Assignment2 – Decision Tree

Due date: Thursday October 17, 11:59 PM
Start early. This is not an easy assignment like assignment 1

Objective

Using data from the census bureau database for the year 1994, build a decision tree that classifies whether a person's yearly income is greater than 50K or less than or equal to 50K. The data was obtained from here (University of California – Irvine).

You must use the **ID3** algorithm and **Information Gain** as the statistical test to determine on what attributes to split data.

Training and Test Data Provided to You

The training and test data are found in the enclosed subfolder named "data". There are 4 files:

- ***census_training.csv***: use this dataset (30110 training examples) to build your decision tree model using ID3 and Info Gain.
- ***census_training_test.csv***: use this dataset (15028 examples) to test the accuracy of your decision tree model.

  The data in *census_training.csv* and *census_training_test.csv* consist of 11 attributes (all categorical) and 1 class label: *high_income*, which can be <=50K or >50K.

- *playtennis*.csv: a small dataset for which you know how the tree looks like. Use this small dataset to validate that your algorithm is somewhat working (tree shape is in Appendix A of this document)
- *emails*.csv: a small dataset for which you know how the tree looks like. Use this small dataset to validate that your tree is somewhat working (tree shape is in Appendix A of this document).

The following pre-processing was done to *census_training.csv* and *census_training_test.csv*:

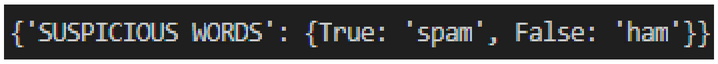
- 3 attributes that existed in the original dataset were removed. These are *fnlwgt*, *capital_gain*, and *capital_loss*.
- 3 attributes that are real-valued in the original dataset were converted to categorical attributes. These are *age, education_num*, and *hours_per_week*.
- Rows containing missing data or unknown data were removed.

Implementation Hints

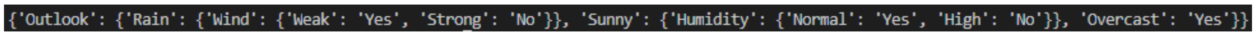
- Build your code incrementally: create helper functions that you are likely to call (calculating entropy, information gain, best feature to choose for a split, etc.). Test those functions on small data for which you know the results of the calculation (e.g., the Information Gain example calculation in PowerPoint DecisionTree.pptx). **Ensure that these are working and doing correct calculations before proceeding** (you don't want to waste 2 days chasing some problem only to find out that your Info Gain formula is incorrect).

- Code your algorithm on small datasets that you know how their resulting decision trees look like: ***playtennis*.csv and *emails.csv* were provided to you for this purpose**. The shapes of the trees are in Figure 1 and 2 of Appendix A at the end of this document.

- When building the decision tree, save it in memory as a dictionary of dictionaries. For example, when printing the Email and PlayTennis trees (dictionaries), you should get something similar to:



```
{'SUSPICIOUS WORDS': {True: 'spam', False: 'ham'}}
```



```
{'Outlook': {'Rain': {'Wind': {'Weak': 'Yes', 'Strong': 'No'}}, 'Sunny': {'Humidity': {'Normal': 'Yes', 'High': 'No'}}, 'Overcast': 'Yes'}}
```

- When you verify that your algorithm is building the correct PlayTennis and Email trees, then you can test your algorithm on training data *census_training.csv*.

  You can also use *census_training.csv* to check your tree accuracy. Testing accuracy against the data you used to build the model with is not a correct check of how good the model is. It is done simply to check that your model captures the trends of the data that was used to build from (and that you did not build a completely wrong tree with arbitrary nodes). You might not get 100% accuracy on such real data (but it should be somewhat 90% or higher). On data that is hand-crafted such as the PlayTennis and Email datasets, you should get 100% accuracy if you test against the training data itself. This is obvious for a small hand-crafted dataset and the tree will be fully consistent with it.

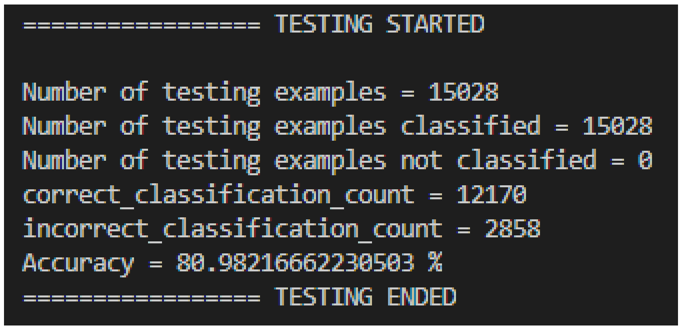
- Now use *census_training_test.csv* to test the accuracy of your Decision Tree model. Data in *census_training_test.csv* is data that your algorithm have never seen. So it is a good set to calculate how accurate its prediction is. Count the number of accurate classifications and the number of inaccurate classifications.

- Enclosed is a file named "DrawDecisionTree.docx". Follow the instructions if you want to draw your tree (but it is not required). After spending many hours you are probably curious to see how the tree looks like☺.

- My implementation takes not more than 3 minutes for training and testing to finish. If you see yourself waiting much longer, you might possibly have some problem in your recursion.

Output of Program

The output should print the accuracy against the test data (i.e., the data from ***census_training_test.csv)***.



```
================= TESTING STARTED

Number of testing examples = 15028
Number of testing examples classified = 15028
Number of testing examples not classified = 0
correct_classification_count = 12170
incorrect_classification_count = 2858
Accuracy = 80.98216662230503 %
================= TESTING ENDED
```

In my case the accuracy is about 81%.

Grading

Your grade will be computed as follows:

1. 40% - Code runs without errors and builds a tree (dictionary of dictionaries).

2. 10% - You calculated the accuracy by checking your model against the test data (***census_training_test.csv***).

3. 20% - Output as shown above (your accuracy might be slightly different).

4. 30% - You are using ID3, Info Gain, and code is well commented.

    a. You are implementing ID3 with Info Gain. Algorithm steps are correct as explained in book and class.
    b. Each function should have a comment as to what it does and what it returns.
    c. There are comments along the steps your program is taking and I am able to easily follow/understand the steps.

Late submission policy:

Up to 2 days late: 10% penalty
Up to 5 days late: 25% penalty
More than 5 days late: Not accepted - you will lose all points of the assignment.

Appendix A

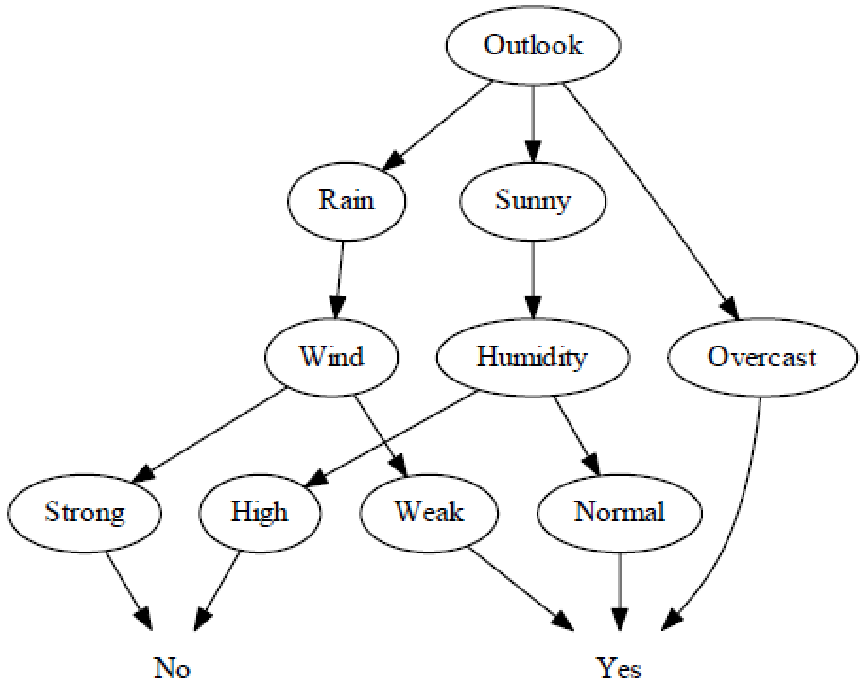


Figure 1: PlayTennis decision tree.

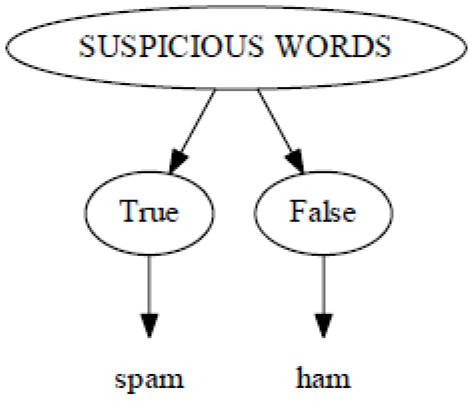


Figure 2: Emails decision tree.