Universitat Politècnica de Catalunya

Supervised and experiential learning

Combining multiple classifiers

Author:

Hanna Lizarzaburu Aguilar

Professor:

Miquel Sànchez Marrè

Spring Term 2022/2023

UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH

Contents

1 Introduction

The goal of this project is to implement the Random Forest and Decision Forest algorithms, which are based on the Decision Tree algorithm. We will vary hyperparameters, such as the number of trees (NT) and the number of random features (F), to see their impact on the performance of each algorithm.

Random Forest is a bagging algorithm that performs resampling with replacement for each tree and uses a number of random features for the splitting of nodes, making it quite robust in handling overfitting. On the other hand, Decision Forest uses the same training dataset (no bootstrap) to train the different trees and a random number of features for each tree.

The implementation will be done entirely in Python 3.10, following the rules of clean code.

2 Datasets

The datasets used are: iris, titanic, glass, wine-red and wine-white. We decided to use 5 datasets instead of 3 to delve deeper into the comparison of the algorithms. A summary of the main variables for each dataset is described in Table 1.

	Total instances	**N. Continuous Attr.**	**N. Discrete Attr.**	**N. Total Attr.**	**N. Classes**
iris	150	4	0	4	3
titanic	1,309	4	3	7	2
glass	214	9	0	9	6
wine-red	1,599	11	0	11	6
wine-white	4,898	11	0	11	7

Table 1: Summary of the datasets used.

In these data sets they do not contain nan values so there was no preprocessing in this regard. The class (y) for each dataset were encoded as integers. For example, the class "survived" from the titanic dataset was transformed into the integers 0 and 1.

3 Ensemble Classifiers

3.1 Overview

- **Decision Tree [3]**: in this work, the CART method (Classification and Regression Trees) was used. This method is based on the recursive division of a training data set into branches using the predictive feature that maximizes the reduction in Gini impurity. Gini impurity is calculated as the weighted sum of the squared probabilities of each class. Once the tree is built, it can be used to predict the class or response value of new test data.

$$Gini(X) = 1 - \sum_{i=1}^{k} (p_x)^2, x \in c_i \tag{1}$$

Formula 1: Loss function: Gini index

- **Random Forest** [2]: it is based on the construction of multiple decision trees randomly using bootstrap samples of the training data set and a random subset of features used in the splitting of the nodes. In classification, each decision tree in the Random Forest makes a prediction, and the class with the highest frequency is selected as the final prediction. The importance of each feature is measured by the average decrease in Gini impurity.
- **Decision Forest** [1]: is similar to Random Forest in that it builds multiple random decision trees. However, in Decision Forest, it does not perform a bootstrap on the data set, but instead takes random subsets of features in the construction of each tree. This makes each tree have the same number of features..

3.2 Implementation

The implementation of the algorithm was created using Python 3.10 and additional packages such as numpy, scikit-learn, and pandas. The required packages to run the code are listed in the requirements.txt file. The code was designed with the principles of clean code in mind.

In the src folder, there are four main scripts that summarize the entire code of the project as follows:

- decision_tree.py: this is the main module for the decision tree classifier, where you can train and predict decision trees. This module uses numpy, dataclasses, and enum libraries.
  - `Class DecisionTree`: is the main class for performing decision tree classification. It has several hyperparameters such as random_state, max_depth, min_samples_split, F, and random_subspace_node. The fit() method is used to train the decision tree using input data X and output data y. Once the tree is trained, the predict() method can be used to classify new data.
  - `Class Node`: it represents nodes in a decision tree and has many attributes such as depth, feature_id, gini value, among others.
  - `Class Operator`: setting operators if the feature is categorical or continuous.

  The pseudo-code for the Decision Tree algorithm is detailed in Algorithm 1 and Algorithm 2.

Algorithm 1 Pseudo-code for Decision Tree. build_tree

```
Procedure of build_tree ( Samples S , Target T ):
if depth > max_depth or n_samples < min_n_samples then
    Return NodeClass()
end if
Let best_gini_gain, best_feature, best_threshold be Best_Split ( S , T )
if  best_feature is empty or best_threshold is empty then
    Return NodeClass()
end if
Let left_data be S <= best_feature
Let right_data be S > best_feature
Let left_tree be build_tree(left_data, right_data, depth + 1)
Let right_tree be build_tree(left_data, right_data, depth + 1)
Let Tree be NodeClass(depth, left_tree, right_tree)
Return Tree
```

Algorithm 2 Pseudo-code for Decision Tree. best_split

```
Procedure of best_split( S , T):
Let best_gini_gain be 0
Let best_feature be None
Let best_threshold be None
For each feature in features:
    Set thresholds as unique values of feature
    For each threshold in thresholds:
        Split the data into left and right side using threshold
        Set left as S <= threshold
        Set ritght as S > threshold
        Calculate the gini gain with left and right of T
        If the gini gain is better than the best_gini_gain then
            Replace the current of best_feature and best_threshold
        with the feature and threshold respectively.
        EndIf
Return best_gini_gain, best_feature and best_threshold
```

- forest_ensemble.py: implements the RandomForest and DecisionForest classes for classification problems. The Feature Importance is estimated by the frequency of its appearance in the constructed Random Forest or Decision Forest.
  - `Class RandomForest`: it has several hyperparameters such as the number of trees to use in the forest, the maximum depth of each tree, and the minimum number of samples required to split a node. It also allows for bootstrapping and random feature selection. The class uses a DecisionTree class as the base learner and trains each tree using a bootstrap sample of the data. The predict method returns the mode of the predictions from all trees in the forest. Additionally, the class includes a property method to determine the number of random features to use at each split of the decision trees.

  - `Class DecisionForest`: it has the same attributes and methods as the previous class except that it does not implement bootstrapping and has a method to randomly choose the features for each tree.

The pseudo-code for the RandomForest and the DecisionForest classes can be seen in Algorithm 3 and Algorithm 4.

Algorithm 3 Pseudo-code for Random Forest

```
Procedure of RandomForest(Samples S, Target T):
Let Random_Subspace_Node be True
Set trees as list of Trees calcukate using DecisionTree(S, T)
For tree in trees:
    Let new_s and new_s be Make_Bootstrap(S, T)
    Set tree.fit(new_s, new_s) as a trained_tree
    Set feature_importance as trained_tree.feature_importance
    Add the trained_tree to the trained_tree_list
    Add the feature_importance to the feature_importance_list
Return trained_tree_list and feature_importance_list
```

Algorithm 4 Pseudo-code for Decision Forest

```
Procedure of best_split( S , T):
Let best_gini_gain be 0
Let best_feature be None
Let best_threshold be None
For each feature in features:
    Set thresholds as unique values of feature
    For each threshold in thresholds:
        Split the data into left and right side using threshold
        Set left as S <= threshold
        Set ritght as S > threshold
        Calculate the gini gain with left and right of T
        If gini gain is better than the best_gini_gain then
            Replace the current of best_feature and best_threshold
        with the feature and threshold respectively.
        EndIf
Return best_gini_gain, best_feature and best_threshold
```

- forest_tools.py: provides useful classes and functions to be used in the RandomForest and DecisionForests classes.

- main.py: contains functions and classes for running experiments with Random Forest and Decision Forest algorithms on different datasets.

  - `function print_verbose`: prints a dictionary in a verbose format.
  - `function run_experiment`: runs an experiment using the Experiment class to test the accuracy of the Random Forest or Decision Forest algorithms.

  - `function forest_interprete`: runs a series of experiments with Random Forest and Decision Forest algorithms with different hyperparameters on the given train and test data, and returns a dataframe with the results.
  - `Class Dataset`: main class for handling datasets.
  - `Class Experiment`: class for experimenting with Random Forest and Decision Forest algorithms.
- analysis.py: contains functions to plot and analyze the results related for all datasets.

Implementing the code has been a great challenge due to the large amount of lines we had to write. To ensure that our algorithms work correctly, we created a testing section.We conducted two types of tests: one using the dataset seen in class, Human Identification (although only with 9 instances), to verify if our code provided the same results for Gini index, best split, and final tree; and another one where we compared the test accuracy of the DecisionTree, RandomForest, and DecisionForest classes from the sklearn library with that of our algorithm.

The code can be found in the "tests" directory and instructions on how to run it are outlined in the project's README.md file.

4 Results

The data were split into a training dataset (75%) and a testing dataset (25%). The training dataset was used to fit the algorithms, which was then applied to the testing dataset to obtain accuracy results.

A summary table for each dataset is shown below, containing the type of the algorithm, the number of tree, number of random features, train accuracy, test accuracy, and feature importance, ordered according to their importance, displaying the feature name and its corresponding value.

We present an average of the train and test accuracy for all cases. But It is important to note that these values are only a reference of the ranges in which accuracies are found. The aim is to evaluate the impact of the NT and F hyperparameters, as well as the type of algorithm used. To aid in this process, we have prepared some graphs which will be presented in the following section and will help facilitate the analysis to be performed.

4.1 Iris dataset

The training process took 44.66 seconds and the resulting accuracy averaged across all cases was 0.983 $\pm$0.060. The model was evaluated on the test dataset, resulting in a test accuracy of 0.933 $\pm$0.073.

The results are shown in Table 2 below[1].

[1]All the results can be found both in the tables of this report and in the directory *data/processed* in csv format

algorithm	n_trees	F	train_acc	test_acc	feature_importance
random forest	1	1	0.929	0.816	”[(’petal width (cm)’, 0.503), (’sepal width (cm)’, 0.222), (’sepal length (cm)’, 0.212), (’petal length (cm)’, 0.062)]”
random forest	1	2	0.982	0.868	”[(’petal length (cm)’, 0.625), (’sepal length (cm)’, 0.292), (’petal width (cm)’, 0.083), (’sepal width (cm)’, 0.0)]”
random forest	1	FMethodRF.LOG	0.955	0.895	”[(’petal width (cm)’, 0.5), (’sepal length (cm)’, 0.25), (’petal length (cm)’, 0.208), (’sepal width (cm)’, 0.042)]”
random forest	1	FMethodRF.SQRT	0.973	0.868	”[(’petal length (cm)’, 0.5), (’sepal length (cm)’, 0.333), (’petal width (cm)’, 0.111), (’sepal width (cm)’, 0.056)]”
random forest	10	1	0.982	0.974	”[(’sepal width (cm)’, 0.386), (’petal length (cm)’, 0.294), (’petal width (cm)’, 0.169), (’sepal length (cm)’, 0.151)]”
random forest	10	2	1.0	0.921	”[(’petal length (cm)’, 0.459), (’petal width (cm)’, 0.304), (’sepal length (cm)’, 0.156), (’sepal width (cm)’, 0.081)]”
random forest	10	FMethodRF.LOG	1.0	0.974	”[(’petal width (cm)’, 0.542), (’petal length (cm)’, 0.341), (’sepal length (cm)’, 0.081), (’sepal width (cm)’, 0.036)]”
random forest	10	FMethodRF.SQRT	1.0	0.921	”[(’petal length (cm)’, 0.452), (’petal width (cm)’, 0.35), (’sepal length (cm)’, 0.141), (’sepal width (cm)’, 0.057)]”
random forest	25	1	1.0	0.921	”[(’petal length (cm)’, 0.388), (’sepal width (cm)’, 0.214), (’petal width (cm)’, 0.2), (’sepal length (cm)’, 0.198)]”
random forest	25	2	1.0	0.974	”[(’petal length (cm)’, 0.411), (’petal width (cm)’, 0.358), (’sepal length (cm)’, 0.174), (’sepal width (cm)’, 0.057)]”
random forest	25	FMethodRF.LOG	1.0	0.974	”[(’petal width (cm)’, 0.449), (’petal length (cm)’, 0.343), (’sepal length (cm)’, 0.166), (’sepal width (cm)’, 0.042)]”
random forest	25	FMethodRF.SQRT	1.0	0.974	”[(’petal length (cm)’, 0.439), (’petal width (cm)’, 0.396), (’sepal length (cm)’, 0.101), (’sepal width (cm)’, 0.065)]”
random forest	50	1	1.0	0.974	”[(’petal length (cm)’, 0.297), (’sepal length (cm)’, 0.249), (’petal width (cm)’, 0.237), (’sepal width (cm)’, 0.217)]”
random forest	50	2	1.0	0.974	”[(’petal length (cm)’, 0.436), (’petal width (cm)’, 0.375), (’sepal length (cm)’, 0.132), (’sepal width (cm)’, 0.057)]”
random forest	50	FMethodRF.LOG	1.0	0.974	”[(’petal width (cm)’, 0.426), (’petal length (cm)’, 0.385), (’sepal length (cm)’, 0.139), (’sepal width (cm)’, 0.049)]”
random forest	50	FMethodRF.SQRT	1.0	0.974	”[(’petal length (cm)’, 0.397), (’petal width (cm)’, 0.379), (’sepal length (cm)’, 0.163), (’sepal width (cm)’, 0.06)]”
random forest	75	1	1.0	0.974	”[(’petal length (cm)’, 0.307), (’petal width (cm)’, 0.278), (’sepal length (cm)’, 0.219), (’sepal width (cm)’, 0.196)]”
random forest	75	2	1.0	0.921	”[(’petal length (cm)’, 0.477), (’petal width (cm)’, 0.342), (’sepal length (cm)’, 0.123), (’sepal width (cm)’, 0.058)]”
random forest	75	FMethodRF.LOG	1.0	0.921	”[(’petal length (cm)’, 0.483), (’petal width (cm)’, 0.323), (’sepal length (cm)’, 0.144), (’sepal width (cm)’, 0.051)]”
random forest	75	FMethodRF.SQRT	1.0	0.974	”[(’petal length (cm)’, 0.433), (’petal width (cm)’, 0.377), (’sepal length (cm)’, 0.141), (’sepal width (cm)’, 0.049)]”
random forest	100	1	1.0	0.974	”[(’petal width (cm)’, 0.312), (’petal length (cm)’, 0.254), (’sepal width (cm)’, 0.242), (’sepal length (cm)’, 0.192)]”
random forest	100	2	1.0	0.974	”[(’petal length (cm)’, 0.419), (’petal width (cm)’, 0.388), (’sepal length (cm)’, 0.145), (’sepal width (cm)’, 0.047)]”
random forest	100	FMethodRF.LOG	1.0	0.974	”[(’petal length (cm)’, 0.437), (’petal width (cm)’, 0.374), (’sepal length (cm)’, 0.145), (’sepal width (cm)’, 0.044)]”
random forest	100	FMethodRF.SQRT	1.0	0.974	”[(’petal width (cm)’, 0.433), (’petal length (cm)’, 0.377), (’sepal length (cm)’, 0.133), (’sepal width (cm)’, 0.057)]”
decision forest	1	0.25	0.598	0.579	”[(’sepal width (cm)’, 1.0), (’petal width (cm)’, 0.0), (’petal length (cm)’, 0.0), (’sepal length (cm)’, 0.0)]”
decision forest	1	0.5	0.982	0.895	”[(’petal width (cm)’, 0.851), (’sepal width (cm)’, 0.149), (’petal length (cm)’, 0.0), (’sepal length (cm)’, 0.0)]”
decision forest	1	0.75	1.0	0.974	”[(’petal width (cm)’, 0.75), (’petal length (cm)’, 0.167), (’sepal width (cm)’, 0.083), (’sepal length (cm)’, 0.0)]”
decision forest	1	FMethodDF.RUNIF	0.982	0.895	”[(’petal width (cm)’, 0.851), (’sepal width (cm)’, 0.149), (’petal length (cm)’, 0.0), (’sepal length (cm)’, 0.0)]”
decision forest	10	0.25	0.92	0.711	”[(’sepal length (cm)’, 0.4), (’sepal width (cm)’, 0.3), (’petal width (cm)’, 0.2), (’petal length (cm)’, 0.1)]”
decision forest	10	0.5	1.0	0.947	”[(’petal width (cm)’, 0.401), (’petal length (cm)’, 0.354), (’sepal length (cm)’, 0.166), (’sepal width (cm)’, 0.079)]”
decision forest	10	0.75	1.0	0.974	”[(’petal width (cm)’, 0.471), (’petal length (cm)’, 0.397), (’sepal width (cm)’, 0.069), (’sepal length (cm)’, 0.063)]”
decision forest	10	FMethodDF.RUNIF	1.0	0.947	”[(’petal length (cm)’, 0.283), (’sepal width (cm)’, 0.255), (’petal width (cm)’, 0.246), (’sepal length (cm)’, 0.215)]”
decision forest	25	0.25	0.964	0.895	”[(’sepal length (cm)’, 0.32), (’petal width (cm)’, 0.28), (’sepal width (cm)’, 0.24), (’petal length (cm)’, 0.16)]”
decision forest	25	0.5	1.0	0.947	”[(’petal width (cm)’, 0.371), (’petal length (cm)’, 0.309), (’sepal length (cm)’, 0.204), (’sepal width (cm)’, 0.116)]”
decision forest	25	0.75	1.0	0.974	”[(’petal width (cm)’, 0.44), (’petal length (cm)’, 0.408), (’sepal width (cm)’, 0.083), (’sepal length (cm)’, 0.068)]”
decision forest	25	FMethodDF.RUNIF	1.0	0.974	”[(’petal width (cm)’, 0.384), (’petal length (cm)’, 0.324), (’sepal width (cm)’, 0.161), (’sepal length (cm)’, 0.131)]”
decision forest	50	0.25	0.964	0.895	”[(’sepal length (cm)’, 0.3), (’petal width (cm)’, 0.28), (’sepal width (cm)’, 0.24), (’petal length (cm)’, 0.18)]”
decision forest	50	0.5	1.0	0.947	”[(’petal width (cm)’, 0.388), (’petal length (cm)’, 0.33), (’sepal length (cm)’, 0.187), (’sepal width (cm)’, 0.095)]”
decision forest	50	0.75	1.0	0.974	”[(’petal width (cm)’, 0.443), (’petal length (cm)’, 0.398), (’sepal width (cm)’, 0.085), (’sepal length (cm)’, 0.075)]”
decision forest	50	FMethodDF.RUNIF	1.0	0.974	”[(’petal width (cm)’, 0.398), (’petal length (cm)’, 0.339), (’sepal length (cm)’, 0.146), (’sepal width (cm)’, 0.117)]”
decision forest	75	0.25	0.964	0.895	”[(’petal width (cm)’, 0.333), (’sepal width (cm)’, 0.253), (’sepal length (cm)’, 0.253), (’petal length (cm)’, 0.16)]”
decision forest	75	0.5	1.0	0.947	”[(’petal width (cm)’, 0.403), (’petal length (cm)’, 0.317), (’sepal length (cm)’, 0.181), (’sepal width (cm)’, 0.099)]”
decision forest	75	0.75	1.0	0.974	”[(’petal width (cm)’, 0.462), (’petal length (cm)’, 0.38), (’sepal width (cm)’, 0.089), (’sepal length (cm)’, 0.069)]”
decision forest	75	FMethodDF.RUNIF	1.0	0.974	”[(’petal width (cm)’, 0.429), (’petal length (cm)’, 0.286), (’sepal width (cm)’, 0.181), (’sepal length (cm)’, 0.104)]”
decision forest	100	0.25	0.973	0.947	”[(’petal width (cm)’, 0.31), (’sepal length (cm)’, 0.25), (’sepal width (cm)’, 0.24), (’petal length (cm)’, 0.2)]”
decision forest	100	0.5	1.0	0.947	”[(’petal length (cm)’, 0.374), (’petal width (cm)’, 0.364), (’sepal length (cm)’, 0.168), (’sepal width (cm)’, 0.095)]”
decision forest	100	0.75	1.0	0.974	”[(’petal length (cm)’, 0.433), (’petal width (cm)’, 0.406), (’sepal width (cm)’, 0.09), (’sepal length (cm)’, 0.071)]”
decision forest	100	FMethodDF.RUNIF	1.0	0.974	”[(’petal width (cm)’, 0.367), (’petal length (cm)’, 0.356), (’sepal length (cm)’, 0.153), (’sepal width (cm)’, 0.124)]”

Table 2: Iris dataset results

4.2 Titanic dataset

For this dataset, the attributes "Name of passenger" and "N. Ticket" were removed because they are not relevant to the classification.

The training process took 510.62 seconds and the resulting accuracy averaged across all cases was 0.852 ±0.070. The model was evaluated on the test dataset, resulting in a test accuracy of 0.789 ±0.095.

Table 3 shows the results for this dataset.

4.3 Glass dataset

The training process took 408.79 seconds and the resulting accuracy averaged across all cases was 0.979 ±0.052. The model was evaluated on the test dataset, resulting in a test accuracy of 0.654 ±0.058.

The results for this dataset are shown in Table 4.

4.4 Wine-red dataset

The training process took 3361.57 seconds and the resulting accuracy averaged across all cases was 0.981 ±0.054. The model was evaluated on the test dataset, resulting in a test accuracy of 0.656 ±0.053.

The results are shown in Table 5 below.

4.5 Wine-white dataset

The training process took 10,112.20 seconds and the resulting accuracy averaged across all cases was 0.983 ±0.046. The model was evaluated on the test dataset, resulting in a test accuracy of 0.627 ±0.046.

The results can be seen in Table 6.

algorithm	n_trees	F	train_acc	test_acc	feature_importance
random forest	1	1	0.704	0.631	”[(’Pclass’, 0.366), (’Age’, 0.334), (’Embarked’, 0.111), (’Fare’, 0.087), (’Sex’, 0.059), (’SibSp’, 0.043), (’Parch’, 0.0)]”
random forest	1	2	0.765	0.72	”[(’Embarked’, 0.336), (’Pclass’, 0.163), (’Fare’, 0.138), (’Parch’, 0.131), (’Sex’, 0.117), (’Age’, 0.087), (’SibSp’, 0.029)]”
random forest	1	FMethodRF.LOG	0.825	0.726	”[(’Fare’, 0.596), (’Sex’, 0.136), (’SibSp’, 0.091), (’Age’, 0.063), (’Embarked’, 0.045), (’Pclass’, 0.039), (’Parch’, 0.031)]”
random forest	1	FMethodRF.SQRT	0.807	0.787	”[(’Pclass’, 0.346), (’Parch’, 0.242), (’Embarked’, 0.135), (’Sex’, 0.117), (’Age’, 0.097), (’Fare’, 0.04), (’SibSp’, 0.022)]”
random forest	10	1	0.848	0.866	”[(’Parch’, 0.215), (’Sex’, 0.167), (’Age’, 0.164), (’Pclass’, 0.131), (’SibSp’, 0.129), (’Embarked’, 0.102), (’Fare’, 0.092)]”
random forest	10	2	0.853	0.829	”[(’Sex’, 0.213), (’Age’, 0.187), (’Pclass’, 0.172), (’Parch’, 0.151), (’Fare’, 0.128), (’Embarked’, 0.098), (’SibSp’, 0.05)]”
random forest	10	FMethodRF.LOG	0.878	0.832	”[(’Fare’, 0.23), (’Age’, 0.18), (’Sex’, 0.168), (’Parch’, 0.128), (’Embarked’, 0.101), (’SibSp’, 0.097), (’Pclass’, 0.095)]”
random forest	10	FMethodRF.SQRT	0.882	0.82	”[(’Fare’, 0.298), (’Sex’, 0.173), (’Age’, 0.163), (’Parch’, 0.147), (’Embarked’, 0.079), (’Pclass’, 0.072), (’SibSp’, 0.068)]”
random forest	25	1	0.813	0.787	”[(’SibSp’, 0.169), (’Age’, 0.159), (’Fare’, 0.154), (’Embarked’, 0.147), (’Sex’, 0.129), (’Pclass’, 0.129), (’Parch’, 0.113)]”
random forest	25	2	0.885	0.863	”[(’Fare’, 0.2), (’Sex’, 0.169), (’Pclass’, 0.163), (’Parch’, 0.139), (’Age’, 0.131), (’SibSp’, 0.106), (’Embarked’, 0.091)]”
random forest	25	FMethodRF.LOG	0.898	0.875	”[(’Fare’, 0.269), (’Age’, 0.179), (’Pclass’, 0.143), (’Sex’, 0.142), (’Parch’, 0.115), (’SibSp’, 0.094), (’Embarked’, 0.059)]”
random forest	25	FMethodRF.SQRT	0.877	0.857	”[(’Fare’, 0.196), (’Sex’, 0.171), (’Pclass’, 0.17), (’Age’, 0.144), (’Embarked’, 0.116), (’Parch’, 0.109), (’SibSp’, 0.095)]”
random forest	50	1	0.862	0.878	”[(’Age’, 0.178), (’Sex’, 0.155), (’Fare’, 0.152), (’SibSp’, 0.137), (’Parch’, 0.133), (’Embarked’, 0.124), (’Pclass’, 0.121)]”
random forest	50	2	0.886	0.875	”[(’Fare’, 0.24), (’Sex’, 0.193), (’Age’, 0.135), (’Pclass’, 0.128), (’Parch’, 0.124), (’SibSp’, 0.093), (’Embarked’, 0.087)]”
random forest	50	FMethodRF.LOG	0.88	0.848	”[(’Fare’, 0.259), (’Sex’, 0.175), (’Age’, 0.166), (’Pclass’, 0.134), (’Parch’, 0.094), (’SibSp’, 0.088), (’Embarked’, 0.085)]”
random forest	50	FMethodRF.SQRT	0.893	0.881	”[(’Fare’, 0.245), (’Age’, 0.167), (’Sex’, 0.159), (’Pclass’, 0.132), (’Parch’, 0.121), (’SibSp’, 0.095), (’Embarked’, 0.081)]”
random forest	75	1	0.863	0.884	”[(’Age’, 0.167), (’Fare’, 0.164), (’Sex’, 0.157), (’SibSp’, 0.139), (’Parch’, 0.134), (’Pclass’, 0.12), (’Embarked’, 0.118)]”
random forest	75	2	0.889	0.854	”[(’Fare’, 0.237), (’Sex’, 0.184), (’Age’, 0.16), (’Pclass’, 0.136), (’SibSp’, 0.097), (’Parch’, 0.093), (’Embarked’, 0.092)]”
random forest	75	FMethodRF.LOG	0.874	0.866	”[(’Fare’, 0.222), (’Sex’, 0.18), (’Age’, 0.158), (’Pclass’, 0.131), (’Parch’, 0.119), (’Embarked’, 0.098), (’SibSp’, 0.091)]”
random forest	75	FMethodRF.SQRT	0.883	0.857	”[(’Fare’, 0.236), (’Sex’, 0.17), (’Age’, 0.152), (’Pclass’, 0.124), (’Parch’, 0.122), (’SibSp’, 0.099), (’Embarked’, 0.097)]”
random forest	100	1	0.856	0.866	”[(’Fare’, 0.151), (’Sex’, 0.147), (’Age’, 0.146), (’SibSp’, 0.143), (’Pclass’, 0.142), (’Embarked’, 0.136), (’Parch’, 0.135)]”
random forest	100	2	0.882	0.848	”[(’Fare’, 0.236), (’Age’, 0.167), (’Sex’, 0.165), (’Pclass’, 0.141), (’SibSp’, 0.116), (’Parch’, 0.089), (’Embarked’, 0.085)]”
random forest	100	FMethodRF.LOG	0.885	0.869	”[(’Fare’, 0.225), (’Sex’, 0.174), (’Age’, 0.168), (’Pclass’, 0.121), (’Parch’, 0.118), (’SibSp’, 0.1), (’Embarked’, 0.095)]”
random forest	100	FMethodRF.SQRT	0.887	0.875	”[(’Fare’, 0.203), (’Age’, 0.167), (’Sex’, 0.165), (’Pclass’, 0.157), (’Parch’, 0.133), (’SibSp’, 0.093), (’Embarked’, 0.082)]”
decision forest	1	0.25	0.846	0.881	”[(’Sex’, 1.0), (’Embarked’, 0.0), (’Fare’, 0.0), (’Parch’, 0.0), (’SibSp’, 0.0), (’Age’, 0.0), (’Pclass’, 0.0)]”
decision forest	1	0.5	0.904	0.878	”[(’Fare’, 0.589), (’Sex’, 0.333), (’Embarked’, 0.077), (’Parch’, 0.0), (’SibSp’, 0.0), (’Age’, 0.0), (’Pclass’, 0.0)]”
decision forest	1	0.75	0.83	0.628	”[(’Sex’, 0.333), (’Fare’, 0.26), (’Age’, 0.202), (’Pclass’, 0.185), (’Embarked’, 0.019), (’Parch’, 0.0), (’SibSp’, 0.0)]”
decision forest	1	FMethodDF.RUNIF	0.846	0.881	”[(’Sex’, 1.0), (’Embarked’, 0.0), (’Fare’, 0.0), (’Parch’, 0.0), (’SibSp’, 0.0), (’Age’, 0.0), (’Pclass’, 0.0)]”
decision forest	10	0.25	0.663	0.558	”[(’Sex’, 0.3), (’Pclass’, 0.3), (’Fare’, 0.1), (’Parch’, 0.1), (’SibSp’, 0.1), (’Age’, 0.1), (’Embarked’, 0.0)]”
decision forest	10	0.5	0.88	0.765	”[(’Fare’, 0.376), (’Sex’, 0.167), (’Age’, 0.163), (’Parch’, 0.093), (’Embarked’, 0.091), (’Pclass’, 0.085), (’SibSp’, 0.026)]”
decision forest	10	0.75	0.878	0.753	”[(’Sex’, 0.267), (’Fare’, 0.24), (’Age’, 0.19), (’Pclass’, 0.154), (’SibSp’, 0.068), (’Parch’, 0.057), (’Embarked’, 0.025)]”
decision forest	10	FMethodDF.RUNIF	0.832	0.695	”[(’Fare’, 0.219), (’Pclass’, 0.179), (’Age’, 0.176), (’Sex’, 0.167), (’SibSp’, 0.131), (’Parch’, 0.103), (’Embarked’, 0.025)]”
decision forest	25	0.25	0.774	0.683	”[(’Sex’, 0.32), (’Fare’, 0.2), (’Embarked’, 0.12), (’Parch’, 0.12), (’Pclass’, 0.12), (’SibSp’, 0.08), (’Age’, 0.04)]”
decision forest	25	0.5	0.908	0.814	”[(’Fare’, 0.395), (’Sex’, 0.173), (’Age’, 0.153), (’Parch’, 0.098), (’Embarked’, 0.071), (’Pclass’, 0.065), (’SibSp’, 0.044)]”
decision forest	25	0.75	0.914	0.787	”[(’Fare’, 0.29), (’Sex’, 0.28), (’Age’, 0.175), (’Pclass’, 0.126), (’SibSp’, 0.064), (’Parch’, 0.046), (’Embarked’, 0.018)]”
decision forest	25	FMethodDF.RUNIF	0.932	0.851	”[(’Fare’, 0.322), (’Sex’, 0.227), (’Age’, 0.155), (’Pclass’, 0.11), (’SibSp’, 0.077), (’Parch’, 0.059), (’Embarked’, 0.05)]”
decision forest	50	0.25	0.709	0.61	”[(’Sex’, 0.24), (’Embarked’, 0.16), (’SibSp’, 0.16), (’Fare’, 0.14), (’Parch’, 0.14), (’Pclass’, 0.14), (’Age’, 0.02)]”
decision forest	50	0.5	0.862	0.707	”[(’Fare’, 0.294), (’Age’, 0.175), (’Parch’, 0.137), (’Sex’, 0.133), (’Pclass’, 0.097), (’SibSp’, 0.09), (’Embarked’, 0.074)]”
decision forest	50	0.75	0.932	0.817	”[(’Fare’, 0.295), (’Sex’, 0.24), (’Age’, 0.178), (’Pclass’, 0.124), (’SibSp’, 0.079), (’Parch’, 0.061), (’Embarked’, 0.023)]”
decision forest	50	FMethodDF.RUNIF	0.886	0.784	”[(’Fare’, 0.257), (’Sex’, 0.22), (’Pclass’, 0.135), (’Age’, 0.126), (’SibSp’, 0.106), (’Parch’, 0.088), (’Embarked’, 0.067)]”
decision forest	75	0.25	0.663	0.558	”[(’Sex’, 0.24), (’Embarked’, 0.173), (’Pclass’, 0.173), (’Fare’, 0.147), (’SibSp’, 0.133), (’Parch’, 0.107), (’Age’, 0.027)]”
decision forest	75	0.5	0.875	0.744	”[(’Fare’, 0.304), (’Age’, 0.165), (’Sex’, 0.133), (’Pclass’, 0.129), (’Parch’, 0.104), (’Embarked’, 0.083), (’SibSp’, 0.083)]”
decision forest	75	0.75	0.925	0.796	”[(’Fare’, 0.288), (’Sex’, 0.249), (’Age’, 0.172), (’Pclass’, 0.134), (’SibSp’, 0.076), (’Parch’, 0.056), (’Embarked’, 0.025)]”
decision forest	75	FMethodDF.RUNIF	0.914	0.802	”[(’Fare’, 0.319), (’Sex’, 0.218), (’Age’, 0.151), (’Pclass’, 0.141), (’SibSp’, 0.064), (’Parch’, 0.059), (’Embarked’, 0.048)]”
decision forest	100	0.25	0.663	0.558	”[(’Sex’, 0.21), (’Pclass’, 0.17), (’Embarked’, 0.15), (’Fare’, 0.15), (’Parch’, 0.14), (’SibSp’, 0.13), (’Age’, 0.05)]”
decision forest	100	0.5	0.862	0.713	”[(’Fare’, 0.27), (’Age’, 0.174), (’Pclass’, 0.143), (’Sex’, 0.14), (’Parch’, 0.105), (’SibSp’, 0.087), (’Embarked’, 0.079)]”
decision forest	100	0.75	0.923	0.79	”[(’Fare’, 0.287), (’Sex’, 0.247), (’Age’, 0.181), (’Pclass’, 0.132), (’SibSp’, 0.071), (’Parch’, 0.059), (’Embarked’, 0.024)]”
decision forest	100	FMethodDF.RUNIF	0.913	0.823	”[(’Fare’, 0.246), (’Sex’, 0.217), (’Age’, 0.172), (’Pclass’, 0.13), (’SibSp’, 0.102), (’Parch’, 0.08), (’Embarked’, 0.052)]”

Table 3: Titanic dataset results

algorithm	n_trees	F	train_acc	test_acc	feature_importance
random forest	1	1	0.856	0.556	"[('Si', 0.454), ('Mg', 0.161), ('K', 0.101), ('Al', 0.077), ('Ca', 0.071), ('RI', 0.058), ('Ba', 0.037), ('Fe', 0.026), ('Na', 0.015)]"
random forest	1	2	0.831	0.648	"[('Mg', 0.336), ('Na', 0.186), ('Ca', 0.186), ('Al', 0.094), ('RI', 0.087), ('K', 0.065), ('Fe', 0.034), ('Si', 0.01), ('Ba', 0.001)]"
random forest	1	FMethodRF.LOG	0.825	0.611	"[('Mg', 0.353), ('RI', 0.186), ('Al', 0.121), ('Ba', 0.111), ('K', 0.087), ('Fe', 0.068), ('Na', 0.038), ('Ca', 0.021), ('Si', 0.014)]"
random forest	1	FMethodRF.SQRT	0.838	0.556	"[('Al', 0.426), ('Si', 0.195), ('K', 0.128), ('Na', 0.125), ('Ca', 0.086), ('Mg', 0.032), ('RI', 0.009), ('Fe', 0.0), ('Ba', 0.0)]"
random forest	10	1	0.962	0.704	"[('Si', 0.161), ('Na', 0.149), ('RI', 0.147), ('K', 0.127), ('Al', 0.112), ('Ba', 0.096), ('Fe', 0.078), ('Ca', 0.071), ('Mg', 0.058)]"
random forest	10	2	0.994	0.667	"[('Al', 0.186), ('K', 0.169), ('Na', 0.134), ('Mg', 0.123), ('Ba', 0.117), ('RI', 0.102), ('Ca', 0.071), ('Si', 0.062), ('Fe', 0.037)]"
random forest	10	FMethodRF.LOG	0.988	0.611	"[('Al', 0.335), ('Mg', 0.165), ('Si', 0.135), ('RI', 0.079), ('K', 0.07), ('Na', 0.067), ('Ba', 0.067), ('Ca', 0.066), ('Fe', 0.016)]"
random forest	10	FMethodRF.SQRT	0.994	0.667	"[('Mg', 0.22), ('Al', 0.218), ('Na', 0.123), ('RI', 0.122), ('Ca', 0.12), ('Ba', 0.09), ('K', 0.048), ('Si', 0.038), ('Fe', 0.02)]"
random forest	25	1	0.962	0.667	"[('K', 0.154), ('Mg', 0.147), ('RI', 0.133), ('Ba', 0.113), ('Al', 0.112), ('Na', 0.091), ('Fe', 0.091), ('Ca', 0.081), ('Si', 0.078)]"
random forest	25	2	1.0	0.685	"[('Al', 0.203), ('Mg', 0.128), ('Ba', 0.117), ('K', 0.114), ('RI', 0.106), ('Na', 0.106), ('Ca', 0.1), ('Si', 0.096), ('Fe', 0.028)]"
random forest	25	FMethodRF.LOG	1.0	0.722	"[('Al', 0.198), ('Mg', 0.171), ('Ca', 0.148), ('Si', 0.108), ('RI', 0.103), ('Na', 0.098), ('K', 0.081), ('Ba', 0.077), ('Fe', 0.016)]"
random forest	25	FMethodRF.SQRT	1.0	0.667	"[('Mg', 0.172), ('Al', 0.167), ('Ca', 0.152), ('Na', 0.134), ('Ba', 0.109), ('RI', 0.103), ('K', 0.08), ('Si', 0.064), ('Fe', 0.019)]"
random forest	50	1	0.975	0.667	"[('RI', 0.148), ('Al', 0.131), ('Si', 0.121), ('Na', 0.109), ('Mg', 0.105), ('Ba', 0.101), ('Ca', 0.096), ('K', 0.095), ('Fe', 0.094)]"
random forest	50	2	1.0	0.722	"[('Al', 0.205), ('Ca', 0.144), ('Na', 0.129), ('Mg', 0.122), ('Ba', 0.115), ('RI', 0.101), ('Si', 0.089), ('K', 0.076), ('Fe', 0.019)]"
random forest	50	FMethodRF.LOG	1.0	0.741	"[('Al', 0.196), ('Mg', 0.173), ('Ba', 0.15), ('Na', 0.111), ('RI', 0.108), ('Ca', 0.094), ('K', 0.085), ('Si', 0.072), ('Fe', 0.012)]"
random forest	50	FMethodRF.SQRT	1.0	0.685	"[('Al', 0.191), ('Mg', 0.155), ('Ba', 0.152), ('Ca', 0.11), ('RI', 0.107), ('Na', 0.099), ('K', 0.097), ('Si', 0.081), ('Fe', 0.01)]"
random forest	75	1	0.975	0.667	"[('Si', 0.146), ('RI', 0.132), ('K', 0.119), ('Ca', 0.118), ('Na', 0.114), ('Ba', 0.103), ('Al', 0.1), ('Fe', 0.084), ('Mg', 0.083)]"
random forest	75	2	1.0	0.685	"[('Mg', 0.16), ('Al', 0.158), ('Ba', 0.135), ('Ca', 0.122), ('Na', 0.111), ('K', 0.104), ('Si', 0.1), ('RI', 0.089), ('Fe', 0.02)]"
random forest	75	FMethodRF.LOG	1.0	0.722	"[('Al', 0.178), ('Mg', 0.17), ('Ba', 0.144), ('Ca', 0.122), ('RI', 0.114), ('K', 0.091), ('Na', 0.088), ('Si', 0.075), ('Fe', 0.017)]"
random forest	75	FMethodRF.SQRT	1.0	0.722	"[('Al', 0.233), ('Mg', 0.18), ('Ba', 0.127), ('RI', 0.109), ('Ca', 0.103), ('Na', 0.095), ('Si', 0.08), ('K', 0.061), ('Fe', 0.013)]"
random forest	100	1	0.988	0.63	"[('K', 0.138), ('Si', 0.126), ('RI', 0.125), ('Al', 0.12), ('Na', 0.118), ('Ca', 0.106), ('Mg', 0.093), ('Ba', 0.09), ('Fe', 0.083)]"
random forest	100	2	1.0	0.685	"[('Al', 0.177), ('Mg', 0.155), ('RI', 0.138), ('Ba', 0.114), ('Ca', 0.108), ('Na', 0.1), ('K', 0.1), ('Si', 0.08), ('Fe', 0.028)]"
random forest	100	FMethodRF.LOG	1.0	0.685	"[('Al', 0.218), ('Mg', 0.152), ('Ba', 0.15), ('Ca', 0.116), ('RI', 0.108), ('Na', 0.107), ('K', 0.071), ('Si', 0.066), ('Fe', 0.013)]"
random forest	100	FMethodRF.SQRT	1.0	0.704	"[('Al', 0.222), ('Mg', 0.153), ('Ca', 0.129), ('Na', 0.117), ('Ba', 0.108), ('RI', 0.1), ('K', 0.086), ('Si', 0.074), ('Fe', 0.012)]"
decision forest	1	0.25	0.806	0.537	"[('K', 0.755), ('Fe', 0.245), ('Ba', 0.0), ('Ca', 0.0), ('Si', 0.0), ('Al', 0.0), ('Mg', 0.0), ('Na', 0.0), ('RI', 0.0)]"
decision forest	1	0.5	1.0	0.519	"[('Na', 0.409), ('Si', 0.347), ('K', 0.239), ('Fe', 0.005), ('Ba', 0.0), ('Ca', 0.0), ('Al', 0.0), ('Mg', 0.0), ('RI', 0.0)]"
decision forest	1	0.75	1.0	0.537	"[('Ba', 0.335), ('Si', 0.204), ('Ca', 0.188), ('K', 0.165), ('Na', 0.099), ('Fe', 0.009), ('Al', 0.0), ('Mg', 0.0), ('RI', 0.0)]"
decision forest	1	FMethodDF.RUNIF	1.0	0.537	"[('Ba', 0.335), ('Si', 0.204), ('Ca', 0.188), ('K', 0.165), ('Na', 0.099), ('Fe', 0.009), ('Al', 0.0), ('Mg', 0.0), ('RI', 0.0)]"
decision forest	10	0.25	1.0	0.556	"[('RI', 0.204), ('Al', 0.197), ('Mg', 0.178), ('K', 0.126), ('Ba', 0.078), ('Si', 0.075), ('Ca', 0.049), ('Fe', 0.049), ('Na', 0.043)]"
decision forest	10	0.5	1.0	0.704	"[('Al', 0.219), ('Ba', 0.176), ('Mg', 0.126), ('Si', 0.112), ('Na', 0.104), ('Ca', 0.095), ('RI', 0.094), ('K', 0.069), ('Fe', 0.006)]"
decision forest	10	0.75	1.0	0.63	"[('Al', 0.254), ('Ba', 0.242), ('Si', 0.124), ('Mg', 0.119), ('Na', 0.07), ('RI', 0.066), ('Ca', 0.062), ('K', 0.061), ('Fe', 0.003)]"
decision forest	10	FMethodDF.RUNIF	1.0	0.685	"[('Al', 0.271), ('Ba', 0.248), ('Mg', 0.11), ('RI', 0.109), ('Si', 0.096), ('Ca', 0.067), ('K', 0.061), ('Na', 0.033), ('Fe', 0.006)]"
decision forest	25	0.25	1.0	0.611	"[('Ca', 0.146), ('Ba', 0.132), ('RI', 0.129), ('Na', 0.121), ('Mg', 0.116), ('Al', 0.105), ('Si', 0.103), ('K', 0.102), ('Fe', 0.047)]"
decision forest	25	0.5	1.0	0.704	"[('Al', 0.182), ('Ba', 0.171), ('Mg', 0.153), ('Na', 0.142), ('Ca', 0.119), ('Si', 0.094), ('RI', 0.067), ('K', 0.063), ('Fe', 0.008)]"
decision forest	25	0.75	1.0	0.685	"[('Al', 0.28), ('Ba', 0.217), ('Mg', 0.121), ('Si', 0.104), ('Na', 0.086), ('Ca', 0.078), ('RI', 0.057), ('K', 0.053), ('Fe', 0.003)]"
decision forest	25	FMethodDF.RUNIF	1.0	0.648	"[('Al', 0.22), ('Ba', 0.188), ('Mg', 0.169), ('Na', 0.139), ('Ca', 0.099), ('Si', 0.083), ('RI', 0.069), ('K', 0.03), ('Fe', 0.003)]"
decision forest	50	0.25	1.0	0.63	"[('Al', 0.161), ('Ca', 0.143), ('Ba', 0.123), ('RI', 0.122), ('Mg', 0.12), ('Si', 0.12), ('Na', 0.117), ('K', 0.062), ('Fe', 0.031)]"
decision forest	50	0.5	1.0	0.704	"[('Al', 0.222), ('Ba', 0.193), ('Mg', 0.17), ('Ca', 0.104), ('Si', 0.092), ('Na', 0.085), ('RI', 0.081), ('K', 0.049), ('Fe', 0.005)]"
decision forest	50	0.75	1.0	0.667	"[('Al', 0.302), ('Ba', 0.232), ('Mg', 0.133), ('Si', 0.099), ('Ca', 0.075), ('Na', 0.067), ('RI', 0.05), ('K', 0.039), ('Fe', 0.003)]"
decision forest	50	FMethodDF.RUNIF	1.0	0.648	"[('Al', 0.244), ('Ba', 0.187), ('Ca', 0.157), ('Mg', 0.132), ('Si', 0.084), ('K', 0.07), ('Na', 0.064), ('RI', 0.058), ('Fe', 0.004)]"
decision forest	75	0.25	1.0	0.611	"[('Ca', 0.188), ('Al', 0.139), ('Na', 0.122), ('RI', 0.121), ('Si', 0.113), ('Mg', 0.095), ('K', 0.095), ('Ba', 0.09), ('Fe', 0.037)]"
decision forest	75	0.5	1.0	0.667	"[('Al', 0.192), ('Ba', 0.166), ('Mg', 0.144), ('Ca', 0.126), ('RI', 0.119), ('Si', 0.089), ('Na', 0.088), ('K', 0.067), ('Fe', 0.009)]"
decision forest	75	0.75	1.0	0.704	"[('Al', 0.277), ('Ba', 0.209), ('Mg', 0.145), ('Si', 0.097), ('Ca', 0.08), ('Na', 0.078), ('RI', 0.066), ('K', 0.042), ('Fe', 0.004)]"
decision forest	75	FMethodDF.RUNIF	1.0	0.63	"[('Al', 0.207), ('Ba', 0.203), ('Mg', 0.145), ('Ca', 0.126), ('RI', 0.102), ('Si', 0.084), ('Na', 0.077), ('K', 0.052), ('Fe', 0.004)]"
decision forest	100	0.25	1.0	0.611	"[('Ca', 0.165), ('Al', 0.156), ('RI', 0.137), ('Na', 0.119), ('Si', 0.115), ('K', 0.1), ('Mg', 0.09), ('Ba', 0.073), ('Fe', 0.046)]"
decision forest	100	0.5	1.0	0.667	"[('Al', 0.21), ('Ba', 0.148), ('Mg', 0.146), ('RI', 0.122), ('Ca', 0.112), ('Na', 0.093), ('Si', 0.085), ('K', 0.073), ('Fe', 0.01)]"
decision forest	100	0.75	1.0	0.722	"[('Al', 0.27), ('Ba', 0.206), ('Mg', 0.145), ('Si', 0.09), ('Na', 0.086), ('Ca', 0.083), ('RI', 0.075), ('K', 0.041), ('Fe', 0.005)]"
decision forest	100	FMethodDF.RUNIF	1.0	0.704	"[('Al', 0.203), ('Ba', 0.167), ('Ca', 0.123), ('Mg', 0.117), ('Na', 0.11), ('Si', 0.098), ('RI', 0.089), ('K', 0.063), ('Fe', 0.031)]"

Table 4: Glass dataset results

algorithm	n_trees	F	train_acc	test_acc	feature_importance
random forest	1	1	0.797	0.532	”[('alcohol', 0.467), ('volatile acidity', 0.206), ('citric acid', 0.085), ('total sulfur dioxide', 0.05), ('sulphates', 0.045), ('free sulfur dioxide', 0.044), ('density', 0.037), ('chlorides', 0.022), ('residual sugar', 0.018), ('fixed acidity', 0.015), ('pH', 0.012)]”
random forest	1	2	0.813	0.54	”[('total sulfur dioxide', 0.348), ('fixed acidity', 0.195), ('chlorides', 0.123), ('alcohol', 0.12), ('free sulfur dioxide', 0.053), ('sulphates', 0.048), ('density', 0.041), ('residual sugar', 0.034), ('volatile acidity', 0.023), ('citric acid', 0.011), ('pH', 0.006)]”
random forest	1	FMethodRF.LOG	0.813	0.575	”[('sulphates', 0.378), ('density', 0.207), ('volatile acidity', 0.13), ('free sulfur dioxide', 0.114), ('total sulfur dioxide', 0.068), ('alcohol', 0.047), ('fixed acidity', 0.027), ('chlorides', 0.011), ('residual sugar', 0.01), ('pH', 0.008), ('citric acid', 0.002)]”
random forest	1	FMethodRF.SQRT	0.8	0.565	”[('citric acid', 0.407), ('alcohol', 0.238), ('total sulfur dioxide', 0.127), ('volatile acidity', 0.075), ('pH', 0.053), ('fixed acidity', 0.034), ('sulphates', 0.028), ('residual sugar', 0.012), ('density', 0.011), ('chlorides', 0.009), ('free sulfur dioxide', 0.006)]”
random forest	10	1	0.981	0.665	”[('total sulfur dioxide', 0.158), ('pH', 0.139), ('sulphates', 0.133), ('alcohol', 0.1), ('free sulfur dioxide', 0.082), ('residual sugar', 0.082), ('fixed acidity', 0.072), ('citric acid', 0.066), ('density', 0.064), ('chlorides', 0.057), ('volatile acidity', 0.046)]”
random forest	10	2	0.985	0.682	”[('alcohol', 0.205), ('sulphates', 0.131), ('total sulfur dioxide', 0.129), ('chlorides', 0.122), ('density', 0.09), ('fixed acidity', 0.086), ('free sulfur dioxide', 0.056), ('volatile acidity', 0.05), ('residual sugar', 0.049), ('pH', 0.041), ('citric acid', 0.041)]”
random forest	10	FMethodRF.LOG	0.98	0.658	”[('alcohol', 0.246), ('total sulfur dioxide', 0.158), ('volatile acidity', 0.152), ('sulphates', 0.123), ('citric acid', 0.07), ('density', 0.069), ('free sulfur dioxide', 0.043), ('chlorides', 0.04), ('pH', 0.035), ('fixed acidity', 0.034), ('residual sugar', 0.031)]”
random forest	10	FMethodRF.SQRT	0.976	0.655	”[('alcohol', 0.187), ('sulphates', 0.139), ('density', 0.12), ('volatile acidity', 0.117), ('total sulfur dioxide', 0.1), ('chlorides', 0.088), ('citric acid', 0.087), ('free sulfur dioxide', 0.055), ('residual sugar', 0.047), ('pH', 0.029), ('fixed acidity', 0.029)]”
random forest	25	1	0.999	0.668	”[('volatile acidity', 0.115), ('chlorides', 0.108), ('pH', 0.107), ('total sulfur dioxide', 0.1), ('citric acid', 0.1), ('density', 0.092), ('fixed acidity', 0.087), ('sulphates', 0.085), ('alcohol', 0.073), ('free sulfur dioxide', 0.07), ('residual sugar', 0.063)]”
random forest	25	2	0.999	0.698	”[('alcohol', 0.185), ('sulphates', 0.159), ('volatile acidity', 0.093), ('fixed acidity', 0.091), ('total sulfur dioxide', 0.085), ('chlorides', 0.081), ('citric acid', 0.077), ('density', 0.068), ('residual sugar', 0.068), ('free sulfur dioxide', 0.049), ('pH', 0.045)]”
random forest	25	FMethodRF.LOG	0.999	0.695	”[('alcohol', 0.194), ('sulphates', 0.146), ('volatile acidity', 0.135), ('total sulfur dioxide', 0.121), ('density', 0.093), ('citric acid', 0.066), ('chlorides', 0.055), ('residual sugar', 0.05), ('free sulfur dioxide', 0.05), ('fixed acidity', 0.047), ('pH', 0.043)]”
random forest	25	FMethodRF.SQRT	1.0	0.68	”[('alcohol', 0.234), ('sulphates', 0.195), ('volatile acidity', 0.106), ('total sulfur dioxide', 0.101), ('density', 0.101), ('fixed acidity', 0.06), ('pH', 0.05), ('citric acid', 0.047), ('chlorides', 0.037), ('free sulfur dioxide', 0.035), ('residual sugar', 0.034)]”
random forest	50	1	1.0	0.702	”[('pH', 0.138), ('chlorides', 0.105), ('free sulfur dioxide', 0.101), ('density', 0.097), ('alcohol', 0.095), ('volatile acidity', 0.082), ('total sulfur dioxide', 0.081), ('residual sugar', 0.076), ('sulphates', 0.076), ('citric acid', 0.076), ('fixed acidity', 0.072)]”
random forest	50	2	1.0	0.692	”[('alcohol', 0.142), ('total sulfur dioxide', 0.124), ('volatile acidity', 0.122), ('sulphates', 0.116), ('density', 0.107), ('fixed acidity', 0.093), ('chlorides', 0.069), ('residual sugar', 0.061), ('citric acid', 0.06), ('pH', 0.056), ('free sulfur dioxide', 0.049)]”
random forest	50	FMethodRF.LOG	1.0	0.708	”[('alcohol', 0.174), ('total sulfur dioxide', 0.136), ('volatile acidity', 0.129), ('sulphates', 0.124), ('density', 0.104), ('chlorides', 0.072), ('citric acid', 0.071), ('fixed acidity', 0.065), ('residual sugar', 0.053), ('free sulfur dioxide', 0.037), ('pH', 0.035)]”
random forest	50	FMethodRF.SQRT	1.0	0.688	”[('sulphates', 0.184), ('alcohol', 0.168), ('total sulfur dioxide', 0.138), ('volatile acidity', 0.126), ('density', 0.097), ('fixed acidity', 0.062), ('citric acid', 0.052), ('chlorides', 0.051), ('residual sugar', 0.043), ('pH', 0.041), ('free sulfur dioxide', 0.038)]”
random forest	75	1	1.0	0.688	”[('chlorides', 0.113), ('alcohol', 0.109), ('free sulfur dioxide', 0.096), ('volatile acidity', 0.094), ('residual sugar', 0.093), ('total sulfur dioxide', 0.089), ('fixed acidity', 0.086), ('sulphates', 0.086), ('citric acid', 0.083), ('density', 0.082), ('pH', 0.069)]”
random forest	75	2	1.0	0.71	”[('alcohol', 0.159), ('volatile acidity', 0.128), ('density', 0.127), ('sulphates', 0.121), ('total sulfur dioxide', 0.101), ('citric acid', 0.085), ('fixed acidity', 0.077), ('chlorides', 0.057), ('pH', 0.049), ('residual sugar', 0.048), ('free sulfur dioxide', 0.048)]”
random forest	75	FMethodRF.LOG	1.0	0.708	”[('alcohol', 0.183), ('sulphates', 0.152), ('volatile acidity', 0.13), ('total sulfur dioxide', 0.126), ('density', 0.092), ('fixed acidity', 0.075), ('citric acid', 0.065), ('chlorides', 0.058), ('residual sugar', 0.042), ('free sulfur dioxide', 0.038), ('pH', 0.038)]”
random forest	75	FMethodRF.SQRT	1.0	0.69	”[('alcohol', 0.182), ('sulphates', 0.17), ('volatile acidity', 0.159), ('total sulfur dioxide', 0.109), ('density', 0.084), ('chlorides', 0.066), ('citric acid', 0.059), ('fixed acidity', 0.049), ('free sulfur dioxide', 0.045), ('pH', 0.04), ('residual sugar', 0.036)]”
random forest	100	1	1.0	0.692	”[('residual sugar', 0.104), ('fixed acidity', 0.104), ('chlorides', 0.1), ('sulphates', 0.097), ('density', 0.097), ('total sulfur dioxide', 0.096), ('alcohol', 0.094), ('citric acid', 0.087), ('volatile acidity', 0.078), ('pH', 0.072), ('free sulfur dioxide', 0.07)]”
random forest	100	2	1.0	0.7	”[('volatile acidity', 0.162), ('alcohol', 0.154), ('density', 0.107), ('sulphates', 0.107), ('total sulfur dioxide', 0.092), ('citric acid', 0.092), ('chlorides', 0.073), ('fixed acidity', 0.07), ('residual sugar', 0.053), ('free sulfur dioxide', 0.045), ('pH', 0.044)]”
random forest	100	FMethodRF.LOG	1.0	0.705	”[('alcohol', 0.192), ('sulphates', 0.166), ('volatile acidity', 0.135), ('total sulfur dioxide', 0.123), ('density', 0.086), ('citric acid', 0.065), ('chlorides', 0.059), ('fixed acidity', 0.051), ('pH', 0.042), ('residual sugar', 0.041), ('free sulfur dioxide', 0.04)]”
random forest	100	FMethodRF.SQRT	1.0	0.712	”[('alcohol', 0.212), ('sulphates', 0.155), ('volatile acidity', 0.133), ('total sulfur dioxide', 0.121), ('density', 0.085), ('citric acid', 0.068), ('chlorides', 0.059), ('fixed acidity', 0.05), ('free sulfur dioxide', 0.043), ('residual sugar', 0.038), ('pH', 0.036)]”
decision forest	1	0.25	0.975	0.492	”[('total sulfur dioxide', 0.523), ('density', 0.477), ('alcohol', 0.0), ('sulphates', 0.0), ('pH', 0.0), ('free sulfur dioxide', 0.0), ('chlorides', 0.0), ('residual sugar', 0.0), ('citric acid', 0.0), ('volatile acidity', 0.0), ('fixed acidity', 0.0)]”
decision forest	1	0.5	1.0	0.595	”[('alcohol', 0.523), ('total sulfur dioxide', 0.251), ('density', 0.136), ('free sulfur dioxide', 0.064), ('chlorides', 0.026), ('sulphates', 0.0), ('pH', 0.0), ('residual sugar', 0.0), ('citric acid', 0.0), ('volatile acidity', 0.0), ('fixed acidity', 0.0)]”
decision forest	1	0.75	1.0	0.63	”[('alcohol', 0.398), ('sulphates', 0.164), ('volatile acidity', 0.147), ('density', 0.093), ('total sulfur dioxide', 0.061), ('pH', 0.058), ('chlorides', 0.057), ('free sulfur dioxide', 0.022), ('residual sugar', 0.0), ('citric acid', 0.0), ('fixed acidity', 0.0)]”
decision forest	1	FMethodDF.RUNIF	1.0	0.572	”[('total sulfur dioxide', 0.482), ('density', 0.365), ('chlorides', 0.091), ('free sulfur dioxide', 0.062), ('alcohol', 0.0), ('sulphates', 0.0), ('pH', 0.0), ('residual sugar', 0.0), ('citric acid', 0.0), ('volatile acidity', 0.0), ('fixed acidity', 0.0)]”
decision forest	10	0.25	0.99	0.568	”[('density', 0.169), ('residual sugar', 0.145), ('free sulfur dioxide', 0.13), ('chlorides', 0.122), ('total sulfur dioxide', 0.119), ('fixed acidity', 0.109), ('alcohol', 0.075), ('volatile acidity', 0.074), ('pH', 0.031), ('citric acid', 0.026), ('sulphates', 0.0)]”
decision forest	10	0.5	1.0	0.672	”[('alcohol', 0.428), ('volatile acidity', 0.114), ('total sulfur dioxide', 0.095), ('sulphates', 0.09), ('fixed acidity', 0.05), ('residual sugar', 0.043), ('pH', 0.042), ('citric acid', 0.04), ('density', 0.039), ('free sulfur dioxide', 0.035), ('chlorides', 0.025)]”
decision forest	10	0.75	1.0	0.64	”[('alcohol', 0.361), ('sulphates', 0.176), ('volatile acidity', 0.133), ('total sulfur dioxide', 0.095), ('residual sugar', 0.061), ('density', 0.047), ('pH', 0.04), ('chlorides', 0.035), ('citric acid', 0.02), ('free sulfur dioxide', 0.017), ('fixed acidity', 0.015)]”
decision forest	10	FMethodDF.RUNIF	1.0	0.655	”[('alcohol', 0.286), ('sulphates', 0.165), ('volatile acidity', 0.131), ('density', 0.107), ('fixed acidity', 0.068), ('total sulfur dioxide', 0.066), ('citric acid', 0.049), ('residual sugar', 0.048), ('chlorides', 0.045), ('pH', 0.021), ('free sulfur dioxide', 0.013)]”
decision forest	25	0.25	0.997	0.6	”[('residual sugar', 0.145), ('density', 0.13), ('sulphates', 0.113), ('volatile acidity', 0.11), ('free sulfur dioxide', 0.103), ('fixed acidity', 0.092), ('chlorides', 0.088), ('total sulfur dioxide', 0.083), ('alcohol', 0.055), ('citric acid', 0.041), ('pH', 0.039)]”
decision forest	25	0.5	1.0	0.68	”[('sulphates', 0.236), ('alcohol', 0.231), ('volatile acidity', 0.115), ('total sulfur dioxide', 0.086), ('density', 0.077), ('citric acid', 0.048), ('residual sugar', 0.048), ('fixed acidity', 0.046), ('chlorides', 0.043), ('pH', 0.04), ('free sulfur dioxide', 0.03)]”
decision forest	25	0.75	1.0	0.668	”[('alcohol', 0.313), ('sulphates', 0.211), ('volatile acidity', 0.138), ('total sulfur dioxide', 0.082), ('residual sugar', 0.062), ('density', 0.052), ('chlorides', 0.046), ('pH', 0.032), ('citric acid', 0.024), ('fixed acidity', 0.022), ('free sulfur dioxide', 0.019)]”
decision forest	25	FMethodDF.RUNIF	1.0	0.648	”[('alcohol', 0.24), ('sulphates', 0.191), ('volatile acidity', 0.14), ('density', 0.091), ('total sulfur dioxide', 0.074), ('residual sugar', 0.059), ('free sulfur dioxide', 0.054), ('pH', 0.046), ('fixed acidity', 0.04), ('citric acid', 0.039), ('chlorides', 0.026)]”
decision forest	50	0.25	0.999	0.62	”[('total sulfur dioxide', 0.119), ('fixed acidity', 0.115), ('volatile acidity', 0.102), ('citric acid', 0.099), ('density', 0.095), ('sulphates', 0.095), ('residual sugar', 0.086), ('chlorides', 0.084), ('alcohol', 0.083), ('free sulfur dioxide', 0.082), ('pH', 0.039)]”
decision forest	50	0.5	1.0	0.695	”[('alcohol', 0.21), ('sulphates', 0.19), ('total sulfur dioxide', 0.115), ('volatile acidity', 0.11), ('density', 0.078), ('fixed acidity', 0.065), ('citric acid', 0.064), ('chlorides', 0.047), ('residual sugar', 0.043), ('pH', 0.04), ('free sulfur dioxide', 0.038)]”
decision forest	50	0.75	1.0	0.675	”[('alcohol', 0.291), ('sulphates', 0.217), ('volatile acidity', 0.124), ('total sulfur dioxide', 0.088), ('density', 0.069), ('residual sugar', 0.056), ('chlorides', 0.041), ('citric acid', 0.031), ('pH', 0.03), ('fixed acidity', 0.029), ('free sulfur dioxide', 0.023)]”
decision forest	50	FMethodDF.RUNIF	1.0	0.675	”[('alcohol', 0.222), ('sulphates', 0.153), ('volatile acidity', 0.106), ('total sulfur dioxide', 0.088), ('density', 0.082), ('residual sugar', 0.078), ('fixed acidity', 0.063), ('citric acid', 0.061), ('chlorides', 0.049), ('free sulfur dioxide', 0.049), ('pH', 0.048)]”
decision forest	75	0.25	0.999	0.638	”[('total sulfur dioxide', 0.138), ('volatile acidity', 0.124), ('fixed acidity', 0.112), ('sulphates', 0.107), ('chlorides', 0.086), ('residual sugar', 0.082), ('alcohol', 0.081), ('free sulfur dioxide', 0.079), ('density', 0.074), ('citric acid', 0.073), ('pH', 0.044)]”
decision forest	75	0.5	1.0	0.68	”[('alcohol', 0.195), ('sulphates', 0.183), ('total sulfur dioxide', 0.126), ('volatile acidity', 0.126), ('fixed acidity', 0.068), ('density', 0.065), ('chlorides', 0.059), ('citric acid', 0.05), ('pH', 0.046), ('free sulfur dioxide', 0.043), ('residual sugar', 0.039)]”
decision forest	75	0.75	1.0	0.675	”[('alcohol', 0.279), ('sulphates', 0.207), ('volatile acidity', 0.135), ('total sulfur dioxide', 0.097), ('density', 0.064), ('residual sugar', 0.057), ('chlorides', 0.041), ('pH', 0.034), ('fixed acidity', 0.033), ('citric acid', 0.028), ('free sulfur dioxide', 0.025)]”
decision forest	75	FMethodDF.RUNIF	1.0	0.682	”[('alcohol', 0.214), ('sulphates', 0.165), ('volatile acidity', 0.121), ('total sulfur dioxide', 0.103), ('fixed acidity', 0.084), ('residual sugar', 0.065), ('density', 0.06), ('chlorides', 0.053), ('pH', 0.051), ('citric acid', 0.043), ('free sulfur dioxide', 0.042)]”
decision forest	100	0.25	0.998	0.638	”[('total sulfur dioxide', 0.126), ('volatile acidity', 0.122), ('fixed acidity', 0.106), ('alcohol', 0.101), ('density', 0.098), ('sulphates', 0.096), ('chlorides', 0.086), ('residual sugar', 0.075), ('citric acid', 0.07), ('free sulfur dioxide', 0.068), ('pH', 0.051)]”
decision forest	100	0.5	1.0	0.695	”[('alcohol', 0.201), ('sulphates', 0.161), ('volatile acidity', 0.137), ('total sulfur dioxide', 0.111), ('density', 0.082), ('fixed acidity', 0.07), ('chlorides', 0.057), ('citric acid', 0.052), ('pH', 0.046), ('residual sugar', 0.042), ('free sulfur dioxide', 0.039)]”
decision forest	100	0.75	1.0	0.678	”[('alcohol', 0.289), ('sulphates', 0.19), ('volatile acidity', 0.142), ('total sulfur dioxide', 0.095), ('density', 0.067), ('residual sugar', 0.055), ('chlorides', 0.04), ('fixed acidity', 0.035), ('pH', 0.035), ('citric acid', 0.029), ('free sulfur dioxide', 0.023)]”
decision forest	100	FMethodDF.RUNIF	1.0	0.698	”[('alcohol', 0.224), ('sulphates', 0.187), ('volatile acidity', 0.126), ('total sulfur dioxide', 0.099), ('density', 0.088), ('citric acid', 0.052), ('chlorides', 0.051), ('residual sugar', 0.05), ('pH', 0.047), ('fixed acidity', 0.044), ('free sulfur dioxide', 0.031)]”

Table 5: Wine-red dataset results

algorithm	n_trees	F	train_acc	test_acc	feature_importance
random forest	1	1	0.823	0.511	"[('citric acid', 0.346), ('alcohol', 0.282), ('sulphates', 0.064), ('chlorides', 0.063), ('free sulfur dioxide', 0.056), ('fixed acidity', 0.046), ('volatile acidity', 0.043), ('residual sugar', 0.039), ('pH', 0.023), ('density', 0.019), ('total sulfur dioxide', 0.018)]"
random forest	1	2	0.837	0.498	"[('volatile acidity', 0.344), ('residual sugar', 0.14), ('fixed acidity', 0.139), ('alcohol', 0.074), ('citric acid', 0.069), ('free sulfur dioxide', 0.054), ('chlorides', 0.053), ('pH', 0.047), ('density', 0.035), ('total sulfur dioxide', 0.027), ('sulphates', 0.017)]"
random forest	1	FMethodRF.LOG	0.834	0.551	"[('volatile acidity', 0.394), ('alcohol', 0.176), ('total sulfur dioxide', 0.141), ('chlorides', 0.082), ('free sulfur dioxide', 0.069), ('pH', 0.031), ('citric acid', 0.027), ('fixed acidity', 0.024), ('residual sugar', 0.023), ('sulphates', 0.021), ('density', 0.012)]"
random forest	1	FMethodRF.SQRT	0.835	0.533	"[('alcohol', 0.403), ('volatile acidity', 0.267), ('chlorides', 0.066), ('citric acid', 0.054), ('free sulfur dioxide', 0.052), ('pH', 0.042), ('residual sugar', 0.04), ('fixed acidity', 0.025), ('total sulfur dioxide', 0.023), ('sulphates', 0.019), ('density', 0.009)]"
random forest	10	1	0.985	0.64	"[('volatile acidity', 0.223), ('sulphates', 0.13), ('total sulfur dioxide', 0.124), ('residual sugar', 0.109), ('citric acid', 0.082), ('alcohol', 0.068), ('fixed acidity', 0.068), ('chlorides', 0.061), ('free sulfur dioxide', 0.056), ('density', 0.05), ('pH', 0.029)]"
random forest	10	2	0.988	0.636	"[('density', 0.149), ('free sulfur dioxide', 0.141), ('alcohol', 0.135), ('volatile acidity', 0.126), ('citric acid', 0.097), ('total sulfur dioxide', 0.093), ('chlorides', 0.079), ('pH', 0.057), ('residual sugar', 0.054), ('sulphates', 0.038), ('fixed acidity', 0.032)]"
random forest	10	FMethodRF.LOG	0.987	0.642	"[('alcohol', 0.246), ('density', 0.156), ('volatile acidity', 0.119), ('chlorides', 0.09), ('free sulfur dioxide', 0.081), ('residual sugar', 0.071), ('pH', 0.066), ('total sulfur dioxide', 0.065), ('citric acid', 0.039), ('fixed acidity', 0.035), ('sulphates', 0.032)]"
random forest	10	FMethodRF.SQRT	0.99	0.629	"[('density', 0.199), ('alcohol', 0.176), ('chlorides', 0.144), ('residual sugar', 0.083), ('total sulfur dioxide', 0.079), ('volatile acidity', 0.075), ('free sulfur dioxide', 0.072), ('citric acid', 0.068), ('sulphates', 0.043), ('fixed acidity', 0.033), ('pH', 0.027)]"
random forest	25	1	0.999	0.655	"[('volatile acidity', 0.131), ('fixed acidity', 0.11), ('density', 0.104), ('free sulfur dioxide', 0.091), ('total sulfur dioxide', 0.09), ('pH', 0.087), ('alcohol', 0.085), ('chlorides', 0.085), ('sulphates', 0.079), ('residual sugar', 0.072), ('citric acid', 0.065)]"
random forest	25	2	0.999	0.651	"[('chlorides', 0.14), ('density', 0.133), ('alcohol', 0.123), ('volatile acidity', 0.105), ('citric acid', 0.102), ('total sulfur dioxide', 0.096), ('free sulfur dioxide', 0.078), ('residual sugar', 0.074), ('pH', 0.069), ('sulphates', 0.049), ('fixed acidity', 0.029)]"
random forest	25	FMethodRF.LOG	1.0	0.65	"[('alcohol', 0.166), ('density', 0.163), ('volatile acidity', 0.109), ('chlorides', 0.097), ('free sulfur dioxide', 0.094), ('total sulfur dioxide', 0.092), ('residual sugar', 0.089), ('citric acid', 0.066), ('pH', 0.047), ('sulphates', 0.046), ('fixed acidity', 0.033)]"
random forest	25	FMethodRF.SQRT	0.999	0.647	"[('density', 0.176), ('alcohol', 0.172), ('chlorides', 0.144), ('volatile acidity', 0.122), ('free sulfur dioxide', 0.084), ('citric acid', 0.072), ('total sulfur dioxide', 0.07), ('residual sugar', 0.057), ('pH', 0.041), ('sulphates', 0.034), ('fixed acidity', 0.029)]"
random forest	50	1	1.0	0.669	"[('citric acid', 0.109), ('density', 0.107), ('volatile acidity', 0.103), ('residual sugar', 0.101), ('sulphates', 0.098), ('total sulfur dioxide', 0.088), ('fixed acidity', 0.087), ('pH', 0.086), ('chlorides', 0.084), ('free sulfur dioxide', 0.078), ('alcohol', 0.06)]"
random forest	50	2	1.0	0.657	"[('alcohol', 0.146), ('chlorides', 0.146), ('density', 0.107), ('total sulfur dioxide', 0.104), ('volatile acidity', 0.092), ('free sulfur dioxide', 0.09), ('pH', 0.084), ('citric acid', 0.083), ('residual sugar', 0.065), ('sulphates', 0.045), ('fixed acidity', 0.038)]"
random forest	50	FMethodRF.LOG	1.0	0.656	"[('alcohol', 0.154), ('density', 0.142), ('chlorides', 0.14), ('volatile acidity', 0.136), ('free sulfur dioxide', 0.094), ('total sulfur dioxide', 0.083), ('residual sugar', 0.076), ('citric acid', 0.071), ('pH', 0.044), ('sulphates', 0.032), ('fixed acidity', 0.028)]"
random forest	50	FMethodRF.SQRT	1.0	0.666	"[('alcohol', 0.211), ('chlorides', 0.128), ('density', 0.127), ('volatile acidity', 0.116), ('total sulfur dioxide', 0.078), ('citric acid', 0.076), ('free sulfur dioxide', 0.076), ('residual sugar', 0.068), ('pH', 0.052), ('fixed acidity', 0.034), ('sulphates', 0.033)]"
random forest	75	1	1.0	0.666	"[('citric acid', 0.111), ('pH', 0.11), ('density', 0.105), ('fixed acidity', 0.104), ('volatile acidity', 0.099), ('chlorides', 0.087), ('total sulfur dioxide', 0.085), ('alcohol', 0.083), ('free sulfur dioxide', 0.083), ('sulphates', 0.074), ('residual sugar', 0.06)]"
random forest	75	2	1.0	0.667	"[('chlorides', 0.145), ('density', 0.137), ('alcohol', 0.135), ('free sulfur dioxide', 0.102), ('volatile acidity', 0.095), ('citric acid', 0.084), ('residual sugar', 0.084), ('total sulfur dioxide', 0.077), ('pH', 0.061), ('sulphates', 0.044), ('fixed acidity', 0.036)]"
random forest	75	FMethodRF.LOG	1.0	0.658	"[('alcohol', 0.203), ('density', 0.146), ('volatile acidity', 0.126), ('chlorides', 0.113), ('free sulfur dioxide', 0.082), ('total sulfur dioxide', 0.072), ('residual sugar', 0.065), ('citric acid', 0.062), ('pH', 0.054), ('sulphates', 0.039), ('fixed acidity', 0.036)]"
random forest	75	FMethodRF.SQRT	1.0	0.657	"[('alcohol', 0.198), ('density', 0.167), ('volatile acidity', 0.115), ('chlorides', 0.106), ('total sulfur dioxide', 0.086), ('free sulfur dioxide', 0.08), ('pH', 0.06), ('citric acid', 0.058), ('residual sugar', 0.054), ('sulphates', 0.039), ('fixed acidity', 0.038)]"
random forest	100	1	1.0	0.662	"[('chlorides', 0.121), ('volatile acidity', 0.103), ('density', 0.098), ('sulphates', 0.096), ('total sulfur dioxide', 0.09), ('free sulfur dioxide', 0.088), ('residual sugar', 0.086), ('alcohol', 0.085), ('citric acid', 0.082), ('fixed acidity', 0.077), ('pH', 0.075)]"
random forest	100	2	1.0	0.661	"[('alcohol', 0.146), ('density', 0.13), ('volatile acidity', 0.113), ('chlorides', 0.105), ('total sulfur dioxide', 0.093), ('free sulfur dioxide', 0.089), ('residual sugar', 0.085), ('pH', 0.08), ('citric acid', 0.08), ('fixed acidity', 0.044), ('sulphates', 0.036)]"
random forest	100	FMethodRF.LOG	1.0	0.657	"[('alcohol', 0.18), ('density', 0.148), ('volatile acidity', 0.115), ('chlorides', 0.106), ('total sulfur dioxide', 0.091), ('free sulfur dioxide', 0.089), ('citric acid', 0.075), ('residual sugar', 0.069), ('pH', 0.049), ('sulphates', 0.039), ('fixed acidity', 0.037)]"
random forest	100	FMethodRF.SQRT	1.0	0.663	"[('alcohol', 0.222), ('density', 0.135), ('volatile acidity', 0.13), ('chlorides', 0.095), ('free sulfur dioxide', 0.08), ('citric acid', 0.076), ('residual sugar', 0.072), ('total sulfur dioxide', 0.07), ('pH', 0.048), ('sulphates', 0.036), ('fixed acidity', 0.036)]"
decision forest	1	0.25	0.974	0.524	"[('density', 0.693), ('total sulfur dioxide', 0.307), ('alcohol', 0.0), ('sulphates', 0.0), ('pH', 0.0), ('free sulfur dioxide', 0.0), ('chlorides', 0.0), ('residual sugar', 0.0), ('citric acid', 0.0), ('volatile acidity', 0.0), ('fixed acidity', 0.0)]"
decision forest	1	0.5	1.0	0.558	"[('alcohol', 0.585), ('free sulfur dioxide', 0.193), ('density', 0.108), ('chlorides', 0.062), ('total sulfur dioxide', 0.053), ('sulphates', 0.0), ('pH', 0.0), ('residual sugar', 0.0), ('citric acid', 0.0), ('volatile acidity', 0.0), ('fixed acidity', 0.0)]"
decision forest	1	0.75	1.0	0.579	"[('alcohol', 0.542), ('volatile acidity', 0.152), ('free sulfur dioxide', 0.123), ('density', 0.068), ('chlorides', 0.033), ('sulphates', 0.032), ('pH', 0.025), ('total sulfur dioxide', 0.024), ('residual sugar', 0.0), ('citric acid', 0.0), ('fixed acidity', 0.0)]"
decision forest	1	FMethodDF.RUNIF	1.0	0.558	"[('alcohol', 0.585), ('free sulfur dioxide', 0.193), ('density', 0.108), ('chlorides', 0.062), ('total sulfur dioxide', 0.053), ('sulphates', 0.0), ('pH', 0.0), ('residual sugar', 0.0), ('citric acid', 0.0), ('volatile acidity', 0.0), ('fixed acidity', 0.0)]"
decision forest	10	0.25	0.974	0.567	"[('density', 0.212), ('chlorides', 0.145), ('free sulfur dioxide', 0.144), ('residual sugar', 0.133), ('alcohol', 0.072), ('total sulfur dioxide', 0.072), ('fixed acidity', 0.062), ('pH', 0.061), ('volatile acidity', 0.057), ('citric acid', 0.043), ('sulphates', 0.0)]"
decision forest	10	0.5	1.0	0.632	"[('alcohol', 0.524), ('free sulfur dioxide', 0.098), ('volatile acidity', 0.09), ('pH', 0.048), ('residual sugar', 0.041), ('chlorides', 0.038), ('density', 0.036), ('citric acid', 0.033), ('total sulfur dioxide', 0.031), ('fixed acidity', 0.031), ('sulphates', 0.029)]"
decision forest	10	0.75	1.0	0.633	"[('alcohol', 0.503), ('volatile acidity', 0.109), ('free sulfur dioxide', 0.096), ('density', 0.064), ('pH', 0.043), ('chlorides', 0.042), ('citric acid', 0.037), ('residual sugar', 0.03), ('total sulfur dioxide', 0.028), ('sulphates', 0.025), ('fixed acidity', 0.024)]"
decision forest	10	FMethodDF.RUNIF	1.0	0.612	"[('alcohol', 0.293), ('density', 0.162), ('chlorides', 0.129), ('free sulfur dioxide', 0.118), ('residual sugar', 0.111), ('volatile acidity', 0.079), ('pH', 0.041), ('sulphates', 0.021), ('total sulfur dioxide', 0.018), ('fixed acidity', 0.017), ('citric acid', 0.011)]"
decision forest	25	0.25	0.981	0.581	"[('density', 0.173), ('residual sugar', 0.131), ('free sulfur dioxide', 0.126), ('chlorides', 0.106), ('volatile acidity', 0.091), ('total sulfur dioxide', 0.071), ('sulphates', 0.065), ('fixed acidity', 0.062), ('citric acid', 0.062), ('alcohol', 0.057), ('pH', 0.055)]"
decision forest	25	0.5	1.0	0.646	"[('alcohol', 0.28), ('volatile acidity', 0.117), ('density', 0.095), ('free sulfur dioxide', 0.094), ('chlorides', 0.081), ('residual sugar', 0.073), ('citric acid', 0.066), ('pH', 0.055), ('total sulfur dioxide', 0.054), ('sulphates', 0.047), ('fixed acidity', 0.039)]"
decision forest	25	0.75	1.0	0.636	"[('alcohol', 0.44), ('volatile acidity', 0.119), ('free sulfur dioxide', 0.089), ('density', 0.071), ('chlorides', 0.067), ('residual sugar', 0.044), ('citric acid', 0.038), ('sulphates', 0.036), ('total sulfur dioxide', 0.035), ('pH', 0.033), ('fixed acidity', 0.027)]"
decision forest	25	FMethodDF.RUNIF	1.0	0.642	"[('alcohol', 0.265), ('volatile acidity', 0.129), ('density', 0.125), ('chlorides', 0.122), ('free sulfur dioxide', 0.109), ('residual sugar', 0.076), ('pH', 0.048), ('total sulfur dioxide', 0.035), ('sulphates', 0.033), ('citric acid', 0.03), ('fixed acidity', 0.029)]"
decision forest	50	0.25	0.984	0.606	"[('density', 0.13), ('free sulfur dioxide', 0.114), ('chlorides', 0.112), ('citric acid', 0.109), ('total sulfur dioxide', 0.103), ('residual sugar', 0.09), ('volatile acidity', 0.089), ('alcohol', 0.085), ('fixed acidity', 0.071), ('sulphates', 0.051), ('pH', 0.048)]"
decision forest	50	0.5	1.0	0.659	"[('alcohol', 0.258), ('density', 0.106), ('free sulfur dioxide', 0.101), ('volatile acidity', 0.099), ('chlorides', 0.096), ('citric acid', 0.08), ('total sulfur dioxide', 0.069), ('residual sugar', 0.063), ('pH', 0.048), ('sulphates', 0.04), ('fixed acidity', 0.039)]"
decision forest	50	0.75	1.0	0.651	"[('alcohol', 0.402), ('density', 0.109), ('volatile acidity', 0.106), ('free sulfur dioxide', 0.1), ('chlorides', 0.063), ('residual sugar', 0.053), ('citric acid', 0.043), ('total sulfur dioxide', 0.034), ('pH', 0.034), ('sulphates', 0.029), ('fixed acidity', 0.027)]"
decision forest	50	FMethodDF.RUNIF	1.0	0.66	"[('alcohol', 0.221), ('volatile acidity', 0.122), ('free sulfur dioxide', 0.114), ('density', 0.108), ('total sulfur dioxide', 0.093), ('chlorides', 0.088), ('citric acid', 0.077), ('residual sugar', 0.063), ('sulphates', 0.046), ('pH', 0.034), ('fixed acidity', 0.033)]"
decision forest	75	0.25	0.987	0.604	"[('free sulfur dioxide', 0.123), ('chlorides', 0.114), ('total sulfur dioxide', 0.113), ('density', 0.105), ('volatile acidity', 0.102), ('residual sugar', 0.093), ('alcohol', 0.086), ('citric acid', 0.081), ('fixed acidity', 0.07), ('sulphates', 0.061), ('pH', 0.053)]"
decision forest	75	0.5	1.0	0.656	"[('alcohol', 0.241), ('free sulfur dioxide', 0.119), ('chlorides', 0.118), ('volatile acidity', 0.105), ('density', 0.091), ('total sulfur dioxide', 0.079), ('citric acid', 0.066), ('residual sugar', 0.057), ('pH', 0.049), ('fixed acidity', 0.038), ('sulphates', 0.038)]"
decision forest	75	0.75	1.0	0.656	"[('alcohol', 0.386), ('free sulfur dioxide', 0.111), ('volatile acidity', 0.111), ('density', 0.104), ('chlorides', 0.072), ('residual sugar', 0.052), ('citric acid', 0.038), ('pH', 0.035), ('total sulfur dioxide', 0.035), ('sulphates', 0.028), ('fixed acidity', 0.028)]"
decision forest	75	FMethodDF.RUNIF	1.0	0.656	"[('alcohol', 0.269), ('free sulfur dioxide', 0.142), ('volatile acidity', 0.112), ('density', 0.101), ('residual sugar', 0.077), ('chlorides', 0.075), ('total sulfur dioxide', 0.056), ('pH', 0.055), ('citric acid', 0.047), ('fixed acidity', 0.034), ('sulphates', 0.031)]"
decision forest	100	0.25	0.99	0.613	"[('density', 0.128), ('chlorides', 0.11), ('free sulfur dioxide', 0.109), ('alcohol', 0.107), ('total sulfur dioxide', 0.105), ('volatile acidity', 0.099), ('residual sugar', 0.086), ('citric acid', 0.075), ('fixed acidity', 0.064), ('pH', 0.06), ('sulphates', 0.057)]"
decision forest	100	0.5	1.0	0.66	"[('alcohol', 0.25), ('density', 0.115), ('free sulfur dioxide', 0.114), ('chlorides', 0.112), ('volatile acidity', 0.103), ('total sulfur dioxide', 0.067), ('citric acid', 0.063), ('residual sugar', 0.056), ('pH', 0.046), ('fixed acidity', 0.039), ('sulphates', 0.035)]"
decision forest	100	0.75	1.0	0.654	"[('alcohol', 0.4), ('volatile acidity', 0.112), ('free sulfur dioxide', 0.108), ('density', 0.105), ('chlorides', 0.064), ('residual sugar', 0.052), ('pH', 0.037), ('citric acid', 0.036), ('total sulfur dioxide', 0.033), ('fixed acidity', 0.028), ('sulphates', 0.027)]"
decision forest	100	FMethodDF.RUNIF	1.0	0.66	"[('alcohol', 0.299), ('free sulfur dioxide', 0.115), ('volatile acidity', 0.099), ('density', 0.083), ('residual sugar', 0.076), ('total sulfur dioxide', 0.073), ('chlorides', 0.067), ('citric acid', 0.062), ('fixed acidity', 0.052), ('pH', 0.041), ('sulphates', 0.033)]"

Table 6: Wine-white dataset results

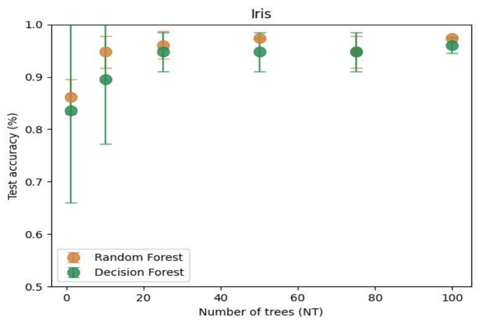
5 Discussion and Interpretations

In this section, we discuss the interpretation of the results we obtained. To facilitate the visualization and summary of the results, the following figures have been generated.

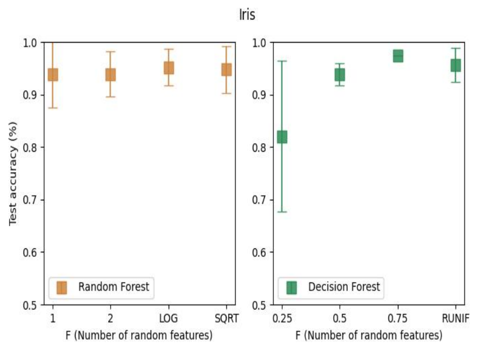
- The first figure shows the average test accuracy and its corresponding standard deviation for each number of trees and algorithm. The error bars correspond to the standard deviation.
- The second figure displays the average test accuracy and its corresponding standard deviation for each value of the variable F and for each algorithm. The error bars correspond to the standard deviation.
- The third figure presents the average feature importances for each type of algorithm, in order to compare their influence.

5.1 Iris dataset

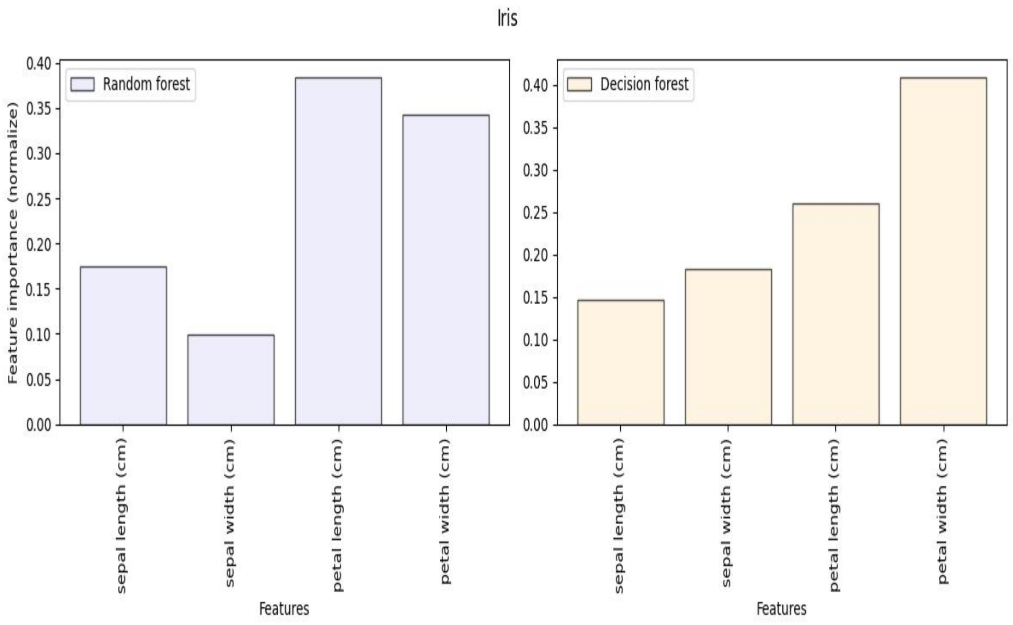
The Iris dataset performed well both in train and test. In Figure 1, it can be seen that the random forest algorithm outperforms the decision tree algorithm for lower numbers of trees, but as the number of trees increases, both algorithms tend to perform similarly. However, when using an NT of 1 or 2, both algorithms show significantly less accuracy. On the other hand, the way F is selected seems to have no effect on the random forest algorithm, but it does affect the decision forest algorithm when F is set to 25%. Generally, for both algorithms, the most relevant features are petal width and length.



(a) Number of trees (NT) vs test accuracy for each algorithm.



(b) Random feature number (F) vs test accuracy for each algorithm.



(c) Feature importance mean for each algorithm.

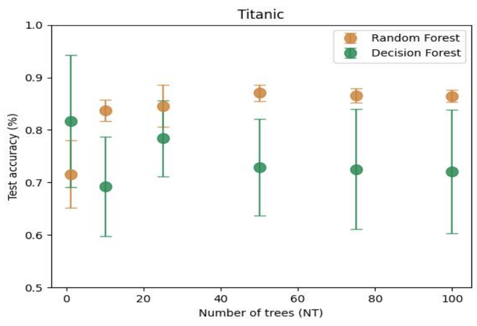
Figure 1: Iris dataset results.

5.2 Titanic dataset

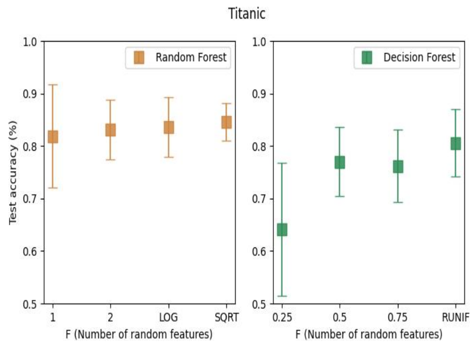
The Titanic dataset is the only dataset that includes categorical variables.

The training set has an average accuracy of 0.85, compared to the test set average of 0.80, which is reasonable and indicates that overfitting is not occurring. In Figure 2a, it can be observed that for all values of NT, the Random Forest algorithm performs better than the Decision Forest algorithm. Moreover, in Figure 2b, it can be noted that the error bars are much larger when F is equal to 1, compared to the cases of 2 or log, and that sqrt has much less standard deviation than the other methods. This means that, even when using NT of 10, 50, or 100, it is better to select F using the sqrt method. For Decision Forest, the same thing happens as with the Iris dataset: when F is 25%, the accuracy decreases, and the error bars are larger because only a quarter of the features are used to train the model. In this case, it can be seen that from 0.5, the accuracy is more or less the same, and if F uses the runif method, slightly higher accuracy can be achieved.

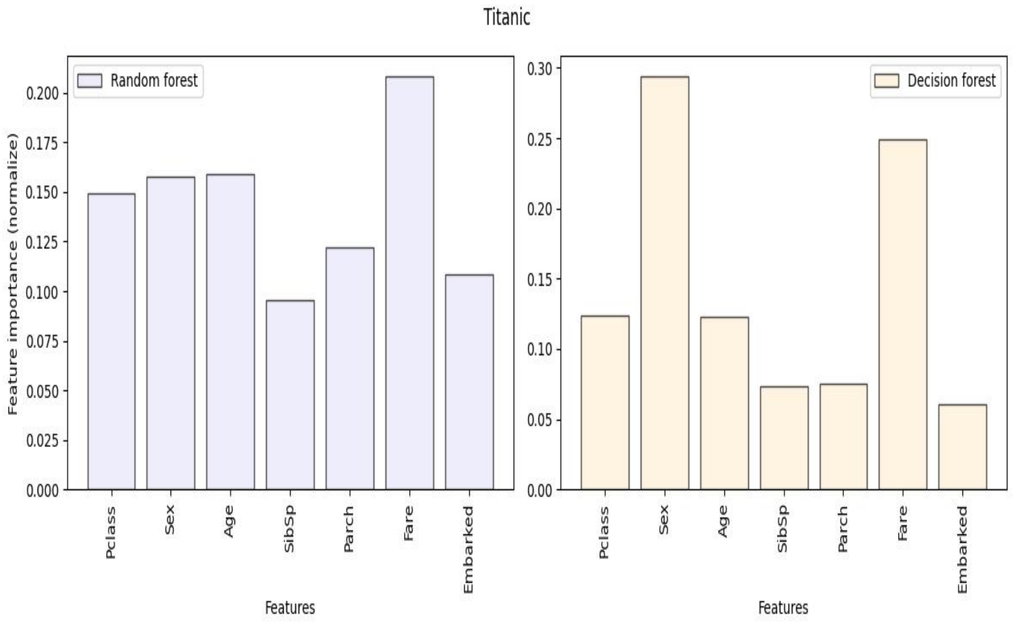
Regarding feature importance, it is evident that for the Decision Forest algorithm, the Sex and Fare features are the most relevant, while in Random Forest, there is a more equal distribution, with Fare standing out above the others (2c).



(a) Number of trees (NT) vs test accuracy for each algorithm.



(b) Random feature number (F) vs test accuracy for each algorithm.



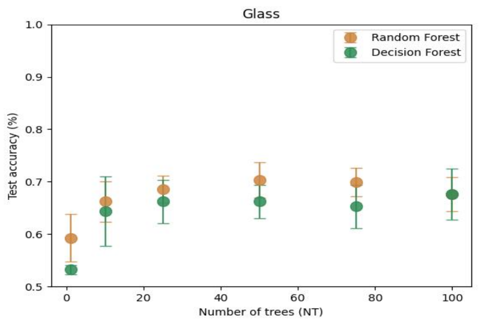
(c) Feature importance mean for each algorithm.

Figure 2: Titanic dataset results.

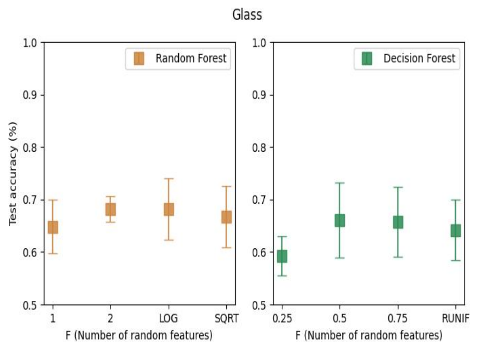
5.3 Glass dataset

Although the training precision is very high at 0.98, the test precision of 0.66 suggests the presence of overfitting.

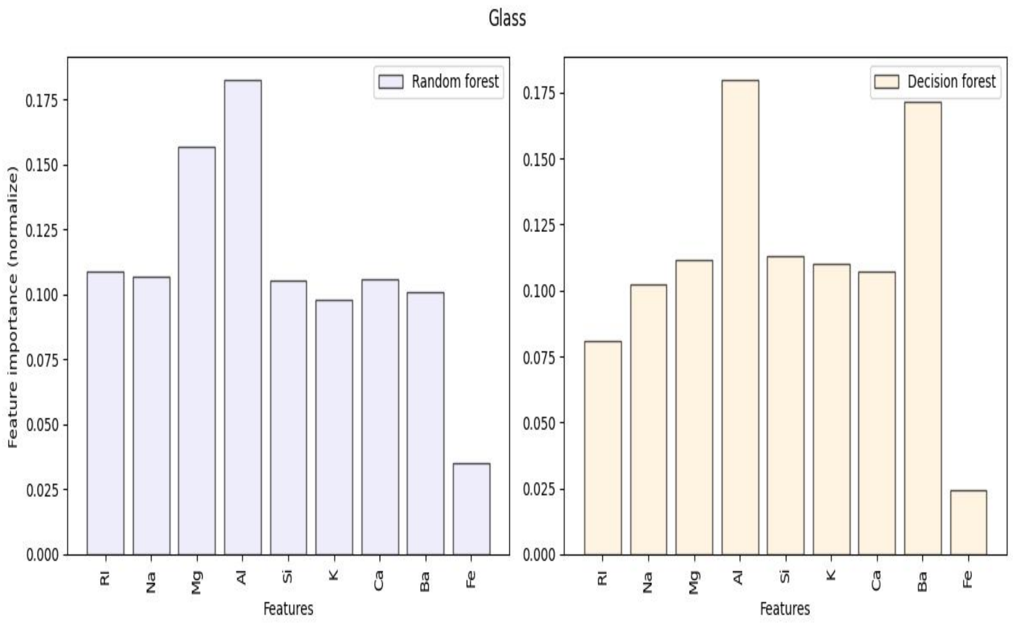
In Figure 3a, it can be observed that the accuracy increases for both algorithms from NT 25 onwards, although the trend of Random Forest to have better performance remains. In Figure 3b, the behavior of lower accuracy in the test set when F is equal to 25% in Decision Forest and when F is equal to 1 in Random Forest it continues to be observed. This makes a lot of sense since the model is trained with fewer features. Regarding the importance of the features Figure 3c, Mg and Al are highlighted in Random Forest, while in Decision Forest, Al and Ba are. Additionally, the least relevant feature is Fe for both. This means that magnesium, aluminum, and barium are the compounds that help the most in classifying the different types of glass, which we remember are for building windows, vehicle windows, containers, tableware, and headlights.



(a) Number of trees (NT) vs test accuracy for each algorithm.



(b) Random feature number (F) vs test accuracy for each algorithm.



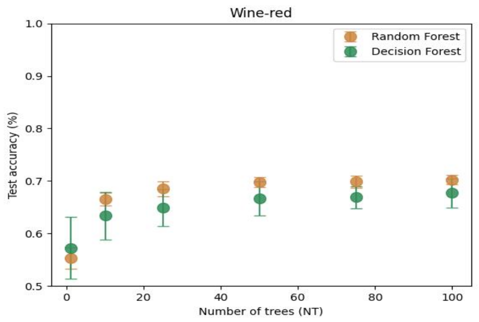
(c) Feature importance mean for each algorithm.

Figure 3: Glass dataset results.

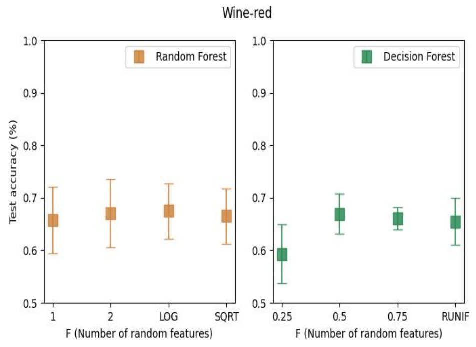
5.4 Wine-red dataset

Although the training precision is very high at 0.98, the test precision of 0.66 suggests the presence of overfitting.

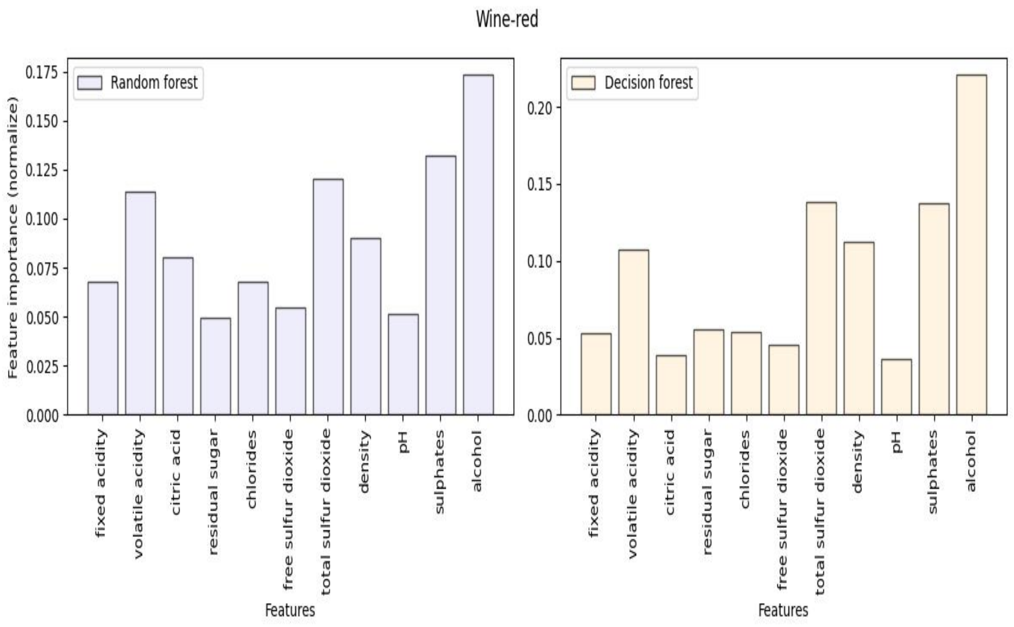
The trends in this dataset are the same as those presented in the glass dataset: Random Forest always gives us a higher accuracy in the test set than Decision Forest for all NT values, and from NT 50, the accuracy in the test set is more or less the same. This means that even if we train the model with an NT of 25 or 100, the results are similar. This is useful if, for example, we have limited capacity on our computer, as we could evaluate the model with 25 trees and still get good results. In Figure 4b, it can be seen that the same thing happens for Decision Forest as in the other datasets for F equal to 25%, but for example, the error bars for F of 0.75 are quite small, indicating that if we train our model with 75% of the features, even if we use NT of 10, 50 or 100, the results are similar. Regarding the importance of features for both algorithms, the degree of alcohol, sulfates, total sulfur dioxide, and volatile acidity stand out, which makes sense since the degree of alcohol is an important variable for distinguishing what type of wine it is.



(a) Number of trees (NT) vs test accuracy for each algorithm.



(b) Random feature number (F) vs test accuracy for each algorithm.



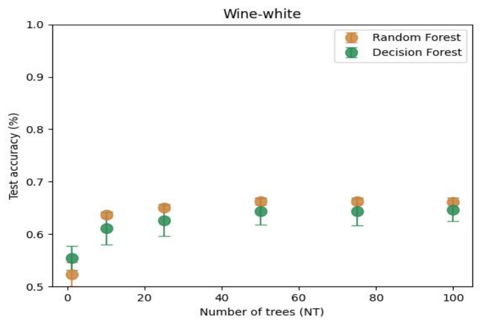
(c) Feature importance mean for each algorithm.

Figure 4: Wine-red dataset results.

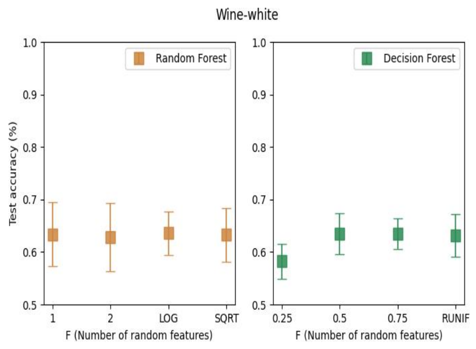
5.5 Wine-white dataset

Although the training precision is very high at 0.98, the test precision of 0.63 suggests the presence of overfitting.

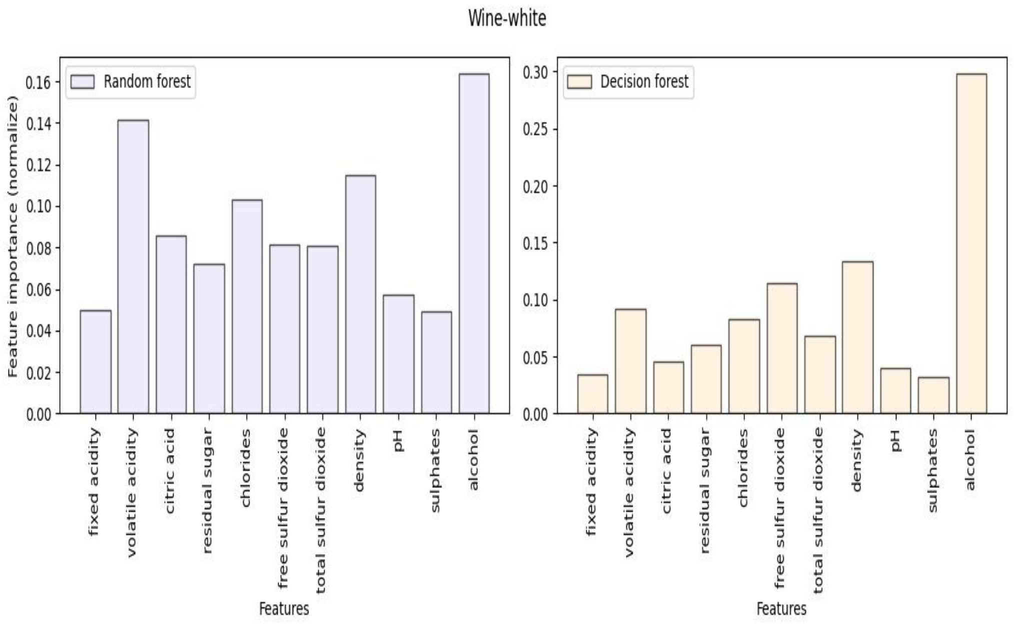
In this dataset, it is very interesting to note that the error bars are small for both algorithms, which means that the results are very similar regardless of whether we train the model with F equal to 1, 2, 50%, or any other value depending on the algorithm. It is repeated that Random Forest performs better than Decision Forest. It is also observed that the test accuracy increases up to NT 50 and then remains more or less constant. In Figure 5b, it can be seen that the log and sqrt methods have small error bars for Random Forest, while for Decision Forest, the F value equal to 0.75 has small error bars. As for the importance of the features, in Random Forest, two features stand out above the others: alcohol and volatile acidity, while in Decision Forest, alcohol is the most important feature.



(a) Number of trees (NT) vs test accuracy for each algorithm.



(b) Random feature number (F) vs test accuracy for each algorithm.



(c) Feature importance mean for each algorithm.

Figure 5: Wine-white dataset results.

6 Conclusions

Taking into account the five datasets, we can draw the following conclusions:

- Random Forest performs better than Decision Forests.
- It is not recommended to use NT values of 1 or 2 as it limits the potential of ensemble algorithms by only having 1 or 2 trees to vote.
- From a value of NT equal to 50, the accuracy in the set of tests remains more or less constant, so it is recommended to use this value as a minimum.
- Using log or sqrt methods to find the F value in Random Forest is much better than using 1 or 2.
- A value of F equal to 25% for Decision Forest has poor performance, and it is recommended to use at least 50% of the features.

- The importance of the features is similar regardless of the algorithm used. This variable is useful for understanding which feature is more relevant when training the model.
- Implementing an algorithm from scratch is an excellent exercise for gaining a deeper understanding of the AI models.

7 References

[1] Ho, T. The random subspace method for constructing decision forests. *Pattern Analysis And Machine Intelligence.* **20**, 832-844 (1998)

[2] Breiman, R. & Learning, M. 45(1). (5-32,2001)

[3] Breiman, L., Friedman, J., Olshen, R. & Stone, C. Classi- fication and Regression Trees. (Wadsworth,1984)

[4] UCI Machine Learning Repository. url: https://archive.ics.uci.edu/ml/index.php.