
Task 1

Code completed in decision_tree.py and uci_data.py.

Full log files are included in a logs directory because why not. The specified log files without the excluded data are included in the root directory of the zip. Log file names are formatted:

<training_file>-<test_file>-<option>-<pruning_thr>.log

The specific files are:

pendigits_training.txt-pendigits_test.txt-1-50.log

pendigits_training.txt-pendigits_test.txt-3-50.log

pendigits_training.txt-pendigits_test.txt-optimal-50.log

b. TODO

c. TODO

Task 2

Example functions:

Entropy(counts)

Weight(numerator, denominator)

a.

wait = 80 no_wait = 20 total = 100

entropy = -(80/100)*log2(80/100) - (20/100)*log2(20/100) = 0.7219

b.

entropy([80,20])

- weight(35,100)*entropy([20,15])

- weight(65,100)*entropy([60,5])

= 0.7219 - 0.35*0.8112 - 0.65*0.8112 = 0.1228

c.

Because we know all the examples at this node is during the weekend, every example will go to node H. If every decision goes to one node, there is no information gained.

d.

A -> C -> F. The tree says the patron will wait.

e.

A -> B -> E -> H. The tree says the patron will not wait.

Task 3

A =

entropy([5,5])

- weight(3,10)*entropy([3,0])

- weight(4,10)*entropy([1,3])

- weight(3,10)*entropy([1,2])

= 1.0 - 0.3*0.0 - 0.4*0.8113 - 0.3*0.9183 = 0.4000

B =

entropy([5,5])

- weight(4,10)*entropy([1,3])

- weight(4,10)*entropy([3,1])

- weight(2,10)*entropy([1,1])

= 1.0 - 0.4*0.8113 - 0.4*0.8113 - 0.2*1.0 = 0.1510

C =

entropy([5,5])

- weight(5,10)*entropy([1,4])

- weight(4,10)*entropy([3,1])

- weight(1,10)*entropy([1,0])

= 1.0 - 0.5*0.7219 - 0.4*0.8113 - 0.1*0.0 = 0.3145

A achieves the highest information gain.

Task 4

a.

The lowest entropy is 0 when every example has the same label.

The highest entropy is 2.0 when every label has the same number of examples.

b.

The lowest possible information gain is 0 when the entropy of node N is 0.

The highest possible information gain is 2.0 when each class directly corelates to an attribute.

Task 5

If the accuracy of the classifier is 28% and if there are only two options for a game(win or lose) you can negate the predicted class from the classifier and invert the accuracy. By choosing the opposite of the classifier you can get an accuracy of 72%