

NYCU Pattern Recognition, Homework 3

Deadline: Apr. 26, 23:59

Part. 1, Coding (80%):

For this coding assignment, you are required to implement the Decision Tree and Random Forest algorithms using only NumPy. Afterward, you will need to train your model on the provided dataset and evaluate its performance on the validation data.

(30%) Decision Tree

Requirements:

- Implement the **Gini** and **Entropy** for measuring the "best" splitting of the data.
- Implement the Decision Tree algorithm ([CART, Classification and Regression Trees](#)) with the following 3 arguments:
- **Criterion**: The function to measure the quality of a split of the data. Your model should support "gini" for the Gini impurity and "entropy" for the information gain.
- **Max_depth**: The maximum depth of the tree. If Max_depth=None, then nodes are expanded until all leaves are pure. Max_depth=1 equals splitting data once.
- **Max_features**: The number of features to consider when looking for the best split. If None, then max_features=n_features.
- For more detailed descriptions of the arguments, please refer to [Scikit-learn](#).
- Your model should produce the same results when rebuilt with the same arguments, and there is no need to prune the trees.
- You can use the recursive method to build the nodes.

Criteria:

1. (5%) Compute the Entropy and Gini index of the array provided in the sample code, using the formulas on page 6 of the HW3 slide.

```
[ '+' '+' '+' '+' '+' '-' ]: entropy = 0.6500224216483541
[ '+' '+' '+' '-' '-' '-' ]: entropy = 1.0
[ '+' '-' '-' '-' '-' '-' ]: entropy = 0.6500224216483541

[ '+' '+' '+' '+' '+' '-' ]: gini index = 0.2777777777777777
[ '+' '+' '+' '-' '-' '-' ]: gini index = 0.5
[ '+' '-' '-' '-' '-' '-' ]: gini index = 0.2777777777777777
```

2. (10%) Show the accuracy score of the validation data using criterion='gini' and max_features=None for max_depth=3 and max_depth=10, respectively.
accuracy score = 0.73125, baseline = 0.73

```
print("Q2-1 max_depth=3: ", acc)
```

✓ 2.4s

```
Q2-1 max_depth=3: 0.73125
```

accuracy score = 0.86375, baseline = 0.85

```
print("Q2-2 max_depth=10:
```

✓ 4.8s

```
Q2-2 max_depth=10: 0.86375
```

3. (10%) Show the accuracy score of the validation data using `max_depth=3` and `max_features=None`, for `criterion='gini'` and `criterion='entropy'`, respectively.

accuracy score = 0.73125, baseline = 0.73

```
print("Q3-1 criterion='gini'
```

✓ 2.3s

```
Q3-1 criterion='gini': 0.73125
```

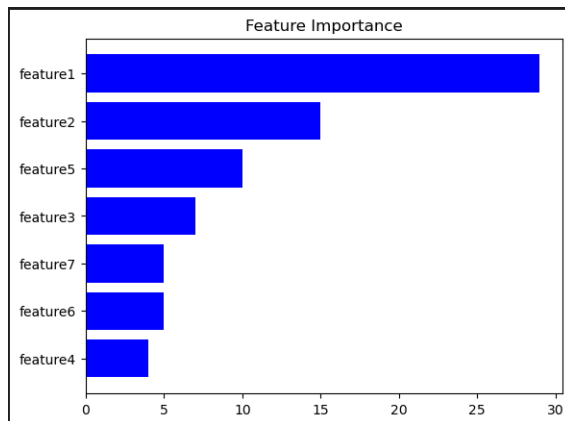
accuracy score = 0.7725, baseline = 0.77

```
print("Q3-2 criterion='entropy'
```

✓ 2.5s

```
Q3-2 criterion='entropy': 0.7725
```

4. (5%) Train your model using `criterion='gini'`, `max_depth=10` and `max_features=None`. Plot the [feature importance](#) of your decision tree model by simply counting the number of times each feature is used to split the data.



(20%) Random Forest

Requirements:

- Fix the random seed.
- Implement the Random Forest algorithm by using the CART you just implemented.
- The Random Forest model should include the following three arguments:
- **N_estimators**: The number of trees in the forest.
- **Max_features**: The number of features to consider when looking for the best split using the decision tree.
- **Bootstrap**: Whether to use bootstrap samples when building trees.
- For more detailed descriptions of the arguments, please refer to [Scikit-learn](https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html).
- Use majority voting to obtain the final prediction.

Criteria:

5. (10%) Show the accuracy score of the validation data using criterion='gini', max_depth=None, max_features=sqrt(n_features), and bootstrap=True, for n_estimators=10 and n_estimators=50, respectively.

accuracy score = 0.8825, baseline = 0.88

```
print("Q6-1 n_estimators=10:
✓ 13.3s
Q6-1 n_estimators=10: 0.8825
```

accuracy score = 0.89875, baseline = 0.89

```
print("Q6-1 n_estimators=50:
✓ 1m 4.0s
Q6-1 n_estimators=50: 0.89875
```

6. (10%) Show the accuracy score of the validation data using criterion='gini', max_depth=None, n_estimators=10, and bootstrap=True, for max_features=sqrt(n_features) and max_features=n_features, respectively. accuracy score = 0.88375, baseline = 0.88

```
print("Q7-1 max_features='sqrt'
✓ 12.4s
Q7-1 max_features='sqrt': 0.88375
```

accuracy score = 0.87875, baseline = 0.86

```
• print("Q7-2 max_features='All'
✓ 23.7s
Q7-2 max_features='All': 0.87875
```

(20%) Train your own model

Requirements:

- Train your model (either Decision Tree or Random Forest).
- Try different parameters and feature engineering to beat the baseline.
- Save your test predictions in a CSV file.

Criteria:

7. (20%) Explain how you chose/design your model and what feature processing you have done in detail. Otherwise, no points will be given.

According to Q4, I know there are at least 3 features important for classifying this data, so I select 4 features for random forest to add sum variance during classification.

Then, I set number of trees less than $C(7, 4)$, and get 28 via experient.

Points	Testing Accuracy
20 points	acc > 0.915
15 points	acc > 0.9
10 points	acc > 0.88
5 points	acc > 0.8
0 points	acc <= 0.8

Part. 2, Questions (30%):

1. Answer the following questions in detail:

a. Why does a decision tree tend to overfit the training set?

There are several reasons why decision trees tend to overfit the training set:

- i. Decision trees have high variance: Decision trees have a tendency to overreact to small fluctuations in the data, which can result in overly complex trees that are tailored to the training set.
- ii. Overly complex trees: Decision trees can grow too large and too complex if there are many features or if the tree depth is not limited, resulting in a model that is overly specific to the training data and does not generalize well to new data.
- iii. Noise in the data: Decision trees are sensitive to noise in the data, which can result in spurious splits that capture noise rather than meaningful patterns in the data.
- iv. Lack of regularization: Decision trees can benefit from regularization techniques, such as pruning or early stopping, that reduce the complexity of the tree and prevent overfitting. Without these techniques, the tree can continue to grow until it perfectly fits the training data.

b. Is it possible for a decision tree to achieve 100% accuracy on the training set?

Yes, if there is no limitation for the depth and width of tree, decision tree can separate each data, that is, make each leaf node only has one data. Then, it can achieve 100% accuracy on the training set.

c. List and describe at least three strategies we can use to reduce the risk of overfitting in a decision tree.

- i. Pruning: This involves removing some of the branches of the decision tree that do not contribute significantly to the overall accuracy of the model. This is typically done by setting a threshold value for the minimum number of instances required to form a leaf node, below which the tree is pruned.
- ii. Regularization: This involves adding a penalty term to the decision tree algorithm that discourages it from creating overly complex models. This can be done by adjusting the hyperparameters of the model, such as the maximum depth of the tree or the minimum number of instances required at each leaf node.
- iii. Cross-validation: This involves partitioning the data into multiple training and validation sets and iteratively training the decision tree model on different combinations of these sets. This can help to identify and prevent overfitting by ensuring that the model generalizes well to new data.

2. For each statement, answer True or False and provide a detailed explanation:

- a. In AdaBoost, weights of the misclassified examples go up by the same multiplicative factor.

False: In AdaBoost, the weights of the misclassified examples are increased by a higher multiplicative factor for more difficult examples (those with higher error rate). Therefore, the weights of misclassified examples do not go up by the same multiplicative factor.

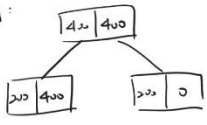
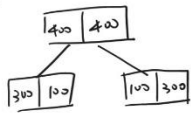
- b. In AdaBoost, weighted training error ϵ_t of the t_{th} weak classifier on training data with weights D_t tends to increase as a function of t .

False: The weighted training error ϵ_t of the t_{th} weak classifier on training data with weights D_t tends to decrease as a function of t , not increase. This is because the goal of AdaBoost is to reduce the training error by iteratively boosting the weights of the misclassified examples.

- c. AdaBoost will eventually give zero training error regardless of the type of weak classifier it uses, provided enough iterations are performed.

False: AdaBoost is a powerful algorithm that can achieve very low training error, but it may not always give zero training error. The training error of the ensemble typically decreases with each iteration of boosting, but it may not reach zero due to limitations of the weak classifier or noise in the data.

3. Consider a data set comprising 400 data points from class C_1 and 400 data points from class C_2 . Suppose that a tree model A splits these into (200, 400) at the first leaf node and (200, 0) at the second leaf node, where (n, m) denotes that n points are assigned to C_1 and m points are assigned to C_2 . Similarly, suppose that a second tree model B splits them into (300, 100) and (100, 300). **Evaluate the misclassification rates for the two trees and hence show that they are equal.** Similarly, **evaluate the cross-entropy** $Entropy = -\sum_{k=1}^K p_k \log_2 p_k$ and **Gini index** $Gini = 1 - \sum_{k=1}^K p_k^2$ for the two trees. Define p_k to be the proportion of data points in region R assigned to class k, where $k = 1, \dots, K$.

<p>A:</p> 	<p>misclassification rate:</p> $200/800 = \frac{1}{4}$	<p>CE:</p> <p>left:</p> $-(\frac{1}{2} \log_2 \frac{1}{2} + \frac{2}{2} \log_2 \frac{2}{2}) = 0.918$ <p>right:</p> 1	<p>Gini:</p> <p>left:</p> $1 - (\frac{1}{2})^2 - (\frac{2}{2})^2 = 0.444$ <p>right:</p> $1 - 1^2 = 0$
<p>B:</p> 	$100/800 + 100/800 = \frac{1}{4}$	<p>left:</p> $-(\frac{3}{4} \log_2 \frac{3}{4} + \frac{1}{4} \log_2 \frac{1}{4}) = 0.811$ <p>right:</p> $-(\frac{1}{4} \log_2 \frac{1}{4} + \frac{3}{4} \log_2 \frac{3}{4}) = 0.811$	<p>left:</p> $1 - (\frac{3}{4})^2 - (\frac{1}{4})^2 = 0.375$ <p>right:</p> 0.375