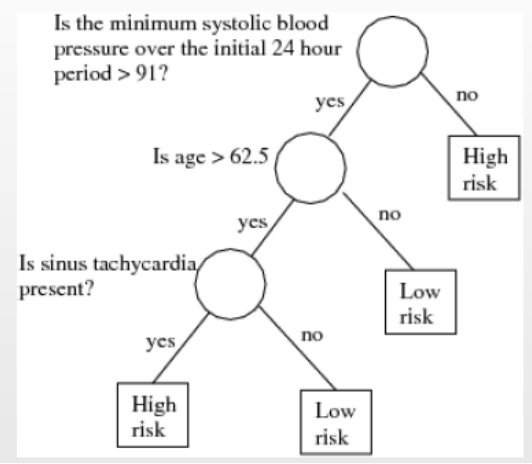
Tree-Based Methods



Rachael Hageman Blair

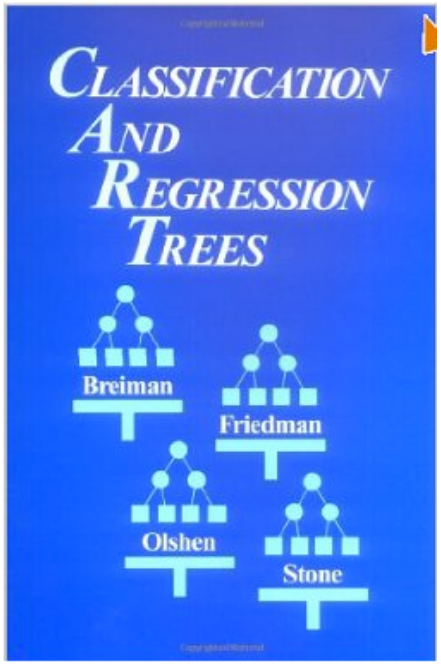
Outline

- Motivation: Real world problems
- Tree Growing
- Tree Pruning
- Metrics
- Example: Digit Recognition
- Issues
- Example: SPAM
- Visualization
- Conclusions

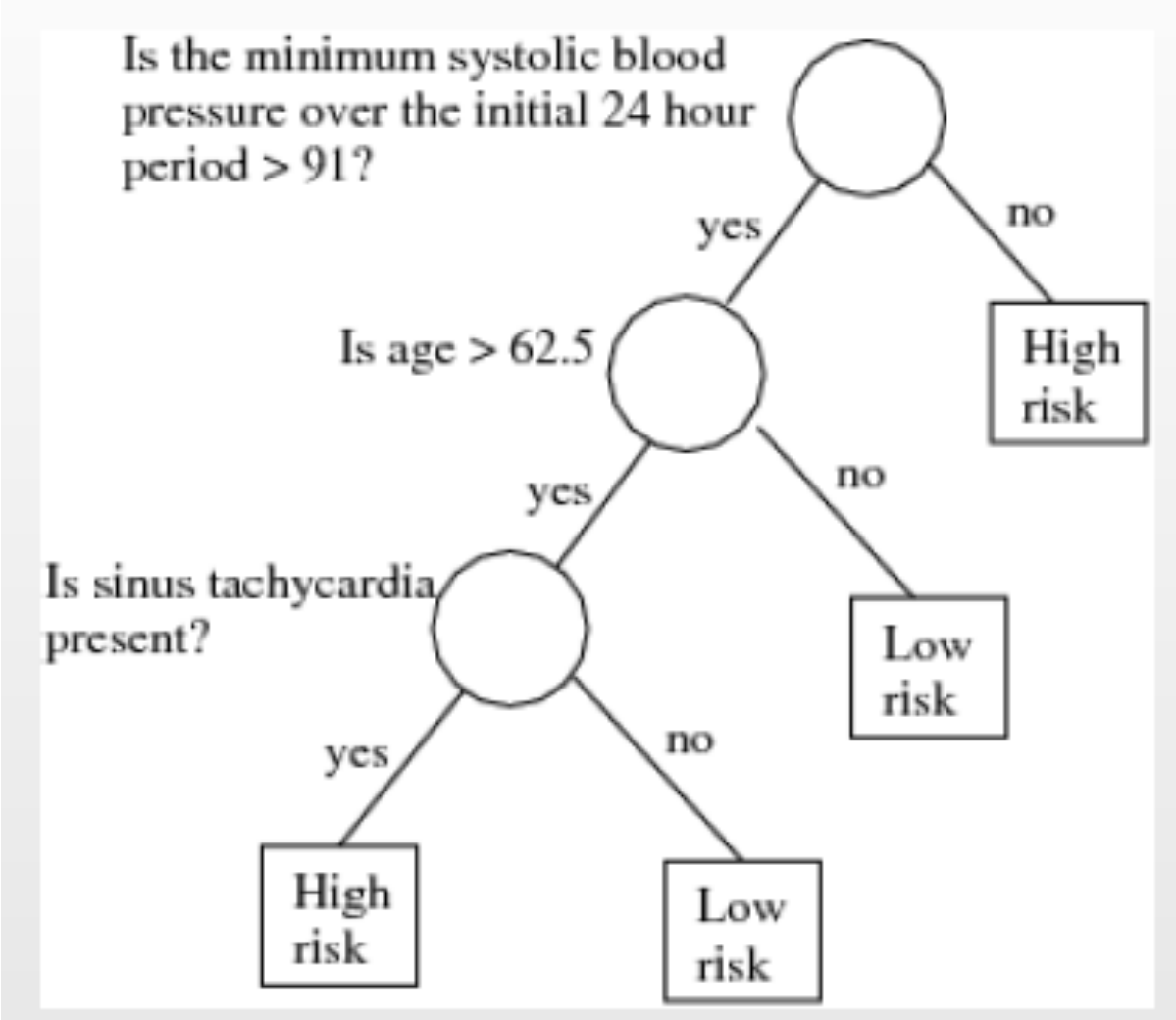
To be continued……

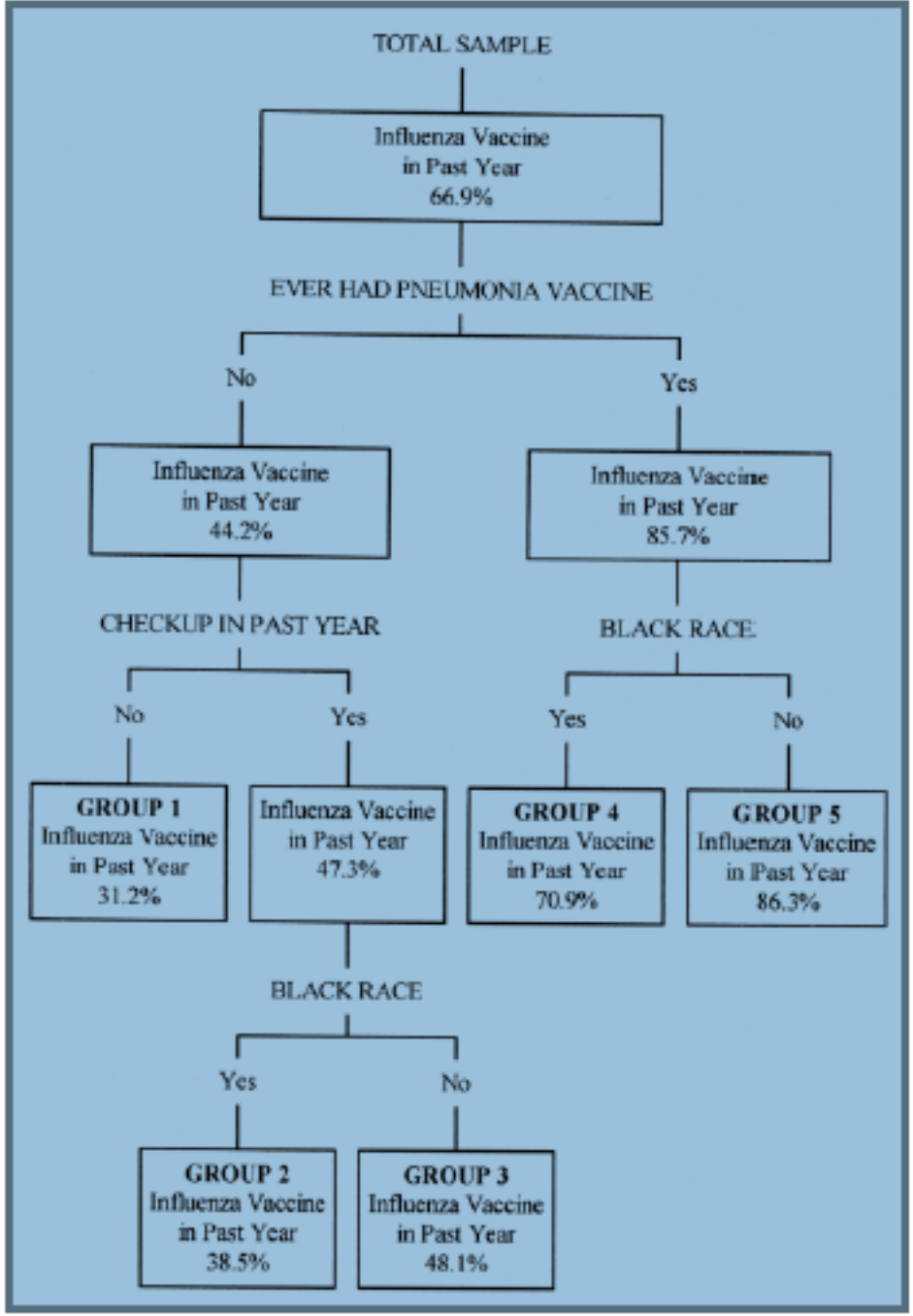
Regression Trees

- Medical Example (CART)
  - Predict high risk patients who will not survive at least 30 days on the basis of the initial 24 hour data.
  - 19 variables are measured in the first 24 hours, including blood pressure, age, etc.



A tree structure classification rule for the medical example





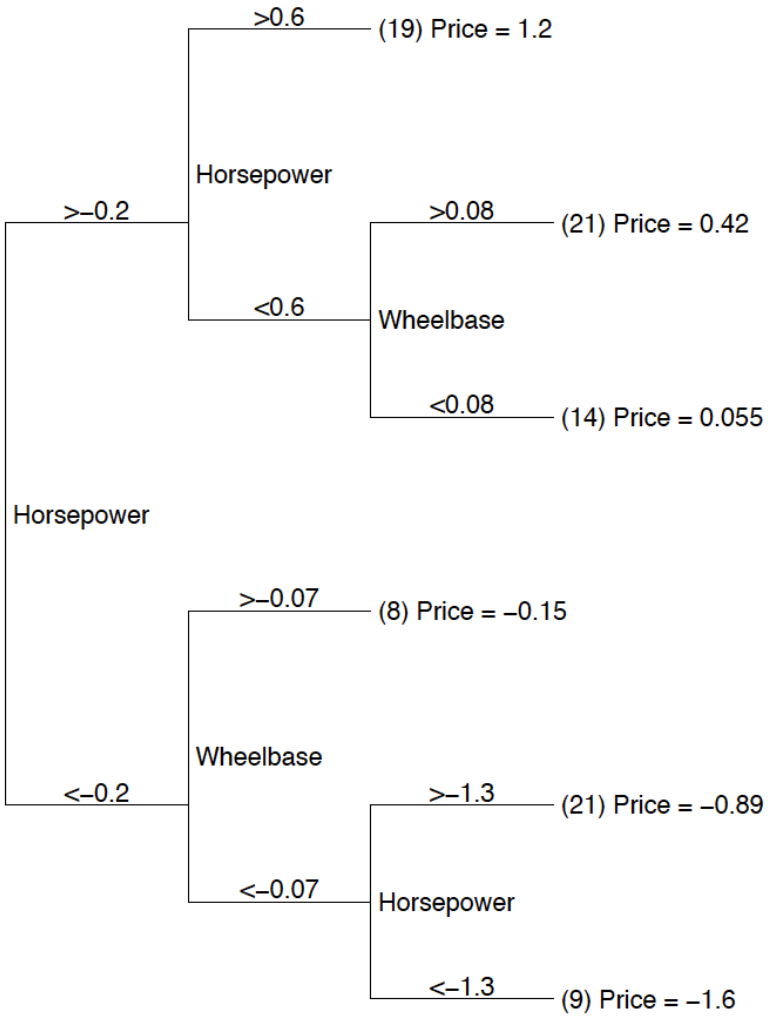
and Regression Tree Analysis in Public Health: Methodological Review and Comparison With Logistic Regression

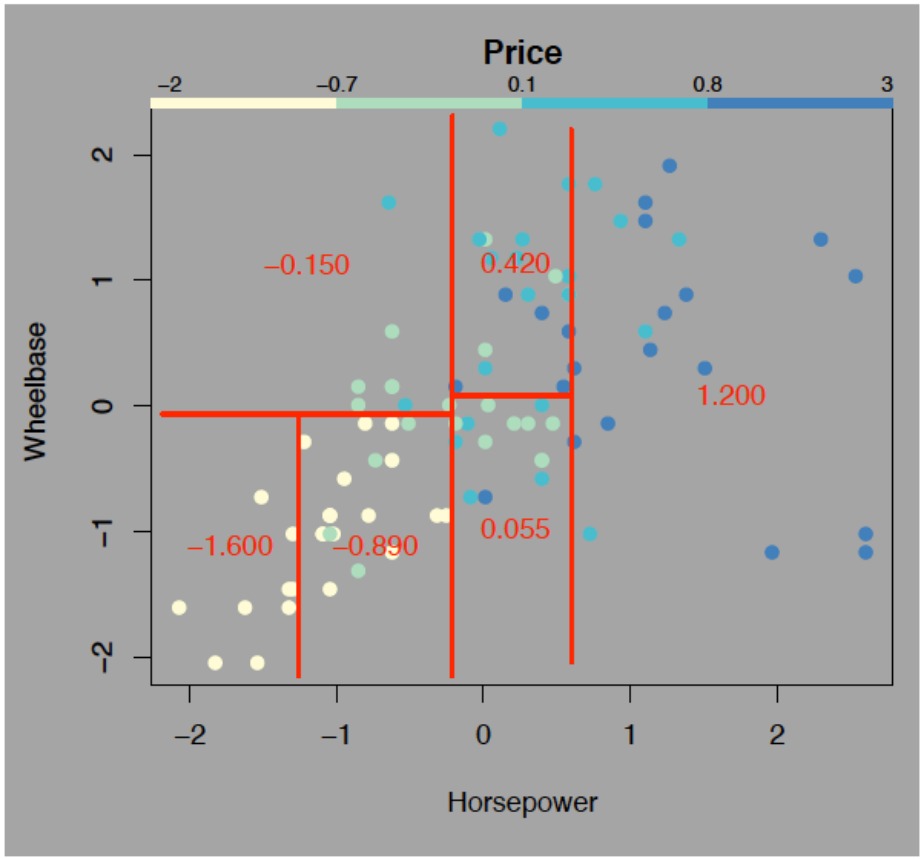
ephenie C. Lemon, Ph.D., Jason Roy, Ph.D., and Melissa A. Clark, Ph.D.
Brown University School of Medicine

Peter D. Friedmann, M.D., M.P.H.
Brown University School of Medicine and
Rhode Island Hospital, Providence, RI

William Rakowski, Ph.D.
Brown University School of Medicine

A tree structure classification rule for car price





Tree Based Methods

- Denote the feature space by χ . The input vector, $X \in \chi$ contains p features, some of which may be categorical.
- Tree structured classifiers are constructed by repeated splits of subsets of χ into descendant subsets, beginning with χ itself.
- Terminology: node, terminal node (leaf node), parent node, child node.
- The union of regions occupied by two child nodes is the region occupied by their parent node.
- Every leaf node is assigned a class. A query/prediction is associated class of the leaf it lands on.

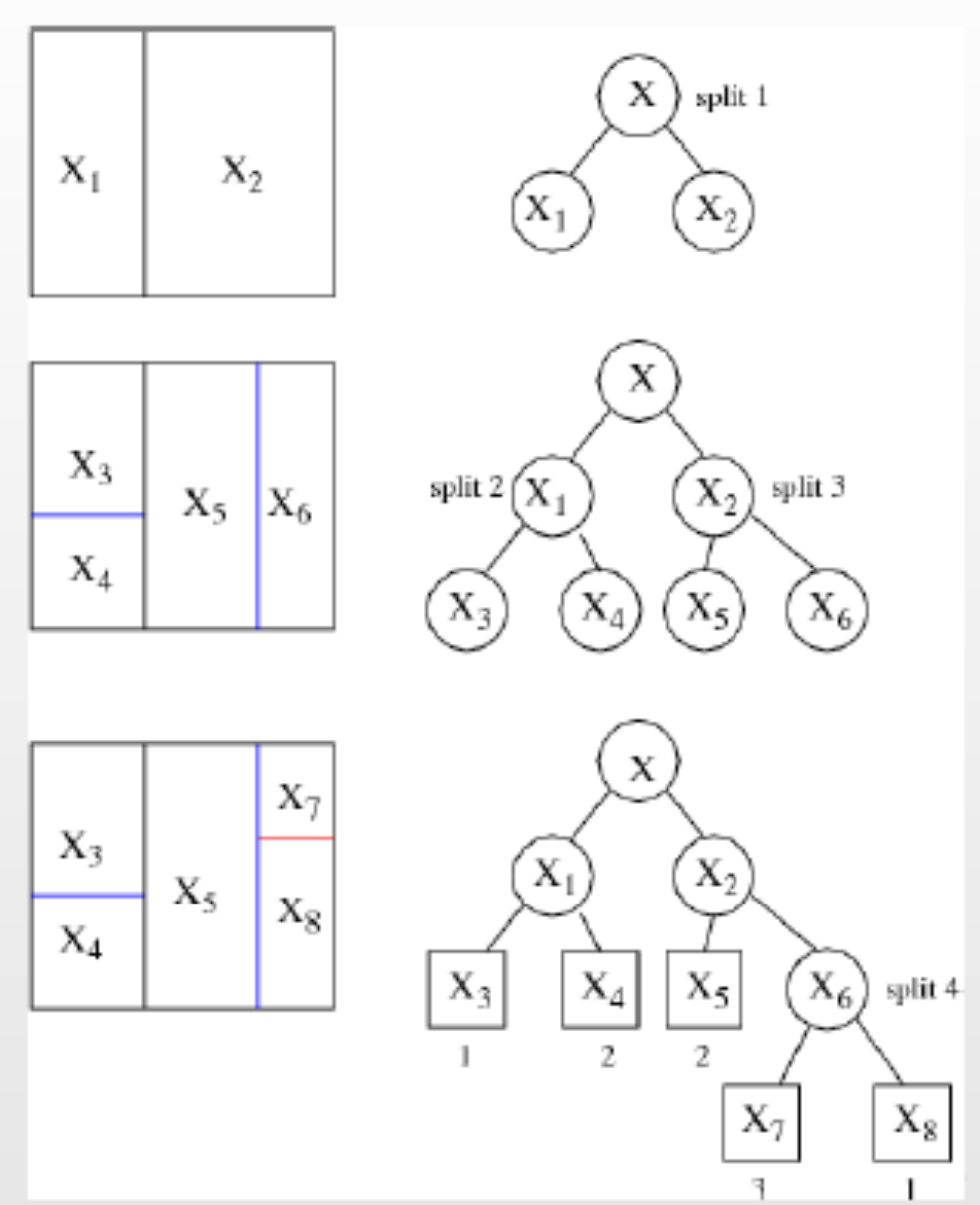
Terminology

- A node is denoted by t. Its left child node is denoted by t_L And the right by t_R.

- The collection of all the nodes is denoted by T; and the collection of all the leaf nodes by $\tilde{T}$.

- A split is denoted by s. The set of splits is denoted by S.

A visualization



An Alternative to Non-linear Regression

Divide and Conquer.

- Partition the space into subspaces where the models and interactions are manageable.

- Decision Trees: Use recursive partitioning to subdivide the space until simple models can be fit.

Regression Trees

- Objective: Partition the feature space and fit simpler models in each partition.

$Y \sim$ Response

$X \in [0,1] \sim$ Explanatory

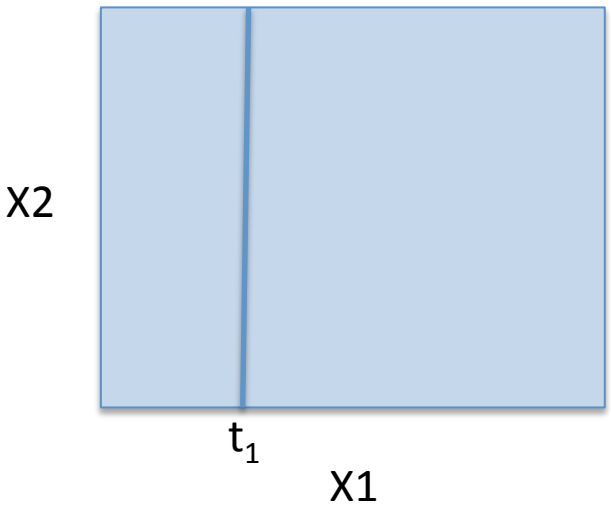
Decision Rules:

1) $X_1 = t_1$

2) $X_1 \leq t_1$ split at $X_2 = t_2$

3) $X_1 > t_1$ split at $X_1 = t_3$

4) $X_1 \leq t_3$ split at $X_2 = t_4$



Regression Trees

- Objective: Partition the feature space and fit simpler models in each partition.

$Y \sim$ Response

$X \in [0,1] \sim$ Explanatory

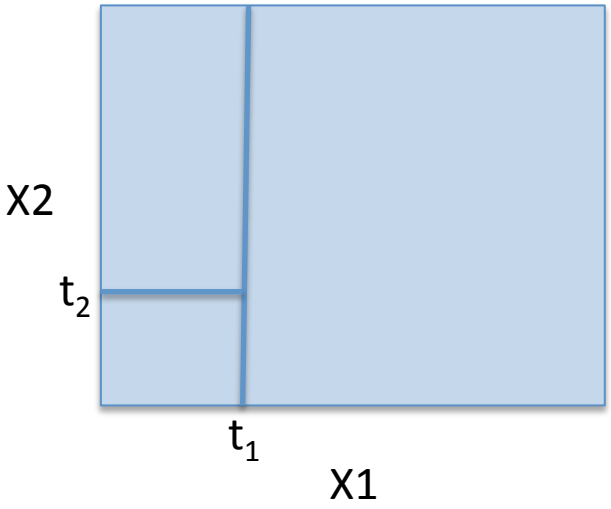
Decision Rules:

1) $X_1 = t_1$

2) $X_1 \leq t_1$ split at $X_2 = t_2$

3) $X_1 > t_1$ split at $X_1 = t_3$

4) $X_1 \leq t_3$ split at $X_2 = t_4$



Regression Trees

- Objective: Partition the feature space and fit simpler models in each partition.

$Y \sim$ Response

$X \in [0,1] \sim$ Explanatory

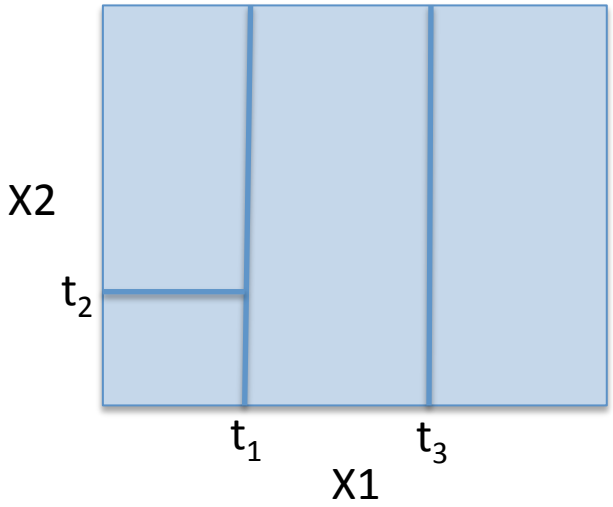
Decision Rules:

1) $X_1 = t_1$

2) $X_1 \leq t_1$ split at $X_2 = t_2$

3) $X_1 > t_1$ split at $X_1 = t_3$

4) $X_1 \leq t_3$ split at $X_2 = t_4$



Regression Trees

- Objective: Partition the feature space and fit simpler models in each partition.

$Y \sim$ Response

$X \in [0,1] \sim$ Explanatory

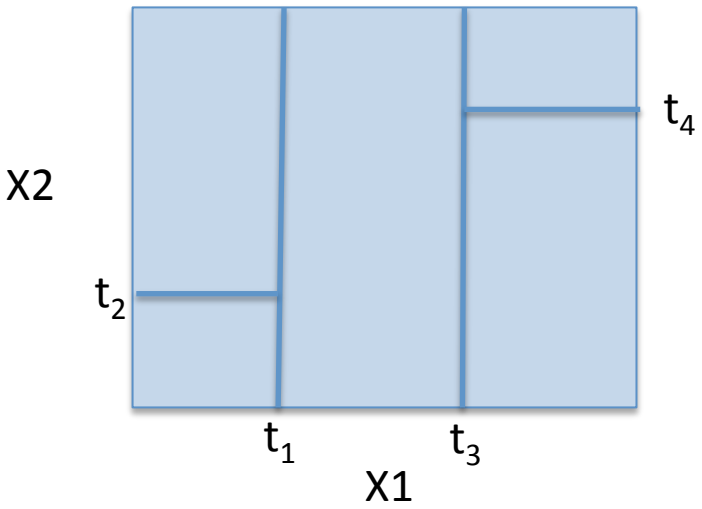
Decision Rules:

1) $X_1 = t_1$

2) $X_1 \leq t_1$ split at $X_2 = t_2$

3) $X_1 > t_1$ split at $X_1 = t_3$

4) $X_1 \leq t_3$ split at $X_2 = t_4$



Regression Trees

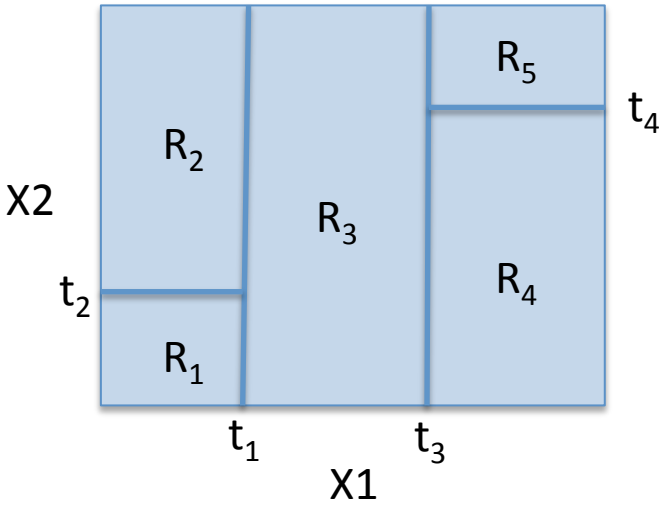
- Objective: Partition the feature space and fit simpler models in each partition.

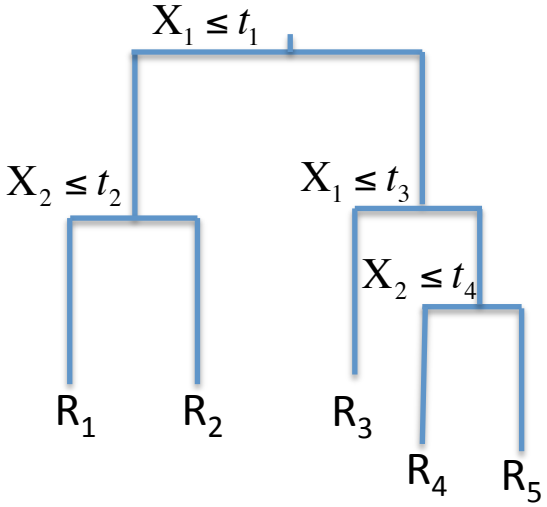
$Y \sim$ Response

$X \in [0,1] \sim$ Explanatory

Decision Rules:

1) $X_1 = t_1$
2) $X_1 \le t_1$ split at $X_2 = t_2$
3) $X_1 > t_1$ split at $X_1 = t_3$
4) $X_1 \le t_3$ split at $X_2 = t_4$





$$f(X) = \sum_{m=1}^{5} c_m I\{(X_1, X_2) \in R_m\}$$

Advantages to Regression Trees

- Predictions are as fast as looking up constants on a tree.
- Important variables "fall out" can be identified by tree inspection.
- When data is missing, we can carry out calculations to a certain branch …. still informative to some extent.
- If the true regression surface is non-smooth, the regression tree provides a non-smooth response.
- If the true regression surface is smooth, the regression tree can provide a good approximation.
- Fast and Reliable algorithms.

R_5

How to Grow a Tree?

Similar to Clustering:

- *Clustering* maximize the information (I) the cluster gives about the features I[C;X].
- *Regression Trees* maximize I[C;Y], Y is the dependent variable and C determines which leaf of the tree we end up in.

The idea – Find one binary question which maximizes the information we get about Y; Use it as a root node; do the same thing for the child nodes and continue.

The three elements of tree growing

The construction of a tree involves the following three elements:

1. The selection of the splits.
2. The decisions when to declare a node terminal or continue splitting it.
3. The assignment of each terminal node to a class.

Tree growing

In particular, we need to decide the following:

1. A set of Q binary questions of the form:

   Is $X \in A,\ A \subseteq \chi.$?

2. A goodness of split criterion $\phi(s,t)$, that can be evaluated for any split s of any node t .
3. A stop-splitting rule.
4. A rule for assigning every terminal node to a class.

Standard Set of Questions

- The input vector $X = \left(X_1, X_2, \ldots, X_p\right)$ contains features of different types.
- Each split depends on the value of only a *unique* variable.

- $\mathcal{Q}$ includes questions of the form:

$$\left\{\text{is } X_j \leq c\,?\right\}$$

for all real valued *c*.

Note: the training data is finite, there are only finitely many distinct splits that can be generated from the question $\left\{\text{is } X_j \leq c\,?\right\}$.

Standard Set of Questions

- If X_j is categorical, taking values, say in $\{1,2,\ldots,M\}$, the Q contains questions of the form

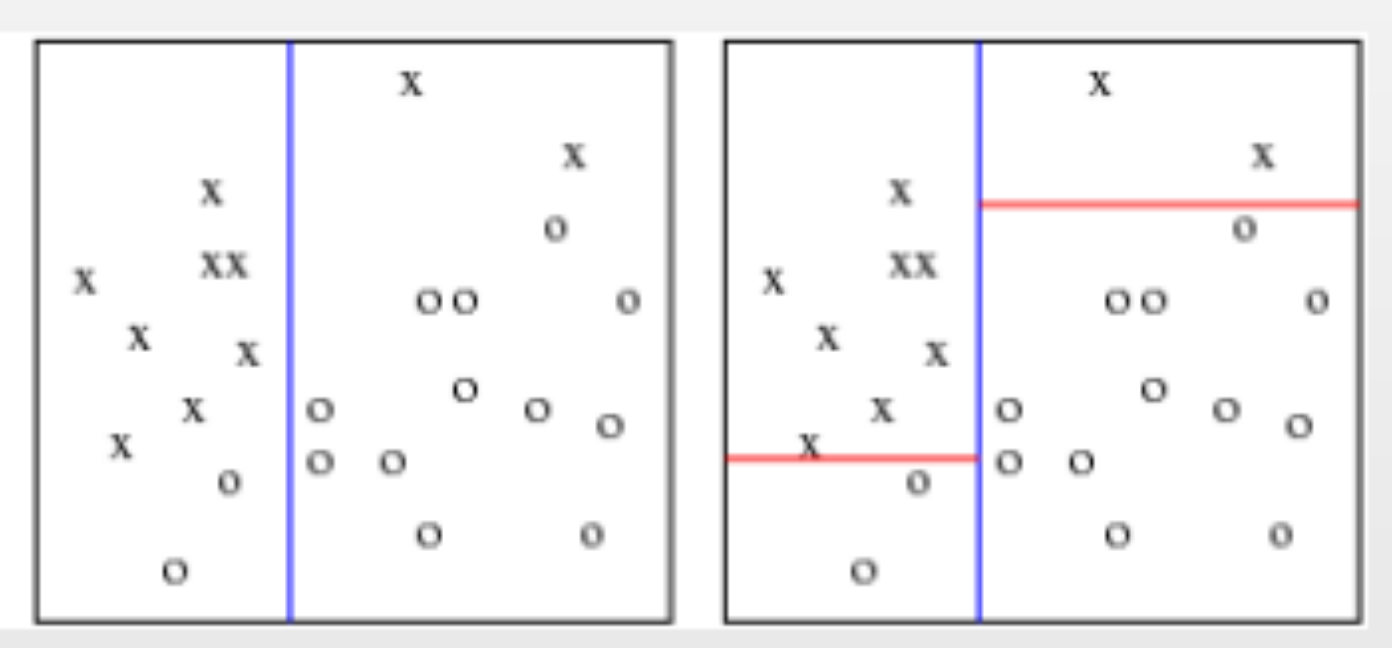
$$\{\text{is } X_j \in A?\}$$

where A ranges over all subsets $\{1,2,\ldots,M\}$.

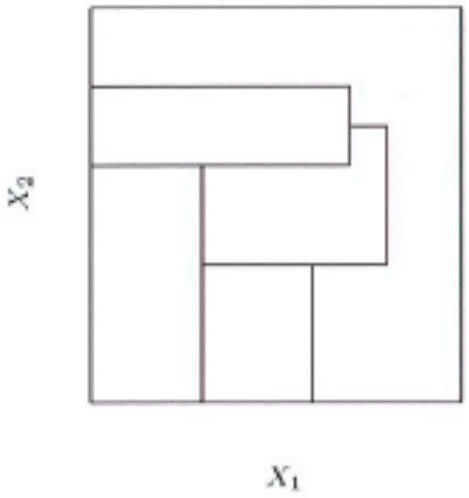
- The splits for all *p* variables constitute the standard set of questions.

Goodness of Split

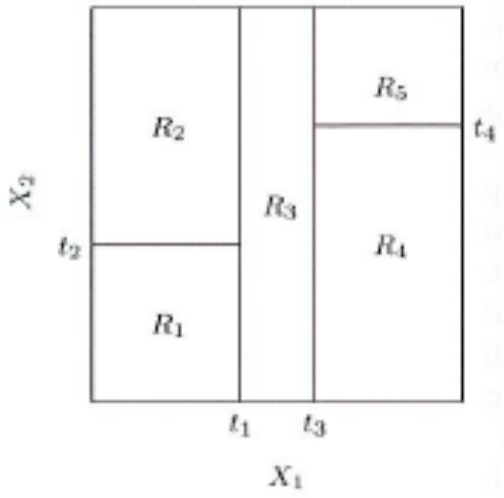
- A goodness of split is measured by an impurity function defined for each node.
- Intuitively, we want each leaf node to be “pure”, that is, one class dominates.

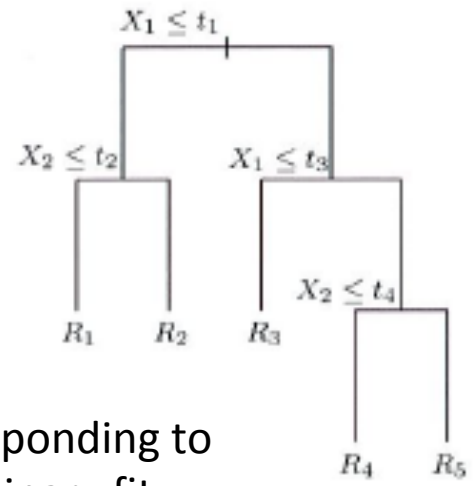


Not a Recursive binary fit

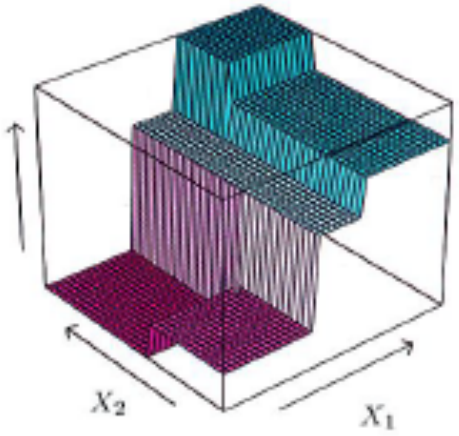


Recursive binary fit





Tree corresponding to
Recursive binary fit



Perspective plot corresponding to
Recursive binary fit

The main idea – recursive binary partitions

- Split the space into two regions, model the response by the mean of *Y* in each region. Choose the split points to achieve the best fit.
- Split one or more of those regions into two regions.
- Continue until some stopping rule is applied.

The corresponding regression model predicts Y with a constant c_m in region R_m, that is,

$$f(X) = \sum_{m=1}^{5} c_m I\{(X_1, X_2) \in R_m\}.$$

Tree-based methods

- Key advantage – easy to interpret, (if-else).
- Difficult to draw region breakdown in more than two dimensions, but we can always fully decompose the space.
- Popular in medical science, mimics doctor thinking…..
  Divide the population into strata of high and low outcome, on the basis of patient characteristics.

How to Grow a Regression Tree?

Data: *p* inputs and a response, for each of the N observations: that is (x_i, y_i) for $i = 1,2,\ldots,N$ with $x_i = (x_{i1}, x_{i2}, \ldots, x_{iN})$.

Objective: automatically decide on splitting variables and split points to minimize some criterion.

Suppose we have partitioned the space into M regions $R_1, R_2, \ldots, R_M$ and we want to model the response as constant c_m in each region: $f(X) = \sum_{m=1}^{M} c_m I\{(X_1, X_2) \in R_m\}$. If we use RSS, then the best estimate is simply the average in the region:

$$\hat{c}_m = ave(y_i \mid x_i \in R_m) \ .$$

How to Grow a Regression Tree?

- Search for binary partition via minimize sum of squares is computationally intractable.

Greedy approach:

- Starting with all of the data, consider splitting variable j and split point S, and define the pair of half planes:

$$R_1(j,s) = \{X \mid X_j \le s\} \text{ and } R_2(j,s) = \{X \mid X_j > s\}.$$

- We seek the splitting variable j and split point S that solve:

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

- For any choice j and S, the inner minimization is solved by:

$$\hat{c}_1 = ave\left(y_i \mid x_i \in R_1(j,s)\right) \text{ and } \hat{c}_2 = ave\left(y_i \mid x_i \in R_2(j,s)\right).$$

How to Grow a Regression Tree?

- For each splitting variable, the determination of the split point s can be done quickly by scanning through all of the inputs to find the best pair (j, s) .
- Having found the best split, partition the data into the two resulting regions and repeat the process on each of the two regions…..etc.

When to stop?

Tree size → Model complexity

Can split the tree efficiently and easily.

Splitting it too much can over-fit the data.

Small trees may miss important structure.

Tree Pruning

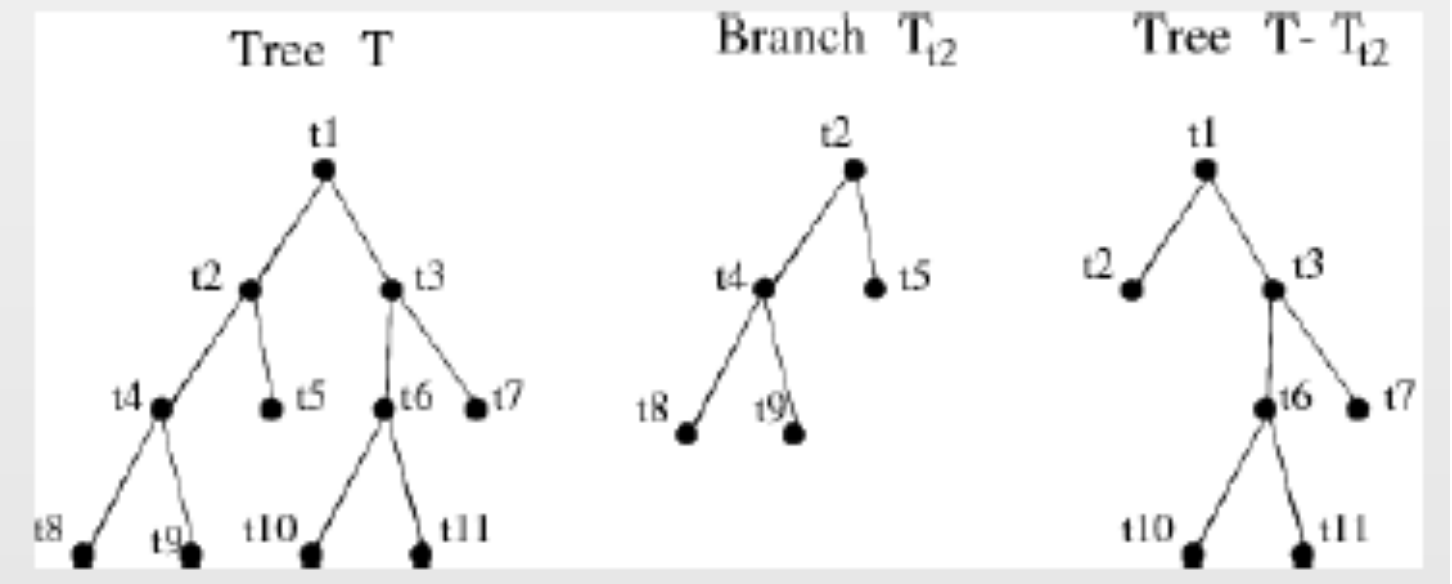
- Grow a very large tree $T_{\max}$.
  - Until all terminal nodes are pure (contain only one class) or contain only identical measurement vectors.
  - When the number of data in each terminal node is no greater than a particular threshold, e.g., 5 or even 1.
  - As long as the tree is sufficiently large, the size of the initial tree is not critical.

Tree Pruning

Terminology

1. Descendent: a node t' is a descendent of node t if there is a connected path down the tree leading from t to t'.
2. Ancestor: t is an ancestor of t' if t' is its descendant.
3. A branch T_t of T with root node $t \in T$ consists of the node t and all descendants t in T.
4. Pruning a branch T_t from a tree T consists of deleting from all descendants of t, that is, cutting off all of T_t except its root node. The tree pruned this way is denoted by $T - T_t$.
5. If T' is obtained from T by successively pruning off branches, the T' it is called a pruned subtree of T and denoted $T' \subset T$.

Tree Pruning



- Even for a moderate size $T_{\max}$, there is an enormously large number of subtrees and an even larger number of ways to prune the initial tree.
- An automated pruning procedure is needed to find the optimal pruning, and manage the computational demand.

When to stop?

Cost Complexity Pruning:

Let $T \subset T_0$ be any tree that can be obtained by pruning T_0.

Let $N_m = \#\{x_i \in R_m\}$

$$\hat{c}_m = \frac{1}{N_m} \sum_{x_i \in R_m} y_i$$

Impurity Measure

$$Q_m(T) = \frac{1}{N_m} \sum_{x_i \in R_m} (y_i - \hat{c}_m)^2$$

Governs the tradeoff between fit and model complexity.

Cost Complexity Criterion (want to minimize):

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

where $|T|$ is the number of terminal nodes in T.

When to stop?

For a single alpha, a spectrum of trees is obtained.

Tree	T_1	T_2	T_3	T_4	T_5	T_6	T_7	T_8	T_9	T_{10}	T_{11}	T_{12}	T_{13}
$\lvert\tilde{T}_k\rvert$	71	63	58	40	34	19	10	9	7	6	5	2	1

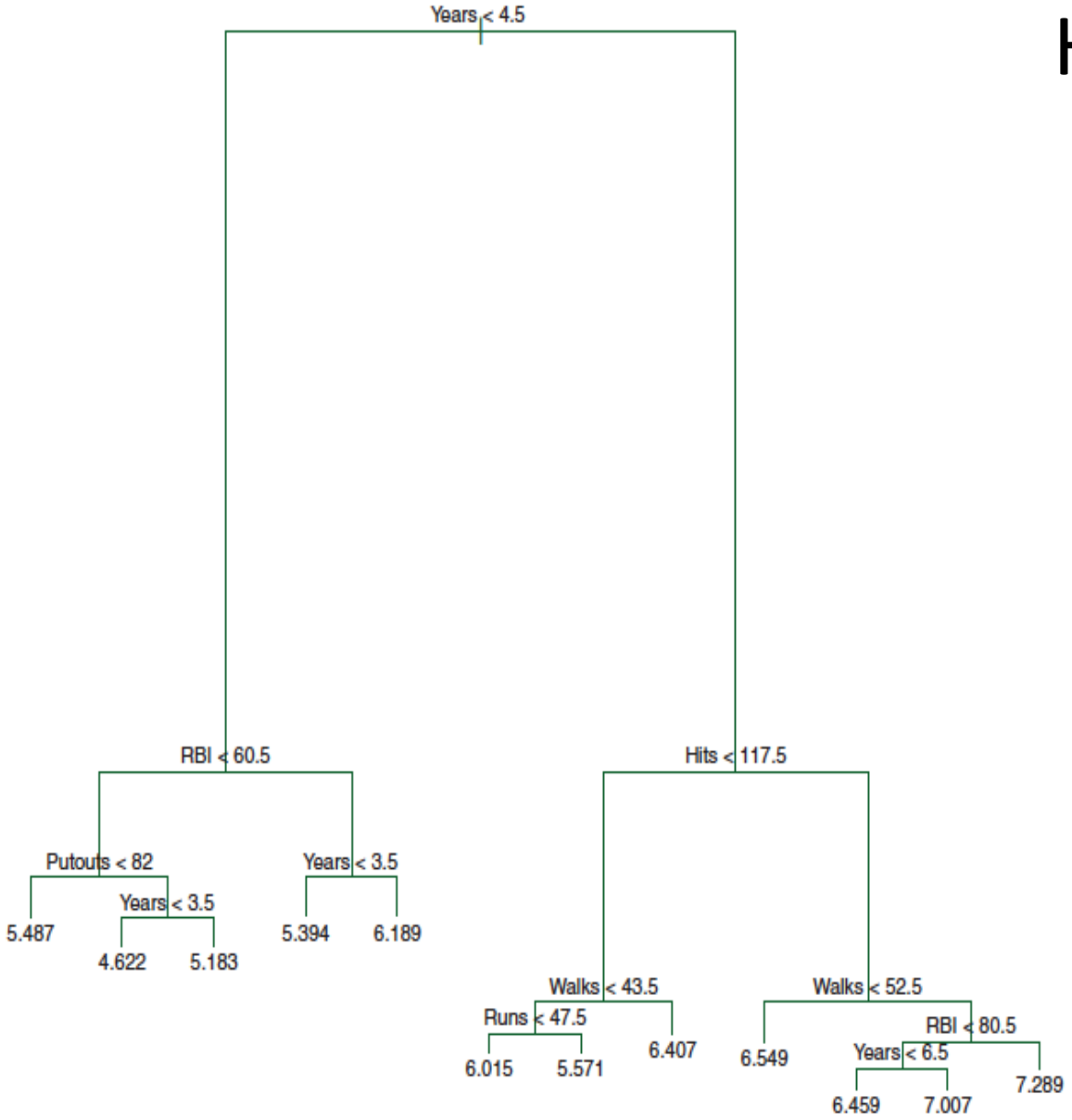
The question is, which alpha, and which subtree to use?

- K-fold cross validation for the selection of alpha.

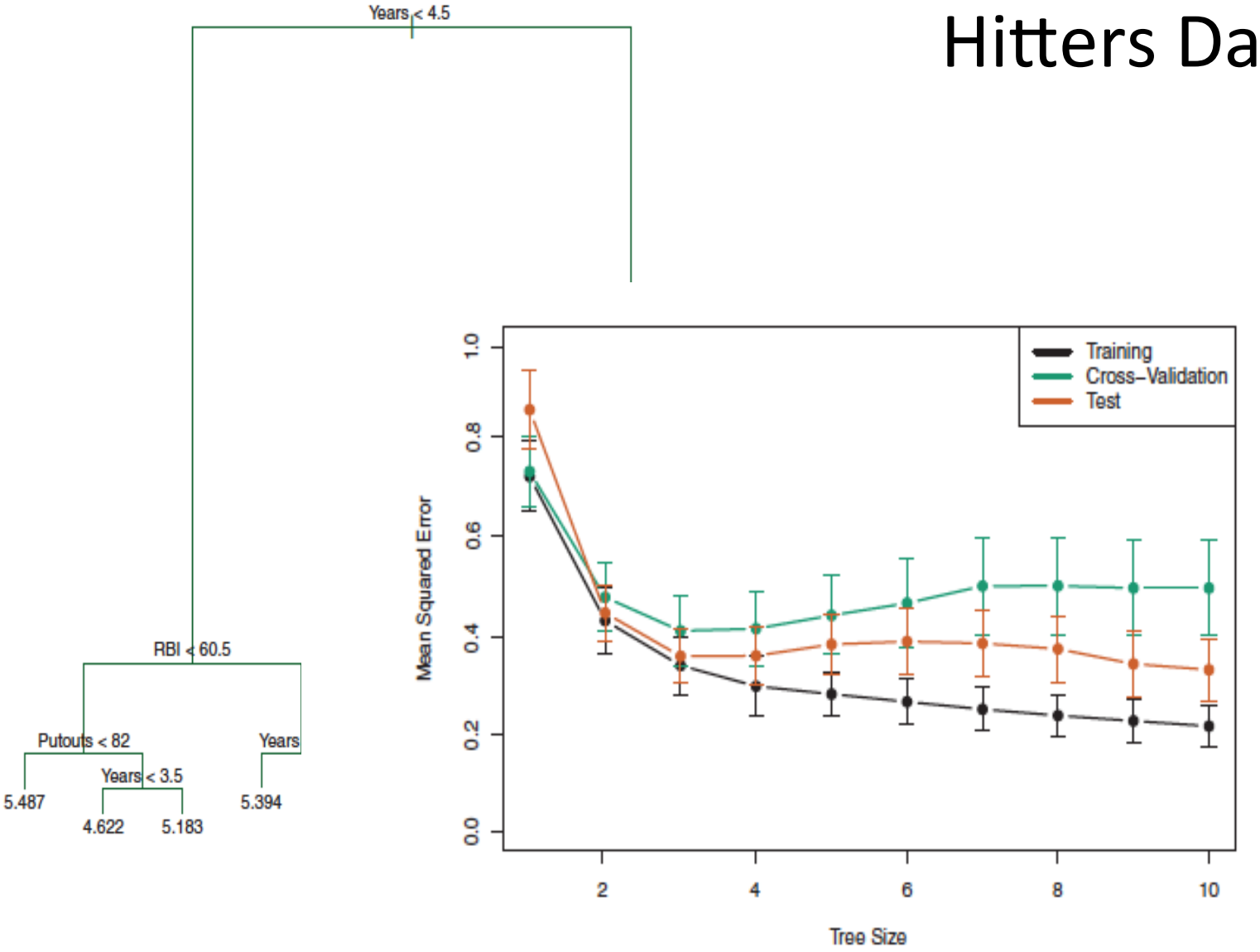
Stability an issue –

- Trees may be very different structurally, but perform the same in terms of classification.

Hitters Data



Hitters Data



Classification Trees

- Impurity function for regression trees not appropriate for classification problems.
- For a node m, representing a region R_m with N_m observations, let,

$$\hat{p}_{mk} = \sum_{x_i \in R_m} I\left(y_i = k\right)$$

  Be the proportion of class k observations in node m.
- The observations in node *m* are classified to the majority class: $k(m) = \arg\max_k \hat{p}_{mk}$.

Classification Trees

Measures of node impurity, $Q_m(T)$:

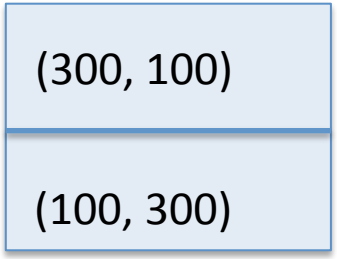
- Cross Entropy or deviance: $-\sum_{k=1}^{K} \hat{p}_{mk} \log \hat{p}_{mk}$.
- Misclassification error: $\frac{1}{N} \sum_{i \in R_m} I\left(y_i \neq k(m)\right) = 1 - \hat{p}_{mk(m)}$.
- Gini index: $\sum_{k=1}^{K} \hat{p}_{mk}\left(1 - \hat{p}_{mk}\right)$.

Small values if $\hat{p}_{mk}$ are close to zero or one - desirable.

Classification Trees

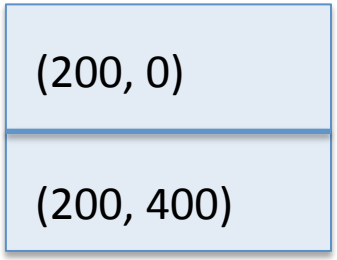
Preferred error splits: suppose 2 class problem 400 observations in each class (400, 400).

Split via misclassification



Error rate 25%

Optimal Split via cross entropy or gini index

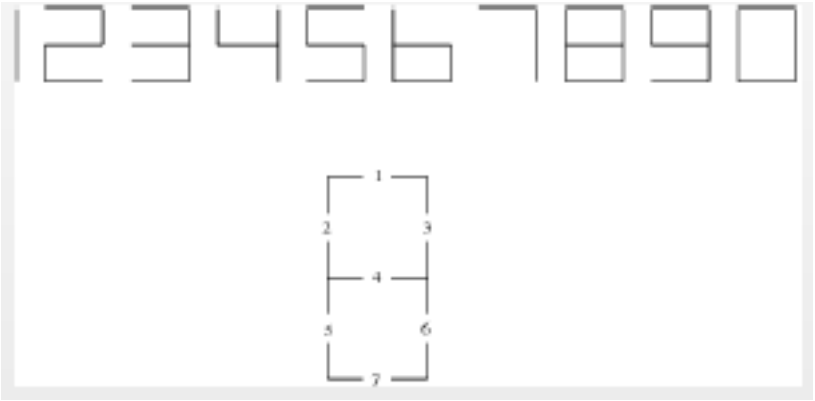


Error rate 25%

(Cross-entropy or gini index for growing.
Misclassification rate for pruning.)

Example – Digit Recognition with CART

- The ten digits are shown by different on-off combinations of seven horizontal or vertical lights.



- Each digit is represented by a 7-dimensional vector of zeros and ones. The *i*th sample is: $x_{ij} = (x_{i1}, x_{i2}, \ldots, x_{i7})$. If $x_{ij} = 1$ the jth light is on, else $x_{ij} = 0$ and it is off.

Example – Digit Recognition with CART

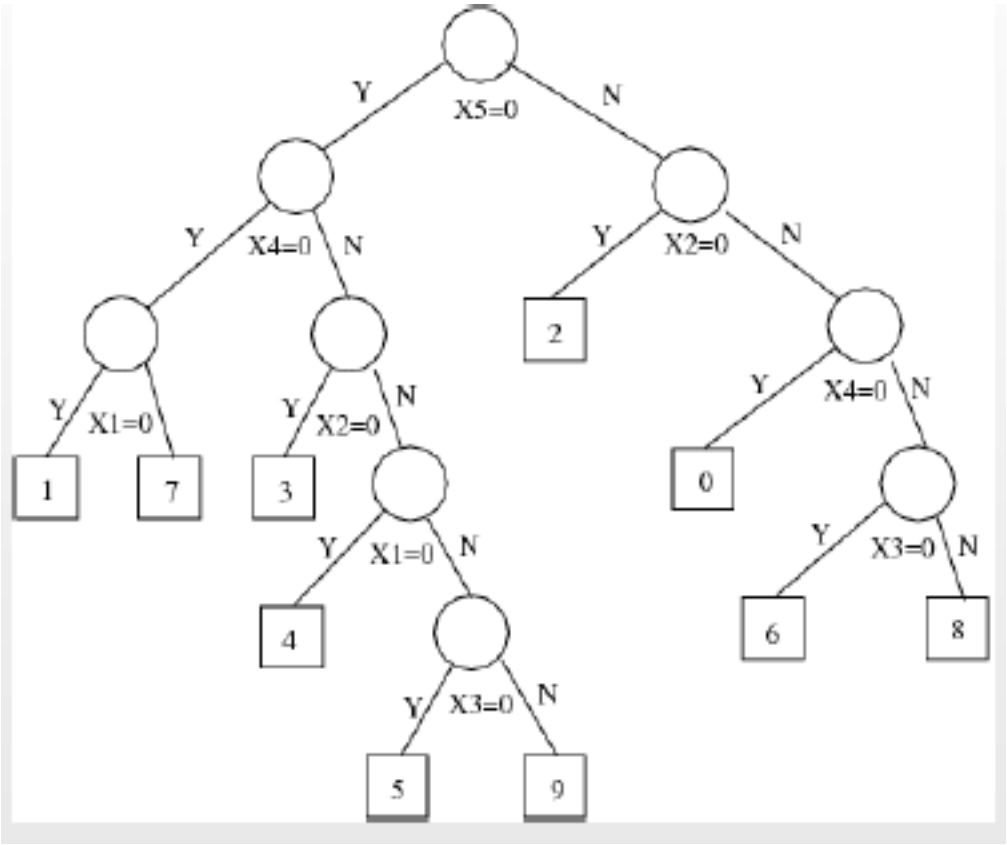
Digit	x_1	x_2	x_3	x_4	x_5	x_6	x_7
1	0	0	1	0	0	1	0
2	1	0	1	1	1	0	1
3	1	0	1	1	0	1	1
4	0	1	1	1	0	1	0
5	1	1	0	1	0	1	1
6	1	1	0	1	1	1	1
7	1	0	1	0	0	1	0
8	1	1	1	1	1	1	1
9	1	1	1	1	0	1	1
0	1	1	1	0	1	1	1

Example – Digit Recognition with CART

- Data is generated by a malfunctioning calculator.
- Each of the seven digits has a probability 0.1 of being in the wrong state independently.
- Training set has 200 samples.

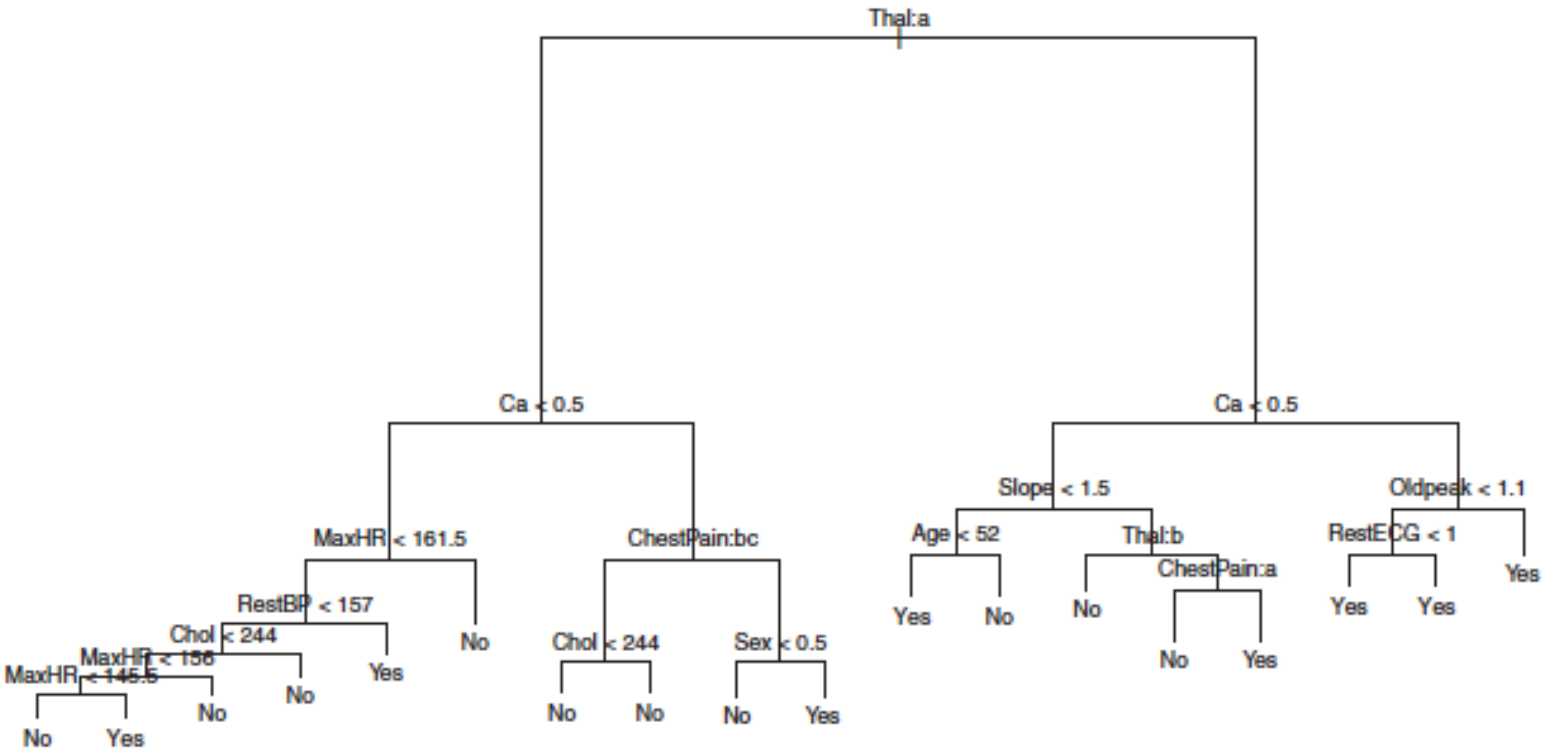
A tree structured classifier is fit:

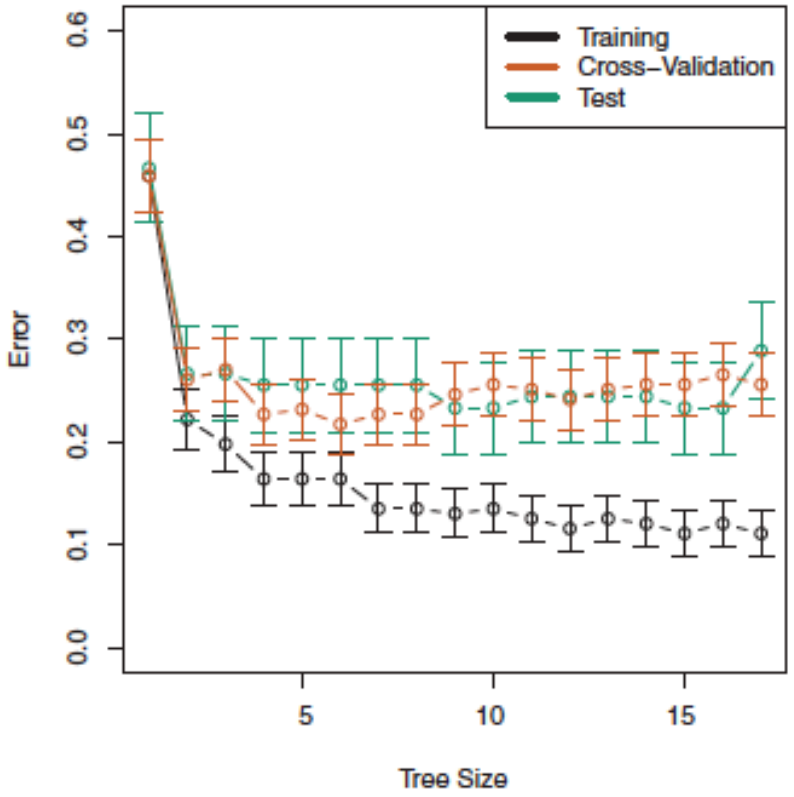
- The set of questions contains:
  $\{\text{is } x_{.j} = 0,\ j=1,2,\ldots,7\}$
- The (two-ing rule) is used in splitting.
- The pruning cross-validation method is used to choose the right size tree.

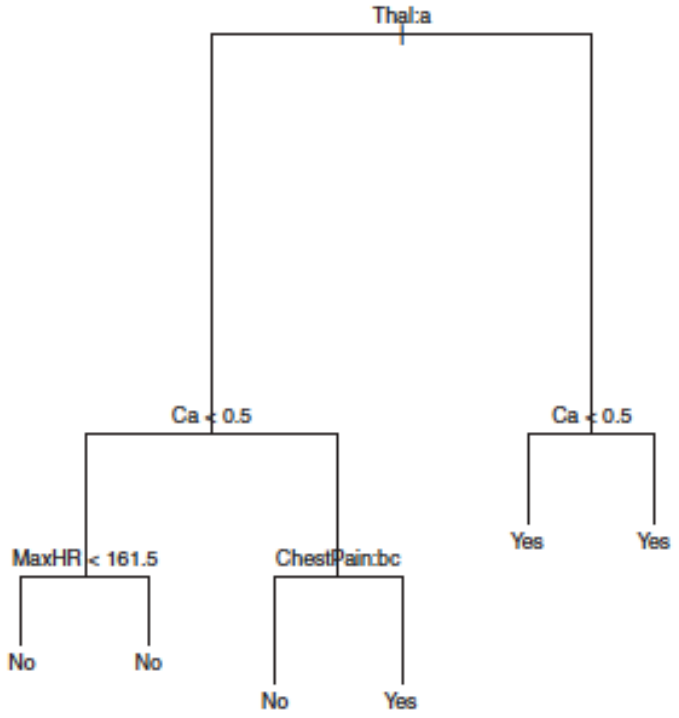


- Accidently every digit occupies one leaf. In practice any class can occupy many or no leaf nodes.
- $X_{.6}$ and $X_{.7}$ are never used.

Error rate ~ 30%


Thal:a
Ca < 0.5
Ca < 0.5
MaxHR < 161.5
ChestPain:bc
Slope < 1.5
Oldpeak < 1.1
Age < 52
Thal:b
RestECG < 1
Yes
RestBP < 157
Chol < 244
Chol < 244
Sex < 0.5
ChestPain:a
Yes
No
No
Yes
Yes
No
MaxHR < 156
MaxHR < 145.5
No
No
No
No
No
Yes
No
Yes
No
Yes
No
Yes


Training
Cross–Validation
Test
Error
0.0
0.1
0.2
0.3
0.4
0.5
0.6
5
10
15
Tree Size


Thal:a
Ca < 0.5
Ca < 0.5
MaxHR < 161.5
ChestPain:bc
Yes
Yes
No
No
No
Yes

Other Details ….

- **Loss Function** –

  In some settings the consequence of misclassifying in a certain way is more serious.

  e.g., a person has cancer vs. they do not.

  e.g., a person has a heart attack vs. they do not.

- To account for this, a (KxK) matrix is formed to convey the loss incurred for classifying a class K observation as K'. Note L(k,k)=0.
- Can embed it into the modeling process, $\sum_{k \neq k'} L_{kk'} \hat{p}_{mk} \hat{p}_{mk'}$
- For a two class case, better to weight the observations directly.

Other Details ….

- **Categorical predictors** – are problematic. The partitioning algorithm will tend to favor predictors with many levels.

- **Missing values** – certain variables are missing in some training samples. Approach - use surrogate splits:

  - Suppose the best split for the node t is s, which involves a question on X_m. Find another split s' on a variable X_j, such that $j \neq m$, which is most similar to s. Similarly, the second best surrogate split, the third and so on.

  - Exploiting the correlation between predictors to try to alleviate the missing data problem.

Other Details ….

- **Why binary splits?** – we might consider multi-way splits at each step, not shown to be very useful. The data becomes fragmented too quickly, leaving insufficient data at the level below. Multi-way splits can be achieved by a series of binary splits.

- **Instability** – inherent error propagation due to the hierarchical structure of the tree. High variance and instability. Solutions – more stable split criterion will alleviate but not remove the error problem. Bagging will reduce the variation, but reduce the ability to interpret.

Other Details ….

- **Difficulty capturing additive structure** – If an additive structure exists within the data, there is no assurance that it will be captured. For example,

$$Y = c_1 I\left(X_1 < t_1\right) + c_2 I\left(X_2 < t_2\right) + \varepsilon.$$

Suppose the first split is near X_1 is near t_1, with enough data, the model may make a split at X_2 near t_2. However, the model is given no special encouragement.

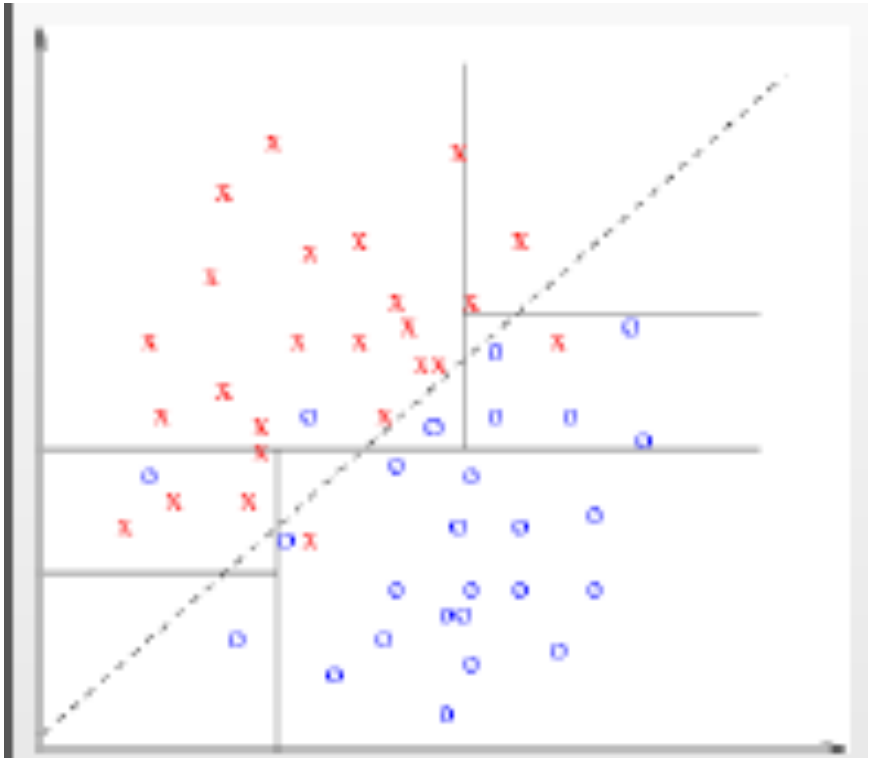
If there were 10 additive variables in the true model, this would be a more severe problem.

Other Details ….

Variable combinations –

- Splits perpendicular to the coordinate axis are not always the most optimal.
- A strategy may be to us linear combinations of the variables:

$$\text{Is } \sum a_j x_j \leq c?$$



- Computation increases.
- Model complexity increases.

Example

Example: Predicting Spam Email

The data: 460 email messages, Y binary response ("email" or "spam"), 48 quantitative predictors based on the percentage of times a particular word appears, 6 quantitative predictors concerning characters, average length of uninterrupted capitals, longest sequence of uninterrupted capitals, total sum of uninterrupted sequences of capitals.

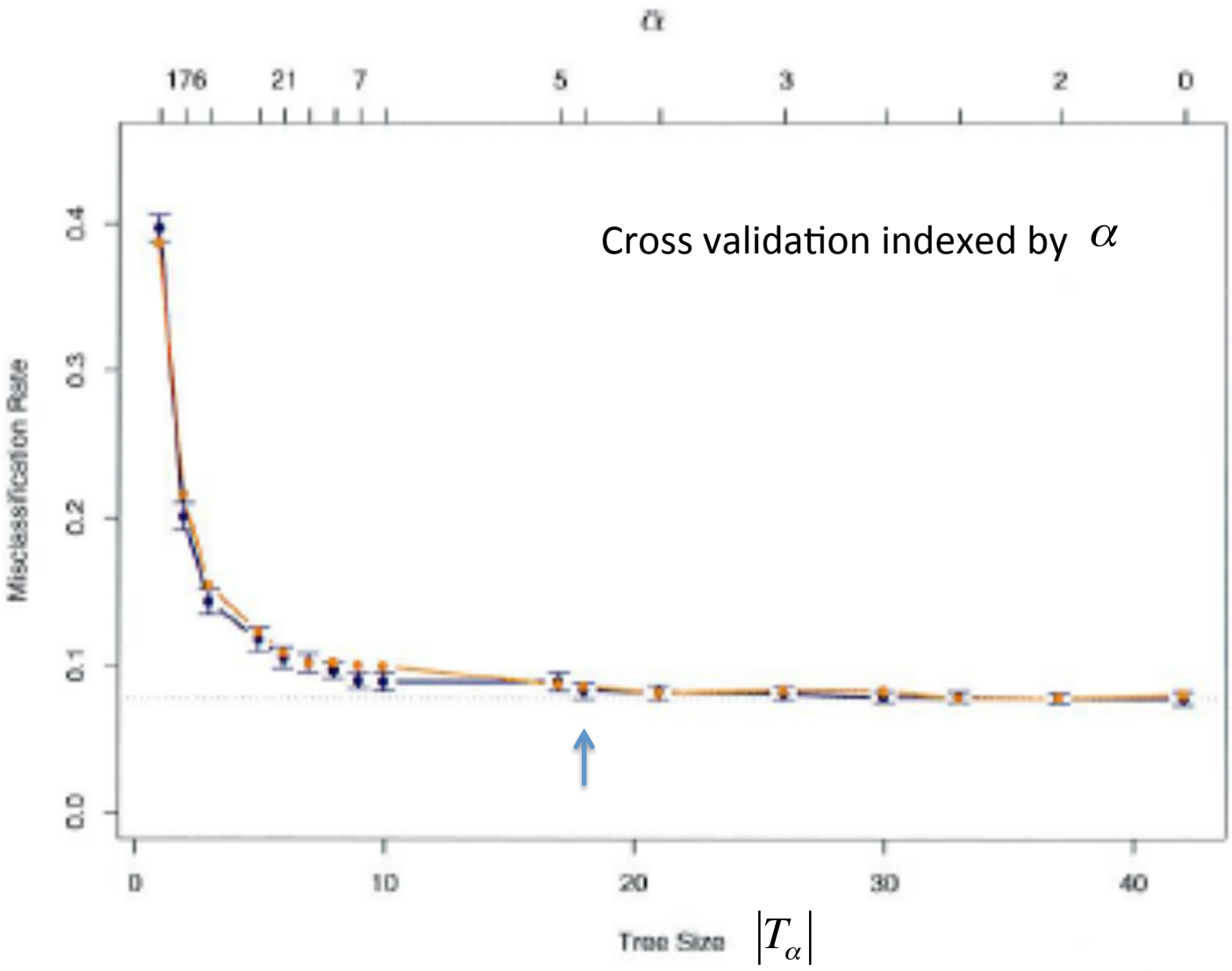
Test data: 1536 observations

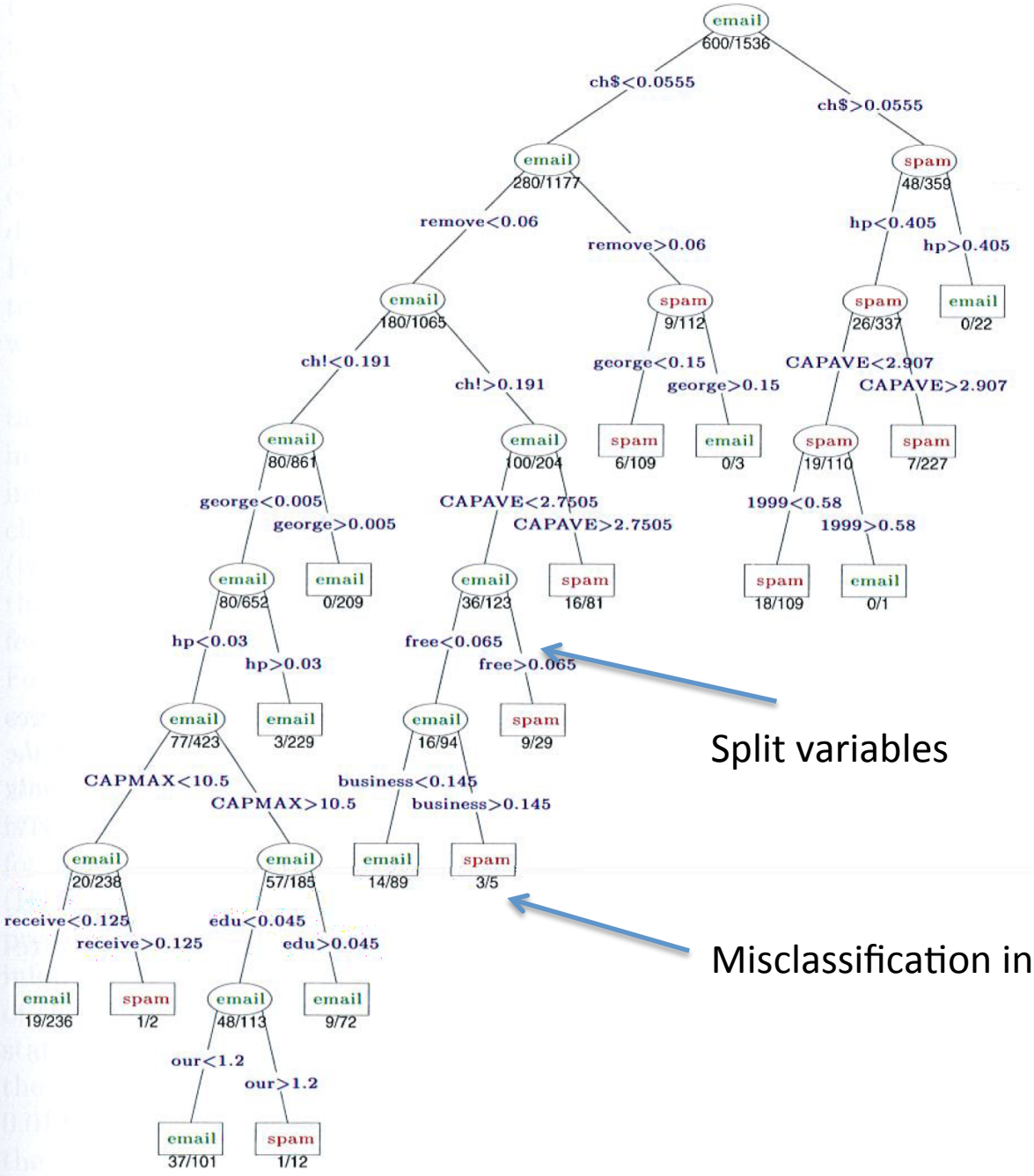
Training data: 3065 observations

Before fitting, log transformed X.

CART strategy: deviance measure to grow the tree and misclassification rate to prune it.

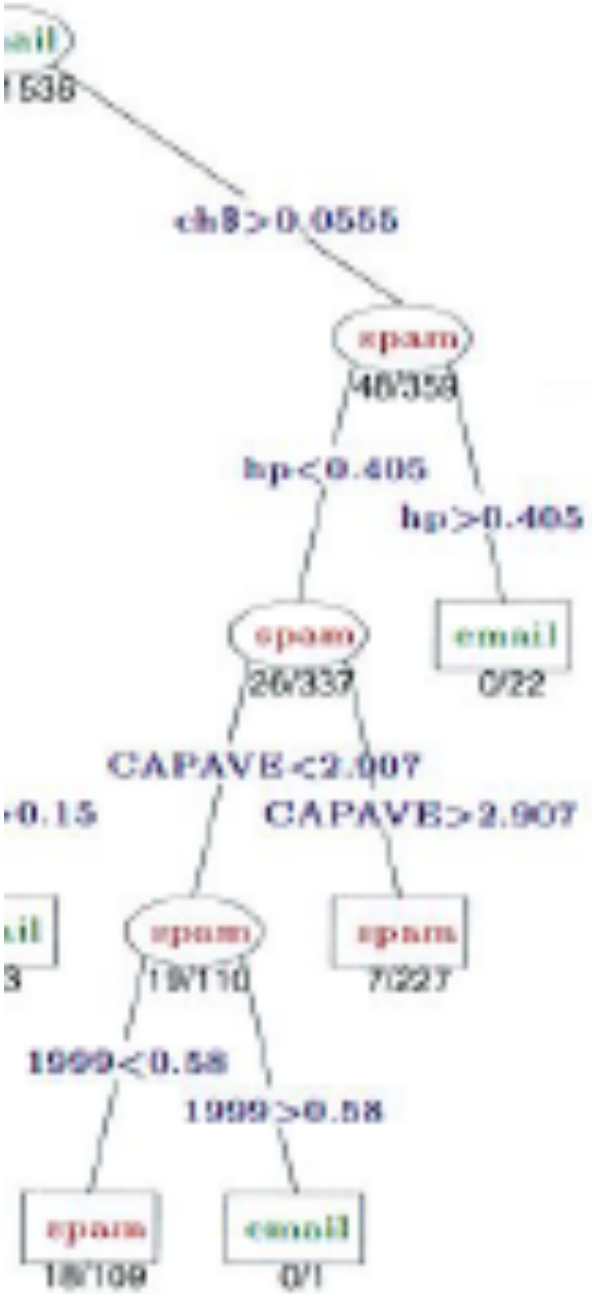
Example


α
176
21
7
5
3
2
0
Misclassification Rate
0.0
0.1
0.2
0.3
0.4
0
10
20
30
40
Tree Size
Cross validation indexed by α
$|T_\alpha|$



Split variables

Misclassification in the test data


ail
1536
ch$>0.0555
spam
46/359
hp<0.405
hp>0.405
spam
26/337
email
0/22
CAPAVE<2.907
0.15
CAPAVE>2.907
ail
3
spam
19/110
spam
7/227
1999<0.58
1999>0.58
spam
18/109
email
0/1

Example

Compared to logistic regression

Logistic

True Class	Predicted Class: email (0)	Predicted Class: spam (1)
email (0)	58.3%	2.5%
spam (1)	3.0%	36.3%

Test error: 5.5%

CART

	Email (0)	Spam (1)
Email (0)	57.3%	4.0%
Spam (1)	5.3%	33.4%

Test error: 9.3%

Example

Sensitivity: Probability of predicting a disease given true state of the disease.

Specificity: Probability of predicting non-disease given the true state is non-disease.

In our example: "spam" → "disease"

Predicted

TRUE	Email (0)	Spam (1)
Email (0)	57.3%	4.0%
Spam (1)	5.3%	33.4%

Test error: 9.3%

$$Sensitivity = 100 \times \frac{33.4}{33.4 + 5.3} = 86.3\%$$

$$Specificity = 100 \times \frac{57.3}{57.3 + 4.0} = 93.4\%$$

Example

- By modifying the losses L_{10} and L_{01} we increase the sensitivity and decrease the sensitivity of the rule, or visa-versa.
- We can achieve this by letting, e.g., $L_{01}>1$ with $L_{10}=1$.
- The Bayes' rule in each terminal node classifies to class 1 (spam) if the proportion of spam is $\geq L_{01}/(L_{10}+L_{01})$., and zero otherwise.
- Better to embed it from the start.

	Email (0)	Spam (1)
Email (0)	57.3%	4.0%
Spam (1)	5.3%	33.4%

Test error: 9.3%

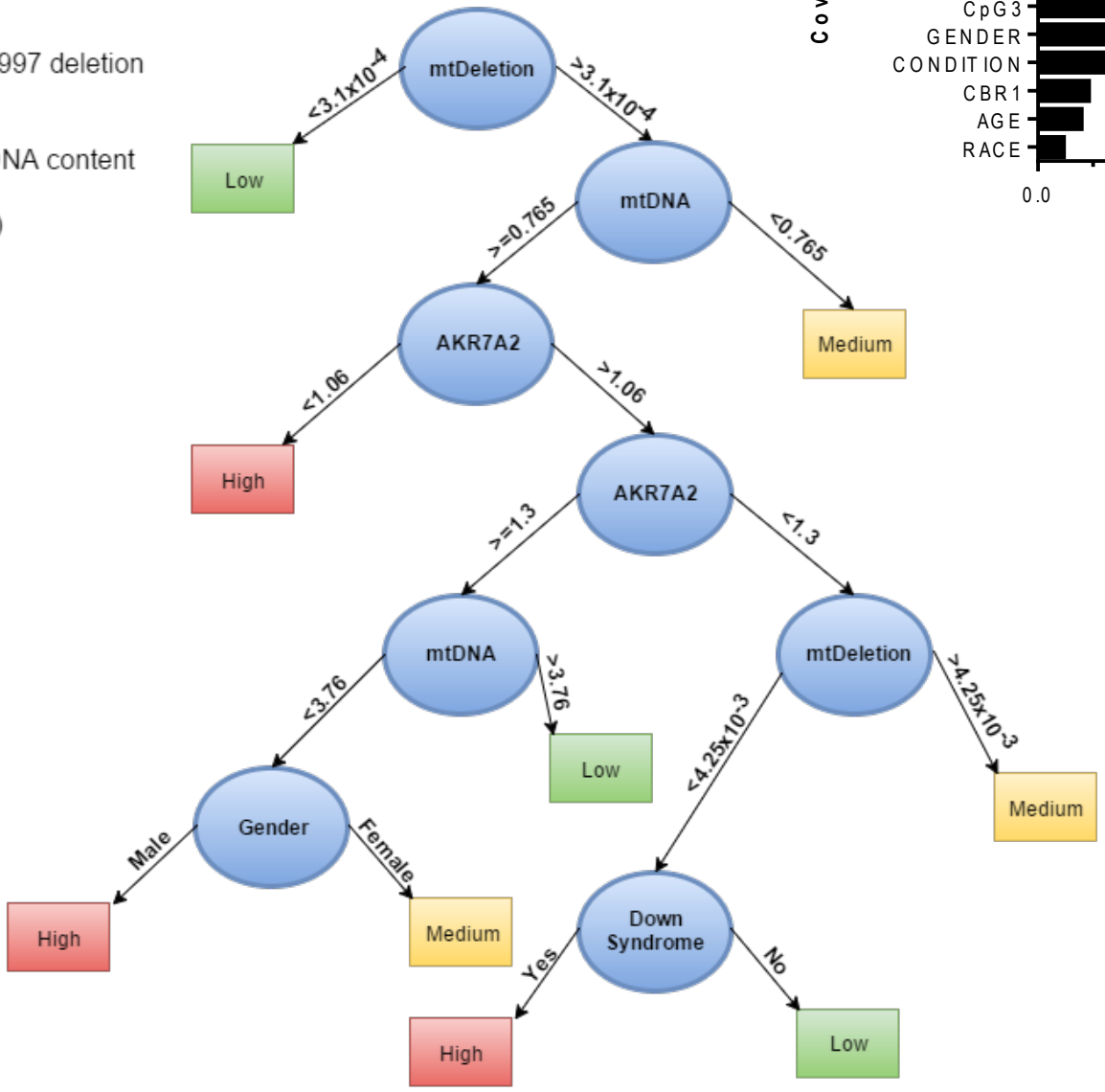
$$Sensitivity = 100 \times \frac{33.4}{33.4+5.3} = 86.3\%$$

$$Specificity = 100 \times \frac{57.3}{57.3+4.0} = 93.4\%$$

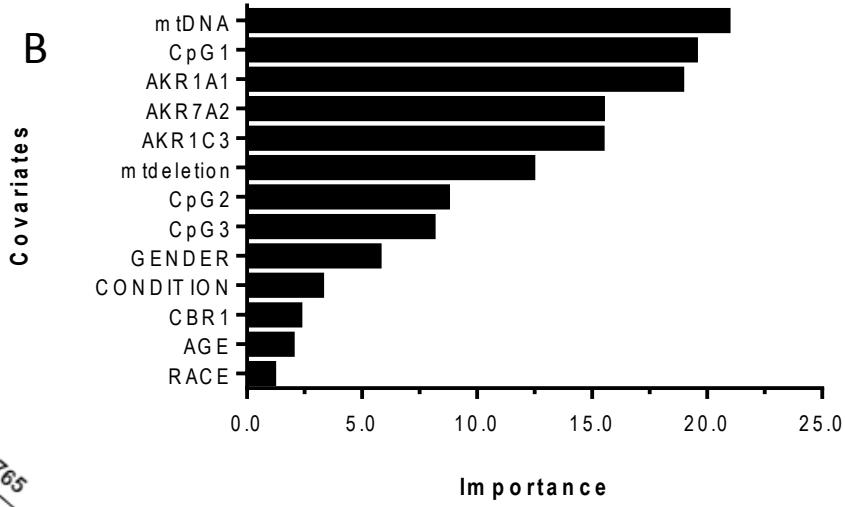
Figure 1: (A) CART model for DA (B) Variable importance for CART model.

A

mtDeletion-
Mitochondrial 4997 deletion frequency
mtDNA-
Mitochondrial DNA content
AKR7A2-
protein (nmol/g)



B



```
node), split, n, loss, yval, (yprob)
      * denotes terminal node

  1) root 44 28 Medium (0.3181818 0.3181818 0.3636364)
    2) mtdeletion< 0.0003155 7  1 Low (0.1428571 0.8571429 0.0000000) *
    3) mtdeletion>=0.0003155 37 21 Medium (0.3513514 0.2162162 0.4324324)
      6) mtDNA>=0.765 32 19 High (0.4062500 0.2500000 0.3437500)
       12) AKR7A2< 1.06 3  0 High (1.0000000 0.0000000 0.0000000) *
       13) AKR7A2>=1.06 29 18 Medium (0.3448276 0.2758621 0.3793103)
         26) AKR7A2>=1.3 13  7 High (0.4615385 0.2307692 0.3076923)
           52) mtDNA< 3.76 10  4 High (0.6000000 0.0000000 0.4000000)
            104) GENDER=0 4  0 High (1.0000000 0.0000000 0.0000000) *
            105) GENDER=1 6  2 Medium (0.3333333 0.0000000 0.6666667) *
           53) mtDNA>=3.76 3  0 Low (0.0000000 1.0000000 0.0000000) *
         27) AKR7A2< 1.3 16  9 Medium (0.2500000 0.3125000 0.4375000)
           54) mtdeletion< 0.004245 9  4 Low (0.2222222 0.5555556 0.2222222)
            108) CONDITION=1 3  1 High (0.6666667 0.3333333 0.0000000) *
            109) CONDITION=0 6  2 Low (0.0000000 0.6666667 0.3333333) *
           55) mtdeletion>=0.004245 7  2 Medium (0.2857143 0.0000000 0.7142857) *
      7) mtDNA< 0.765 5  0 Medium (0.0000000 0.0000000 1.0000000) *
```

Figure 2. A screenshot of the tree splitting for the DA decision tree.

Node number 1: Primary splits:
mtdeletion < 0.0003155 to the left, improve=6.115836, (0 missing)
AKR1C3 < 0.45 to the left, improve=5.225031, (5 missing)
AKR1A1 < 1.235 to the right, improve=5.158927, (8 missing)
CpG1 < 0.23 to the right, improve=4.392002, (3 missing)
AKR7A2 < 1.06 to the left, improve=4.227505, (12 missing)
Surrogate splits:
AGE < 28 to the left, agree=0.955, adj=0.714, (0 split)

Node number 3: 37 observations
Primary splits:
mtDNA < 0.765 to the right, improve=4.715842, (0 missing)
AKR7A2 < 1.06 to the left, improve=4.005800, (11 missing)
CpG1 < 0.23 to the right, improve=3.778485, (2 missing)
AKR1C3 < 0.45 to the left, improve=3.624650, (5 missing)
AKR1A1 < 1.185 to the right, improve=3.466523, (6 missing)

Advantages of the tree structure approach

- Handles both categorical and ordered variables in a simple and natural way.
- Automatic stepwise variable selection and complexity reduction.
- It provides an estimate of the misclassification rate for a query sample.
- It is invariant under all monotone transformations of individual ordered variables.
- Robust to outliers and misclassified points in the training set.
- Easy to interpret.