City University of Hong Kong
2024 - 2025 Semester B
CS3481 Fundamentals of Data Science

Random Forest and Naive Bayes
Classifier Assignment

Report

1. Introduction

In this assignment, we construct multiple decision trees using the Wine dataset to classify wine samples into three distinct classes. Each decision tree is configured with different train-test split ratios, and keeps the impurity measures constant. The models are evaluated by comparing their tree structure, classification accuracy, and confusion matrices to identify patterns in misclassification.

2. Dataset Description

- **Features**
  - 13 chemical attributes (e.g., alcohol, malic acid, flavonoids).
- **Classes**
  - 3 types of wines (Class 0, 1, 2).
- **Class Distribution**
  - Class 0: 59 instances
  - Class 1: 71 instances
  - Class 2: 48 instances
  - Total: 178 instances

3. Results and Analysis

3.1. Task (a): Constructing Random Forest Models with Varying Number of Trees

For all tasks below, the wine dataset was partitioned into training and test sets with ratios of 7:3 (test_size = 0.3). The data preparation:

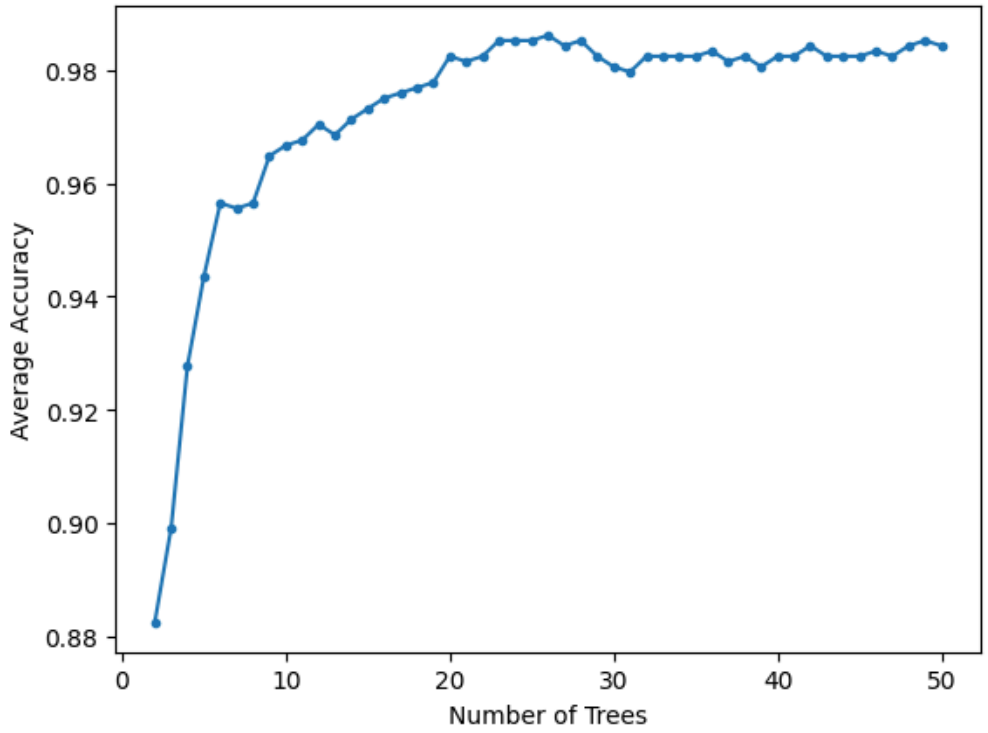
```
# Data Preparation
wine_dataset = load_wine()

X = wine_dataset.data
Y = wine_dataset.target

X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.3, random_state=42)
```

Since the wine dataset is small, a test range with 2–50 trees is chosen for constructing different random forest models in this task. Testing up to 50 trees can balance thoroughness and efficiency. Also, 20 loops are added on each random forest model and calculate the average accuracy of that model, in order to account for randomness in bootstrap sampling. Also, maximum depth of trees are set to 3 to prevent overfitting individual trees.

The average accuracy of random forest model with different number of tree:



Considering 2 to 26 trees, we can see the average accuracy of random forest models is increasing with the number of trees in the model. But after 26 trees, the average accuracy fluctuates around 0.98, which has no further improvement with increasing the number of trees.

We can find that the model achieves peak performance at 26 trees, which has 98.6% average accuracy for the testing data. Therefore, the optimal n_estimators is 26.

3.2. Task (b): Comparing Performance of Individual Trees With Original Random Forest Model

From task (a), we found the optimal n_estimators is 26. In this task, 4 individual trees in the model will be selected for comparing.

The performance of the random forest model with n_estimators=26 and max_depth=3:

Training Accuracy	99.19354838709677%
Forest Testing Accuracy	98.14814814814815%
Average Tree Accuracy	84.75783475783475%
Median Tree Accuracy	85.18518518518519%

Individual Tree Accuracies (Selected Examples):

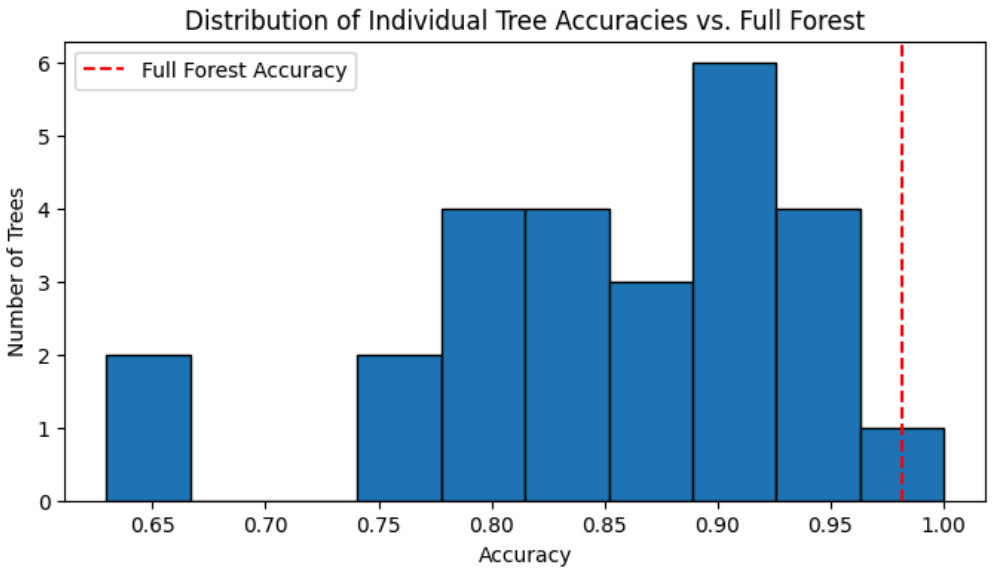
Tree	**Accuracy**
Tree 7	74.07407407407407%
Tree 12	62.96296296296297%
Tree 13	94.44444444444444%
Tree 21	100%

Accuracies ranged from 62.96% (Tree 12) to 100% (Tree 21), showing the impact of randomness in bootstrap sampling and feature selection.

While Tree 21 achieved 100% accuracy on the test set, this is likely due to overfitting its specific bootstrap sample. The forest's 98.15% accuracy is more trustworthy than Tree 21. While Tree 12 achieved 62.96% accuracy on the test set, this is likely due to the poor feature selection or unlucky sampling.

For the 4 selected trees, there are 3 trees which have lower accuracy than the forest’s accuracy. It seems the random forest model may have higher accuracy than most of the individual trees.

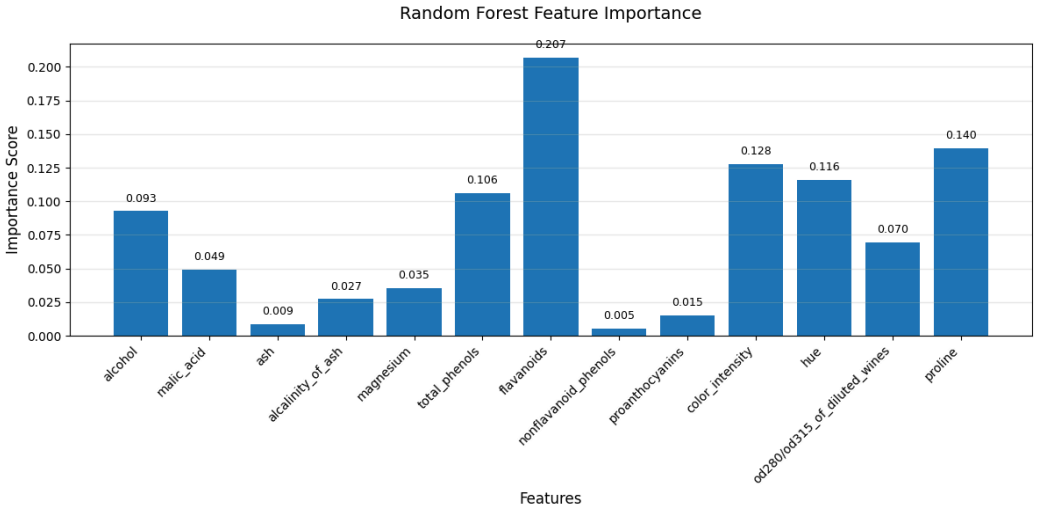
The result of comparing performance of all individual trees with full forest:



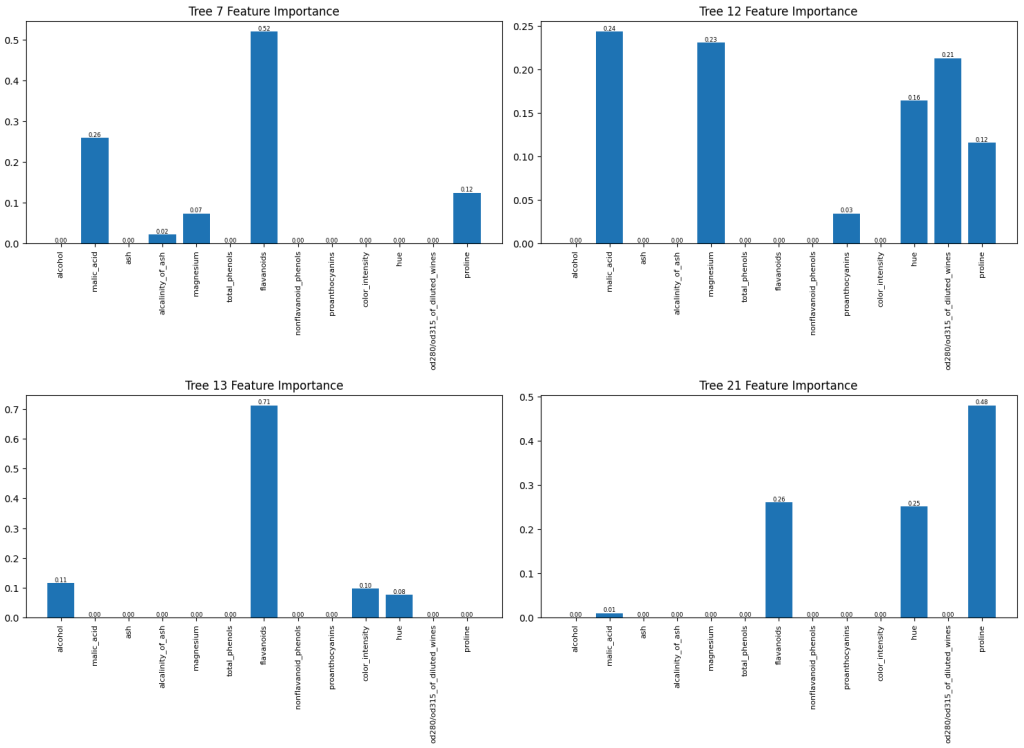
Almost all trees (25/26) performed below the forest's accuracy. From the diagram, the random forest significantly outperformed the average individual tree. It seems the model will always prefer the full forest over any single tree, since most trees perform poorly alone.

3.3. Task (c): Comparing Feature Importances

Distribution of full forest's feature importance:



Distribution of each selected tree's feature importance:



Top 3 feature importances of the full forest and the 4 selected trees:

Model	**1st (Weight)**	**2nd (Weight)**	**3rd (Weight)**
Full Forest	flavanoids (0.21)	proline (0.14)	color_intensity (0.13)
Tree 7	flavanoids (0.52)	malic_acid (0.26)	proline (0.12)
Tree 12	malic_acid (0.24)	magnesium (0.23)	od280/od315_of_diluted_wines (0.21)
Tree 13	flavanoids (0.71)	alcohol (0.11)	color_intensity (0.10)
Tree 21	proline (0.48)	flavanoids (0.26)	hue (0.25)

For the full forest, it has balanced importance across features, with the importances of flavanoids, proline and color_intensity within the range 0.13 to 0.21. For Tree 13, it is dominated by flavanoids with 0.71 weight, ignoring other predictive features. For the Tree 21, it seems to be over-relies on proline with 0.48 weight, likely leading to overfitting.

The feature “flavanoids” seems to be the most important in the classification, since it has the highest appearance frequency among all features in the 4 selected trees. Therefore, the top 1 feature importances of the full forest is flavanoids.

It can be observed that the full forest combines many trees' views to ease the problem of being unreliable on using only the individual tree, ensuring more stable performance of the model.

3.4. Task (d): Comparing Naive Bayes vs. Random Forest

Random Forest: Use the best model from Task (a) (n_estimators=26, max_depth=3)
Naïve Bayes: Train a Gaussian Naïve Bayes classifier (default parameters).

The performance of random forest model and naive bayes model:

Model	Training Accuracy	Test Accuracy
Random Forest	99.19354838709677%	98.14814814814815%
Naive Bayes	97.58064516129032%	100%

Naïve Bayes showed perfect classification on the test set despite slightly lower training accuracy with 97.58%, suggesting exceptional alignment with the test data's characteristics. Random Forest showed consistent performance with 99.19% training accuracy and 98.15% test accuracy, indicating robust generalization.

Confusion matrix on Random Forest:

```
[19  0  0]
[ 0 20  1]
[ 0  0 14]
```

Confusion matrix on Naïve Bayes:

```
[19  0  0]
[ 0 21  0]
[ 0  0 14]
```

It shows the single misclassification occurred in Class 1, which Random Forest confused with Class 2. Also, Class 0 is perfectly separable by both models.

In conclusion, the perfect result on testing using Naïve Bayes should be contextualized. Random Forest ensures stable train-test performance and ability to capture feature interactions, leading to more generally reliable results for most scenarios.