Machine Learning COMP 5630/ COMP 6630/ COMP 6630 - D01

Instructor: Dr. Shubhra ("Santu") Karmaker
TA 1: Dongji Feng
TA 2: Souvika Sarkar
Department of Computer Science and Software Engineering
Auburn University
Fall, 2022

August 29, 2022

LINK TO MY REPOSITORY CONTAINING CODE:

Assignment #1 Decision Tree

Tasks

1. **Decision Tree Basics [50 pts]:** The goal of this assignment is to test and reinforce your understanding of Decision Tree Classifiers.

   (a) **[5 pts]** How many unique, perfect binary trees of depth 3 can be drawn if we have 5 attributes. By depth, we mean depth of the splits, not including the nodes that only contain a label (see Figure 1). So a tree that checks just one attribute is a depth 1 tree. By perfect binary tree, we mean every node has either 0 or 2 children, and every leaf is at the same depth. Note also that a tree with the same attributes but organized

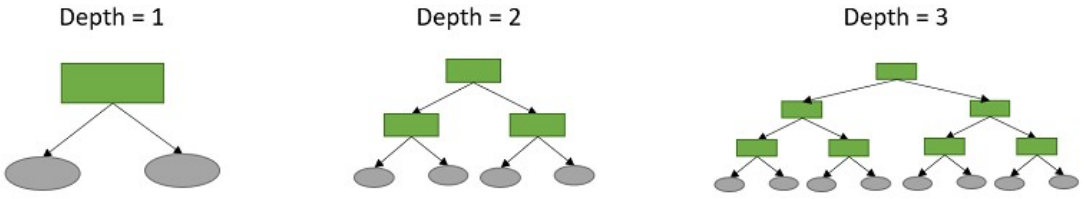


Figure 1: Example of perfect binary trees with different depths.

   at different depths are considered "unique". Do not include trees that test the same attribute along the same path in the tree.

   The number of distinct decision trees with 5 attributes is <u>6480</u> distinct perfect binary trees with depth 3.

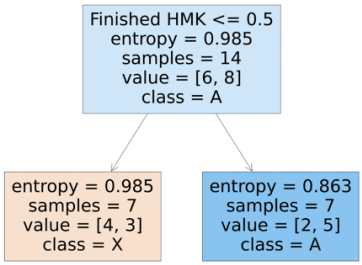
[5 pts] In general, for a problem with A attributes, how many unique perfect D depth trees can be drawn? Assume *A>>D*

$$\prod_{D=1}^{3} (A - D + 1)^{2^{D-1}}$$

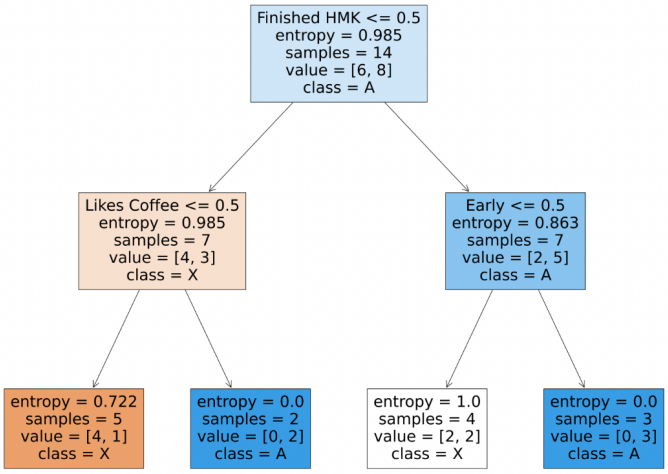
This was used to calculate the above answer

(b) **[20 pts]** Consider the following dataset for this problem. Given the five attributes on the left, we want to predict if the student got an A in the course. Create 2 decision trees for this dataset. For the first, only go to depth 1. For the second go to depth 2. For all trees, use the ID3 entropy algorithm from class. For each node of the tree, show the decision, the number of positive and negative examples and show the entropy at that node.

i **Decision Tree: Depth 1**



ii **Decision Tree: Depth 2**



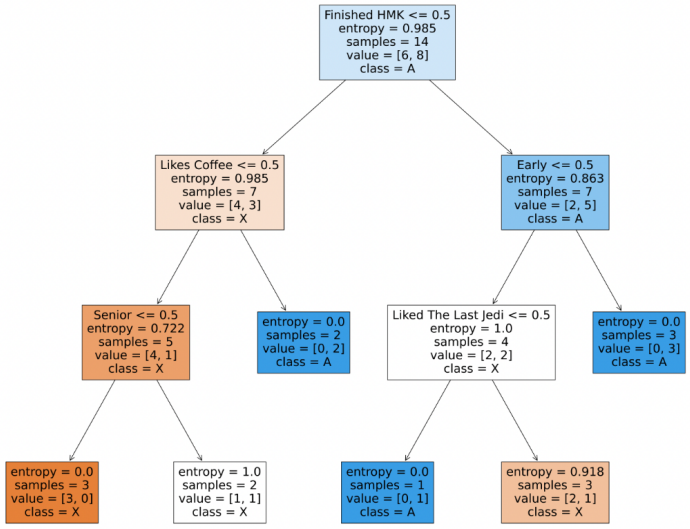
Hint: There are a lot of calculations here. You may want to do this programatically.

Early	Finished HMK	Senior	Likes Coffee	Liked The Last Jedi	A
1	1	0	0	1	1
1	1	1	0	1	1

0	0	1	0	0	0
0	1	1	0	1	0
0	1	1	0	0	1
0	0	1	1	1	1
1	0	0	0	1	0
0	1	0	1	1	1
0	0	1	0	1	1
1	0	0	0	0	0
1	1	1	0	0	1
0	1	1	1	1	0
0	0	0	0	1	0
1	0	0	1	0	1

Table 1: Toy Data-set for Task 1: Decision Tree Basics.

(c) **[10 pts]** Make one more decision tree. Use the same procedure as in (b), but make it depth 3. Now, given these three trees, which would you prefer if you wanted to predict the grades of 10 new students who are not included in this data-set? Justify your choice.



In order to predict the grades of 10 new students, not included in the training dataset, I would choose to use a decision tree of depth 2. The reason for this selection is the limited number of instances or datapoints used to create a decision tree. There are 2^5 = 32 possible different instances that can be presented to the decision tree. Despite this, only 14 instances are provided to train the tree, and 10 new cases are presented to predict. These 10 instances could all be classified incorrectly and show that the original data used to train a decision tree wasn't indicative of actual trends. A depth of 3 runs

the risk of overfitting too much to the 14 instances used to build the tree. A depth of 1 is too general, as the "Finished HMK" alone incorrectly classifies over 40% training set.

(d) **[10 pts]** Recall the definition of the "realizable" case. "For some fixed concept class *C*, such as decision trees, a realizable case is one where the algorithm gets a sample consistent with some concept $c \in C$. In other words, for decision trees, a case is realizable if there is some tree that perfectly classifies the data-set.

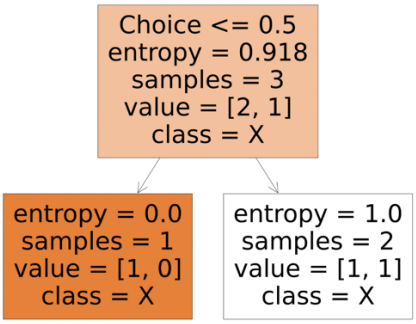
If the number of attributes A is sufficiently large, under what condition would a dataset not be realizable for decision trees of no fixed depth? Prove that the dataset is unrealizable if and only if that condition is true.

My understanding is that a realizable case occurs when the decision tree is able to perfectly model a dataset. Meaning that a certain combination of feature values leads to a specific classification. With that understanding, an unrealizable case occurs when the same combination of feature values has a different target classification.

So, when the condition of two instances have same feature values but different classifications in the same dataset is true, a case is definitively unrealizable as a decision tree could not classify both instances correctly. To elaborate, two identical instances would follow the same exact decision path through a decision tree both ending at the same classification. However, this would be incorrect as we have previously established those two identical instances have different classifications.

Simple Example:

Choice	*Class*
0	*0*
0	*1*
1	*0*



Decision tree that perfectly classifies dataset is unrealizeable.

2. **Application on Real-Word Data-set [50 pts]:** In this task, you will build a decision tree classifier using a real-word data-set called Census-Income Data Set available publicly for downloading at Dataset. This data set contains weighted census data extracted from the 1994 and 1995 Current Population Surveys conducted by the U.S. Census Bureau. The data contains 41 demographic and employment related variables.

   The instance weight indicates the number of people in the population that each record represents due to stratified sampling. To do real analysis and derive conclusions, this field must be used. **This attribute should *not* be used in the classifiers!!!**

   One instance per line with comma delimited fields. There are 199,523 instances in the data file and 99,762 in the test file.

   The data was split into train/test in approximately $\frac{2}{3}$, $\frac{1}{3}$ proportions using MineSet's MIndUtil mineset-to-mlc. Below are your tasks:

   (a) **[20 pts]** Train a decision tree classifier using the data file. Feel free to use existing python/java packages/libraries you may like. You cannot touch the test file in this part. Vary the cut-off depth from 2 to 10 and report the training accuracy for each cut-off depth *k*. Based on your results, select an optimal *k*.

   See command line output below…

   (b) **[15 pts]** Using the trained classifier with optimal cut-off depth *k*, classify the 99,762 instances from the test file and report the testing accuracy (portion of testing instances classified correctly).

```
(venv) christophercasey@Christophers-MacBook-Air DecisionTree % python3 DecisionTree.py

Decision tree built and tested with test/train split of Training Dataset Only
Accuracy of Decision Tree built with training set having depth  2 :  94.02831725347701
Accuracy of Decision Tree built with training set having depth  3 :  94.02831725347701
Accuracy of Decision Tree built with training set having depth  4 :  94.6422754040847
Accuracy of Decision Tree built with training set having depth  5 :  94.6573111138955
Accuracy of Decision Tree built with training set having depth  6 :  95.04573361734118
Accuracy of Decision Tree built with training set having depth  7 :  95.16100739255732
Accuracy of Decision Tree built with training set having depth  8 :  95.14847763438166
Accuracy of Decision Tree built with training set having depth  9 :  95.16601929582758
Accuracy of Decision Tree built with training set having depth  10 :  95.1058764565844

Optimal Depth has been found to be 9

Decision Tree with predetermined best depth. Model buillt with partial training dataset from earlier testing
Accuracy of classification with Test Data having depth of  9 :  94.82768990196668

Decision Tree with predetermined best depth. Model buillt with entire training dataset
Accuracy of classification with Test Data having depth of  9 :  94.80964695976424
(venv) christophercasey@Christophers-MacBook-Air DecisionTree % python3 DecisionTree.py

Decision tree built and tested with test/train split of Training Dataset Only
Accuracy of Decision Tree built with training set having depth  2 :  94.02831725347701
Accuracy of Decision Tree built with training set having depth  3 :  94.02831725347701
Accuracy of Decision Tree built with training set having depth  4 :  94.6422754040847
Accuracy of Decision Tree built with training set having depth  5 :  94.6573111138955
Accuracy of Decision Tree built with training set having depth  6 :  95.04573361734118
Accuracy of Decision Tree built with training set having depth  7 :  95.1585014409222
Accuracy of Decision Tree built with training set having depth  8 :  95.15098358601679
Accuracy of Decision Tree built with training set having depth  9 :  95.16601929582758
Accuracy of Decision Tree built with training set having depth  10 :  95.08582884350332

Optimal Depth has been found to be 9

Decision Tree with predetermined best depth. Model buillt with partial training dataset from earlier testing
Accuracy of classification with Test Data having depth of  9 :  94.83470660171207

Decision Tree with predetermined best depth. Model buillt with entire training dataset
Accuracy of classification with Test Data having depth of  9 :  94.80363264569675
(venv) christophercasey@Christophers-MacBook-Air DecisionTree %
```

The above output contains 2 runs of my program for redundancy.

(c) **[15 pts]** Do you see any over-fitting issues for this experiment? Report your observations.

In order to detect overfitting, I split the original training data set into a train test split of 80% and 20% respectively. This was to follow the original directions of not using the testing set at all, while finding best depth k. If I had not done this common train test split, just training with the entire training dataset, finding optimal k would have been impossible as accuracy would have increased with k. However, to cater to the issues that could arise when comparing our outcomes, I made sure to test the classifiers final test set accuracy with both a model generated with the entire training set (100%) and the previously used partial training dataset (80%).

Overfitting is the inability for a model to generalize datasets. This can be seen when the training accuracy is greater than the testing accuracy. Further, the depth at which overfitting occurs can be observed when the accuracy scored peaks. After this peak accuracy begins to trend downward. This is as a result of the model catering more and more closely to the training data specifically. Thus, losing its accuracy when applied to new test data. We can measure overfitting with:

$$Training\ Accuracy > Testing\ Accuracy$$

The outputs displayed above show that this is the case. We summarize the above outputs below.

$$\mathbf{95.2} > \mathbf{94.8}$$

Disclaimers: This assignment re-uses some materials from the publicly available website: CMU Introduction to Machine Learning Course, 10-315, Spring 2019. I personally thank Prof. Maria-Florina Balcan for sharing her teaching materials publicly. This assignment is exculively used for instructional purposes.