Introduction to Machine Learning Project 3: Decision Trees
By: Jonathan Klinger

Abstract:
I used the decision tree classifier and decision tree regressor on the image segmentation, abalone and cars datasets and the machines, forest fires, red wine and white wine datasets respectively. For the image segmentation dataset, I was able to acheive 87% accuracy without pruning. For the abalone dataset, I acheived 23% accuracy with pruning. For the cars dataset, I reached 70% accuracy with pruning. Pruning proved to be helpful 2/3 times, but for the image segmentation dataset it dramatically decreased results (10%). For the machines dataset, I identified the optimal MSE early stopping value to be 1000 and acheived 1847 MSE and a 0.88 R-Squared value for my training set. Forest fires and the wine datasets proved to be more challenging regression problems. For forest fires, I identified the optimal MSE early stopping value to be 4000 and acheived 2306 MSE and a -0.01 R-Squared value for my training set. For the red wine dataset, I identified the optimal MSE early stopping value to be 0.4 and acheived 0.5 MSE and a 0.24 R-Squared value for my training set. For the white wine dataset, I identified the optimal MSE early stopping value to be 0.4 and acheived 0.62 MSE and a 0.26 R-Squared value for my training set.

Problem statement, including hypothesis, projecting how you expect each algorithm to perform:
The task at hand is to use a decision tree classifier and decision tree regressor to run predictions on six datasets. My hypotheses going into this project were: 1) the decision tree classifier will not perform much better if we use gain ratio, because after reviewing the data I did not find any categorical attributes with many values; 2) the decision tree classifier will improve if we prune the tree using a validation set; 3) picking the optimal MSE for a stopping criteria for each decision tree regressor will be extremely important for yielding optimal results. Without this, I strongly suspect that the tree will overfit the training data; and 4) with regards to performance of a decision tree vs. other models, I suspect that there are some models more well suited to acheiving better results, but the fact that a decision tree is a universal function approximator will play a huge role in acheiving results close to the results of those other models in many cases.

Implementation of algorithms
The algorithms implemented were a decision tree classifier and a decision tree regressor. Both decision tree are built by using a Node class for each node or leaf. If there is a value for a node, it is identified as a leaf, otherwise the node will have splitting criterium and two children nodes. This represents the hiearchy of a tree, where the leaves have the predicted values and the nodes have splitting criterium to traverse to one of them.

The decision tree classifier is implemented as such: at each splitting point of the tree, we identify the best feature for splitting by picking the feature that has the highest gain. The gain is calculated by computing the entropy of the split and subtracting it from the current root of the subtree's entropy. To make sure that features with a lot of unique values are not always chosen in extreme cases, I divide the gain by the information value of the feature to compute the gain ratio. The max of the gain ratios is then taken. The identification of a leaf node is defined in the stop function, where it checks if the current depth has been reached, the min sample split has been reached, or if the number of labels in the current subset is equal to one. If these stopping criterium have not been met, we continue to split and grow the tree.

The decision tree classifier also allows for pruning. When pruning is enabled, we split out a validation set (10% of the data) and use the rest for 5-fold cross validation. Both the validation sets and the cross validation set have an equal distribution of classes. Once the tree is built,

the pruning function traverses the tree and sets a node to a leaf. If the accuracy increases, the node remains a leaf, otherwise we return it to its original state. It is worth noting that if the tree is turned into a leaf, the leaf value is determined by taking the most common label of the subset of the data.

The decision tree regressor is implemented as such: at each splitting point of the tree we identify the best feature for splitting by picking the feature that has the lowest mean squared error. The mean squared error is computing by calculating the squared error of each side of the split (using the mean as the prediction value) and taking the mean of the sum of those squared errors. Similar to the classifier, we have stopping criterium for the decision tree regressor to prevent overfitting. That is, we return a leaf node if the max depth has been reached or if the splitting results in an MSE that meets our optimal value determined using a validation set.

Experimental Approach/Behavior of Algorithms

I started testing my decision tree classifier on my datasets using gain, followed by testing with the addition of weighting the gains by information value. As I expected the results were roughly the same using both approaches. I attributed this to the fact that there were no categorical attributes in these dataset that possesed many unique values. The one exception was the cars dataset, which I found interesting because there weren't too many categories for each feature. That being said, it was the dataset that had the most features for certain categorical attributes, which is likely why it showed more improvent.

Pruning my classification decision trees lead to increased performance on the cars and abalone datasets but dramatically decreased my results on the image segmentation dataset. I thought this decrease in performance was odd, so I did some digging to try and find out why. In cost-sensitive decision tree pruning, Cai, Durkin and Cai discuss an alternative method to decision tree pruning, stating that "one main problem for many traditional decision tree pruning methods is that when we prune a decision tree, we always assume that all misclassifications are equally probable and equally serious. However, in a real world classification problem, there is also a cost associated with misclassifying examples from each class." This made sense to me and was roughly in line with my own thoughts on the issue.

For the decision tree regression datasets, I started out by computing the mean value of the subset labels to compute the value of the leaf, however I found that the data was noisy and computing the median for the leaf value acheived better results. After seeing the minimal change in results, I wondered if there was a better alternative to just taking the average or the median of the subset of labels. In Functional Models for Regression Tree Leaves, Torgo discusses the limitations of decision tree regression models in the context of its approximation of the leaf values. One of his proposed approaches was to compute the prediction by running a linear regression on the subset of samples in the lead node. Overall, his conclusion was that hybrid models could potentially increase a models performance.

To identify the optimal stopping criteria for each regression, I used a validation set to compute the performance on a grown tree with some value for the mean squared error. I found that using the resulting mean squared error from my original results (not optimizing MSE), served as a good starting point for identifying the optimal value. Identifying the optimal mean squared error improved my results. With no optimal value (0), the model overfit the training data and decreased performance significantly.

Results on 5-fold cross validation

Decision Tree Classifier

Dataset	Accuracy (without pruning)	Accuracy (with pruning)
Image Segmentation	87%	80%
Abalone	20%	23%
Cars	69%	70%

Decision Tree Regressor

Dataset	MSE stopping criteria	Mean Squared Error	R-Squared
Machines	1000	1847	0.88
Forest Fires	4000	2306	-0.01
Red Wine	0.4	0.5	0.24
White Wine	0.4	0.62	0.26

Summary and Conclusions

Overall, our results were not too exciting using decision tree classification and regression, aside from the performance on the machine dataset. However, most of these datasets are notoriously difficult to predict, and the decision tree classifier performed roughly on par with others. The real benefit to decision trees for me, was the fact that they are universal function approximators, which made it much easier to run the same model on a different dataset without much tuning in hyperparamaters. For classification, when it comes to pruning, we must take care with the way that we prune. Testing the resulting pruned tree on a variety of pruning methods is crucial to getting the best results. If possible, adding the cost of a misclassification to a pruning method could be useful. For regression, it is important that we set a stopping criteria so as not to overfit the data. Traversing down the tree, lowers the overall Mean Squared Error to 0 eventually (because the mean squared error of 1 value vs the average is 0), so we must stop the growth of the tree at an ideal mean squared error, which is identified using a validation set testing on many values for the optimal mean squared error. Additionally, the approximation for the leaf value has its flaws for regression trees, in the real world I would definitely consider the hybrid approach mentioned by Torgo.

Sources:

Cost-Sensitive Decision Tree Pruning by J. Cai, J. Durkin & Q. Cai
Functional Models for Regression Tree Leaves by Luís Torgo