Chapter 9: Additive Models and Trees
Section 9.2 Tree Based Models

Clint P. George

Department of Computer and Information Science and Engineering
University of Florida

Elements of Statistical Learning

Mar 15, 2010

Overview

- It is a method based on binary recursive partitioning of the feature space into regions and fitting a tree model
- Some characteristics
  - No distribution assumptions on the variables

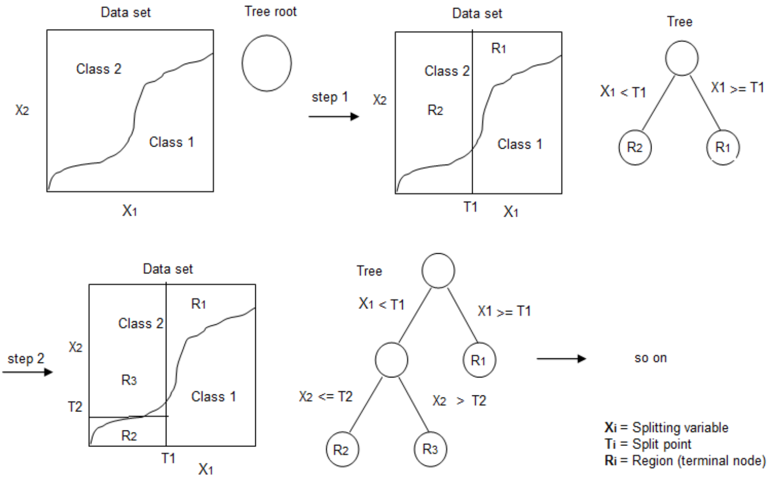
Overview

- It is a method based on binary recursive partitioning of the feature space into regions and fitting a tree model
- Some characteristics
  - No distribution assumptions on the variables
  - The feature space can be fully represented by a single tree

Overview

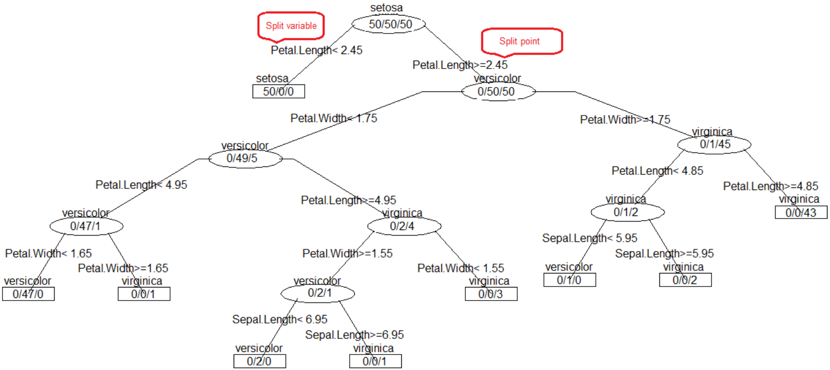
- It is a method based on binary recursive partitioning of the feature space into regions and fitting a tree model
- Some characteristics
  - No distribution assumptions on the variables
  - The feature space can be fully represented by a single tree
  - Interpretable

Overview


Data set
Tree root
Class 2
Class 1
X2
X1
step 1
Data set
R1
Class 2
R2
Class 1
X2
T1
X1
Tree
X1 < T1
X1 >= T1
R2
R1
step 2
Data set
R1
Class 2
R3
Class 1
R2
X2
T2
T1
X1
Tree
X1 < T1
X1 >= T1
R1
X2 <= T2
X2 > T2
R2
R3
so on
Xi = Splitting variable
Ti = Split point
Ri = Region (terminal node)

Example: Classification Tree

- Data set: IRIS - four features, 150 samples, and three classes
- Software: *RPART* (Terry M. Therneau, 1997) package in R



Observations

Node	Split	N_m	loss	prediction	proportions
1	root	150	100	setosa	(0.33 0.33 0.33)
2	$PL < 2.45$	50	0	setosa	(1.00 0.00 0.00)*
3	$PL \geq 2.45$	100	50	versicolor	(0.00 0.50 0.50)
6	$PW < 1.75$	54	5	versicolor	(0.00 0.90 0.09)
-	-	-	-	-	-

Table: Tree split path and node proportions

- The variables actually used in tree construction: Petal.Length, Petal.Width and Sepal.Length
- The tree partitions the sample space into nine regions

Overview

Basic idea of any tree building :

- Grow a large and complicated tree that explains the data
  - decide the splitting variables (predictors) and split points
  - binary recursive partitioning
  - stopping criterion e.g. max number of samples in a leaf node
- Prune the tree to avoid over fitting

References: Bishop (2007); Pham (2006); T. Hastie (2009)

Classification Tree Setup

Suppose, we have a K class classification problem with data $\{x_i, y_i\}_{i=1}^{N}$ where

- $y_i \, \epsilon \, \{1, 2, ...K\}$
- $x_i = (x_{i1}, x_{i2}, ...x_{id})$, d = dimensionality

then the *node proportion* of a class k at node m:

$$\hat{p_{mk}} = \frac{1}{N_m} \sum_{i=1}^{N_m} I(y_i \ = \ k), k = 1, 2, ...K, \tag{1}$$

where N_m = number of observations in a tree node m and $I(A)$ is an indicator function

Classification Rule

Classify the observations in node m to the *majority class*

$$k(m) = argmax_k[\hat{p_{mk}}]$$

A more general rule is to assign node m to the class

$$k(m) = argmin_k[r_{mk}]$$

where r_{mk} is the *expected misclassification cost* for class k

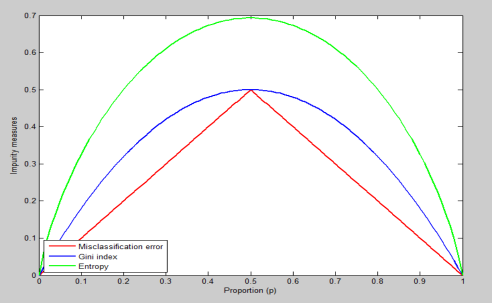
Suppose π_{mj} is the probability of node m as class j and $c(k|j)$ = the cost of classifying a class j sample as class k sample, then

$$r_{mk} = \sum_j c(k|j)\pi_{mj}$$

Impurity Functions

- Our aim is to reduce the node misclassification cost i.e. make all the samples in a node belongs to one class
- This can be seen as reducing the *node impurity*
- Popular functions to measure degree of impurity:
  - Misclassification error = $\frac{1}{N_m}\sum_{i \, \epsilon \, R_m} I(y_i \neq k(m)) = 1 - max\{p_{mk}\}$
  - Gini index = $\sum_{k=1}^{K} p_{mk}(1 - p_{mk})$
  - Cross-entropy (deviance) = $-\sum_{k=1}^{K} p_{mk} log(p_{mk})$

Comparison of Impurity Functions (2 class problem)



1. Encourages the formation of regions in which high proportion of data points assigns to one class
2. Gini index and cross entropy are differentiable and more sensitive to change in node proportions (p_{mk})

Tree Splitting Algorithm

- Assume a predictor x_j
- Let m_{left} and m_{right} be the left and right branches by splitting node m based on x_j
  - when x_j is *continuous* or *ordinal*, m_{left} and m_{right} are given by $x_j < s$ and $x_j \geq s$ for a splitting point s
  - when x_j is categorical we may need exhaustive subset search to find s
- And q_{left} and q_{right} be the proportion of samples in node m assigned into m_{left} and m_{right}

Tree Splitting Algorithm

- For each x_j, find the split by maximizing the decrease in

$$\Delta i_j(s, m) = i(m) - [q_{left} i(m_{left}) + q_{right} i(m_{right})]$$

where

$$i(m) = \sum_{k=1}^{K} \hat{p_{mk}}(1 - \hat{p_{mk}}) = 1 - \sum_{k=1}^{K} [\hat{p_{mk}}^2]$$

Greedy Algorithm

- Scan through all predictors (x_j) to find the best pair (j, s) with *largest decrease* in $\Delta i_j(s, m)$
- Then repeat this splitting procedure recursively for m_{left} and m_{right}
- Define a stopping criteria:
  - Stop when some minimum node size (N_m) is reached
  - Split only when the decrease in cost reaches a threshold
- Tree size:
  - A very large tree may over fit the data
  - A small tree may not structure the important data structure

Cost Complexity Pruning

Focus is to balance *misclassification error* against a measure of *complexity*

- Suppose, we got a large tree T_0 by using the greedy algorithm, m indexes terminal nodes
- Define a sub tree $T \subset T_0$ by pruning nodes from T_0
  - Collapsing the internal nodes by combining the corresponding regions
- Find $T_\alpha \subset T_0$ for a given $\alpha \geq 0$ that minimizes the *cost-complexity* criterion

$$C_\alpha(T) = R(T) + \alpha|T| = \sum_{m=1}^{|T|} N_m i(m) + \alpha|T|$$

  - $|T|$ is the number of terminal nodes in a tree T (model complexity)
  - $i(m)$ is the node impurity

Tuning Parameter (α)

- α can be interpreted as the complexity cost per terminal node.
- α determines the trade-off between the overall misclassification error and the model complexity
  - when α is small, the penalty for having a larger tree is small so T_α is large.
  - when α increases, $|T_\alpha|$ decreases

Weakest link pruning

- Define T_m as a branch of T_i containing a node m and its descendants
- When T_i is pruned at node m
  - its misclassification cost increases by $R(m) - R(T_m)$, where

$$R(T) = \sum_{m=1}^{|T|} N_m i(m)$$

  - and its complexity decreases by $|T_m| - 1$
  - the ratio

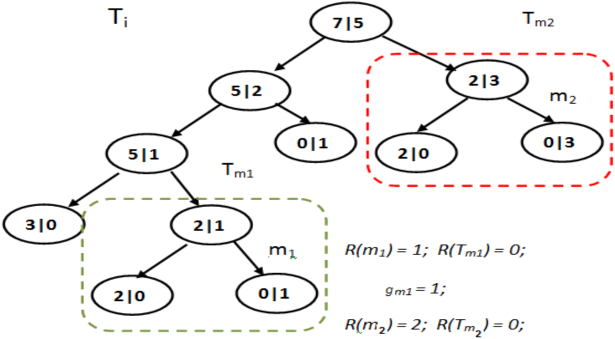
$$g_i(m) = \frac{R(m) - R(T_m)}{|T_m| - 1}$$

    measures the increase in misclassification cost per pruned terminal node

Weakest link pruning

- T_{i+1} is obtained by pruning all nodes in T_i with lowest value of $g_i(m)$ i.e. the *weakest link*
- α_i associated with T_i is given by $\alpha_i = min_m g_i(m)$
- starting with T_0 continue this pruning procedure till it reaches T_l (tree with only the root node)
- then CART identifies the *optimal subtree* from $\{T_i | i = 0, 1, ...l\}$ by selecting
  - the one with minimal classification error (0-SE rule)
  - or the smallest tree with in one standard error of minimum error rate (1-SE rule)
  - one approach is to use cross validation to find out the error

Weakest link pruning: Example



$R(m_1) = 1;\ R(T_{m1}) = 0;$

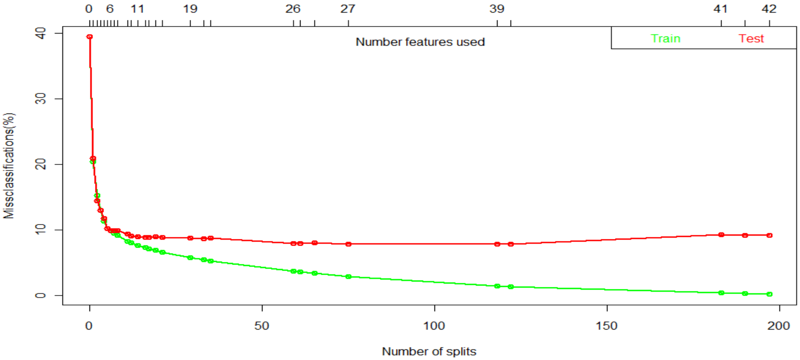
$g_{m1} = 1;$

$R(m_2) = 2;\ R(T_{m_2}) = 0;$

$g_{m2} = 2;$

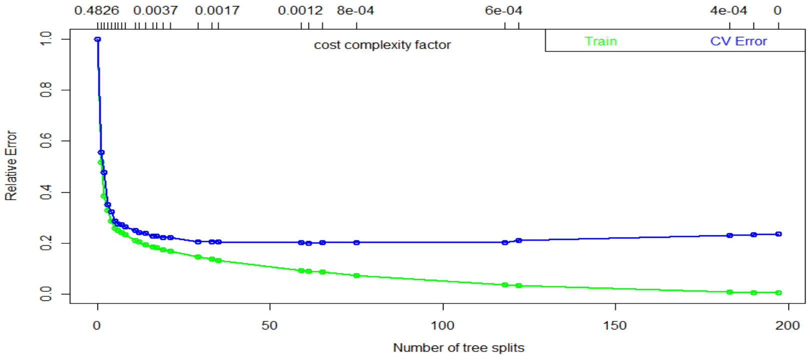
Spam data: Misclassification

- 4601 samples and 57 features (two class problem)
- train set (80 percent of the data set), and the rest for test



Spam data: Relative Error

- Cross validation (10 fold) is done to find best alpha



Regression Trees: Overview

Key differences:

- The outcome variables are continuous
- The criteria for splitting and pruning : squared error
- The calculation of predicted value: it is done by averaging the variables in a tree node

Impurity function: Squared Error

Suppose the feature space is partitioned into M regions $\{R_1, R_2, ...R_M\}$ and the response in each region R_m is represented as a constant c_m, then the regression model can be represented as :

$$f(x) = \sum_{m=1}^{M} c_m I(x \epsilon R_m)$$

If we use *minimization criteria* as Squared Error

$$\sum_{i=1}^{N} (y_i \; - \; f(x_i))^2$$

then best $\hat{c_m}$ will be $\hat{c_m} = ave(y_i | x_i \epsilon R_m)$

Greedy Algorithm

- Start with all of the data $(x_i, y_i)_{i=1}^{N}$, consider a splitting variable j and a split point s, define the regions

$$R_1(j,s) = \{X | X_j \leq s\} \ \ and \ \ R_2(j,s) = \{X | X_j > s\}$$

Greedy Algorithm

- Start with all of the data $(x_i, y_i)_{i=1}^{N}$, consider a splitting variable j and a split point s, define the regions

$$R_1(j,s) = \{X | X_j \leq s\} \; and \; R_2(j,s) = \{X | X_j > s\}$$

- Then the variables j and s can be solved using the greedy criterion

$$min_{j,s}[min_{c_1} \sum_{x_i \epsilon R_1(j,s)} (y_i - c_1)^2 + min_{c_2} \sum_{x_i \, \epsilon \, R_2(j,s)} (y_i - c_2)^2]$$

Greedy Algorithm

- Start with all of the data $(x_i, y_i)_{i=1}^{N}$, consider a splitting variable j and a split point s, define the regions

$$R_1(j,s) = \{X | X_j \leq s\} \text{ and } R_2(j,s) = \{X | X_j > s\}$$

- Then the variables j and s can be solved using the greedy criterion

$$min_{j,s}[min_{c_1} \sum_{x_i \epsilon R_1(j,s)} (y_i - c_1)^2 + min_{c_2} \sum_{x_i \epsilon R_2(j,s)} (y_i - c_2)^2]$$

- For any selected j and s the inner minimization is solved by

$$\hat{c}_1 = ave(y_i | x_i \epsilon R_1(j,s)) \text{ and } \hat{c}_2 = ave(y_i | x_i \epsilon R_2(j,s))$$

Tree Pruning

The tree pruning can be done by *weakest link pruning* using the squared error impurity function

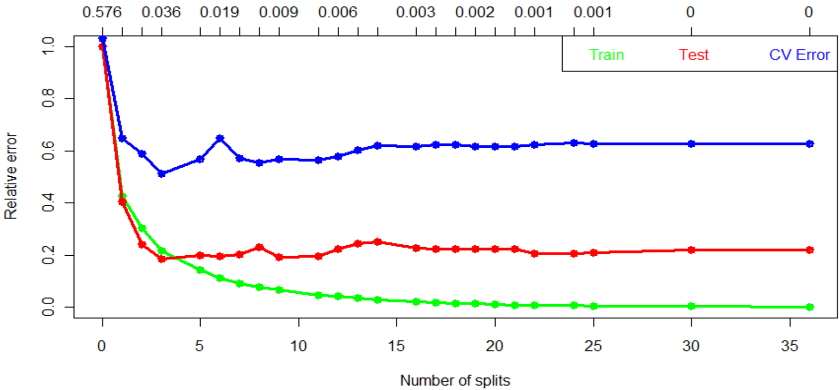
$$i(m) = \frac{1}{N_m} \sum_{x_i \,\epsilon\, R_m} (y_i \, - \, \hat{c_m})^2$$

The cost complexity criterion is similar to the case of classification

$$C_\alpha(T) = \sum_{m=1}^{|T|} N_m i(m) + \alpha |T|$$

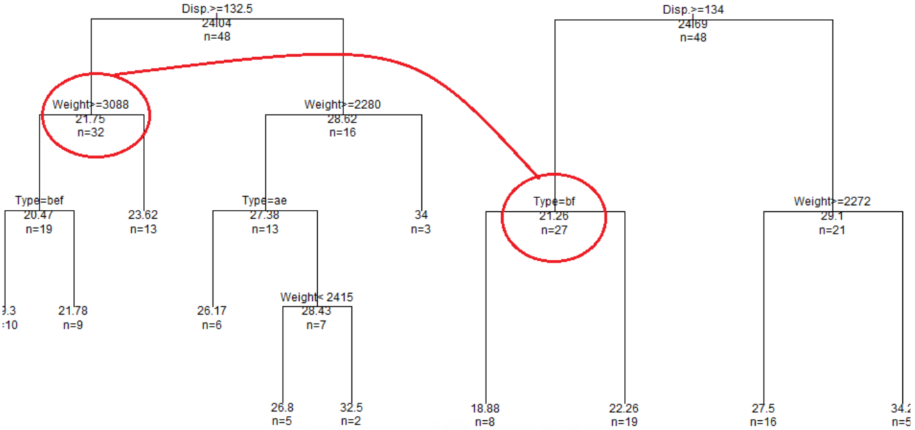
Data set: cars

Data set: 60 data points and 8 features (training set = 2/3 of the data; the rest is for testing)

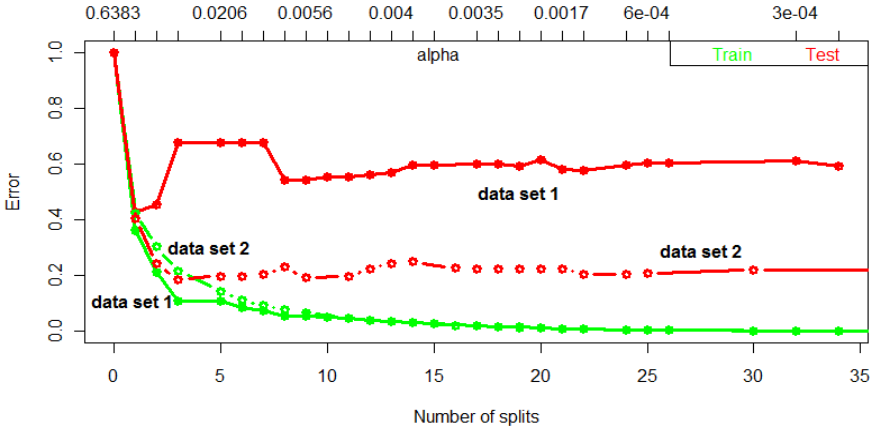


Tree structure's sensitivity to the training set

The left tree is from training set 1 and the right tree is from set 2.



Tree structure's sensitivity to the training set



Summary

Advantages:

- CART makes no distribution assumptions on the variables and supports both categorical and continuous variables
- Binary tree structure offers excellent interpret-ability
- Can be used for ranking the variables (by summing up the impurity measures across all nodes in the tree)

Disadvantages:

- Since CART uses binary tree, it suffers from instability
- Splits are aligned with the axes of the feature space, which may be suboptimal

Bishop, C. M. (2007). *Pattern Recognition and Machine Learning*. Springer, 1 edition.

Pham, H. (2006). *Springer Handbook of Engineering Statistics*. Springer, 1 edition.

T. Hastie, R. Tibshirani, J. F. (2009). *The Elements of Statistical Learning*. Springer, 2 edition.

Terry M. Therneau, Elizabeth J. Atkinson, M. F. (1997). An introduction to recursive partitioning using the rpart routines. Technical report.