[Mini Project 1] **Project Code : NSDT**

Nursery School Application Selection using Decision Tree based Learning Model

Project Duration : 25-Jan-2021 ~~ 08-Feb-2021
Submission Information : (via) CSE-Moodle

Objective:
In the 1980s, there was a period when an excessive enrollment drive in Nursery schools took place in the countries like Ljubljana, Slovenia, and hence there are thousands of applications which are to be shortlisted. In the process, the rejected applications are frequently asked for objective explanations/reasons. Since the selection was of nursery school students, so a deterministic performance criteria was not sufficient to fix a cutoff for the same. The Primary School Education Minister of those countries seek the help of Machine Learning experts to formulate this task as a decision tree learning problem so that not only they have a strategy for proper selection, but also have a strategy for explaining the non-selected candidates.
Your task is to build a decision tree learning model to enable such activity/survey on behalf of the authority. In particular, you shall be doing the following tasks:

1. Based on the dataset (described later), you will write a program to learn a decision tree. You have to use two methods in such decision tree learning:
    a. *Method-1:* Decision tree learning should use *entropy-based information gain* in selecting attribute choices while building the tree
    b. *Method-2:* Decision tree learning should use gini index based criteria for choosing the attribute for splitting.
2. Use decision tree pruning techniques to eliminate overfitting
    Tree pruning should be performed at different depths. For pruning you may create a simple function to run your model using different values for a function maxdepth (say, from 1 to 25) and visualize its results to see how the accuracy differs for each criterion value (i.e. for *gini index* and *information gain*). Based on the best accuracy criteria at a certain depth the tree structure should be printed as output.
3. Finally, both methods should be compared before and after pruning with the given set of test data samples.

Note: *The program can be written in C / C++ / Java / Python programming language from scratch. No machine learning /data science /statistics package / library should be used.*

DataSets:
Training Data Filename: `nursery.csv`
Test Data Filename: `nursery_test.csv`

Training Data Description: The **attribute** Information is given as follows.

- *parents*: usual, pretentious, great_pret
- *has_nurs*: proper, less_proper, improper, critical, very_crit
- *form*: complete, completed, incomplete, foster
- *children*: 1, 2, 3, more
- *housing*: convenient, less_conv, critical
- *finance*: convenient, inconv
- *social*: non-prob, slightly_prob, problematic
- *health*: recommended, priority, not_recom

Output Classes: recommend, very_recom, spec_prior, priority, not_recom

Your Tasks:

1. Decision Tree Model without Pruning:
    a. Implement the standard **ID3 , Gini Decision tree algorithm** as discussed in class, using information gain to choose which attribute to split at each point. Do NOT use scikit-learn for this part.

   b. Test out the implementation of Decision Tree Classifier from scikit-learn package, using information gain.
2. Revised Decision Tree Model with Pruning
   a. To prune the tree, you have to use Reduced Error Pruning. You have to extract 10% of the dataset as a validation set.
   b. Now, to estimate the performance of decision trees with and without pruning you have to use the k-fold stratified cross validation with k = 10 on the remaining 90% of data and to see how the accuracy improves do the learning with ever larger training sets starting with 10 examples up to using all the data. After that , plot the accuracy of the two models.
   c. Finally, use the *entire* training set to learn a decision tree with and without pruning and use the test set to estimate their accuracy.
3. Classification Report
   a. Create a classification report for both the trees in tabular form. (with and without pruning).
   b. You need to calculate precision , recall , f1-score and support of the model you made for each and every class(bad , good , ok , vgood)
   c. Also get the accuracy of the model.

Submission Details: (to be submitted under the specified entry in CSE-Moodle)

1. ZIPPED Code Distribution in CSE-Moodle
2. A brief (2-3 page) report/manual of your work
   (with your hyperparameter tuning results also presented in that report)

Submission Guidelines:

1. You may use one of the following languages: C/C++/Java/Python.
2. Your Programs should run on a Linux Environment.
3. You are **not** allowed to use any library apart from these (Also explore all these libraries if doing in Python, or equivalent of these):

```
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
from sklearn.metrics import classification_report
from sklearn.model_selection import KFold
import operator
from math import log
from collections import Counter
from statistics import mean
```

   Your program should be standalone and should **not** use any *special purpose* library for Machine Learning. Numpy and Pandas may be used. And, you can use libraries for other purposes, such as generation and formatting of data.
4. You should submit the program file and README file and not the output/input file.
5. You should name your file as <GroupNo_ProjectCode.extension>
   (e.g., Group1_NSDT.pdf or Group1_NSDT.zip).
6. The submitted program file *should* have the following header comments:
   # Group Number
   # Roll Numbers : Names of members (listed line wise)
   # Project Number
   # Project Title

You should not use any code available on the Web. Submissions found to be plagiarised or having used ML libraries (except for parts where specifically allowed) will be awarded zero marks.

For any questions about the assignment, contact the following TAs: Jay Bhutada (jbhutada96@gmail.com)