Abstract

A decision tree is a decision support tool that uses a tree-like model of decisions and their possible consequences, including chance event outcomes, resource costs, and utility. It is one way to display an algorithm that only contains conditional control statements. Decision trees are commonly used in operations research, specifically in decision analysis, to help identify a strategy most likely to reach a goal, but are also a popular tool in machine learning.

Decision trees play a significant role in data mining when it comes to classification problems. Alongside decision rules it is the method for deducing classification models to apply to unclassified data. Decision trees for classification have the purpose to classify instances correctly based on attribute values. However, this is nearly never the case since to be completely correct the tree would have to be trained with all possible instances. There are several techniques and methods available to make the decision tree's performance as high as possible. This paper will discuss one of them called grafting.

During learning, when considering potential new partitions within an existing partition p, only training examples within p are considered. Grafting is an algorithm for adding nodes to the tree as a post-process. Its purpose is to increase the probability of correctly classifying instances that fall outside the areas covered by the training data or where there is misclassification data. The opportunity to identify useful additional branches arises due to the top-down divide-and-conquer strategies used in conventional decision tree induction algorithms. It finds the best suited cuts to create new leaves with other classifications than the original. One thing that should be kept in mind is that only branching that does not introduce any classification errors in data already rightly classified is considered. This ensures that the new tree reduces errors instead of introducing them. The main algorithms implementing this idea are C4.5x and C4.5+, the difference between these two algorithms is that test were made to prove that a new cut was good and did not decrease the prediction accuracy and also the ability for multiple cuts, new leaves reclassifying regions and allowing global information to be used when local information does not exist, differs between the two algorithms.

One more different method is called decision tree pruning, for post-processing decision trees, which removes nodes from an inferred decision

tree. Pruning uses only local information and reduces the number of partitions imposed on an instance space by a decision tree. The main difference between pruning and grafting is that, pruning aims at reducing the complexity of the decision tree and still has good prediction accuracy while grafting does this by adding complexity to the tree, and that's why they are complementary to each other. Webb (1997) concludes that pruning and grafting despite being opposites, or because of that, works well in parallel. The grafting algorithm takes into account instances outside the analyzed leaf (global information) while pruning only looks at instances within the analyzed leaf (local information). The primary focus of Webb's [1996] grafting research was to examine the effect of complexity on prediction accuracy. There were some key changes made in the approach of C4.5x for better performance these were: allowing grafting to alter resubstitution performance, the ordered addition of multiple new branches in the place of a single original leaf; the use of a significance test to restrict the selection of new branches; and allowing grafting within leaves occupied by no training examples.

Technical review of paper and it's pros and cons with further research.

A classifier can be viewed as partitioning an instance space. Each partition associates a set of possible objects with a class. A number of recent studies have suggested that predictive accuracy may also be improved by more complex partitioning of an instance space than that formed by standard decision tree induction. Predictive accuracy has been improved both by (i) grafting additional leaves and (ii) developing multiple classifiers that are used in conjunction to classify objects. The reason for increase in accuracy due to complex partitioning of the instance space is because the conventional machine learning technique considers only that area of the instance space which is occupied by the training example and the area which is vacant or unoccupied is assigned to partitions as a side-effect of partitioning occupied area. This occurs without consideration of the available evidence relating to appropriate partitioning of these regions. Explicit examination of such areas may provide evidence as to the most likely class for previously unseen objects that fall therein. If there is such evidence and the

appropriate classification differs from that currently assigned to the region, a new partition can be formed. This is achieved by grafting a new leaf onto the tree.

The primary focus of Webb's grafting research was to examine the effect of complexity on predictive accuracy, but in general we used to discard the complex algorithms and techniques, but C4.5x and C4.5+ prove that a more complex decision tree should not always be discarded.

Talking about the C4.5x algorithm, this algorithm tries to find areas without any training data that the C4.5 algorithm has given a class that might not be the best one from a similarity point of view. When such an area is found it adds branches to the tree to split that area's class space into smaller partitions. It will not do any branching or splitting that worsens the performance in classifying the existing training data.

The algorithm works as:

1. For each leaf l in the tree each attribute a is considered.

2. For each a all possible thresholds below and above the region represented in leaf l are explored.

3. A maximum and minimum value are computed such that only instances with min<value<max for attribute a are considered. In other words only instances with a value for a that can reach leaf l is considered.

4. For each value observed that fall within the range of leaf l but outside the range of the actual instances' values in l, a support is calculated for reclassifying the region above/below that threshold.

5. The support is calculated using binary Laplacian accuracy estimate (P +1 /T+2) where P is the number of instances that belong to the certain class and T is the number of instances at the ancestor level for which min<value<threshold. And then the same is made for threshold<value<max.

6. The single threshold value for the attribute a that has the highest estimate is matched against the evidence for the original classification of the leaf region. If it is higher than a new branch is added that creates a new leaf with a region that was previously a part of the region of l, but is now separated and can be classified as another more likely class even though no training data is present there.

An example to illustrate the algorithm is as follows. An instance space has been divided by the C4.5 algorithm into different classifications with respect to two attributes A and B, as in figure 1. The blue area corresponds to a leaf and there is an uncertainty to which class the question mark seen in the figure belong. The area will be considered to be cut in several pieces by the C4.5+ grafting algorithm to produce less prediction errors. The evidence in support for the majority class and a cut at B=3 is calculated with respect to the ancestor node (B<=5) and is equal to (9+1)/(9+2) = 0.909. This is the highest value and so B=3 is the best cut between 1<=B<=5. This value is stored in a list together with the best cuts for each attribute. The list is later used to create new branches and leaves between the leaf and its parents. Figure 2 illustrates the result after grafting the tree from in figure 1. Three new leaves a, b and c in the figure now belongs to the classes ◊, * and ◊ since the support for those classes were the highest in those new regions. Now it is possible to classify the question mark with a small prediction error.

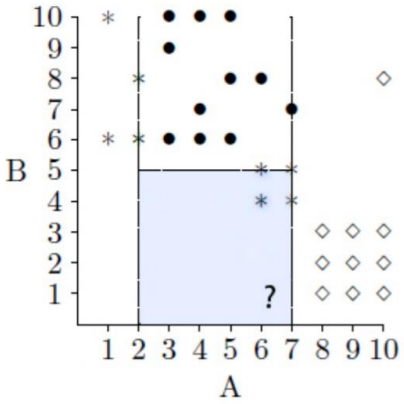

Figure 1

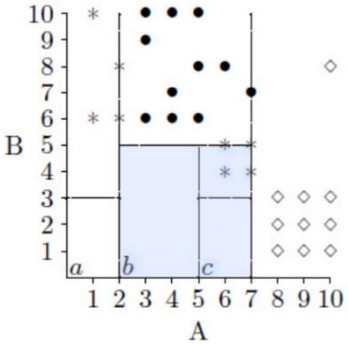

Figure 2

As there are some more techniques like boosting and pruning which serves the same purpose as grafting does with decision tree, but the main advantage of it over other methods is that it is computationally less expensive and provides less prediction error and more accuracy.

The main drawback of this technique is that it may not be able to do the required classification with good accuracy when the data is not linearly separable, because since it is using cuts to perform the grafting, and when the data is dense and large number of classes are required to classify, it may not be able to perform the desired work due to overfitting.

For improving the technique and getting better results, one can do some pre-processing on the data like removal of redundant data, which prevents model and the post-processing from overfitting. One more thing one can do which is a kind of feature reduction, i.e. making the decision tree based on important features and discarding the redundant feature.

For further research one can go for classifying the instance space using some non-linear function instead of linear cuts.

Further suggested work and advancement.

Now as this paper talks about the improvement in predictive accuracy in decision trees, many recent studies and experiments have given the idea that I can be increased by more complex partitioning of an instance space than that formed by standard decision tree induction. In this paper we have seen that the predictive accuracy can be increased by:-

· Grafting additional leaves in the inferred decision tree.

· By developing multiple classifiers that are used in conjunction to classify objects.

So in order to have some technical updates/ suggestion, to increase the predictive accuracy here are some of them:

As we have seen that the idea or technique proposed in the given paper is to reclassify the instance space after decision tree is formed using cuts, now the question arises that what is the possibility that we are able to reclassify the instance space using linear cuts, it might be possible that the data is not evenly spread and not able to be reclassified using linear cuts. So we need some non-linear function to classify these instance spaces to do the required post-processing (i.e. grafting).

Another thing is that, as the algorithm is itself computationally complex and time taking, and doing this computation (i.e. post-processing) on a large data might be more expensive and time taking, so the idea is that we can reduce the dimension and remove the redundant features and considering only important features for deducing the decision tree, because in applications where classifier complexity is

a significant factor, this trade-off deserves careful consideration so that when post-processing(i.e. grafting) is applied it would be less tedious.

Now addressing a problem encountered by grafting systems when they are applied to some datasets containing missing values. For datasets with many missing values, this can result in a large increase in predictive error. A simple solution to this problem is to prevent grafting on attributes for which there are missing values at the current node. This measure is shown to prevent the extreme increase in the predictive error previously identified without general negative side-effects. It also offers the additional benefit of substantially reducing the size of the inferred trees.

One aspect which is missing in the paper and can be an area for future development and research is to develop a grafting technique which can graft new nodes for discrete attributes. One thing also that as grafting adds new nodes to increase predictive accuracy of decision trees, this accuracy can also be increased with less error than grafting alone, by using pruning and grafting together. It is possible that this is due to the ability of pruning to identify partitions of the instance space where the local information is insufficient to create sensible sub-partitions. Grafting can then use non-local information to generate appropriate sub-partitions. Pruning then grafting outperforms either pruning or grafting alone. Grafting then pruning is inappropriate as pruning will remove all nodes added by grafting as these new nodes will have little support in terms of the local evidence that pruning can consider.

As the first job before grafting is that to distinguish the instance space which is occupied by training samples and which are not occupied by training samples, so defining a sharp boundary over the instance space where the training data is present, this give a clear picture where there is instance space with training data and where instance space is without training data, so that when grafting is applied it is able to classify at its full potential.

So these were some of my technical updates/ suggestions on this paper based on my understanding, I hope this might improve the paper.