Practical Work 2: Combining multiple classifiers

Supervised and Experiential Learning

GUILLERMO CREUS BOTELLA

MSC IN ARTIFICIAL INTELLIGENCE

UNIVERSITAT POLITÈCNICA
DE CATALUNYA
BARCELONATECH

BARCELONA, MAY 17, 2022

Contents

1 Introduction

Ensemble learning [7] is a Machine Learning (ML) technique that consists of training an ensemble of classifiers. That is, combining multiple independent classifiers, training them, and aggregating all of the predictions whenever a novel instance needs to be predicted. In this practical work, the classifier that will be the base of the ensemble will be a Classification and Regression Tree (CART) [5], a decision tree that is suited for classification and regression tasks.

The main objective of this practical work is to observe the influence of different ensemble methods in the performance of a classification task. To do that, the methods that will be explored in this work will be the following: Random Forests [1] and Decision Forests [3]. Each model will be trained with different hyperparameter combinations in order to analyze their effect on the classification performance. Regarding the data used to evaluate the model, three datasets will be used: Wines, Breast Cancer Wisconsin (Diagnostic) and Mushroom dataset. The purpose of analyzing the results in three different datasets is double-sided: three different datasets will be analyzed, while evaluating how the models perform in a Small, Medium and Large dataset.

The structure of this report is the following: section 2 will go through the datasets used in this practical work, section 3 will introduce CART, Random Forests and Decision Forests, and sections 4 and 5 will evaluate the results and give a comprehensive conclusion on them. In addition, an extensive appendix can be found in section 6, where the results.

2 Data

In this section, the datasets used for evaluating the ensemble methods will be introduced, as well as the metrics to evaluate them. All of the datasets have been extracted from the UCI Machine Learning Repository [2] and the default task to solve is classification. That is, given some features, the model should output a class out of a given set of classes.

Before introducing the datasets, it is important to mention that in this practical work, the possible size of the dataset will have three categories: Small (# instances ≤ 500), Medium ($500 <$ # of instances ≤ 2000) and Large (# of instances > 2000).

As one can see in table 1, one dataset of each category will be used, so instead of referring to the dataset by its name, the category will define the dataset. For instance, when one refers to the small dataset, it will be referring to the Wine dataset.

Furthermore, one can identify two groups of datasets. The small dataset has 3 classes (multiclass classification) while the others have two classes (binary classification). In the next section, the metrics best suited for these two types of datasets will be addressed.

Table 1: Information of the datasets used

Dataset name	Category	# Instances (N)	# Features (M)	# Classes (c)
Wine	Small	178	13	3
Breast Cancer Wisconsin	Medium	699	32	2
Mushroom	Large	8124	22	2

2.1 Metrics

Since Random Forests and Decision Forests can handle multiclass problems, i.e., ones where there are c target values: $\delta_1, \dots, \delta_c$, with $c > 2$, it is important to explain the metrics [4] that will be used. First of all, accuracy is defined as:

$$\text{Accuracy} = \frac{\sum_{k=1}^{N_c} TP_k}{\#\text{ of instances}}$$

Where TP_k is the number of correctly classified instances of class δ_k, and FP_k is the number of instances that were wrongly predicted as class δ_k.

For every class δ_k, one can define the

$$\text{Precision}_k = \frac{TP_k}{TP_k + FP_k}$$

$$\text{Recall}_k = \frac{TP_k}{\#\text{ of instances of class } \delta_k}$$

$$\text{F1}_k = \frac{2 * \text{Precision}_k * \text{Recall}_k}{\text{Precision}_k + \text{Recall}_k}$$

When one reports these metrics, readily available in [8], it is important to announce whether they are **weighted** by the amount of instances belonging to each class, or if a regular **macro average** is being done, ignoring the amount of instances in each class. This is extremely important in unbalanced datasets, since an extremely low metric in the minimal class can drop the overall metric significantly. However, since all of the datasets in question are balanced, the metrics will be **macro averaged**.

Small dataset

The Wine dataset has been the choice for the small dataset, at 178 instances. It contains the results of a chemical analysis of wines from three different cultivars. Each instance has 13 numerical attributes derived from the analysis: Alcohol, Malic acid, Ash, Alcalinity of ash, Magnesium, Total phenols, Flavanoids, Nonflavanoid phenols, Proanthocyanins, Color intensity, Hue, OD280/OD315 of diluted wines, Proline.

As previously mentioned, the dataset's instances can be classified into three classes: δ_1, δ_2 and δ_3. Its distribution is the following: δ_1 – 59 (33.1%), δ_2 – 71 (39.9%) and δ_3 – 48 (27.0%), which indicates that it is a well behaved dataset, with an even class distribution.

Medium dataset

As it can seen table 1, the Medium dataset will be the Breast Cancer Wisconsin (Diagnostic), with 699 instances. Data was collected by Dr. William H. Wolberg at the University of Wisconsin Hospitals, Madison [6], culminating in a breast cancer database consisting of several attributes: Sample code number, Clump Thickness, Uniformity of Cell Size, Uniformity of Cell Shape, Marginal Adhesion, Single Epithelial Cell Size, Bare Nuclei, Bland Chromatin, Normal Nucleoli, Mitoses. The first one, Sample code number, has been dropped since it does not have predicting power. The others are integer valued (1 – 10) with the column Bare Nuclei being the only one that had 16 missing values. In order to develop code robust to missing values, they were left untouched, and considered as a unique category of the attribute.

The classification aims to predict instances into two groups: δ_0 and δ_1. The first class, δ_0, represents benign cases, while δ_1 represents malignant ones. The distribution is the following: δ_0 – 458 (65.5%) and δ_1 – 241 (34.5%). Since the objective is to predict malignant cases, this problem has been addressed as binary classification.

Large dataset

The large dataset choice has been the Mushroom one. It contains 8124 instances with 22 attributes, all of them categorical. As stated in [2], this data set includes samples corresponding to 23 species of gilled mushrooms in the Agaricus and Lepiota Family. Each species is identified as definitely edible (δ_0) or as definitely poisonous/of unknown edibility and not recommended (δ_1). Therefore, the problem is a binary classification, with the aim of predicting whether a species is edible or not.

The only attribute that contains is “stalk-root”. As in the medium dataset, with the sole purpose of making the algorithms robust to missing values, they will be left as they are, and they will be considered as a unique value.

3 Methods

Since the methods developed in this practical work are a combination of decision trees, the CART method is essential. That is why a whole section (3.1) has been devoted to explain the process behind the training of a CART. Afterwards, the Random Forest and Decision forest methods will be explained via their pseudocode.

3.1 CART

The decision tree used (CART) can be defined by nodes, which divide data in two (binary split), thus, having a left and right node. In addition, the nodes without left and right nodes are called leaves. At each node, given some attributes A_j, the goal is to find the one that best partitions data. To do that, one must rely on equation 2, which computes the Gini index of a sample X, given a split X_1 and X_2 given by attribute A. For instance, if attribute eye color of 5 instances is [blue, blue, brown, green, brown], then one of the possible splits will be: $X_1 = \{x \mid x_{\text{eye color}} \in \{\text{blue, green}\}\}$, $X_2 = \{x \mid x_{\text{eye color}} \in \{\text{brown}\}\}$. If an attribute is numerical, say height: $[180, 174, 174, 165, 191]$, the possible split points will be mid points of the unique values. In the previous case, the possible splitting points are: $[\frac{165+174}{2}, \frac{174+180}{2}, \frac{180+191}{2}] = [169.5, 177, 185.5]$, and a possible split is: $X_1 = \{x \mid x_{\text{height}} \leq 177\}\}$, $X_2 = \{x \mid x_{\text{height}} > 177\}$.

Having defined all that, given a sample X, one can compute the Gini index seen in equation 1, which indicates the degree of separation of a sample. In the extreme case of all samples belonging to the same class, Gini would be 1 and a split would not be necessary. However, the task of CART is to find the attribute A (and the value) that best separates a sample X into X_1 and X_2 (see equation 3).

$$\text{Gini}(X) := 1 - \sum_{i=1}^{c} p^2_{x \in \delta_i} \tag{1}$$

$$\text{Gini}_{\text{attr}}(X, A, X_1, X_2) := \frac{|X_1|}{|X|} \cdot \text{Gini}(X_1) + \frac{|X_2|}{|X|} \cdot \text{Gini}(X_2) \tag{2}$$

$$\widehat{A}, \widehat{X_1}, \widehat{X_2} = \underset{A, X_1, X_2}{\text{argmax}}\, \Delta\text{Gini}(X, A) := \text{Gini}(X) - \text{Gini}_{\text{attr}}(X, A, X_1, X_2) = \underset{A, X_1, X_2}{\text{argmin}}\, \text{Gini}_{\text{attr}}(X, A, X_1, X_2) \tag{3}$$

Now that all of the equations have been defined, CART will start at the root considering F possible random attributes (sample without replacement) and all of their possible splitting points, to choose the tuple (attribute, splitting point) that partitions data best according to the Gini Index. This process will continue on the left and right node until reaching a node where all the elements belong to the same class (leaf), and this will be defined as its value. It could happen that not all of the elements belong to the same class but their attributes are the same. In that case, the most common label will be the value of the leaf node. This algorithm will be defined as CART_F.

The prediction using CART_F is quite simple since it only requires to traverse the tree. Once a leaf node has been reached, its value will be predicted.

3.2 Random Forest

In algorithm 1 one can see the pseudocode of the training of a Random Forest. The main parameters are the number of trees, NT, and the number of random features when splitting each node, F. The process is the following: create NT empty CART_F decision trees. For each decision tree T_i, generate a sample with replacement of size N (number of training instances). Lastly, it will train T_i and update the feature importance of the Random Forest $f_{\text{importance}}$.

Furthermore, in algorithm 2 one can see the prediction mechanism for a Random Forest. For every tree T_i it computes a prediction, and it aggregates them with majority voting.

Pseudocode

Algorithm 1 Random Forest Algorithm (Training)

Input:

- X_{trn}, y_{trn}: Training features and labels
- NT: Number of trees
- F: number of features when splitting nodes

Output: T – Fitted Random Forest, i.e., NT trained trees

```
1: $T \leftarrow [T_1, \ldots, T_{NT}]$ – array of $NT$ empty decision trees
2: $N$ – number of samples in $X_{\text{trn}}$.
3: $M$ – number of features in $X_{\text{trn}}$.
4: $f_{\text{importance}} \leftarrow [0, \ldots, 0]$ – array of length $M$, where position $i$ indicates the frequency of appearance of feature $i$.
5: for each $T_i \in T$ do
6:     $T_i \leftarrow \text{CART}_F$
7:     $X_{\text{bootstrap}}, y_{\text{bootstrap}} \leftarrow$ sample of $X_{\text{trn}}, y_{\text{trn}}$ with replacement of size $N$
8:     Use $X_{\text{bootstrap}}, y_{\text{bootstrap}}$ to fit $T_i$
9:     Update $f_{\text{importance}}$ with the frequencies extracted from $T_i$
10: end for
11: return $T$
```

Algorithm 2 Random Forest (Prediction)

Input:

- X_{tst}: Test instances
- T: Array of decision trees

Output: $\widehat{y}_{\text{tst}}$ – Array with a prediction for each instance of the input dataset

```
1: $\widehat{y}_{\text{tst}} \leftarrow []$
2: for each instance $x_i \in X_{\text{trn}}$ do
3:     predictions$_i \leftarrow []$
4:     for each tree $T_j \in T$ do
5:         Append class prediction of $x_i$ using $T_j$ to predictions$_i$
6:     end for
7:     $\widehat{y}_i \leftarrow$ most common class of predictions$_i$
8:     Append $\widehat{y}_i$ to $\widehat{y}_{\text{tst}}$
9: end for
10: return $\widehat{y}_{\text{tst}}$
```

3.3 Decision Forest

In algorithm 3 one can see the pseudocode of the training of a Decision Forest. The main parameters are the number of trees, NT, and the number of random features chosen to train each tree, F. The methodology is the following: create NT empty CART_F decision trees. For each decision tree, F random columns will be selected (without replacement. Lastly, it will train T_i and update the feature importance of the F chosen features for T_i. Note that CART_F will choose F random features when splitting a node and the number of features of the dataset sent to CART_F is F, thus making the features for the split constant in every decision tree.

In addition, in algorithm 4 one can see the prediction mechanism for a Decision Forest, which is very similar to the Random Forest's. The only difference is that when predicting the class of an instance using tree T_i the F features of the tree will be selected.

Pseudocode

Algorithm 3 Decision Forest Algorithm (Training)

Input:

- X_{trn}, y_{trn}: Training features and labels
- NT: Number of trees
- F: number of features used by each tree

Output: T – Fitted Decision Forest, i.e., NT trained trees

```
1: $T \leftarrow [T_1, \ldots, T_{NT}]$ – array of $NT$ empty decision trees
2: $f_{\text{chosen}} \leftarrow []$ – list that will contain the chosen $F$ features for every decision tree
3: $N$ – number of samples in $X_{\text{trn}}$.
4: $M$ – number of features in $X_{\text{trn}}$.
5: $f_{\text{importance}} \leftarrow [0, \ldots, 0]$ – array of length $M$, where position $i$ indicates the frequency of appearance of feature $i$.
6: for each $T_i \in T$ do
7:     $T_i \leftarrow \text{CART}_F$
8:
9:     $f^i_{\text{chosen}} \leftarrow$ select $F$ random features (sample without replacement)
10:    $X_{\text{feat}} \leftarrow$ select the columns $f^i_{\text{chosen}}$ from $X_{\text{trn}}$
11:    Append $f^i_{\text{chosen}}$ to $f_{\text{chosen}}$
12:
13:    Use $X_{\text{feat}}, y_{\text{trn}}$ to fit $T_i$
14:
15:    Update $f_{\text{importance}}$ with the frequencies extracted from $T_i$ (only modify the $f^i_{\text{chosen}}$ indices)
16: end for
17: return $T$
```

Algorithm 4 Decision Forest (Prediction)

Input:

- X_{tst}: Test instances
- T: Array of NT decision trees
- f_{chosen}: List that contains NT arrays, each of size F, indicating the features that each decision tree uses

Output: $\widehat{y}_{\text{tst}}$ – Array with a prediction for each instance of the input dataset

```
1:  $\widehat{y}_{\text{tst}} \leftarrow []$
2:  for each instance $x_i \in X_{\text{trn}}$ do
3:      predictions$_i \leftarrow []$
4:      for each tree $T_j \in T$ do
5:          $x_i^{\text{chosen}} \leftarrow$ select the $f^j_{\text{chosen}}$ features from instance $x_i$
6:          Append class prediction of $x_i^{\text{chosen}}$ using $T_j$ to predictions$_i$
7:      end for
8:      $\widehat{y}_i \leftarrow$ most common class of predictions$_i$
9:      Append $\widehat{y}_i$ to $\widehat{y}_{\text{tst}}$
10: end for
11: return $\widehat{y}_{\text{tst}}$
```

4 Results

In this section the results of the methods explained in sections 3.2 and 3.3 will be analyzed. In particular, each of the three datasets shown in section 2 will be divided into a training set and a test set (80%-20% split). Then, each model will be trained in the training set and report the metrics obtained in the test set (explained in section 2.1).

Although it is outside the scope of this practical work, in the real world, models need to be selected. To be more precise, one needs to choose the hyperparameters F and NT for each method. The method of choice is 5-fold cross-validation [9], which will divide the dataset in 5 chunks, so it can train the algorithm in four chunks and test in one. Then it will average a metric of choice in these 5 independent validation sets and choose the hyperparameters that work best (see figure 1). In the case of a binary classification task, the F1-Score will be the metric of choice and in a multi-class classification context accuracy will be the chosen metric.

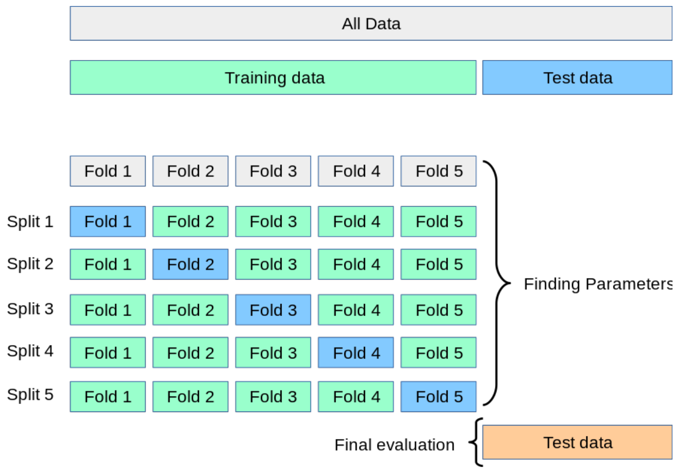

Figure 1: 5-fold cross-validation (credits to [8])

Random Forest

The parameters that will be explored will be the following

- NT : $[1, 10, 25, 50, 75, 100]$
- F : $\{1, 3, \log_2(M+1), \sqrt{M}\}$

Where M is the number of features of the dataset.

Decision Forest

The parameters that will be explored will be the following

- NT : $[1, 10, 25, 50, 75, 100]$
- F : $\{\lfloor \frac{M}{4} \rfloor, \lfloor \frac{M}{2} \rfloor, \lfloor \frac{3 \cdot M}{4} \rfloor$, "Uniform"$\}$

Where M is the number of features of the dataset. Whenever F is uniform, it means that instead of fixing F for all of the decision trees, each one will have a different F. This value will be chosen uniformly between 1 and M.

4.1 Small dataset

The results for this dataset can be found in appendix 6.1. In particular, it is important to check the dictionary that indicates the feature name given its index, which can be found in the Appendix 6.1.1. Also, note that the entries of tables in green will be the largest value in that row, while an orange entry will be the second largest value in that row.

Random Forest

Regarding the feature importance of the Random Forest in table 2 one can see the set of features for every hyperparameter combination ordered from high to low. That is, if the table said $[0, 2, 1]$, then it would mean that the feature with index 0 would be the most important, and the feature with index 1 would be the least important. The dictionary that indicates the feature name given its index can be found in the Appendix 6.1.1.

That being said, whenever the number of chosen features is 1, the variability of features is high. There is not a clear pattern to analyze. Focusing on the column $F = 1$ and $NT = 100$, which is the maximum number of trials, one is not able to see a correlation with the other combinations of NT. On the other hand, when $F = 3$, clearly features Flavanoids (6), Color intensity (9), OD280/OD315 of diluted wines (11) and Proline (12) are the most important, since they are consistently in the top 4 features, regardless of the number of trees. As a consequence, whenever F grows, the feature importance is more predictable.

As for the best hyperparameters, with the results obtained in table 3) with CV, the best combination is achieved whenever $F = 3$ and $NT = 50$ since the accuracy is 0.949. The second best is achieved when $F = 1$ and $NT = 75$, with an accuracy of 0.947. Now that the hyperparameters have been chosen, it is time to see how they have performed in the unseen test set.

The results in the test set can be seen in table 4. The first and second best performing model both achieve a score of 1.0 in the accuracy, precision, recall and F1-Score metrics. Therefore, the selected models are also the best performing ones in the test set. Furthermore, it can be concluded that when $F = 1$ all of the metrics improve when the number of trees grow, up to $NT = 25$, where the scores are perfect. In the case of $F = 3$, there is not a clear pattern between the scores and NT since $NT = 10$, performs better than larger NTs.

Decision Forest

In table 5 one can see the feature importance of the Decision Forest for different combination of hyperparameters. Overall, features Ash (2), Flavanoids (6) Hue (10) and OD280/OD315 of diluted wines (11) are very important, regardless the number of F and NT. In addition, features Alcohol (0), Nonflavanoid phenols (7) and Proanthocyanins (8) seem to be irrelevant for the Decision forest, since they are barely used in the splits. When $F =$ "Uniform", the most important features are robust to NT, since choosing F randomly makes computing the importance smoother.

The CV accuracy is shown in table 6, where $F = 3$ has the highest accuracy. In fact, the best hyperparameter combination is $(F, NT) = (3, 100)$ at 0.937 and the second best is $(F, NT) = (3, 50)$ at 0.935. The selected models, as one can see in table 7 are not the best performers in the test dataset. Their accuracy for both models is 0.944 and they have the second highest accuracy in the test set.

The best performing F in the test set is $F = 6$. Its accuracy is 0.972 for $NT \geq 10$ and 0.944 for $NT = 1$. Although their accuracy is very good in the test set, it does not match the validation accuracy since it is substantially higher. That makes one think of chance being in question since the two best performing models in CV have fairly similar accuracy scores in the test set. Therefore, the model selection is considered to be coherent.

Comparison

The Random Forest is the clear winner in this case since the accuracy it achieves in the test set is 1.00 for the selected models. On the contrary, the Decision Forest's selected model only achieve an accuracy of 0.944 in the test set. In addition, the macro precision, recall and F1-Score also is considerably higher for the Random Forest.

A common aspect in both methods is that $F = 3$ for the best performing model in CV. This makes one think that for this dataset checking 3 features when splitting each node is enough regarding the complexity of the model and whether these features are random or fixed. Furthermore, another thing they have in common is that features Flavanoids (6) and OD280/OD315 of diluted wines (11) are important for both methods.

As a final note, it should be mentioned that accuracy does not seem to improve linearly as the NT increases. There seems to be a cutoff value where performance ceases to improve to oscillate or it stays still.

4.2 Medium dataset

The results for this dataset can be found in appendix 6.2. In particular, it is important to check the dictionary that indicates the feature name given its index, which can be found in the Appendix 6.2.1. Also, note that the entries of tables in green will be the largest value in that row, while an orange entry will be the second largest value in that row.

Random Forest

From table 8 one can see that features Clump Thickness (0), Uniformity of Cell Size (1), Bare Nuclei (5) and Bland Chromatin (6) are among the most important since they constantly show in the top 3 most important features for both values of F. Irregardless of the value of NT and F it is pretty clear that feature Mitoses (8) is not very important. On another note, when $F = 3$, the most important feature is Bare Nuclei (5), the same independent of NT.

In table 9 one can see the F1-Score results of the cross-validation process. The best performing model is $(F, NT) = (1, 100)$ and the second best is $(F, NT) = (1, 50)$. The general trend is that the higher the NT, the higher the F1-Score. Also, if $F = 1$, the results are significantly higher than if $F = 3$. This means that checking one random feature in each node is better than checking three. A possibility could be that Gini is not as effective in this dataset to determine the validity of a partition.

This observation continues in the test results (see table 10), where a lower F implies having a better performance across multiple metrics (with NT fixed). Concerning the selected models, it can be seen that the second selected model in CV $(F, NT) = (1, 100)$ is the best performer in the test set in all of the metrics, with a reported F1-Score of 0.971. In contrast, although the difference is small, the selected model in CV is the third best performer in all of the metrics, with a reported F1-Score of 0.951.

One last thing that should be mentioned is that $F = 1$, makes the model have a higher F1-Score, which comes from a substantially higher recall.

Decision Forest

From table 11 one can see that the most relevant features are: Uniformity of Cell Size (1), Uniformity of Cell Shape (2), Bare Nuclei (5) and Normal Nucleoli (7). The ones that appear as irrelevant are Clump Thickness (0) and Mitoses (8).

From table 12 one can corroborate the findings found in the Random Forest subsection; when one fixes F, NT and F1-Score$_{\text{CV}}$ are positively correlated. That being said, the best and second best models are $(F, NT) = (2, 100)$ and $(F, NT) = (2, 75)$, respectively. Again, the performance seems to decrease if F increases, even if F is chosen uniformly.

Lastly, in table 13 one can see that the best performing models (across all metrics) in the test set are the ones with $F = 2$. The reported F1-Score in the test set for the best model via CV is 0.951 (same for the second best model). One important thing to mention is that when $F = 2$, the results are identical for the majority of NT, thus, implying that the trees built are very similar.

Comparison

One thing that should be mentioned is that feature Clump Thickness (0) is very important in the Random Forest context but it is not in the Decision Forest one. Apart from that, features Uniformity of Cell Size (1) and Bare Nuclei (5) are seen as important in both methods. Feature Mitoses (8) is irrelevant in both methods.

As for the performance, the selected model via cross-validation for Random Forests achieves an F1-Score in the test set of 0.971, while the selected model for Decision Forests reports an F1-Score of 0.951.

4.3 Large dataset

The results for this dataset can be found in appendix 6.3. In particular, it is important to check the dictionary that indicates the feature name given its index, which can be found in the Appendix 6.3.1. Also, note that the entries of tables in green will be the largest value in that row, while an orange entry will be the second largest value in that row.

Random Forest

In table 14 one can see the feature importance for all of the Random Forests. The variability is considerably higher than in the previous datasets, which could be caused by the large number of features; 22, almost doubling the ones from the small and medium datasets. Despite the variability, feature odor (4) is among the most important ones, which makes sense, since one must remember that the main task is to determine whether certain mushroom is edible or not. In contrast, feature veil-type (15) seems to be disregarded by most models.

In table 15 one can observe the F1-Score of the different models in the CV process. Again, with F fixed, as NT increases, so does the F1-Score. Furthermore, F1-Score, when one fixes F is negatively correlated to F. That is why the best and second-best model are $(F, NT) = (1, 100)$ and $(F, NT) = (1, 75)$, respectively.

Lastly, as one can see in table 16, the F1-Score of the best model in CV is not the largest in the test set. In fact, it is the third largest at 0.990. Conversely, the second best model in CV achieves the second largest value in the test set with a value of 0.992.

Decision Forest

The importance of features of Decision Forests (table 17) helps one determine that features cap-color (2), stalk-color-below-ring (14), spore-print-color (19) are considered important. In contrast, feature cap-shape (0) is considered irrelevant.

The cross-validation F1-Scores can be seen in table 18. The first thing that should be mentioned is the fact that some models achieve a 0.0 F1-Score. That is caused by the prediction of all samples belonging to the negative class, causing the precision to go to 0. Regarding the best and second best model, they are achieved when $(F, NT) = (16, 1)$ and $(F, NT) = (\text{Uniform}, 1)$, respectively.

Lastly, as one can see in table 19, the best model in CV continues to be the best in all of the metrics in the test set (reporting an F1-Score of 0.997). In addition, the second best model in CV also continues to be the second best, as far as all the test metrics are concerned (reporting an F1-Score of 0.991). Lastly, most of the configurations that did not predict any positive class continue to do the same in the test set.

Comparison

As a comparison of models it is interesting to notice that Decision Forests have less variability in the feature importance. In addition, the performance of the selected Decision Forest outperforms the Random Forest: 0.997 vs. 0.990.

5 Conclusion

In conclusion, it is important to highlight that all of the objectives were met. Firstly, a decision tree CART was developed so that it could be the base learner of the ensemble methods. Subsequently, the Random Forests and Decision Forest classifiers were developed so that that they could read CSV files, train the different decision trees, and display the feature importance. Both methods, with a wide set of hyperparameters were applied to three different databases: Wine (small), Breast cancer Wisconsin (medium) and Mushroom (large). Subsequently, the metrics in the train, 5-fold CV and test sets were shown and analyzed.

One of the results obtained, contrary to intuition, is that a higher F does not mean better performance. Usually, one would think that having a higher F and choosing the best feature would equal a better performance in the Random Forest. Also, the argument is similar for Decision Forests. However, this is not what was observed. That is why it would be interesting to analyze different split evaluation techniques, such as entropy, in future work. Regarding the hyperparameter NT, usually a higher one implies better performance. However there is a cutoff value and the performance stops increasing from there. Also, feature importance does not seem to be consistent across methods and F, only with different NT and everything else fixed it behaves correctly.

Lastly, the selected models for Random Forests achieved far better performance in the small (1.00 vs. 0.944 accuracy) and medium (0.971 vs. 0.951 F1-Score) datasets. However, in the large dataset the Decision Forest performed slightly better (0.997 vs. 0.990 F1-Score) than the Random Forest. The interesting aspect is that in the large dataset only one decision tree was needed to achieve these results. This makes the dataset seem easy to classify since decision trees have been known to overfit data. That is why, in further work it would be interesting to apply these methods to more datasets to confirm the findings and not be as dataset dependent as now.

References

[1] Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[2] Dheeru Dua and Casey Graff. UCI machine learning repository, 2017.

[3] Tin Kam Ho. The random subspace method for constructing decision forests. *IEEE transactions on pattern analysis and machine intelligence*, 20(8):832–844, 1998.

[4] Mohammad Hossin and Md Nasir Sulaiman. A review on evaluation metrics for data classification evaluations. *International journal of data mining & knowledge management process*, 5(2):1, 2015.

[5] Wei-Yin Loh. Classification and regression trees. *Wiley interdisciplinary reviews: data mining and knowledge discovery*, 1(1):14–23, 2011.

[6] Olvi L Mangasarian and William H Wolberg. Cancer diagnosis via linear programming. Technical report, University of Wisconsin-Madison Department of Computer Sciences, 1990.

[7] David Opitz and Richard Maclin. Popular ensemble methods: An empirical study. *Journal of artificial intelligence research*, 11:169–198, 1999.

[8] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[9] Payam Refaeilzadeh, Lei Tang, and Huan Liu. Cross-validation. *Encyclopedia of database systems*, 5:532–538, 2009.

6 Appendix

6.1 Small dataset

6.1.1 Features

- 0: Alcohol
- 1: Malic acid
- 2: Ash
- 3: Alcalinity of ash
- 4: Magnesium
- 5: Total phenols
- 6: Flavanoids
- 7: Nonflavanoid phenols
- 8: Proanthocyanins
- 9: Color intensity
- 10: Hue
- 11: OD280/OD315 of diluted wines
- 12: Proline

6.1.2 Results

Table 2: Random Forest – Importance of features in the Small dataset (from high to low)

F	1						3					
NT	1	10	25	50	75	100	1	10	25	50	75	100
#1	7	7	4	4	1	7	6	6	6	9	9	9
#2	10	11	10	1	4	1	11	9	9	6	6	6
#3	3	1	12	0	10	4	9	12	12	11	12	12
#4	12	12	8	10	7	6	7	11	11	12	11	11
#5	8	8	7	5	6	12	3	4	1	10	1	0
#6	6	4	0	6	12	10	1	2	4	1	0	10
#7	5	6	9	12	2	2	0	1	2	0	10	1
#8	4	2	1	8	5	5	12	0	0	5	5	5
#9	1	5	11	2	0	8	10	8	10	4	4	8
#10	11	0	2	7	11	3	8	3	3	2	8	4
#11	9	10	5	9	8	0	5	10	8	3	2	3
#12	2	3	6	11	9	9	4	5	5	8	3	2
#13	0	9	3	3	3	11	2	7	7	7	7	7

Table 3: Random Forest – Accuracy in the Small dataset (CV)

F	1						3					
NT	1	10	25	50	75	100	1	10	25	50	75	100
Accuracy$_{\mathrm{CV}}$	0.736	0.882	0.937	0.944	0.947	0.942	0.757	0.903	0.937	0.949	0.945	0.945

Table 4: Random Forest – Metrics in the Small dataset (Test)

F	1						3					
NT	1	10	25	50	75	100	1	10	25	50	75	100
Accuracy	0.833	0.944	1.0	1.0	1.0	1.0	0.833	1.0	0.972	1.0	0.972	0.972
Macro precision	0.827	0.954	1.0	1.0	1.0	1.0	0.84	1.0	0.979	1.0	0.979	0.979
Macro recall	0.863	0.954	1.0	1.0	1.0	1.0	0.863	1.0	0.976	1.0	0.976	0.976
Macro F1-Score	0.839	0.954	1.0	1.0	1.0	1.0	0.849	1.0	0.977	1.0	0.977	0.977

Table 5: Decision forest – Importance of features in the Small dataset (from high to low)

F	3						6						9						Uniform					
NT	1	10	25	50	75	100	1	10	25	50	75	100	1	10	25	50	75	100	1	10	25	50	75	100
#1	11	2	2	2	2	10	6	10	2	10	10	10	2	11	6	6	6	6	12	2	2	2	2	2
#2	6	10	7	10	10	2	10	2	10	2	6	6	11	6	10	12	12	11	11	11	11	11	11	11
#3	4	4	3	7	3	7	11	6	6	9	2	9	9	10	12	2	11	12	9	7	10	1	10	12
#4	12	11	1	1	6	3	8	4	9	6	9	2	6	2	11	11	10	2	6	6	1	5	5	10
#5	10	6	10	9	1	8	4	9	12	12	12	12	4	12	2	10	2	9	3	10	6	7	12	6
#6	9	7	11	3	9	6	2	1	3	1	1	1	1	9	9	9	9	10	2	5	5	10	1	5
#7	8	3	12	11	7	1	12	12	4	11	11	11	12	4	0	4	4	4	10	12	9	6	6	1
#8	7	1	5	8	11	9	9	5	1	3	3	3	10	1	4	0	0	0	8	1	7	12	9	9
#9	5	8	8	12	4	5	7	11	8	8	4	4	8	8	3	3	3	3	7	4	3	9	7	0
#10	3	5	6	6	8	4	5	8	11	4	0	0	7	5	1	1	1	1	5	3	12	3	0	7
#11	2	9	4	4	12	11	3	3	5	7	8	8	5	3	5	8	8	8	4	9	8	0	3	3
#12	1	0	9	5	5	12	1	0	0	5	5	5	3	0	8	5	5	5	1	0	0	4	4	4
#13	0	12	0	0	0	0	0	7	7	0	7	7	0	7	7	7	7	7	0	8	4	8	8	8

Table 6: Decision Forest – Accuracy in the Small dataset (CV)

F	3						6						9						Uniform					
NT	1	10	25	50	75	100	1	10	25	50	75	100	1	10	25	50	75	100	1	10	25	50	75	100
Accuracy$_{\mathrm{CV}}$	0.731	0.831	0.907	0.935	0.931	0.937	0.75	0.923	0.926	0.919	0.917	0.916	0.828	0.893	0.905	0.893	0.889	0.891	0.854	0.912	0.903	0.914	0.916	0.912

Table 7: Decision Forest – Metrics in the Small dataset (Test)

F	3						6						9						Uniform					
NT	1	10	25	50	75	100	1	10	25	50	75	100	1	10	25	50	75	100	1	10	25	50	75	100
Accuracy	0.944	0.944	0.944	0.944	0.944	0.944	0.944	0.972	0.972	0.972	0.972	0.972	0.944	0.944	0.944	0.917	0.917	0.917	0.917	0.944	0.917	0.944	0.944	0.944
Macro precision	0.956	0.954	0.961	0.961	0.961	0.961	0.954	0.979	0.979	0.979	0.979	0.979	0.954	0.961	0.961	0.944	0.944	0.944	0.944	0.961	0.944	0.961	0.961	0.961
Macro recall	0.930	0.954	0.952	0.952	0.952	0.952	0.954	0.976	0.976	0.976	0.976	0.976	0.954	0.952	0.952	0.929	0.929	0.929	0.929	0.952	0.929	0.952	0.952	0.952
Macro F1-Score	0.941	0.954	0.954	0.954	0.954	0.954	0.954	0.977	0.977	0.977	0.977	0.977	0.954	0.954	0.954	0.93	0.93	0.93	0.93	0.954	0.93	0.954	0.954	0.954

6.2 Medium dataset

6.2.1 Features

- 0: Clump Thickness
- 1: Uniformity of Cell Size
- 2: Uniformity of Cell Shape
- 3: Marginal Adhesion
- 4: Single Epithelial Cell Size
- 5: Bare Nuclei
- 6: Bland Chromatin
- 7: Normal Nucleoli
- 8: Mitoses

6.2.2 Results

Table 8: Random Forest – Importance of features in the Medium dataset (from high to low)

F	1						3					
NT	1	10	25	50	75	100	1	10	25	50	75	100
#1	5	2	0	0	0	0	5	5	5	5	5	5
#2	0	1	2	2	1	1	3	1	1	6	0	0
#3	1	0	1	1	5	2	6	2	6	1	1	6
#4	7	7	5	5	2	5	4	6	0	0	6	1
#5	6	4	4	4	3	6	2	0	7	2	2	2
#6	8	5	6	7	4	4	1	3	2	3	4	3
#7	2	3	3	3	6	3	0	7	3	4	3	4
#8	4	8	7	6	7	7	7	4	4	7	7	7
#9	3	6	8	8	8	8	8	8	8	8	8	8

Table 9: Random Forest – F1 Score in the Medium dataset (CV)

F	1						3					
NT	1	10	25	50	75	100	1	10	25	50	75	100
F1-Score$_{\mathrm{CV}}$	0.89	0.931	0.944	0.948	0.946	0.950	0.867	0.92	0.934	0.932	0.937	0.938

Table 10: Random Forest – Metrics in the Medium dataset (Test)

F	1						3					
NT	1	10	25	50	75	100	1	10	25	50	75	100
Accuracy	0.879	0.964	0.971	0.979	0.971	0.964	0.9	0.95	0.943	0.95	0.957	0.964
Precision	0.905	0.942	0.927	0.944	0.943	0.942	0.911	0.94	0.922	0.94	0.941	0.942
Recall	0.745	0.961	1.0	1.0	0.98	0.961	0.804	0.922	0.922	0.922	0.941	0.961
F1-Score	0.817	0.951	0.962	0.971	0.962	0.951	0.854	0.931	0.922	0.931	0.941	0.951

Table 11: Decision forest – Importance of features in the Medium dataset (from high to low)

F	2						4						6						Uniform					
NT	1	10	25	50	75	100	1	10	25	50	75	100	1	10	25	50	75	100	1	10	25	50	75	100
#1	2	2	2	2	2	2	4	7	7	7	7	7	7	0	2	2	1	5	5	5	5	5	5	5
#2	7	7	7	7	4	7	7	4	2	2	1	1	4	2	7	1	2	1	1	1	2	1	1	1
#3	8	4	4	1	1	4	2	2	4	1	4	2	1	7	0	5	5	0	2	2	1	2	2	0
#4	6	1	8	6	7	1	1	6	6	4	2	4	8	1	1	0	0	2	6	4	0	0	0	2
#5	5	8	3	4	6	6	8	1	1	6	6	0	2	4	5	7	7	7	4	0	7	7	4	4
#6	4	6	1	3	3	3	6	0	8	0	0	6	6	6	3	3	3	4	8	3	6	3	3	3
#7	3	5	6	8	8	0	5	8	5	3	5	5	5	3	4	4	4	3	7	6	3	4	7	7
#8	1	3	5	0	0	8	3	5	3	5	3	3	3	5	6	6	6	6	3	8	4	6	6	6
#9	0	0	0	5	5	5	0	3	0	8	8	8	0	8	8	8	8	8	0	7	8	8	8	8

Table 12: Decision Forest – F1 Score in the Medium dataset (CV)

F	2						4						6						Uniform					
NT	1	10	25	50	75	100	1	10	25	50	75	100	1	10	25	50	75	100	1	10	25	50	75	100
F1-Score$_{\text{CV}}$	0.881	0.924	0.937	0.939	0.944	0.948	0.865	0.913	0.932	0.937	0.942	0.942	0.873	0.904	0.92	0.912	0.916	0.915	0.872	0.894	0.914	0.916	0.916	0.918

Table 13: Decision Forest – Metrics in the Medium dataset (Test)

F	2						4						6						Uniform					
NT	1	10	25	50	75	100	1	10	25	50	75	100	1	10	25	50	75	100	1	10	25	50	75	100
Accuracy	0.95	0.964	0.964	0.964	0.964	0.964	0.95	0.964	0.964	0.957	0.957	0.957	0.957	0.943	0.957	0.957	0.964	0.957	0.921	0.95	0.95	0.957	0.957	0.95
Precision	0.923	0.942	0.942	0.942	0.942	0.942	0.94	0.942	0.942	0.941	0.941	0.941	0.941	0.939	0.941	0.941	0.942	0.941	0.917	0.94	0.923	0.941	0.941	0.94
Recall	0.941	0.961	0.961	0.961	0.961	0.961	0.922	0.961	0.961	0.941	0.941	0.941	0.941	0.902	0.941	0.941	0.961	0.941	0.863	0.922	0.941	0.941	0.941	0.922
F1-Score	0.932	0.951	0.951	0.951	0.951	0.951	0.931	0.951	0.951	0.941	0.941	0.941	0.941	0.92	0.941	0.941	0.951	0.941	0.889	0.931	0.932	0.941	0.941	0.931

6.3 Large dataset

6.3.1 Features

- 0: cap-shape
- 1: cap-surface
- 2: cap-color
- 3: bruises?
- 4: odor
- 5: gill-attachment
- 6: gill-spacing
- 7: gill-size
- 8: gill-color
- 9: stalk-shape
- 10: stalk-root
- 11: stalk-surface-above-ring
- 12: stalk-surface-below-ring
- 13: stalk-color-above-ring
- 14: stalk-color-below-ring
- 15: veil-type
- 16: veil-color
- 17: ring-number
- 18: ring-type
- 19: spore-print-color
- 20: population
- 21: habitat

6.3.2 Results

Table 14: Random Forest – Importance of features in the Large dataset (from high to low)

F	1						3						4					
NT	1	10	25	50	75	100	1	10	25	50	75	100	1	10	25	50	75	100
#1	13	1	10	10	4	4	8	20	19	19	4	4	11	19	19	19	4	4
#2	10	10	8	4	10	10	19	21	21	21	21	21	21	4	4	4	19	19
#3	1	8	1	1	2	8	13	19	7	4	19	19	9	6	7	7	7	8
#4	17	19	14	8	1	2	11	4	4	14	8	8	1	21	12	21	8	7
#5	16	14	19	14	8	0	7	7	13	20	7	20	2	9	17	12	21	12
#6	3	9	4	20	0	1	9	13	20	7	20	7	3	18	2	8	12	21
#7	14	2	2	0	20	20	21	2	14	13	11	11	4	1	13	2	2	2
#8	6	6	6	19	19	6	1	3	2	8	13	2	5	2	11	11	17	18
#9	7	18	20	2	6	19	2	8	12	2	10	10	6	7	8	17	14	14
#10	8	13	7	6	21	11	3	9	8	12	14	13	7	11	21	20	18	17
#11	2	11	18	7	14	21	4	10	3	11	12	12	8	8	6	13	11	20
#12	4	7	11	11	12	12	5	11	10	18	18	14	10	3	1	1	20	1
#13	5	3	3	12	11	9	6	12	18	3	2	18	20	10	20	18	1	11
#14	21	21	0	13	7	7	10	14	9	10	6	1	12	20	18	14	13	3
#15	9	20	12	9	18	18	20	6	11	1	9	6	13	12	3	9	3	10
#16	20	12	13	21	9	14	12	18	6	9	3	3	14	13	10	6	10	13
#17	11	16	17	18	13	13	14	15	1	6	1	9	15	14	9	3	6	6
#18	12	0	9	3	17	17	15	5	0	0	0	17	16	16	14	10	9	9
#19	15	4	21	17	3	3	16	16	17	17	17	0	17	17	16	16	16	16
#20	18	17	16	16	16	16	17	17	15	16	16	16	18	5	5	0	0	0
#21	19	5	5	5	5	5	18	1	5	5	15	15	19	15	15	5	5	5
#22	0	15	15	15	15	15	0	0	16	15	5	5	0	0	0	15	15	15

Table 15: Random Forest – F1 Score in the Large dataset (CV)

F	1						3						4					
NT	1	10	25	50	75	100	1	10	25	50	75	100	1	10	25	50	75	100
F1-Score$_{\text{CV}}$	0.982	0.95	0.979	0.988	0.989	0.99	0.936	0.969	0.986	0.977	0.977	0.975	0.951	0.911	0.969	0.964	0.973	0.971

Table 16: Random Forest – Metrics in the Large dataset (Test)

F	1						3						4					
NT	1	10	25	50	75	100	1	10	25	50	75	100	1	10	25	50	75	100
Accuracy	0.808	0.951	0.986	0.988	0.993	0.99	0.963	0.921	0.94	0.959	0.985	0.98	0.78	0.994	0.988	0.988	0.988	0.983
Precision	0.763	0.997	1.0	1.0	1.0	1.0	0.93	1.0	0.999	1.0	1.0	1.0	0.948	1.0	1.0	1.0	1.0	1.0
Recall	0.885	0.904	0.971	0.976	0.985	0.98	1.0	0.839	0.879	0.918	0.969	0.959	0.587	0.988	0.975	0.975	0.975	0.965
F1-Score	0.82	0.948	0.985	0.988	0.992	0.99	0.964	0.912	0.935	0.957	0.984	0.979	0.725	0.994	0.987	0.987	0.987	0.982

Table 17: Decision forest – Importance of features in the Large dataset (from high to low)

F	5						11						16						Uniform					
NT	1	10	25	50	75	100	1	10	25	50	75	100	1	10	25	50	75	100	1	10	25	50	75	100
#1	20	2	19	2	2	2	20	19	19	19	19	19	14	14	14	14	14	14	17	4	19	21	14	14
#2	10	19	2	19	20	19	21	14	14	14	14	14	21	19	19	19	19	19	4	2	14	19	19	19
#3	1	20	20	21	19	21	13	20	20	2	4	4	19	1	1	4	4	4	14	17	4	4	21	21
#4	14	13	14	20	13	13	10	2	2	4	2	2	1	4	4	1	1	1	8	21	20	20	4	2
#5	13	21	13	14	21	20	19	21	4	21	17	8	4	21	7	17	17	17	21	19	7	14	20	4
#6	8	1	1	13	14	14	17	4	21	20	8	7	9	7	21	21	21	21	9	8	21	11	7	7
#7	2	6	21	4	4	11	7	1	1	8	20	17	2	18	17	6	7	7	1	14	17	7	11	18
#8	3	18	3	1	10	8	1	12	7	17	21	1	3	17	13	7	2	2	2	20	2	12	2	11
#9	4	3	4	6	7	10	2	7	9	1	1	21	5	13	6	2	6	6	3	7	11	2	13	20
#10	5	14	6	9	11	7	3	10	17	7	7	20	6	11	20	18	8	8	5	9	8	10	12	13
#11	6	4	11	11	8	4	4	13	13	9	9	18	7	6	2	13	18	18	6	13	13	18	18	8
#12	7	7	9	10	1	18	5	17	6	18	18	11	8	2	10	8	13	13	7	11	1	13	8	17
#13	21	10	7	8	18	1	6	18	8	6	11	9	10	3	11	10	11	11	10	12	10	17	10	12
#14	9	9	10	7	9	6	8	11	10	0	13	6	20	5	18	20	10	20	20	16	9	8	17	6
#15	11	11	8	18	6	9	16	9	12	13	6	16	11	10	9	11	20	10	11	18	12	9	1	10
#16	12	17	17	12	17	17	9	8	18	16	16	13	12	8	8	9	16	16	12	1	16	16	9	1
#17	15	0	18	3	3	3	11	15	0	11	0	0	13	9	12	5	9	9	13	3	18	6	6	9
#18	16	8	12	17	12	12	12	6	11	12	12	12	15	20	5	12	5	5	15	10	6	3	16	3
#19	17	5	16	0	0	0	18	5	5	10	10	10	16	12	15	3	12	12	16	5	5	1	3	16
#20	18	12	0	16	16	16	14	16	15	5	5	5	17	15	3	15	3	3	18	6	15	0	0	0
#21	19	15	5	15	15	5	15	3	3	3	3	3	18	16	16	16	15	15	19	15	3	15	15	5
#22	0	16	15	5	5	15	0	0	16	15	15	15	0	0	0	0	0	0	0	0	0	5	5	15

Table 18: Decision Forest – F1 Score in the Large dataset (CV)

F	5						11						16						Uniform					
NT	1	10	25	50	75	100	1	10	25	50	75	100	1	10	25	50	75	100	1	10	25	50	75	100
F1-Score$_{\mathrm{CV}}$	0.819	0.939	0.913	0.881	0.889	0.884	0.91	0.913	0.0	0.0	0.0	0.0	0.997	0.0	0.0	0.0	0.0	0.0	0.99	0.858	0.806	0.511	0.0	0.0

Table 19: Decision Forest – Metrics in the Large dataset (Test)

F	5						11						16						Uniform					
NT	1	10	25	50	75	100	1	10	25	50	75	100	1	10	25	50	75	100	1	10	25	50	75	100
Accuracy	0.806	0.961	0.924	0.906	0.914	0.913	0.945	0.938	0.507	0.507	0.507	0.507	0.997	0.507	0.507	0.507	0.507	0.507	0.991	0.887	0.828	0.732	0.507	0.507
Precision	0.751	0.989	0.993	1.0	1.0	1.0	0.945	1.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	1.0	1.0	1.0	1.0	0.0	0.0
Recall	0.905	0.931	0.853	0.809	0.826	0.824	0.944	0.874	0.0	0.0	0.0	0.0	0.994	0.0	0.0	0.0	0.0	0.0	0.983	0.772	0.65	0.457	0.0	0.0
F1-Score	0.821	0.959	0.917	0.894	0.905	0.903	0.944	0.933	0.0	0.0	0.0	0.0	0.997	0.0	0.0	0.0	0.0	0.0	0.991	0.871	0.788	0.627	0.0	0.0