Lab 2: Classifiers and their Performance

Nischal Regmi
Everest Engineering College

2022

This lab will focus on using decision tree. The practical concepts learned in the context of building decision trees are applicable to other classifiers too.

Problems

1. **Basics**: The file 'discreteLevelData_discretized.csv' contains binary variables describing various district level indicators for Nepal. The number '0' represents low and '1' high value of the corresponding variables. The objective here is to understand what describes the variation in district level internet access in Nepal from the given dataset. Run the Python script 'lab2_example1.csv' and answer the following questions.

   (a) Taking maximum allowable depth (max_depth) of the tree as 3 and minimum number of samples to split a node (min_samples_split) as 10, build the decision tree and describe what characterizes the districts with high and low internet access. [You need to answer in plain English, in one paragraph.]

   (b) Refer to the official documentation of *sklearn* and describe in brief at least three other hyperparameters (other than max_depth and min_samples_split) that define the structure of a decision tree.

   (c) *sklearn*'s function *DecisionTreeClassifier()* has a parameter called random_state. But the algorithm we studied, which basically is the Hunt's algorithm, does not has randomness. Why has *sklearn* included the random_seed parameter? [Refer to the official documentation]

2. **Training and Test Errors**: Run the Python script 'lab2_example2.csv' and answer the following questions.

   (a) Do you think taking max_depth equal to 25 and other parameters their default values has resulted in *overfitting*? Why?

   (b) Keeping other parameters to their default, what value of max_depth will result in maximum test accuracy? For this you need to calculate the test and train accuracy for different values of max_depth, say from 2 to 30, and plot how the train and test accuracy change with max_depth. [You may plot error instead of accuracy]. Modify the example script to answer this question.

3. **Confusion Matrix**: Interpret the performance of your classifier in the model of question 2 based on the confusion matrix. Does the classifier perform well for each class label?

4. **Predictor Importance**: A simple method to assess which variables are more important in a classifier is the *permutation test.* The idea is simple. To test the importance of a variable x_j, build a classifier $\hat{f}$ from the dataset D and

   - Randomly permute (shuffle) the value of the variable x_j in the data set D, creating a new dataset D'.

- Calculate the accuracy of $\hat{f}$ in D'. If x_j is an important predictor variable, then its permutation should result in significantly lower value of the accuracy.

Your task is apply permutation test for all the predictor variable in your model of question 2.

5. Explore the Internet to find at least two other methods of assessing variable importance in a classification (or, regression) model. Also think how could you implement them in Python.

6. **Cross-Validation**: Use a 5-fold cross validation to find the best value of max_depth, keeping other hyperparameters to their default values. You can use the Python library *sklearn.model_selection.cross_validate*