandomForest, and NaïveBayes classification algorithms with all four test options (Use Training Set, Supplied Test Set, Cross-Validation, and Percentage Split) showed that model performance varies depending on the algorithm and evaluation method.

  - J48 provided high accuracy on the training set but showed reduced performance on unseen data, indicating a tendency toward overfitting without pruning.
  - RandomForest consistently delivered high accuracy across different test options due to its ensemble nature, with better generalization compared to a single decision tree.
  - NaïveBayes performed well in cross-validation and percentage split, especially on categorical and text-like datasets, though its independence assumption limited accuracy in some cases.

  Overall, cross-validation emerged as the most reliable evaluation method for unbiased performance estimation, while RandomForest generally produced the best accuracy and robustness among the tested algorithms.