Paul Johnston

CS 478

Decision Tree

1. This Decision Tree learner uses the ID3 decision tree algorithm and is able to handle unknown data. It uses standard information gain as it’s basic attribute evaluation metric. It uses a node class with the capabilities of making a tree structure. Each node has children, data (features and labels), connecting attribute (meaning the attribute that connects it to its parent), connecting value, (instance of the connecting attribute possible values),remaining attributes, and depth. When training, this learner first computes the base info possible from just dividing the data into the possible output classes of the given node. It then calculates what the info gain would be if it split on each of the remaining attributes. It then creates children nodes based on the attribute that provides the best info gain (base info – attribute info). It then recursively splits on all the children nodes until there are either no more attributes to split on or the child nodes are pure, meaning all their data is of the same output class. In addition to this, the learner uses reduced error pruning to avoid overfitting to the training data. On the lenses data, it is able to predict with an average of 75 percent accuracy

2.

Cross validation results

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Cars dataset | | Voting dataset | |
|  | Training accuracy | Testing accuracy | Training accuracy | Testing accuracy |
| Fold 0 | 1.0 | 0.9651162790697675 | 1.0 | 0.9767441860465116 |
| Fold 1 | 1.0 | 0.9248554913294798 | 1.0 | 0.9545454545454546 |
| Fold 2 | 1.0 | 0.9364161849710982 | 1.0 | 0.8604651162790697 |
| Fold 3 | 1.0 | 0.930635838150289 | 1.0 | 0.9545454545454546 |
| Fold 4 | 1.0 | 0.9017341040462428 | 1.0 | 0.9302325581395349 |
| Fold 5 | 1.0 | 0.8953488372093024 | 1.0 | 0.9318181818181818 |
| Fold 6 | 1.0 | 0.9364161849710982 | 1.0 | 0.9069767441860465 |
| Fold 7 | 1.0 | 0.9190751445086706 | 1.0 | 0.9545454545454546 |
| Fold 8 | 1.0 | 0.9421965317919075 | 1.0 | 0.9302325581395349 |
| Fold 9 | 1.0 | 0.9248554913294798 | 1.0 | 0.9318181818181818 |
| Average |  | 0.9276650087377336 |  | 0.9331923890063425 |

The table above shows this decision tree’s accuracies for 10-fold cross validation. The learner got 100% accuracy for each dataset when training, this is because it would basically memorize the training data, splitting the tree until the nodes are pure. This means that even when the data is shuffled, the tree will still send the same instance down the same path of the tree, resulting in the same output class. I observed from the table above that the fold accuracies were consistent except for fold 5 for the cars dataset. It seemed like a fluke, as it was very close to 90 percent. The other outlier was fold 2 for the voting dataset which I think happened because when it trained on this fold, it happened to grab a set of training data that was more clumped together from the testing set. It trained a lot on the clumped training set and then when it saw novel data that didn’t follow the same clumping as the training, it just had to take the best output possibility from the parent node (because there were no children nodes with the attribute value as the novel instance).

3.

For the cars dataset, the decision tree learned that the best attribute to split on at the beginning was attribute 5 (zero indexed) which was safety. It then learned that for cars with high and medium safety, the best attribute to split on was persons. For cars with low safety, it didn’t have to learn anymore because all cars with low safety in the training set were unacc. The decision tree continued recursing like this on the nodes, for high safety cars which fit 2 persons, it didn’t have to learn anymore because all cars within these parameters were unacc as well. One of the more shallow, uninformative features was the lug boot attribute.

For the vote dataset, the decision tree learned that the best attribute to split on at the beginning was the “physician fee freeze” question. For people who said no to this question, the next most informative question was “adoption of the budget”. For people who said yes to the “physician fee freeze” question, the next most informative question was “synfuels corporation cutback”. For people who didn’t respond the “physician fee freeze” question, the next most informative question was “mx missile”. One of the shallowest, least informative question was the education spending question and the water project cost sharing question.

4.

This decision tree learner handles unknown attributes by creating a new category of attribute values. This allows the tree to, instead of just ignoring these values, infer knowledge from the missing value. For example, in the voting set, not voting for a certain issue may mean that the person who chose not to vote was leaning one way politically. I chose this approach because it allowed the learner to retain information from the unknown values and the numpy unique function allowed for this implementation very nicely.

5.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Cars dataset | | | |
|  | Nodes before Prune | Nodes after Prune | Test accuracy before | Testing accuracy |
| Test 1 | 1.0 | 0.9651162790697675 | 1.0 | 0.9767441860465116 |
| Test 1 | 1.0 | 0.9248554913294798 | 1.0 | 0.9545454545454546 |
| Test 2 | 1.0 | 0.9364161849710982 | 1.0 | 0.8604651162790697 |
| Test 3 | 1.0 | 0.930635838150289 | 1.0 | 0.9545454545454546 |
| Test 4 | 1.0 | 0.9017341040462428 | 1.0 | 0.9302325581395349 |
| Test 5 | 1.0 | 0.8953488372093024 | 1.0 | 0.9318181818181818 |
| Test 6 | 1.0 | 0.9364161849710982 | 1.0 | 0.9069767441860465 |
| Test 7 | 1.0 | 0.9190751445086706 | 1.0 | 0.9545454545454546 |
| Test 8 | 1.0 | 0.9421965317919075 | 1.0 | 0.9302325581395349 |
| Test 9 | 1.0 | 0.9248554913294798 | 1.0 | 0.9318181818181818 |
| Average |  | 0.9276650087377336 |  | 0.9331923890063425 |