

NCTU Pattern Recognition, Homework 3

Deadline: May 26, 23:59

Part. 1, Coding (60%):

In this coding assignment, you are required to implement the decision tree, and random forest algorithm by using only [NumPy](#), then train your model on the provided dataset and evaluate the performance on testing data. Find the sample code and data on the GitHub page

https://github.com/NCTU-VRDL/CS_ILE5065/tree/main/HW3

Please note that only [NumPy](#) can be used to implement your model. You will get zero points by simply calling `sklearn.tree.DecisionTreeClassifier`. And note that all of the model accuracy scores should be over 0.9

1. (5%) Please compute the Entropy and Gini Index of the given array by the formula on the [slides](#).
2. (20%) Implement the Decision Tree algorithm ([CART, Classification and Regression Trees](#)) and train the model by the given arguments, and print the accuracy score on the test data. You should implement **two arguments** for the Decision Tree algorithm, 1) **Criterion**: The function to measure the quality of a split. Your model should support “gini” for the Gini impurity and “entropy” for the information gain.
2) **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 to split data once
 - 2.1. Using Criterion=‘gini’ to train the model and show the accuracy score of test data by Max_depth=3 and Max_depth=10, respectively.
 - 2.2. Using Max_depth=3 to train the model and show the accuracy score of test data by Criterion=‘gini’ and Criterion=‘entropy’, respectively.

*Note: You should get the same results when re-building the model with the same arguments, **no need to prune the trees***

Note: You can find the best split threshold by two methods. First one: 1) Try N-1 threshold values, where the i-th threshold is the average of the i-th and (i+1)-th sorted values. Second one: Use the unique sorted value of the feature as the threshold to split the data.

Hint: You can use the recursive method to build the nodes

3. (15%) Plot the [feature importance](#) of your Decision Tree model. You can use the model for Question 2.1, max_depth=10. (You can simply count the number of a feature used in the tree, instead of the formula in the reference. Find more details on the sample code. (matplotlib is allowed to use))
4. (20%) Implement the [random forest](#) algorithm by using the CART you just implemented for Question 2. You should implement **three arguments** for the random forest model.

- 1) **N_estimators**: The number of trees in the forest.
- 2) **Max_features**: The number of features to consider when looking for the best split
- 3) **Bootstrap**: Whether bootstrap samples are used when building trees

- 4.1. Using Criterion='gini', Max_depth=None, Max_features=sqrt(n_features), Bootstrap=True to train the model and show the accuracy score of test data by n_estimators=10 and n_estimators=100, respectively.
- 4.2. Using Criterion='gini', Max_depth=None, N_estimators=10, Bootstrap=True, to train the model and show the accuracy score of test data by Max_features=sqrt(n_features) and Max_features=n_features, respectively.

Note: Use majority votes to get the final prediction, you may get different results when re-building the random forest model

Part. 2, Questions (40%):

1. (15%) By differentiating the error function below with respect to α_m ,

$$\begin{aligned}
 E &= e^{-\alpha_m/2} \sum_{n \in T_m} w_n^{(m)} + e^{\alpha_m/2} \sum_{n \in M_m} w_n^{(m)} \\
 &= (e^{\alpha_m/2} - e^{-\alpha_m/2}) \sum_{n=1}^N w_n^{(m)} I(y_m(x_n) \neq t_n) + e^{-\alpha_m/2} \sum_{n=1}^N w_n^{(m)}
 \end{aligned}$$

show that the parameters α_m in the AdaBoost algorithm are updated using

$$\alpha_m = \ln \left\{ \frac{1 - \epsilon_m}{\epsilon_m} \right\} \text{ in which } \epsilon_m \text{ is defined by } \epsilon_m = \frac{\sum_{n=1}^N w_n^{(m)} I(y_m(x_n) \neq t_n)}{\sum_{n=1}^N w_n^{(m)}}.$$

2. (15%) 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 (300, 100) assigned to the first leaf node (predicting C_1) and (100, 300) assigned to the second leaf node (predicting C_2), where (n, m) denotes that n points come from class C_1 and m points come from class C_2 . Similarly, suppose that a second tree model B splits them into (200, 400) and (200, 0), respectively. Evaluate the misclassification rates for the two trees and hence show that they are equal. Similarly, evaluate the pruning criterion $C(T) = \sum_{\tau=1}^{|T|} Q_{\tau}(T) + \lambda|T|$ for the cross-entropy case $Q_{\tau}(T) = \sum_{k=1}^K p_{\tau k} \ln(p_{\tau k})$ for the two trees and show that tree B is lower than tree A. Leaf nodes are indexed by

$\tau = 1, \dots, |T|$, with leaf node τ represents a region R_τ , and $p_{\tau k}$ is the proportion of data points in region R_τ assigned to class k , where $k = 1, \dots, K$.

Hint: The answer should contain λ which is the regularization parameter.

3. (10%) Verify that if we minimize the sum-of-squares error between a set of training values $\{t_n\}_{n=1 \sim N}$ (N is number of training data) and a single predictive value t , then the optimal solution for t is given by the mean of the $\{t_n\}_{n=1 \sim N}$.