BHARATHIDASAN UNIVERSITY

Tiruchirappalli- 620024

Tamil Nadu, India.

Programme: M.Sc. Statistics

Course Title: Introduction to Machine learning

Course Code: 23ST01VAC

Unit-V

Regression Trees

Dr. T. Jai Sankar
Associate Professor and Head
Department of Statistics

Ms. E. Devi
Guest Faculty
Department of Statistics

UNIT-V

Regression Trees

Definition:

A regression tree is basically a decision tree that is used for the task of regression which can be used to predict continuous valued outputs instead of discrete outputs.

Basic regression trees partition a data set into smaller subgroups and then fit a simple constant for each observation in the subgroup. The partitioning is achieved by successive binary based on the different predictors. The constant to predict is based on the average response values for all observations that fall in that subgroup. There are many methodologies for constructing regression trees but one of the oldest is known as the classification and regression tree (CART) approach developed by Breiman et al. (1984).

Strengths and weaknesses

There are several advantages to regression trees:

- They are very interpretable.
- Making predictions is fast.
- It’s easy to understand what variables are important in making the prediction. The internal nodes (splits) are those variables that most largely reduced the SSE.
- If some data is missing, we might not be able to go all the way down the tree to a leaf, but we can still make a prediction by averaging all the leaves in the sub-tree we do reach.
- The model provides a non-linear “jagged” response, so it can work when the true regression surface is not smooth. If it is smooth, though, the piecewise-constant surface can approximate it arbitrarily closely (with enough leaves).
- There are fast, reliable algorithms to learn these trees.

But there are also some significant weaknesses:

- Single regression trees have high variance, resulting in unstable predictions.
- Due to the high variance single regression trees have poor predictive accuracy

Tree-based regression

Definition:

Tree-based regression models are known for their simplicity and efficiency when dealing with domains with large number of variables and cases. Regression trees are obtained using a fast divide and conquer greedy algorithm that recursively partitions the given training data into smaller subsets.

This describes two different approaches to induce regression trees. We first present the standard methodology based on the minimization of the squared error. Least squares (LS) regression trees had already been described in detail in the book by Breiman et. al. (1984). Compared to this work we present some simplifications of the splitting criterion that lead to gains in computational efficiency.

We then address the alternative method of using a least absolute deviation (LAD) error criterion to obtain regression trees. Although mentioned in the book of Breiman and colleagues (1984), this methodology was never described in sufficient detail. The LAD criterion is known to be more robust to skewed distributions and outliers than the LS criterion used in standard regression trees.

However, the use of the LAD criterion brings additional computational difficulties to the task of growing a tree. In this we present algorithms based on a theoretical study of the LAD criterion that overcome these difficulties for numeric variables.

With respect to nominal variables we show that the theorem proved by Breiman et. al. (1984) for subset splits in LS trees does not hold for the LAD error criterion. Still, we have experimentally observed that the use of the results of this theorem as a heuristic method of obtaining the best split does not degrade predictive accuracy.

A regression tree can be seen as a kind of additive model

$$\boxed{m(x)= \sum_{i=1}^{I} K_i \times I(x \in D_i)}$$

of the form, where,

K_i is constants; I (.) is an indicator function returning 1 if its argument is true and 0 otherwise; and D_i are disjoint partitions of the training data D such that $\bigcup_{i=1}^{I} D_i$ = D and

$$\bigcap_{i=1}^{I} D\, i = \varphi$$

The algorithm has three main components

- A way to select a split test (the splitting rule).
- A rule to determine when a tree node is terminal (termination criterion).
- A rule for assigning a value to each terminal node. In the following sections we present two different approaches to solve these problems. These alternatives try to minimize either the mean squared error or the mean absolute deviation of the resulting tree.

LEAST SQUARES REGRESSION TREES

Least Squares Regression Trees The most common method for building a regression model based on a sample of an unknown regression surface consists of trying to obtain the model parameters that minimize the least squares error criterion,

$$\frac{1}{n}\sum_{i}^{n} (\mathrm{y_i} - \mathrm{r}(\beta, \mathrm{x_i}))^2$$

where,

n is the sample size;

< x_i, y_i > is a data point ;

and r(β, xi) is the prediction of the regression model r(β, x) for the case < x_i, y_i >

Greedy Algorithms (Greedy Criterion)

The greedy algorithm is a step-by-step method of problem solving which makes the locally optimal choice at each stage. Algorithm 1 outlines a general greedy procedure for the problem of maximizing a set function.

At each iteration, the element with the individual largest contribution is considered as the best available addition to the solution set. The algorithm halts when adding the best available element would fail to increase f. Greedy algorithms have shown great potential in applications such as sparse approximation (Tropp, 2004), signal compression (Tropp et al., 2006), and pattern recognition (Nair et al., 2003). At the same time, it has been shown that they may be applied in a variety of ways to significantly cut computational and memory costs while developing models which are much more accurate.

The greedy pursuit algorithm may be applied to design a sequential procedure of incrementally selecting terms from the dictionary, A and moving them to the solution matrix K.

Resubstitution

Definition:

If all the data is used for training the model and the error rate is evaluated based on outcome vs. actual value from the same training data set, this error is called the resubstitution error. This technique is called the resubstitution validation technique.

The Role of Pruning in Decision Trees

- Pruning is one of the techniques that is used to overcome our problem of Over fitting. Pruning, in its literal sense, is a practice which involves the selective removal of certain parts of a tree(or plant), such as branches, buds, or roots, to improve the tree’s structure, and promote healthy growth. This is exactly what Pruning does to our Decision Trees as well.
- It makes it versatile so that it can adapt if we feed any new kind of data to it, thereby fixing the problem of over fitting. It reduces the size of a Decision Tree which might slightly increase your training error but drastically decrease your testing error, hence making it more adaptable.

Avoiding Over fitting the Data

When we are designing a machine learning model, a model is said to be a good machine learning model, if it generalizes any new input data from the problem domain in a proper way. This helps us to make predictions in the future data, that data model has never seen. Now, suppose we want to check how well our machine learning model learns and generalizes to the new data. For that we have over fitting and under fitting, which are majorly responsible for the poor performances of the machine learning algorithms.

Under fitting

A machine learning algorithm is said to have under fitting when it cannot capture the underlying trend of the data. Under fitting destroys the accuracy of our machine learning model. Its occurrence simply means that our model or the algorithm does not fit the data well enough.

- It usually happens when we have less data to build an accurate model and also when we try to build a linear model with a non-linear data. In such cases the rules of the machine learning model are too easy and flexible to be applied on such a minimal data and therefore the model will probably make a lot of wrong predictions. Under fitting can be avoided by using more data and also reducing the features by feature selection

Over fitting Pruning:

- Over fitting refers to the condition when the model completely fits the training data but fails to generalize the testing unseen data. Over fit condition arises when the model memorizes the noise of the training data and fails to capture important patterns. A perfectly fit decision tree performs well for training data but performs poorly for unseen test data.

There are two types of pruning.,

(i) Pre-pruning

(ii) Post-pruning

Pre-Pruning:

- The pre-pruning technique refers to the early stopping of the growth of the decision tree. The pre-pruning technique involves tuning the hyper parameters of the decision tree model prior to the training pipeline. The hyper parameters of the decision tree including max depth, min samples leaf, min samples split can be tuned to early stop the growth of the tree and prevent the model from over fitting.

Post-Pruning:

- The Post-pruning technique allows the decision tree model to grow to its full depth, then removes the tree branches to prevent the model from over fitting. Cost complexity pruning (ccp) is one type of post-pruning technique. In case of cost complexity pruning, the ccp alpha can be tuned to get the best fit model.

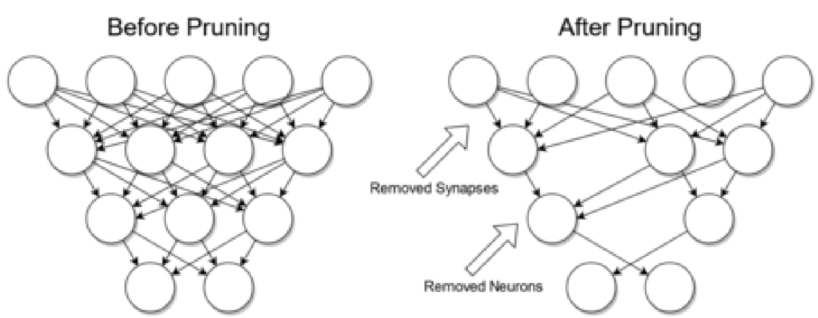
Sub tree replacement and Sub tree raising:

- The two methods for that are *sub tree replacement* and *sub tree raising*. At each node, an algorithm decides whether it should perform sub tree replacement or sub tree raising.
- Sub tree replacement selects a sub tree and replaces it with a single leaf.
- Sub tree raising selects a sub tree and replaces it with the child one
- (ie, a "sub-sub tree" replaces its parent)

Pruning Algorithm:

Definition:

Pruning is a data compression technique in machine learning and search algorithms that reduces the size of decision trees by removing sections of the tree that are non-critical and redundant to classify instances. Pruning reduces the complexity of the final classifier, and hence improves predictive accuracy by the reduction of over fitting.



Criterion used to determine the correct final tree size

- Use a separate set of examples, distinct from the training examples, to evaluate the utility of post-pruning nodes from the tree
- Use all the available data for training, but apply a statistical test to estimate whether expanding (or pruning) a particular node is likely to produce an improvement beyond the training set
- Use measure of the complexity for encoding the training examples and the decision tree, halting growth of the tree when this encoding size is minimized. This approach is called the Minimum Description Length

- MDL — Minimize : size(tree) + size (misclassifications(tree))

  The first of the above approaches is the most common and is often referred to as training and validation set approach.

We discuss the two main variants of this approach below:

In this approach, the available data are separated into two sets of examples: a training set, which is used to form the learned hypothesis, and a separate validation set, which is used to evaluate the accuracy of this hypothesis over subsequent data and in particular, to evaluate the impact of pruning this hypothesis.

1. Reduced Error Pruning

Reduced Error Pruning is an algorithm that has been used as a representative technique in attempts to explain the problems of decision tree learning.

- Reduced-error pruning, is to consider each of the decision nodes in the tree to be candidates for pruning.
- Pruning a decision node consists of removing the sub tree rooted at that node, making it a leaf node, and assigning it the most common classification of the training examples affiliated with that node.
- Nodes are removed only if the resulting pruned tree performs no worse than-the original over the validation set.
- Reduced error pruning has the effect that any leaf node added due to coincidental regularities in the training set is likely to be pruned because these same coincidences are unlikely to occur in the validation set.

2. Rule Post-Pruning

Rule post-pruning involves the following steps:

- Infer the decision tree from the training set, growing the tree until the training data is fit as well as possible and allowing over fitting to occur.
- Convert the learned tree into an equivalent set of rules by creating one rule for each path from the root node to a leaf node.

- Prune (generalize) each rule by removing any preconditions that result in improving its estimated accuracy.
- Sort the pruned rules by their estimated accuracy, and consider them in this sequence when classifying subsequent instances.

There are three main advantages by converting the decision tree to rules before pruning:

- Converting to rules allows distinguishing among the different contexts in which a decision node is used. Because each distinct path through the decision tree node produces a distinct rule, the pruning decision regarding that attribute test can be made differently for each path.
- Converting to rules removes the distinction between attribute tests that occur near the root of the tree and those that occur near the leaves. Thus, it avoid messy bookkeeping issues such as how to reorganize the tree if the root node is pruned while retaining part of the sub tree below this test.
- Converting to rules improves readability. Rules are often easier for to understand.

Cost complexity criterion

- There is often a balance to be achieved in the depth and complexity of the tree to optimize predictive performance on some unseen data. To find this balance, we typically grow a very large tree as defined in the previous section and then prune it back to find an optimal sub tree.
- We find the optimal sub tree by using a cost complexity parameter ($\alpha\alpha$) that penalizes our objective function for the number of terminal nodes of the tree (T) as in below Equation. For a given value of α, we find the smallest pruned tree that has the lowest penalized error. If you are familiar with regularized regression, you will realize the close association to the lasso L1 norm penalty. As with these regularization methods, smaller penalties tend to produce more complex models, which result in larger trees.
- Whereas larger penalties result in much smaller trees. Consequently, as a tree grows larger, the reduction in the SSE must be greater than the cost complexity penalty. Typically, we evaluate multiple models across a spectrum of α and use cross-validation to identify the optimal α and, therefore, the optimal sub tree.

Minimum error

- Minimum error entropy (MEE) is an information theoretic learning approach that minimizes the information contained in the prediction error, which is measured by entropy. It has been successfully used in various machine learning tasks for its robustness to heavy-tailed distributions and outliers.

Pessimistic

- The pessimism bias refers to the tendency to overestimate the likelihood of negative events while underestimating the likelihood of positive events.

Critical value

- Critical value or P Value can be defined as a value that is compared to a test statistic in hypothesis testing to determine whether the null hypothesis is to be rejected or not. If the value of the test statistic is less extreme than the critical value, then the null hypothesis cannot be rejected.
- However, if the test statistic is more extreme than the critical value, the null hypothesis is rejected and the alternative hypothesis is accepted. In other words, the critical value divides the distribution graph into the acceptance and the rejection region. If the value of the test statistic falls in the rejection region, then the null hypothesis is rejected otherwise it cannot be rejected.

*****