Universitat Politècnica de Catalunya

Facultat d'Informàtica de Barcelona

SEL: ENSEMBLE OF CLASSIFIERS

Author:

Marc Pascual Roig

March 2024

Contents

1 Introduction

The aim of this work is to analyse and evaluate the Decision Forest and Random Forest classifiers with different hyper parameters configurations. For that purpose, we have implemented the classifiers using a CART Decision Tree, which can handle both numerical and categorical features.

In this paper, two aspects are subject of study - first, evaluate the performance of the classifiers using different datasets of the UCI Repository; and second, infer and analyse the feature importance in each dataset.

2 Datasets

The datasets chosen for this project are the Iris dataset, the Breast Cancer Wisconsin (Diagnostic) dataset and the Mushroom dataset.

In the following table we show the metadata of the datasets.

Dataset	**N**	**Num Features**	**Type**	**Class percentages**
Iris	150	4	Numerical	33.3% - 33.3% - 33.3%
WDBC	569	30	Numerical	62.6% - 37.3%
Mushroom	8124	22	Categorical	51.8% - 48.2%

Table 1: Dimension and feature information of the datasets.

3 Algorithms

The code implemented consists of three algorithms. The CART Decision Tree algorithm (see Algorithms 1, 2 and 3) is the core algorithm, from which the binary decision trees are built using the *Gini index*.

The Gini index is an impurity measure used to evaluate the quality of a split at a node in a tree. It quantifies the probability of misclassifying a randomly chosen data point's class label within that node. Mathematically, the Gini index Gini(t) for a node t with c classes is calculated as

$$Gini(t) := 1 - \sum_{i=1}^{c} p(i|t)^2.$$

A lower Gini index indicates higher node purity, with most data points belonging to the same class, guiding the selection of optimal splits during tree construction. The CART algorithm seeks splits that minimize the weighted sum of Gini indices across resulting child nodes, promoting the creation of homogeneous subsets for accurate classification.

We consider the following as possible binary splits in a node tree. For categorical features, we consider all possible 2 subsets splits of the unique values, it yields a total of $2^{v_F-1}-1$ splits for each feature, where v_F is the number of unique values for the feature F. For numerical features, we consider all midpoint values for all adjacent values because it only makes sense that each split is a connected component of the original values, it yields a total of $v_F - 1$ splits for each feature.

Observe that the number of possible splits grows linearly against the number of unique values when handling numerical features but it grows exponentially for categorical features. Moreover, at each splitting of the tree we are reducing lots of numerical values (we split them in two sets) while the number of possible splits of categorical values is still very large in the son nodes. Hence, we should be aware of that when dealing with categorical features with lots of unique values.

Algorithm 1 CART Decision Tree

```
Require: NumFeatures, NumRandomFeatures, X=dataset
1: Tree ← $\emptyset$
2: FeatureFreqs ← 0
3: Tree = BuildTree(X)
4: return Tree, FeatureFreqs
```

Algorithm 2 Build Tree Function

```
Require: FeatureFreqs
 1: function BuildTree(X)
 2:     classes ← unique classes of X
 3:     if len(classes)==1 then
 4:         return classes
 5:     end if
 6:     Feature, Combination, LeftTree, RightTree, Impurity ← BestSplit(X)
 7:     if Feature is None then
 8:         return MostCommonClass(X)
 9:     end if
10:     Update FeatureFreqs[Feature]
11:     LeftTree ← BuildTree(LeftTree)
12:     RightTree ← BuildTree(RightTree)
13:     return Feature, Combination, LeftTree, RightTree
14: end function
```

Then, both the Decision Forest algorithm and the Random Forest algorithm use an ensemble of CART Decision Trees in different fashions.

On the one hand, in the Random Forest classifier (see Algorithm 4), the Decision Trees are generated from a bootstrapped sample of the original dataset. Moreover, at each node o the tree the number of features considered for the split is a random choice of features.

On the other hand, in the Decision Forest (see Algorithm 5) each Decision Tree use the same subset of features throughout all the nodes and the samples chosen are the whole dataset.

Algorithm 3 Best Split

```
Require: TotalFeatures, NumRandomFeatures
 1: function BestSplit(X)
 2:     BestImpurity ← CurrentNodeImpurity
 3:     if NumRandomFeatures > 0 then
 4:         Features ← Random(TotalFeatures, NumRandomFeatures)
 5:     else
 6:         Features ← TotalFeatures
 7:     end if
 8:     for each i in range(Features) do
 9:         Combinations[i] ← All possible splits of unique values of feature i
10:     end for
11:     for feature in Features do
12:         for combination in Combinations[features] do
13:             Impurity ← Gini index according to the resulting split
14:             LeftTree ← LeftTree resulting from the split of the partition
15:             RightTree ← RightTree resulting from the split of the partition
16:             if Impurity < BestImpurity then
17:                 BestIndex ← feature
18:                 BestCombination ← combination
19:                 BestLeftTree ← LeftTree
20:                 BestRightTree ← RightTree
21:                 BestImpurity ← Impurity
22:             end if
23:         end for
24:     end for
25:     return BestIndex, BestCombination, BestLeftTree, BestRightTree, BestImpurity
26: end function
```

Algorithm 4 Random Forest

```
Require: NT=number of trees, F=number of features used in each node, X=dataset, TotalFeatures
 1: Estimators ← ∅
 2: FeatureFreq ← 0
 3: for i in range(NT) do
 4:     X' ← random bootstrapped sample of X
 5:     Tree, frequencies ← CART DecisionTree(X', TotalFeatures, F)
 6:     Estimators = Estimators + Tree
 7:     FeatureFreq = FeatureFreq + frequencies
 8: end for
 9: return Estimators, FeatureFreq
```

4 Methodology

We will evaluate the algorithms using a 70% - 30% split for the training and test sets, respectively. We use a random seed throughout the experiments for reproducibility.

The general procedure will consist of different configurations of the fit method of both classifiers, which will be performed on the above commented datasets. Namely, we will execute both

Algorithm 5 Decision Forest Fit method

```
Require: NT=number of trees, F=number of features used in each node, X=dataset, TotalFeatures
 1: Estimators ← $\emptyset$
 2: FeatureFreq ← 0
 3: for i in range(NT) do
 4:     RandomFeatures ← Random(TotalFeatures, F)
 5:     X' ← X[:, RandomFeatures]
 6:     Tree, frequencies ← CART DecisionTree(X', F)
 7:     Estimators = Estimators + Tree
 8:     FeatureFreq = FeatureFreq + frequencies
 9: end for
10: return Estimators, FeatureFreq
```

classifiers with combinations of the **NT** (Number of trees that configure the ensemble) and **F** (Number of features) parameters. Each execution will result in a set of frequencies of the appearance of the features in the trees.

The accuracy of the predictions of the test set against the set of hyper parameters will be subject of study, as well as the importance of each feature in the decision process. In order to evaluate the importance of each feature we will take into account the number of remaining samples at each node weighted by the decrease of impurity:

$$Importance(i) := \sum_{j \in nodes\ containing\ i} \Delta Gini\ index \times Num\ Samples\ at\ j.$$

Regarding the prediction method, we have taken into account the following considerations. In a given node of a decision tree, if the value of the feature of the instance that we are classifying does not match any of the values in the node of the tree, then we continue with the node which contains more instances. Notice that this happens if a value has not been seen in the training set.

To determine the final class of the ensemble we use the majority class of the set of trees as the voting policy.

For each dataset, we will analyse the following set of hyper parameters. Regarding the Number of Trees (**NT**) of the ensembles we will try:

$$1, 10, 25, 50, 75, 100.$$

Regarding the Decision Forest, the number of features used in each Decision Tree will be:

$$Int(\frac{M}{4}), int(\frac{M}{2}), int(\frac{3M}{4}), Random(1,\ M),$$

where M is the number of features of the dataset.

Regarding the Random Forest, the number of features used in each node of the tree will be:

$$1, 2, Int(log_2(M) + 1), int(\sqrt{M}),$$

where M is the number of features of the dataset.

That yields a total of 48 executions for each dataset.

5 Results

5.1 Iris dataset

We see that both classifiers achieve a stable performance when $NT \geq 10$ and $F \geq 2$. See Tables 2 and 3. When these values are achieved, we do not see any improvement in the performance. Apart from that, the only noticeable difference between the algorithms is that the Random Forest performs better when only 1 feature is selected. This is expected because the Decision Forest lacks of explainability since only one single feature is used throughout the whole tree, whereas Random Forest pick the feature at random in each node. Selecting only one feature is risky because the algorithm might select irrelevant features and get stacked quickly.

Almost all the configurations lead to the same ordering of the features (See Tables 4 and 5): petal length - petal width - sepal width - sepal length. Features describing *Petal* are definitely more important than features describing *Sepal*, and it seems that features describing *width* are more important than the ones describing *length*, although there is some discrepancy regarding *sepal*. We conclude that the differences among the 3 different iris classes are found in the petal characteristics rather than the sepal.

Num Trees	**F=1**	**F=2**	**F=3**
1	75.56	82.22	93.33
10	91.11	91.11	91.11
25	91.11	91.11	91.11
50	91.11	91.11	91.11
75	88.89	91.11	91.11
100	88.89	91.11	91.11

Table 2: Accuracy scores with Random Forest of the Iris dataset.

Num Trees	**F=1**	**F=2**	**F=3**	**F=random**
1	51.11	93.33	91.11	51.11
10	73.33	91.11	91.11	91.11
25	75.56	91.11	91.11	91.11
50	75.56	91.11	91.11	91.11
75	80.0	91.11	91.11	91.11
100	80.0	91.11	91.11	91.11

Table 3: Accuracy scores with Decision Forest of the Iris dataset.

NT, F	**Features ordering**
1, 1	sepal length - sepal width - petal length - petal width
1, 2	petal length - petal width - sepal width - sepal length
1, 3	petal width - petal length - sepal width - sepal length
10, 1	petal length - sepal length - petal width - sepal width
10, 2	petal length - petal width - sepal length - sepal width
10, 3	petal width - petal length - sepal width - sepal length
25, 1	petal length - petal width - sepal length - sepal width
25, 2	petal length - petal width - sepal length - sepal width
25, 3	petal length - petal width - sepal width - sepal length
50, 1	petal length - petal width - sepal length - sepal width
50, 2	petal length - petal width - sepal length - sepal width
50, 3	petal length - petal width - sepal width - sepal length
75, 1	petal length - petal width - sepal length - sepal width
75, 2	petal length - petal width - sepal length - sepal width
75, 3	petal length - petal width - sepal width - sepal length
100, 1	petal length - petal width - sepal length - sepal width
100, 2	petal length - petal width - sepal length - sepal width
100, 3	petal length - petal width - sepal width - sepal length

Table 4: Features of the Iris dataset ordered by importance according to Random Forest.

NT, F	Features ordering
1, 1	sepal length - petal width - petal length - sepal width
1, 2	petal width - sepal length - petal length - sepal width
1, 3	petal width - sepal width - sepal length - petal length
1, random	sepal length - petal width - petal length - sepal width
10, 1	sepal length - petal length - petal width - sepal width
10, 2	petal length - petal width - sepal length - sepal width
10, 3	petal length - petal width - sepal width - sepal length
10, random	petal length - petal width - sepal length - sepal width
25, 1	sepal length - petal length - petal width - sepal width
25, 2	petal length - petal width - sepal length - sepal width
25, 3	petal length - petal width - sepal width - sepal length
25, random	petal length - petal width - sepal length - sepal width
50, 1	sepal length - petal length - petal width - sepal width
50, 2	petal length - petal width - sepal length - sepal width
50, 3	petal length - petal width - sepal length - sepal width
50, random	petal length - petal width - sepal length - sepal width
75, 1	petal length - sepal length - petal width - sepal width
75, 2	petal length - petal width - sepal length - sepal width
75, 3	petal length - petal width - sepal length - sepal width
75, random	petal length - petal width - sepal length - sepal width
100, 1	petal length - petal width - sepal length - sepal width
100, 2	petal length - petal width - sepal length - sepal width
100, 3	petal length - petal width - sepal width - sepal length
100, random	petal length - petal width - sepal length - sepal width

Table 5: Features of the Iris dataset ordered by importance according to Decision Forest.

5.2 Breast Cancer Wisconsin (Diagnostic)

Similarly, we do not observe noticeable changes in the performance of the classifiers (see Table 6 and 7) when $F > 1$ or $NT \geq 10$. Observe that Random Forest still manages to perform well even when we only consider 1 feature out of the 30 features of the dataset.

The set of features is composed by 10 different features computed for 3 different cells nucleus types each. Briefly, these are the features descriptions:

1. **Radius:** Mean of distances from the center to points on the perimeter.
2. **Texture:** Standard deviation of gray-scale values.
3. **Perimeter:** Total length of the boundary of the object.
4. **Area:** Total area of the object.
5. **Smoothness:** Local variation in radius lengths.
6. **Compactness:** $\frac{\text{Perimeter}^2}{\text{Area}} - 1.0$
7. **Concavity:** Severity of concave portions of the contour.
8. **Concave Points:** Number of concave portions of the contour.
9. **Symmetry:** Measure of symmetry.
10. **Fractal Dimension:** ”Coastline approximation” - 1, a measure of the complexity of the object's perimeter.

Then, the diagnosis is either Benign or Malignant.

The features that appear to be more relevant according to both classifiers are:

- 22: perimeter3

- 23: area3
- 20: radius3
- 27: concave_points3
- 7: concave_points1
- 3: area1
- 2: perimeter1

Again, we find some slight discrepancies but most of the models agree on these features being the most important ones, both in the Random Forest and Decision Forest. See Table 8 and 9 for details. Therefore, we infer that cells nucleus of type 3 and 1 are the discriminant nucleus in the decision process, while nucleus of type 2 do not give us evidence for a diagnosis. Being the perimeter, area, radius and concave points the attributes that determine whether the diagnosis is Benign or Malignant. See how Figure 2 has 4 features highlighted as important for cell nucleus 3; and 4 features highlighted as important - less important than cell 3 - for cell nucleus 1.

We also remark that - given that we are dealing with a medical dataset - we should take into account the number of False negatives in order to have a complete evaluation of the models in addition to the accuracy results shown.

Num Trees	**F=1**	**F=2**	**F=5**	**F=7**
1	89.47	90.64	94.15	90.06
10	95.32	95.32	96.49	95.91
25	95.91	97.08	95.91	94.74
50	95.32	96.49	95.32	95.91
75	95.32	96.49	95.32	95.91
100	94.74	96.49	95.91	95.32

Table 6: Accuracy scores with Random Forest of the WDBC dataset.

Num Trees	**F=7**	**F=15**	**F=22**	**F=random**
1	92.98	93.57	89.47	92.98
10	94.74	94.15	94.15	93.57
25	94.15	95.32	94.74	93.57
50	96.49	95.91	94.15	94.15
75	96.49	94.74	93.57	94.15
100	96.49	94.15	94.15	94.15

Table 7: Accuracy scores with Decision Forest of the WDBC dataset.

NT, F	Features ordering
1, 1	20 - 8 - 23 - 3 - 21
1, 2	20 - 23 - 21 - 13 - 2
1, 5	23 - 24 - 1 - 21 - 7
1, 7	23 - 22 - 1 - 21 - 10
10, 1	2 - 20 - 21 - 12 - 27
10, 2	7 - 3 - 20 - 13 - 2
10, 5	23 - 22 - 27 - 3 - 2
10, 7	22 - 20 - 23 - 27 - 7
25, 1	23 - 2 - 20 - 27 - 7
25, 2	20 - 7 - 3 - 23 - 2
25, 5	22 - 23 - 27 - 3 - 0
25, 7	22 - 20 - 27 - 23 - 7
50, 1	2 - 22 - 20 - 10 - 7
50, 2	20 - 3 - 22 - 7 - 23
50, 5	22 - 23 - 27 - 20 - 6
50, 7	22 - 23 - 20 - 27 - 7
75, 1	2 - 20 - 3 - 22 - 23
75, 2	20 - 3 - 22 - 23 - 2
75, 5	22 - 23 - 20 - 27 - 6
75, 7	22 - 23 - 20 - 27 - 7
100, 1	2 - 20 - 22 - 3 - 23
100, 2	20 - 22 - 23 - 2 - 7
100, 5	22 - 20 - 23 - 27 - 3
100, 7	22 - 23 - 27 - 20 - 7

Table 8: Top 5 features of the WDBC dataset ordered by importance according to Random Forest.

NT, F	Features ordering
1, 7	20 - 7 - 24 - 23 - 3
1, 15	20 - 27 - 1 - 24 - 7
1, 22	20 - 27 - 21 - 11 - 3
1, random	22 - 21 - 7 - 3 - 17
10, 7	22 - 23 - 20 - 7 - 3
10, 15	22 - 20 - 23 - 27 - 21
10, 22	22 - 20 - 27 - 21 - 1
10, random	22 - 3 - 20 - 6 - 2
25, 7	22 - 20 - 23 - 7 - 3
25, 15	22 - 20 - 23 - 27 - 7
25, 22	22 - 20 - 27 - 21 - 23
25, random	22 - 23 - 20 - 27 - 21
50, 7	23 - 22 - 20 - 27 - 7
50, 15	22 - 20 - 23 - 27 - 7
50, 22	22 - 20 - 27 - 23 - 21
50, random	22 - 20 - 27 - 23 - 21
75, 7	22 - 27 - 23 - 20 - 7
75, 15	22 - 20 - 23 - 27 - 7
75, 22	22 - 20 - 27 - 23 - 21
75, random	22 - 20 - 27 - 23 - 21
100, 7	27 - 23 - 22 - 20 - 7
100, 15	22 - 20 - 23 - 27 - 7
100, 22	22 - 20 - 27 - 21 - 23
100, random	22 - 20 - 27 - 23 - 21

Table 9: Top 5 features of the WDBC dataset ordered by importance according to Decision Forest.

5.3 Mushroom

For the mushroom dataset, the Random Forest has a flawless performance when $F > 1$ and $NT \geq 2$. The Decision Forest has similar results, although being less consistent throughout the executions with different hyperparameters. See Table 10 and 11.

The set of categorical features is the following:

1. **cap-shape:** Shape of the mushroom cap (bell=b, conical=c, convex=x, flat=f, knobbed=k,

sunken=s).

2. **cap-surface:** Surface texture of the mushroom cap (fibrous=f, grooves=g, scaly=y, smooth=s).
3. **cap-color:** Color of the mushroom cap (brown=n, buff=b, cinnamon=c, gray=g, green=r, pink=p, purple=u, red=e, white=w, yellow=y).
4. **bruises?:** Presence of bruises on the mushroom (bruises=t, no=f).
5. **odor:** Odor of the mushroom (almond=a, anise=l, creosote=c, fishy=y, foul=f, musty=m, none=n, pungent=p, spicy=s).
6. **gill-attachment:** Attachment type of gills to the stem (attached=a, descending=d, free=f, notched=n).
7. **gill-spacing:** Spacing between gills (close=c, crowded=w, distant=d).
8. **gill-size:** Size of the gills (broad=b, narrow=n).
9. **gill-color:** Color of the gills (black=k, brown=n, buff=b, chocolate=h, gray=g, green=r, orange=o, pink=p, purple=u, red=e, white=w, yellow=y).
10. **stalk-shape:** Shape of the mushroom stalk (enlarging=e, tapering=t).
11. **stalk-root:** Root type of the stalk (bulbous=b, club=c, cup=u, equal=e, rhizomorphs=z, rooted=r, missing=?).
12. **stalk-surface-above-ring:** Surface texture of the stalk above the ring (fibrous=f, scaly=y, silky=k, smooth=s).
13. **stalk-surface-below-ring:** Surface texture of the stalk below the ring (fibrous=f, scaly=y, silky=k, smooth=s).
14. **stalk-color-above-ring:** Color of the stalk above the ring (brown=n, buff=b, cinnamon=c, gray=g, orange=o, pink=p, red=e, white=w, yellow=y).
15. **stalk-color-below-ring:** Color of the stalk below the ring (brown=n, buff=b, cinnamon=c, gray=g, orange=o, pink=p, red=e, white=w, yellow=y).
16. **veil-type:** Type of veil covering the gills (partial=p, universal=u).
17. **veil-color:** Color of the veil (brown=n, orange=o, white=w, yellow=y).
18. **ring-number:** Number of rings on the stalk (none=n, one=o, two=t).
19. **ring-type:** Type of ring on the stalk (cobwebby=c, evanescent=e, flaring=f, large=l, none=n, pendant=p, sheathing=s, zone=z).
20. **spore-print-color:** Color of the spore print (black=k, brown=n, buff=b, chocolate=h, green=r, orange=o, purple=u, white=w, yellow=y).
21. **population:** Abundance of mushrooms (abundant=a, clustered=c, numerous=n, scattered=s, several=v, solitary=y).
22. **habitat:** Habitat where the mushroom is typically found (grasses=g, leaves=l, meadows=m, paths=p, urban=u, waste=w, woods=d).

And then the mushroom is either labelled as edible or poisonous.

Regarding the importance of the features, we see that the features *odor, spore-print-color, stalk-root, gill-size or gill-color* are chosen as the most important ones by both classifiers. Actually, *odor and spore-print-color* are consistently chosen as the first and second most recurrent features, and *odor* being clearly the favorite of Decision Forest (see 3). In Figure 4 we see how there are features that are not even selected at all by the Random Forest. It appears that the set of discriminant features is a reduced set of the original features, which is enough to tell whether the mushroom is edible or poisonous.

Num Trees	**F=1**	**F=2**	**F=4**	**F=5**
1	96.6	100.0	99.71	97.62
10	94.87	100.0	100.0	100.0
25	98.85	100.0	100.0	100.0
50	98.44	100.0	100.0	100.0
75	99.47	100.0	100.0	100.0
100	98.93	100.0	100.0	100.0

Table 10: Accuracy scores with Random Forest of the Mushroom dataset.

Num Trees	**F=5**	**F=11**	**F=16**	**F=random**
1	94.79	96.27	96.43	96.43
10	99.55	98.36	97.46	99.22
25	99.96	100.0	97.74	99.51
50	100.0	99.84	98.73	99.38
75	100.0	100.0	99.3	99.55
100	100.0	100.0	99.34	99.55

Table 11: Accuracy scores with Decision Forest of the Mushroom dataset.

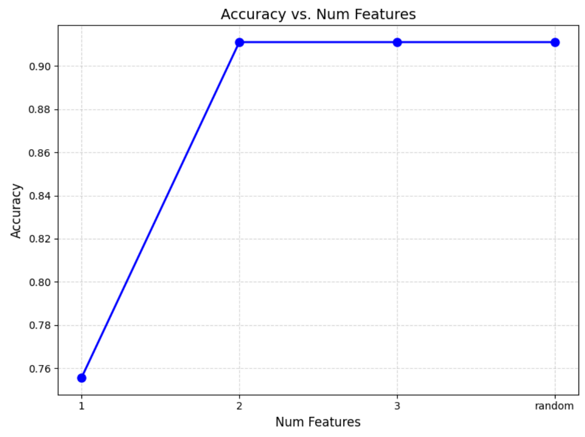
NT, F	**Features ordering**
1, 1	population - bruises - ring-type - habitat - cap-color
1, 2	odor - bruises - gill-color - spore-print-color - gill-size
1, 4	gill-color - odor - gill-size - spore-print-color - stalk-color-above-ring
1, 5	odor - gill-color - stalk-color-above-ring - spore-print-color - stalk-root
10, 1	odor - stalk-root - population - stalk-surface-above-ring - stalk-surface-below-ring
10, 2	odor - gill-size - stalk-root - spore-print-color - ring-type
10, 4	odor - spore-print-color - stalk-root - gill-color - stalk-surface-above-ring
10, 5	odor - spore-print-color - gill-color - gill-size - stalk-root
25, 1	odor - gill-color - stalk-root - gill-size - stalk-surface-above-ring
25, 2	odor - spore-print-color - gill-size - stalk-surface-above-ring - stalk-surface-below-ring
25, 4	odor - spore-print-color - stalk-root - stalk-surface-below-ring - gill-color
25, 5	odor - spore-print-color - stalk-root - gill-color - stalk-surface-above-ring
50, 1	odor - stalk-root - gill-color - stalk-surface-above-ring - habitat
50, 2	odor - spore-print-color - stalk-root - stalk-surface-above-ring - gill-size
50, 4	odor - spore-print-color - stalk-root - stalk-surface-above-ring - gill-size
50, 5	odor - spore-print-color - stalk-root - gill-color - stalk-surface-above-ring
75, 1	odor - gill-color - stalk-root - ring-type - stalk-surface-above-ring
75, 2	odor - spore-print-color - stalk-root - gill-size - stalk-surface-above-ring
75, 4	odor - spore-print-color - stalk-root - stalk-surface-above-ring - stalk-surface-below-ring
75, 5	odor - spore-print-color - stalk-root - gill-color - stalk-surface-above-ring
100, 1	odor - stalk-root - gill-color - spore-print-color - ring-type
100, 2	odor - spore-print-color - stalk-root - gill-size - gill-color
100, 4	odor - spore-print-color - stalk-root - stalk-surface-above-ring - gill-color
100, 5	odor - spore-print-color - stalk-root - gill-color - stalk-surface-above-ring

Table 12: Top 5 features of the Mushroom dataset ordered by importance according to Random Forest.

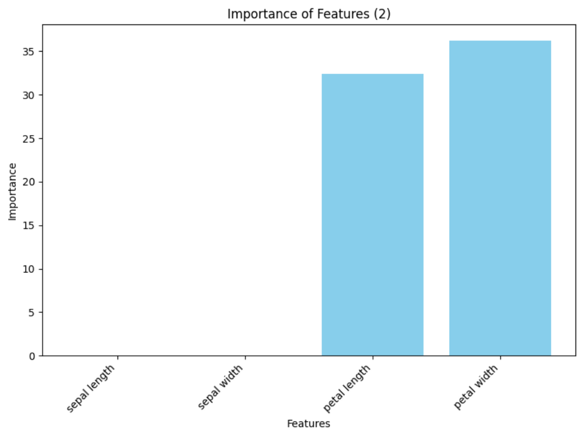
NT, F	**Features ordering**
1, 5	gill-color - population - gill-size - bruises
1, 11	odor - gill-color - stalk-root - population
1, 16	odor - spore-print-color - stalk-root - population
1, random	odor - spore-print-color - stalk-root - bruises
10, 5	odor - spore-print-color - gill-color - stalk-surface-below-ring
10, 11	odor - spore-print-color - gill-color - population
10, 16	odor - spore-print-color - habitat - stalk-root
10, random	odor - spore-print-color - stalk-root - bruises
25, 5	odor - spore-print-color - gill-color - stalk-root
25, 11	odor - spore-print-color - stalk-root - gill-size
25, 16	odor - spore-print-color - stalk-root - gill-size
25, random	odor - spore-print-color - stalk-root - gill-color
50, 5	odor - spore-print-color - gill-color - stalk-root
50, 11	odor - spore-print-color - stalk-root - gill-size
50, 16	odor - spore-print-color - stalk-root - gill-size
50, random	odor - spore-print-color - stalk-root - population
75, 5	odor - spore-print-color - stalk-root - gill-color
75, 11	odor - spore-print-color - stalk-root - gill-size
75, 16	odor - spore-print-color - stalk-root - gill-size
75, random	odor - spore-print-color - stalk-root - gill-size
100, 5	odor - spore-print-color - stalk-root - gill-color
100, 11	odor - spore-print-color - stalk-root - gill-size
100, 16	odor - spore-print-color - stalk-root - gill-size
100, random	odor - spore-print-color - stalk-root - gill-size

Table 13: Top 5 features of the Mushroom dataset ordered by importance according to Decision Forest.

6 Plots

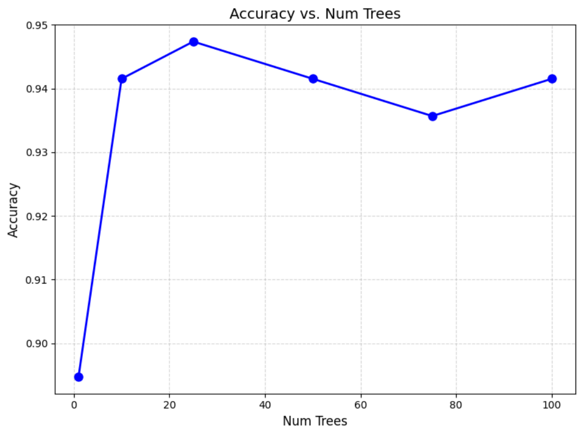


(a) Accuracy vs number of features with Decision Forest. Number of trees = 50.

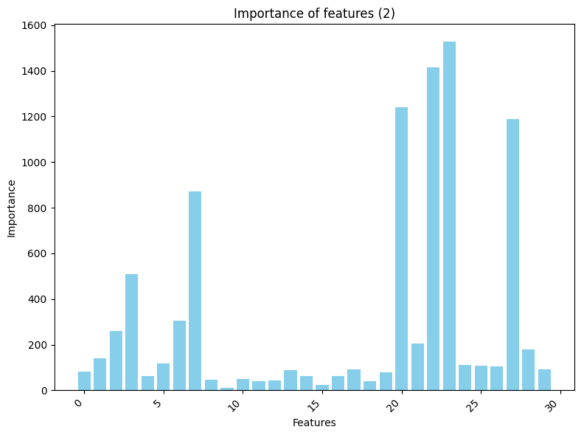


(b) Feature importance with Random Forest (model with best accuracy)

Figure 1: Iris dataset plots.

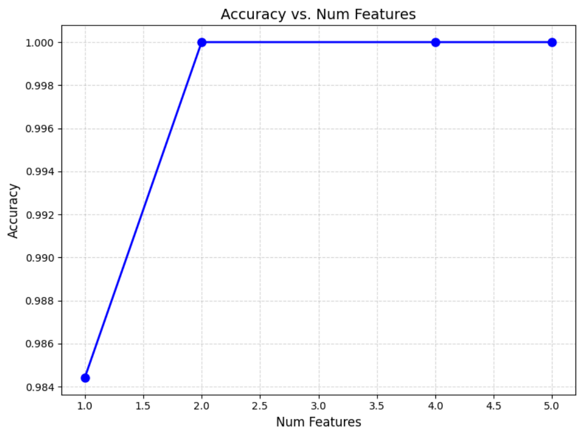


(a) Accuracy vs number of trees with Decision Forest. Number of features = 22.

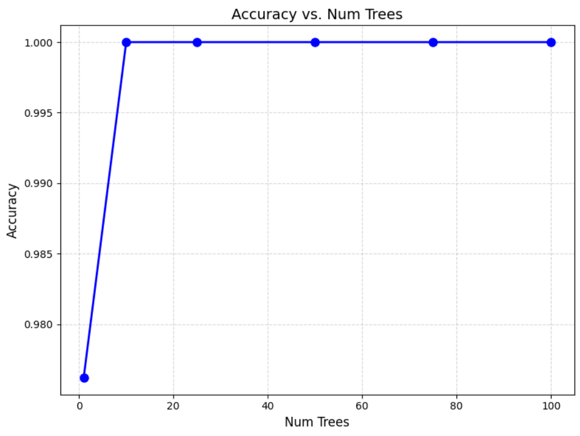


(b) Feature importance with Decision Forest (model with best accuracy).

Figure 2: WDBC dataset plots.

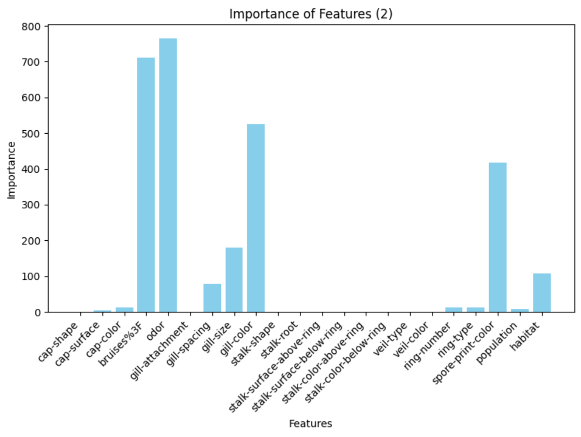


(a) Accuracy vs number of features with Random Forest. Number of trees = 50.

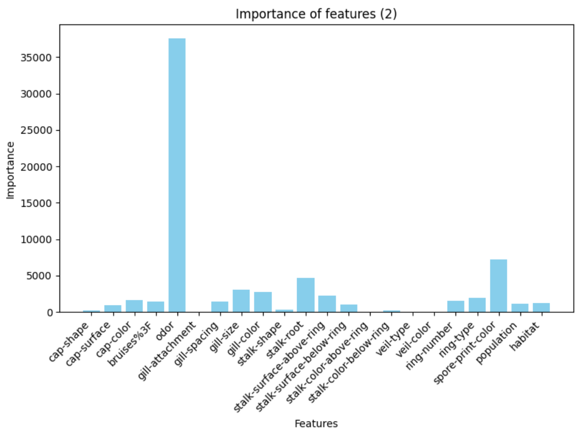


(b) Accuracy vs number of trees with Random Forest. Number of features = 4.

Figure 3: Mushroom dataset plots.



(a) Feature importance with Random Forest (model with best accuracy).



(b) Feature importance with Decision Forest (model with best accuracy).

Figure 4: Mushroom dataset plots.

7 Conclusions

Overall, we did not notice any remarkable differences regarding the performance between the Random Forest and the Decision Forest algorithms. Moreover, although they operate slightly different, the features highlighted by both algorithms in terms of importance are the same most of the times.

Our results also indicate that it is not necessary to consider many trees in the ensemble since the performance remains equal up to some threshold of the number of trees. In the same direction, our results also indicate that there are no improvements after a few features are considered. Specifically, the classifiers might actually performs slightly better when fewer features are selected, although there is some randomness associated, too. Anyways, there is no evidence whatsoever that selecting a high number of features improves the results.

The claim that Random Forest and Decision Forest perform well with a low number of trees and features selected is supported by the fact that the ordering of the features importance does not change much when the number of trees considered is larger than 10, and the features selected are larger or equal than $Int(\frac{M}{4})$ and 2, for the Decision Forest and Random Forest, respectively.

References

[1] D.T. Pham and M.S. Aksoy. Rules: A simple rule extraction system. *Expert Systems with Applications*, 8(1):59–65, 1995.