Statistical Learning– MATH 6333
Set 6 (Tree-Based Methods)

Tamer Oraby

UTRGV

tamer.oraby@utrgv.edu

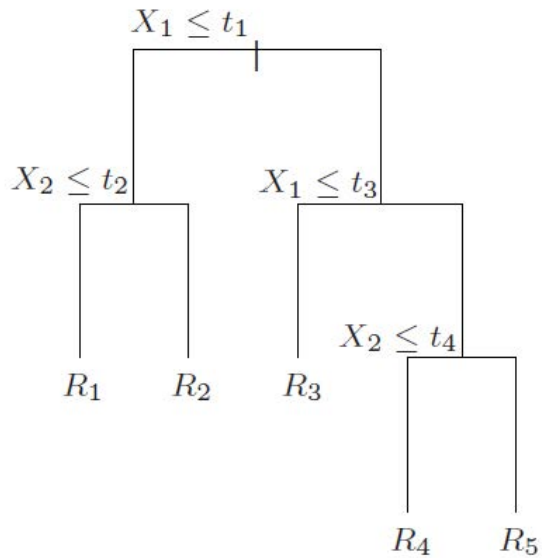
* Last updated November 10, 2021

Tree-Based Methods

Classification and Regression Tree (CART)

- Another method for regression (Y is continuous) and classification (Y is categorical).
- It produces recursively a binary partition of the input space with constant predicted outputs c_m for R_m.
- The predicted regression surface

$$\hat{f}(x) = \sum_{m=1}^{5} \hat{c}_m I((X_1, X_2) \in R_m)$$

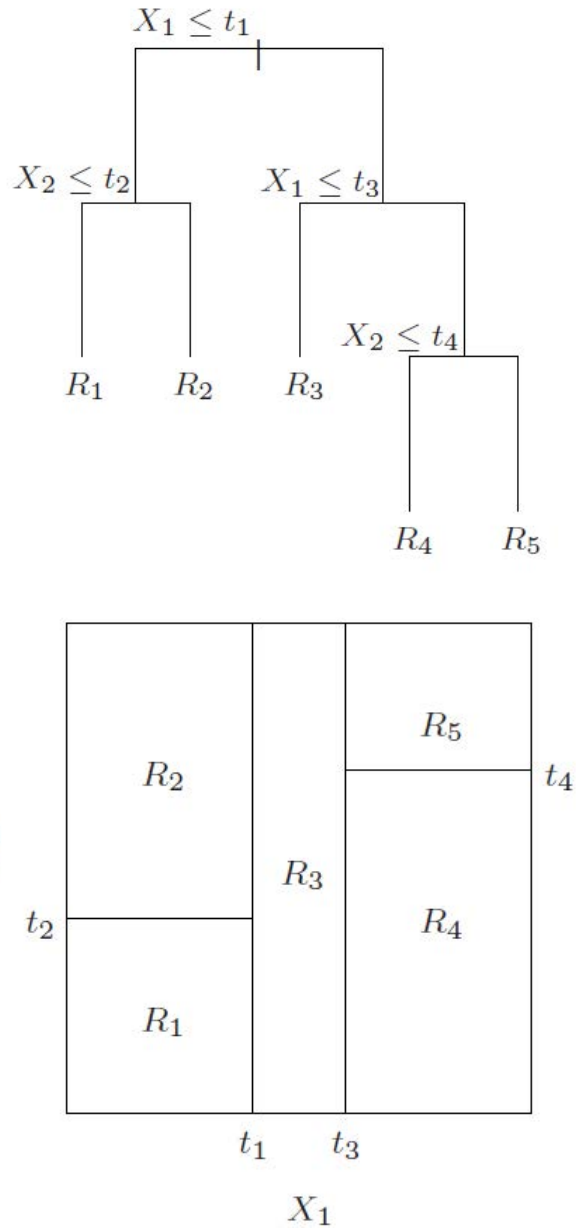


Tree-Based Methods

Classification and Regression Tree (CART)

- Another method for regression (Y is continuous) and classification (Y is categorical).
- It produces recursively a binary partition of the input space with constant predicted outputs c_m for R_m.
- The predicted regression surface

$$\hat{f}(x) = \sum_{m=1}^{5} \hat{c}_m I((X_1, X_2) \in R_m)$$

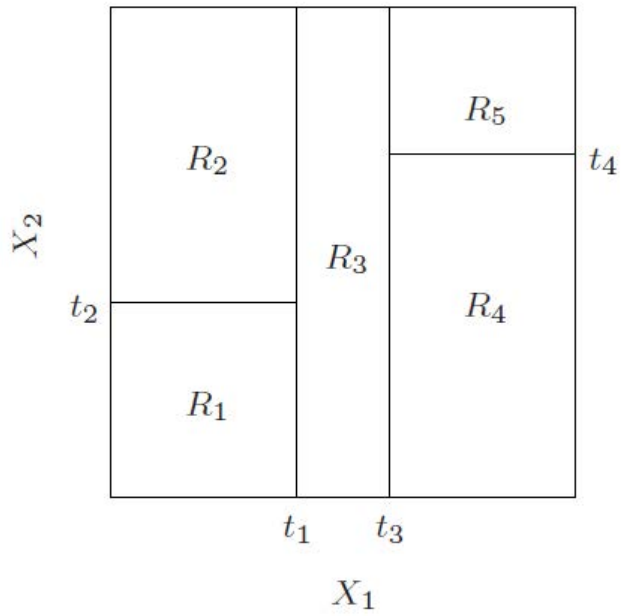


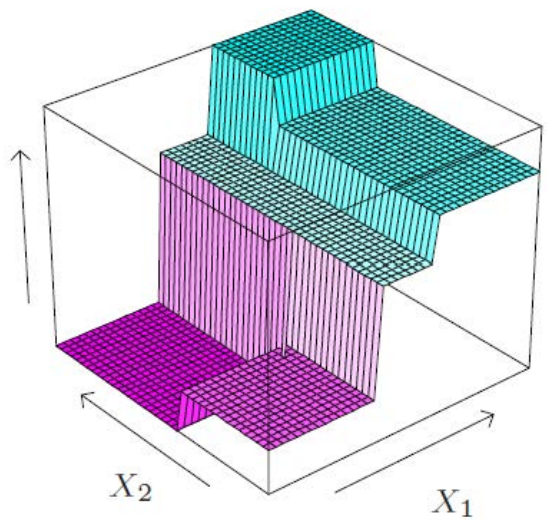
Tree-Based Methods

Classification and Regression Tree (CART)

- Another method for regression (Y is continuous) and classification (Y is categorical).
- It produces recursively a binary partition of the input space with constant predicted outputs c_m for R_m.
- The predicted regression surface

$$\hat{f}(x) = \sum_{m=1}^{5} \hat{c}_m I((X_1, X_2) \in R_m)$$





Regression Trees

Regression Trees

- If Y is continuous, then $\hat{c} = \{\hat{c}_m, m = 1, 2, \ldots, M\}$ could be found using the method of least squares by minimizing

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

- It results in

$$\hat{c}_m = average(y_i | x_i \in R_m) = \frac{1}{N_m} \sum_{i: x_i \in R_m} y_i$$

  where $N_m = |\{i : x_i \in R_m\}|$

- But, finding the binary partition $\{R_m, m = 1, 2, \ldots, M\}$ is computationally infeasible.
- Unless, we use a greedy (short-sighted) algorithm to "grow a regression tree top-down."

Regression Trees

- If Y is continuous, then $\hat{c} = \{\hat{c}_m, m = 1, 2, \ldots, M\}$ could be found using the method of least squares by minimizing

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

- It results in

$$\hat{c}_m = average(y_i | x_i \in R_m) = \frac{1}{N_m} \sum_{i:x_i \in R_m} y_i$$

where $N_m = |\{i : x_i \in R_m\}|$

- But, finding the binary partition $\{R_m, m = 1, 2, \ldots, M\}$ is computationally infeasible.
- Unless, we use a greedy (short-sighted) algorithm to "grow a regression tree top-down."

Regression Trees

- If Y is continuous, then $\hat{c} = \{\hat{c}_m, m = 1, 2, \ldots, M\}$ could be found using the method of least squares by minimizing

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

- It results in

$$\hat{c}_m = average(y_i | x_i \in R_m) = \frac{1}{N_m} \sum_{i:x_i \in R_m} y_i$$

  where $N_m = |\{i : x_i \in R_m\}|$

- But, finding the binary partition $\{R_m, m = 1, 2, \ldots, M\}$ is computationally infeasible.

- Unless, we use a greedy (short-sighted) algorithm to "grow a regression tree top-down."

Regression Trees

- If Y is continuous, then $\hat{c} = \{\hat{c}_m, m = 1, 2, \ldots, M\}$ could be found using the method of least squares by minimizing

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

- It results in

$$\hat{c}_m = average(y_i | x_i \in R_m) = \frac{1}{N_m} \sum_{i:x_i \in R_m} y_i$$

  where $N_m = |\{i : x_i \in R_m\}|$

- But, finding the binary partition $\{R_m, m = 1, 2, \ldots, M\}$ is computationally infeasible.
- Unless, we use a greedy (short-sighted) algorithm to "grow a regression tree top-down."

Regression Trees

Recursive binary splitting

- selecting variable X_j and cut-off point s that define the two regions

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

- Then find optimal j, s using that minimize RSS

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

- Internally, for the optimal j, s,

$$\hat{c}_1 = average(y_i | x_i \in R_1(j,s))$$

and

$$\hat{c}_2 = average(y_i | x_i \in R_2(j,s))$$

Regression Trees

Recursive binary splitting

- selecting variable X_j and cut-off point s that define the two regions

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

- Then find optimal j, s using that minimize RSS

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

- Internally, for the optimal j, s,

$$\hat{c}_1 = average(y_i | x_i \in R_1(j,s))$$

and

$$\hat{c}_2 = average(y_i | x_i \in R_2(j,s))$$

Regression Trees

Recursive binary splitting

- selecting variable X_j and cut-off point s that define the two regions

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

- Then find optimal j, s using that minimize RSS

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

- Internally, for the optimal j, s,

$$\hat{c}_1 = average(y_i | x_i \in R_1(j,s))$$

and

$$\hat{c}_2 = average(y_i | x_i \in R_2(j,s))$$

Regression Trees

Recursive binary splitting

The algorithm continues by

- dividing one of the resulting regions into further two divided regions via similar optimization problem and then
- further divide one of the resulting three regions and so on and so forth till there is no more than 5 observations in each region.
- But, when to stop growing the tree? (Large trees lead to overfitting, and small trees are less effective.)
- May be using a threshold for RSS below which the algorithm stops splitting. But early sub-optimal stopping is possible.
- The answer is by stopping and pruning trees.

Regression Trees

Recursive binary splitting

The algorithm continues by

- dividing one of the resulting regions into further two divided regions via similar optimization problem and then
- further divide one of the resulting three regions and so on and so forth till there is no more than 5 observations in each region.
- But, when to stop growing the tree? (Large trees lead to overfitting, and small trees are less effective.)
- May be using a threshold for RSS below which the algorithm stops splitting. But early sub-optimal stopping is possible.
- The answer is by stopping and pruning trees.

Regression Trees

Recursive binary splitting

The algorithm continues by

- dividing one of the resulting regions into further two divided regions via similar optimization problem and then
- further divide one of the resulting three regions and so on and so forth till there is no more than 5 observations in each region.
- But, when to stop growing the tree? (Large trees lead to overfitting, and small trees are less effective.)
- May be using a threshold for RSS below which the algorithm stops splitting. But early sub-optimal stopping is possible.
- The answer is by stopping and pruning trees.

Regression Trees

Recursive binary splitting

The algorithm continues by

- dividing one of the resulting regions into further two divided regions via similar optimization problem and then
- further divide one of the resulting three regions and so on and so forth till there is no more than 5 observations in each region.
- But, when to stop growing the tree? (Large trees lead to overfitting, and small trees are less effective.)
- May be using a threshold for RSS below which the algorithm stops splitting. But early sub-optimal stopping is possible.
- The answer is by stopping and pruning trees.

Regression Trees

Recursive binary splitting

The algorithm continues by

- dividing one of the resulting regions into further two divided regions via similar optimization problem and then
- further divide one of the resulting three regions and so on and so forth till there is no more than 5 observations in each region.
- But, when to stop growing the tree? (Large trees lead to overfitting, and small trees are less effective.)
- May be using a threshold for RSS below which the algorithm stops splitting. But early sub-optimal stopping is possible.
- The answer is by stopping and pruning trees.

Regression Trees

Tree Pruning

Stopping and pruning goes by

- growing a tree T_0 and stop splitting when a selected minimum node size is reached.
- Prune the tree using a cost-complexity pruning, *aka* weakest link pruning. (Pruning works backward by collapsing internal (non-terminal) nodes back to get a subtree $T \subset T_0$.) Let $|T|$ be the number of terminal nodes in T.
- Find tuning parameter $\alpha \geq 0$ (by CV) and the subtree $T_\alpha \subset T_0$ that minimize the cost complexity criterion

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

where $\hat{c}_m = \frac{1}{N_m} \sum_{i:x_i \in R_m} y_i$ and $N_m = |\{i : x_i \in R_m\}|$.

Regression Trees

Tree Pruning

Stopping and pruning goes by

- growing a tree T_0 and stop splitting when a selected minimum node size is reached.
- Prune the tree using a cost-complexity pruning, *aka* weakest link pruning. (Pruning works backward by collapsing internal (non-terminal) nodes back to get a subtree $T \subset T_0$.) Let $|T|$ be the number of terminal nodes in T.
- Find tuning parameter $\alpha \geq 0$ (by CV) and the subtree $T_\alpha \subset T_0$ that minimize the cost complexity criterion

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

where $\hat{c}_m = \frac{1}{N_m} \sum_{i:x_i \in R_m} y_i$ and $N_m = |\{i : x_i \in R_m\}|$.

Regression Trees

Tree Pruning

Stopping and pruning goes by

- growing a tree T_0 and stop splitting when a selected minimum node size is reached.
- Prune the tree using a cost-complexity pruning, *aka* weakest link pruning. (Pruning works backward by collapsing internal (non-terminal) nodes back to get a subtree $T \subset T_0$.) Let $|T|$ be the number of terminal nodes in T.
- Find tuning parameter $\alpha \geq 0$ (by CV) and the subtree $T_\alpha \subset T_0$ that minimize the cost complexity criterion

$$C_\alpha(T) = \sum_{m=1}^{|T|} \sum_{i:x_i \in R_m} (y_i - \hat{c}_m)^2 + \alpha|T|$$

where $\hat{c}_m = \frac{1}{N_m} \sum_{i:x_i \in R_m} y_i$ and $N_m = |\{i : x_i \in R_m\}|$.

Regression Trees

Tree Pruning

Stopping and pruning goes by

- growing a tree T_0 and stop splitting when a selected minimum node size is reached.
- Prune the tree using a cost-complexity pruning, *aka* weakest link pruning. (Pruning works backward by collapsing internal (non-terminal) nodes back to get a subtree $T \subset T_0$.) Let $|T|$ be the number of terminal nodes in T.
- Find tuning parameter $\alpha \geq 0$ (by CV) and the subtree $T_\alpha \subset T_0$ that minimize the cost complexity criterion

$$C_\alpha(T) = \sum_{m=1}^{|T|} \overbrace{\sum_{i:x_i \in R_m} (y_i - \hat{c}_m)^2}^{N_m Q_m(T)} + \alpha |T|$$

where $\hat{c}_m = \frac{1}{N_m} \sum_{i:x_i \in R_m} y_i$ and $N_m = |\{i : x_i \in R_m\}|$.

Regression Trees

Tree Pruning

Stopping and pruning goes by

- growing a tree T_0 and stop splitting when a selected minimum node size is reached.
- Prune the tree using a cost-complexity pruning, *aka* weakest link pruning. (Pruning works backward by collapsing internal (non-terminal) nodes back to get a subtree $T \subset T_0$.) Let $|T|$ be the number of terminal nodes in T.
- Find tuning parameter $\alpha \geq 0$ (by CV) and the subtree $T_\alpha \subset T_0$ that minimize the cost complexity criterion

$$C_\alpha(T) = \sum_{m=1}^{|T|} \overbrace{\sum_{i:x_i \in R_m} (y_i - \hat{c}_m)^2}^{N_m Q_m(T) \text{ impurity measure}} + \alpha|T|$$

where $\hat{c}_m = \frac{1}{N_m} \sum_{i:x_i \in R_m} y_i$ and $N_m = |\{i : x_i \in R_m\}|$.

Regression Trees

Tree Pruning

What happens when α is small or large in

$$C_\alpha(T) = \sum_{m=1}^{|T|} \sum_{i:x_i \in R_m} (y_i - \hat{c}_m)^2 + \alpha|T|$$

?

At $\alpha = 0$, the results is T_0.

Regression Trees

Tree Pruning

What happens when α is small or large in

$$C_\alpha(T) = \sum_{m=1}^{|T|} \sum_{i:x_i \in R_m} (y_i - \hat{c}_m)^2 + \alpha|T|$$

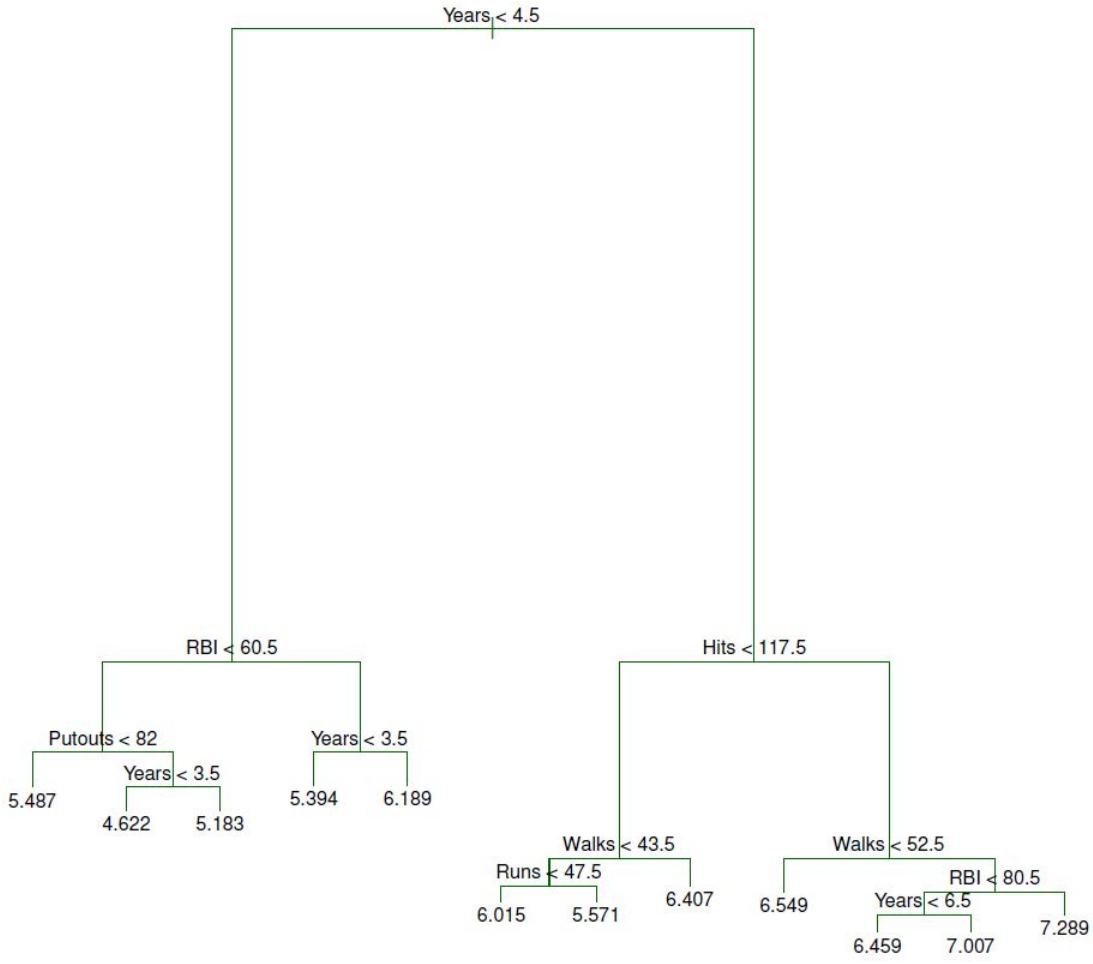
?

At $\alpha = 0$, the results is T_0.

Regression Trees

Example (Hitters data)

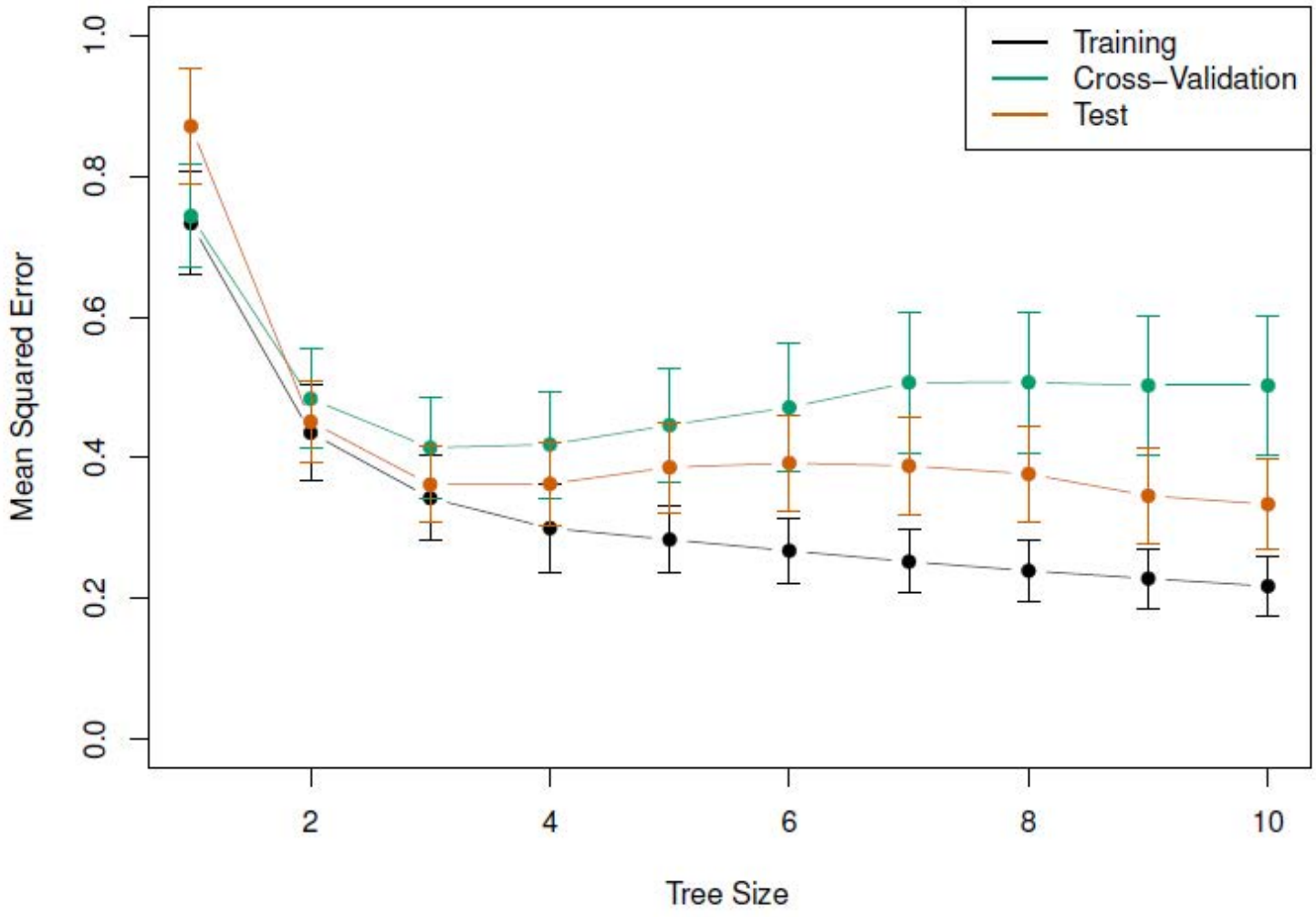
To predict baseball player's Salary ($Y = \log(\text{Salary}/1,000)$) based on Years in a major league and previous year's number of Hits, etc., RT gives T_0



Regression Trees

Example (Hitters data)

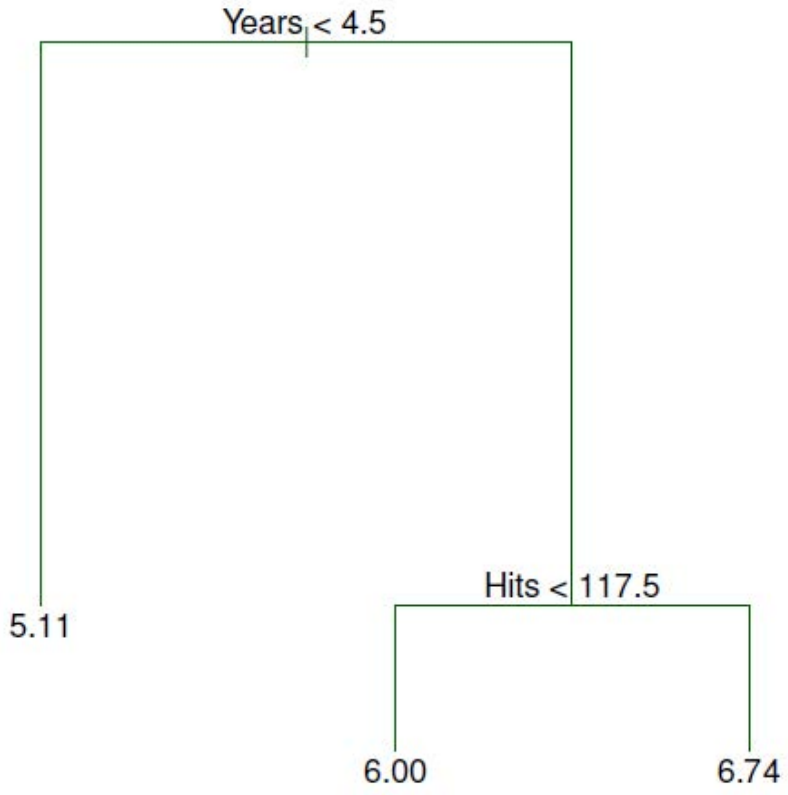
A cross-validation will find the optimal tree size $|T| = 3$ (number of terminal nodes.)

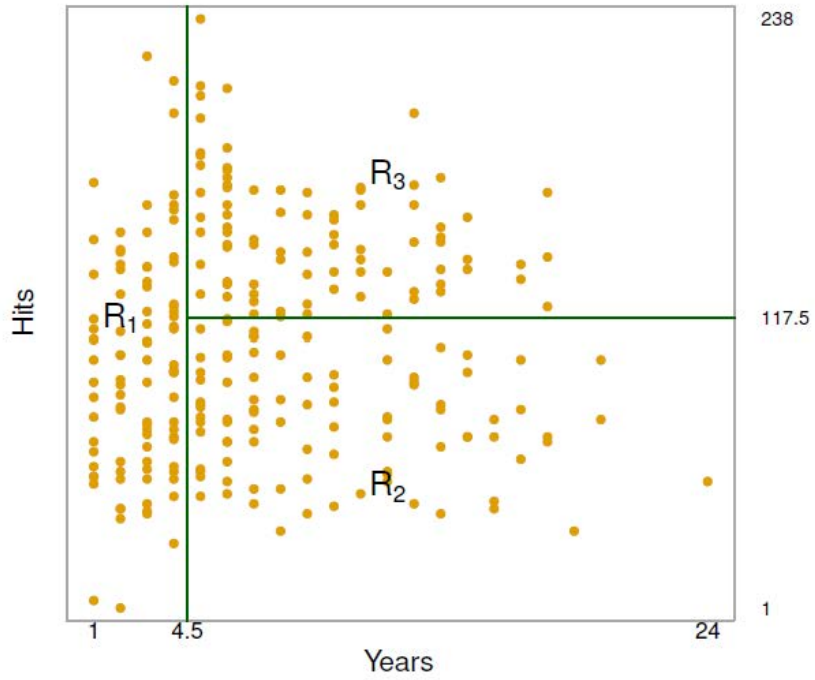


Regression Trees

Example (Hitters data)

T_α is





Classification Trees

Classification Trees

- If Y is categorical with K classes, then the class proportion $\hat{p} = \{\hat{p}_{mk}, m = 1, 2, \ldots, M \text{ and } k = 1, 2, \ldots, K\}$ could be found to be

$$\hat{p}_{mk} = \frac{1}{N_m} \sum_{i: x_i \in R_m} I(y_i = k)$$

and

$$k(m) = argmax_k \, \hat{p}_{mk}$$

- They are found by minimizing objective functions that include different measures of impurity $Q_m(T)$
  1. Misclassification error: $\frac{1}{N_m} \sum_{i: x_i \in R_m} I(y_i \neq k) = 1 - max_k \hat{p}_{mk}$
  2. Gini index (total variance): $\sum_{k \neq k'} \hat{p}_{mk} \hat{p}_{mk'} = \sum_{k=1}^{K} \hat{p}_{mk}(1 - \hat{p}_{mk})$ as a measure of purity which is small if the mth node is pure.
  3. Cross-entropy (deviance): $-\sum_{k=1}^{K} \hat{p}_{mk} \log(\hat{p}_{mk})$ which is small if the mth node is pure.

Classification Trees

- If Y is categorical with K classes, then the class proportion $\hat{p} = \{\hat{p}_{mk}, m = 1, 2, \ldots, M \text{ and } k = 1, 2, \ldots, K\}$ could be found to be

$$\hat{p}_{mk} = \frac{1}{N_m} \sum_{i: x_i \in R_m} I(y_i = k)$$

and

$$k(m) = argmax_k \, \hat{p}_{mk}$$

- They are found by minimizing objective functions that include different measures of impurity $Q_m(T)$

1. Misclassification error: $\frac{1}{N_m} \sum_{i: x_i \in R_m} I(y_i \neq k) = 1 - max_k \hat{p}_{mk}$
2. Gini index (total variance): $\sum_{k \neq k'} \hat{p}_{mk} \hat{p}_{mk'} = \sum_{k=1}^{K} \hat{p}_{mk}(1 - \hat{p}_{mk})$ as a measure of purity which is small if the mth node is pure.
3. Cross-entropy (deviance): $-\sum_{k=1}^{K} \hat{p}_{mk} \log(\hat{p}_{mk})$ which is small if the mth node is pure.

Classification Trees

- If Y is categorical with K classes, then the class proportion $\hat{p} = \{\hat{p}_{mk}, m = 1, 2, \ldots, M \text{ and } k = 1, 2, \ldots, K\}$ could be found to be

$$\hat{p}_{mk} = \frac{1}{N_m} \sum_{i: x_i \in R_m} I(y_i = k)$$

and

$$k(m) = argmax_k \, \hat{p}_{mk}$$

- They are found by minimizing objective functions that include different measures of impurity $Q_m(T)$

1. Misclassification error: $\frac{1}{N_m} \sum_{i: x_i \in R_m} I(y_i \neq k) = 1 - max_k \hat{p}_{mk}$
2. Gini index (total variance): $\sum_{k \neq k'} \hat{p}_{mk} \hat{p}_{mk'} = \sum_{k=1}^{K} \hat{p}_{mk}(1 - \hat{p}_{mk})$ as a measure of purity which is small if the mth node is pure.
3. Cross-entropy (deviance): $-\sum_{k=1}^{K} \hat{p}_{mk} \log(\hat{p}_{mk})$ which is small if the mth node is pure.

Classification Trees

- If Y is categorical with K classes, then the class proportion $\hat{p} = \{\hat{p}_{mk}, m = 1, 2, \ldots, M \text{ and } k = 1, 2, \ldots, K\}$ could be found to be

$$\hat{p}_{mk} = \frac{1}{N_m} \sum_{i: x_i \in R_m} I(y_i = k)$$

and

$$k(m) = argmax_k \, \hat{p}_{mk}$$

- They are found by minimizing objective functions that include different measures of impurity $Q_m(T)$

1. Misclassification error: $\frac{1}{N_m} \sum_{i: x_i \in R_m} I(y_i \neq k) = 1 - max_k \hat{p}_{mk}$
2. Gini index (total variance): $\sum_{k \neq k'} \hat{p}_{mk} \hat{p}_{mk'} = \sum_{k=1}^{K} \hat{p}_{mk}(1 - \hat{p}_{mk})$ as a measure of purity which is small if the mth node is pure.
3. Cross-entropy (deviance): $-\sum_{k=1}^{K} \hat{p}_{mk} \log(\hat{p}_{mk})$ which is small if the mth node is pure.

Classification Trees

- If Y is categorical with K classes, then the class proportion $\hat{p} = \{\hat{p}_{mk}, m = 1, 2, \ldots, M \text{ and } k = 1, 2, \ldots, K\}$ could be found to be

$$\hat{p}_{mk} = \frac{1}{N_m} \sum_{i: x_i \in R_m} I(y_i = k)$$

and

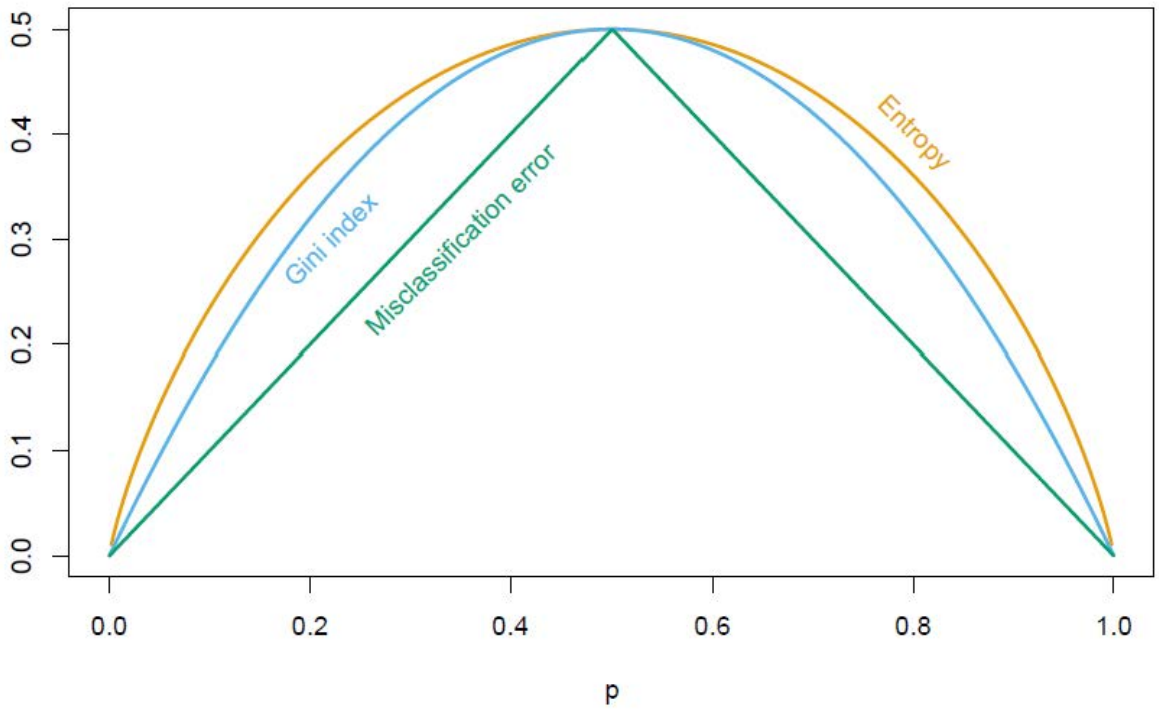
$$k(m) = argmax_k \, \hat{p}_{mk}$$

- They are found by minimizing objective functions that include different measures of impurity $Q_m(T)$

1. Misclassification error: $\frac{1}{N_m} \sum_{i: x_i \in R_m} I(y_i \neq k) = 1 - max_k \hat{p}_{mk}$
2. Gini index (total variance): $\sum_{k \neq k'} \hat{p}_{mk} \hat{p}_{mk'} = \sum_{k=1}^{K} \hat{p}_{mk}(1 - \hat{p}_{mk})$ as a measure of purity which is small if the mth node is pure.
3. Cross-entropy (deviance): $-\sum_{k=1}^{K} \hat{p}_{mk} \log(\hat{p}_{mk})$ which is small if the mth node is pure.

Classification Trees

Two class impurity functions with scaled cross-entropy function to go through (.5, .5)



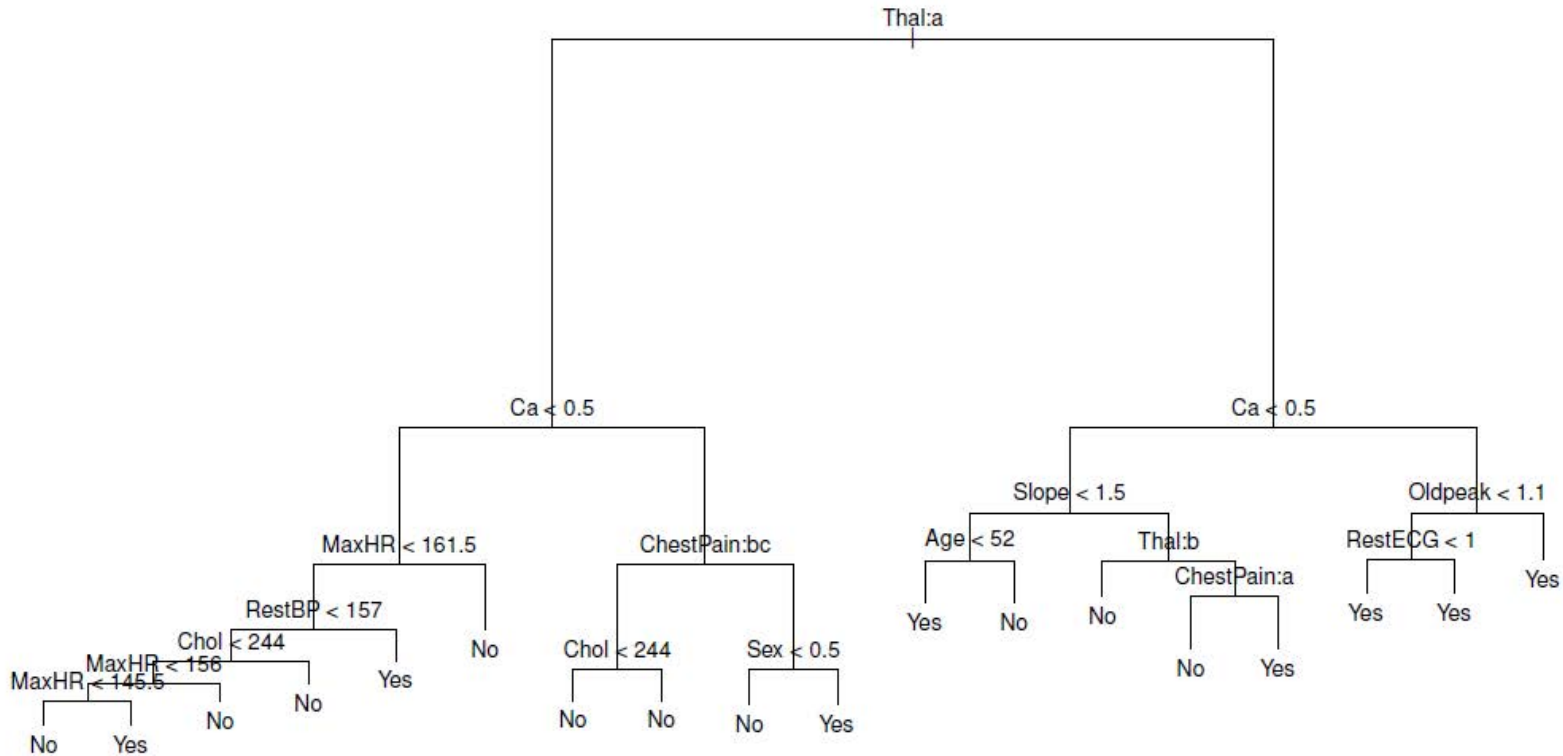
The functions are

1. Misclassification error: $1 - max(p, 1 - p)$
2. Gini index (total variance): $2p(1 - p)$
3. Cross-entropy (deviance): $-p \log(p) - (1 - p) \log(1 - p)$

Classification Trees

Example (Another Heart data)

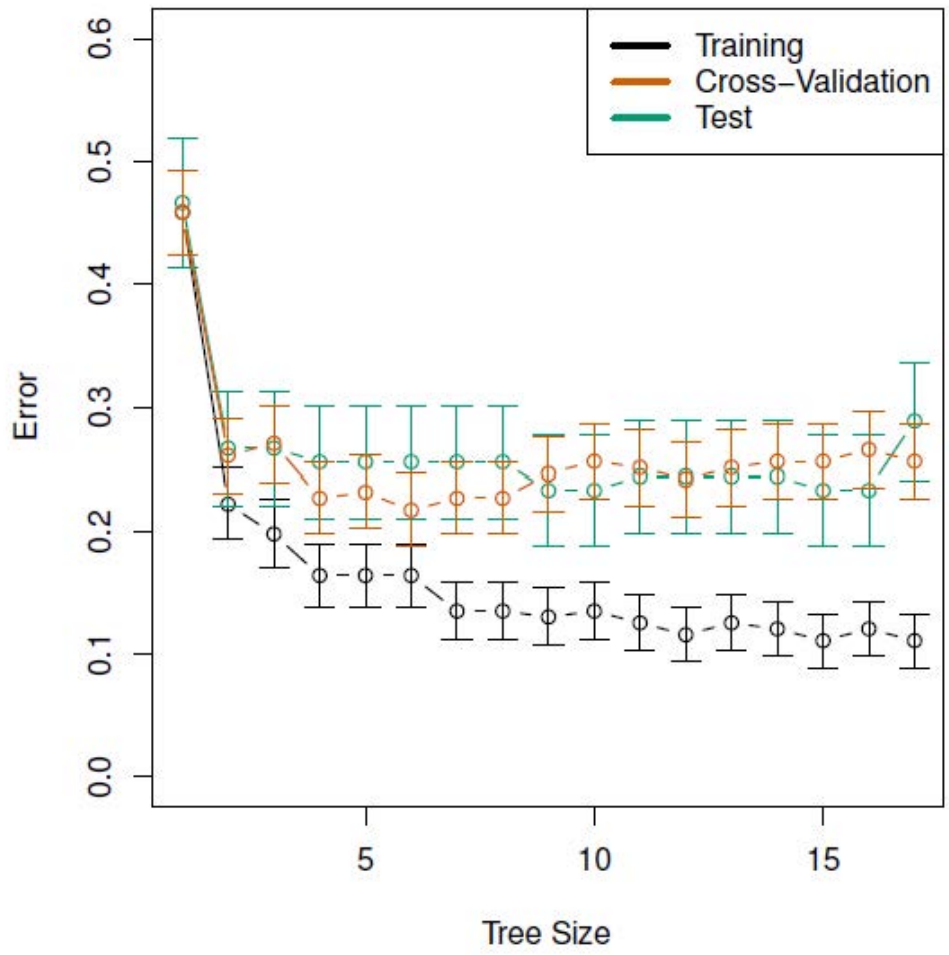
To predict heart disease HD (Y =Yes or No) based on 13 predictors Age, Sex, Chol, Thal,ChestPain, etc., CT gives T_0



Classification Trees

Example (Another Heart data)

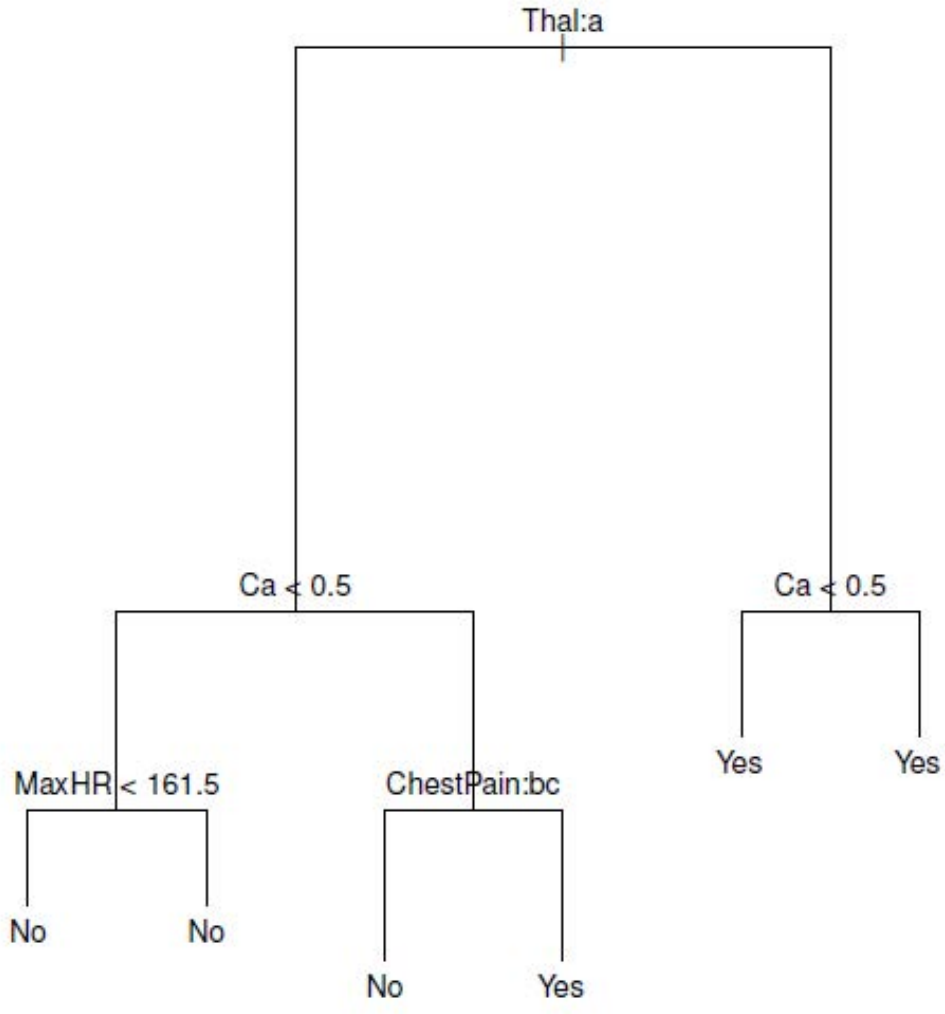
A cross-validation will find the optimal tree size $|T| = 6$ (number of terminal nodes.)



Classification Trees

Example (Another Heart data)

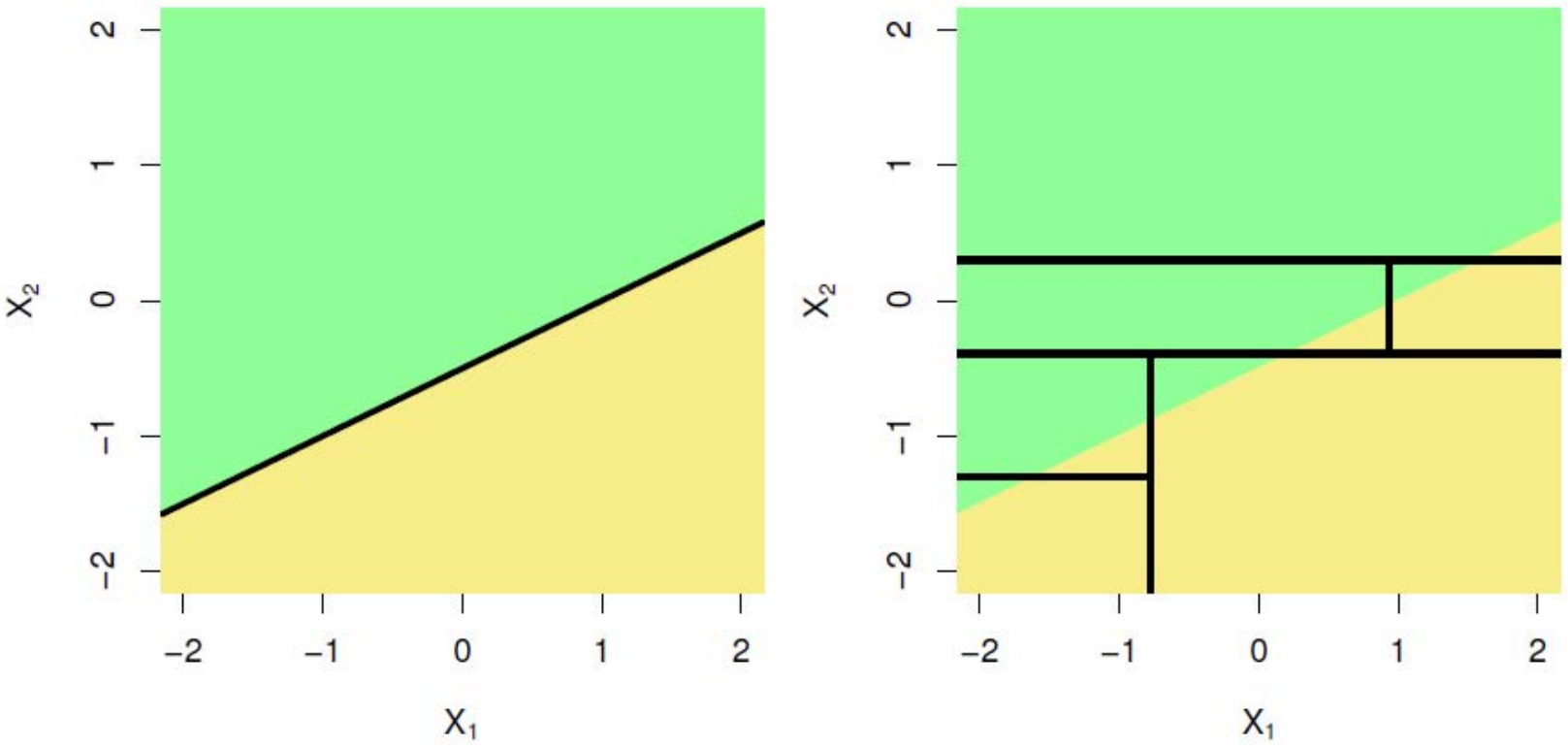
T_α is



Tree-based methods vs. Linear models

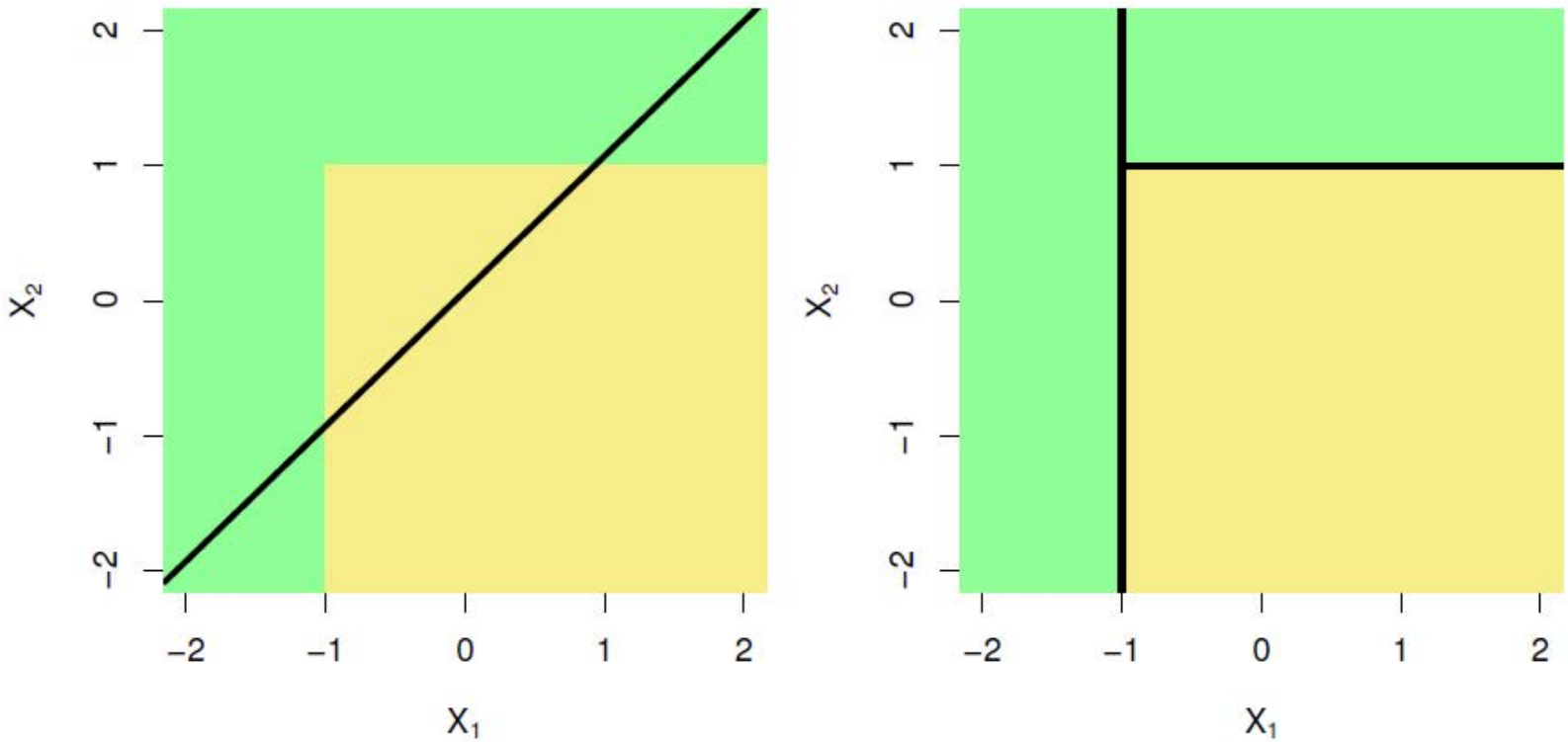
Tree-based Methods

True linear decision boundary



Tree-based Methods

True non-linear decision boundary



Tree-based Methods

Trees

- ⇑ easy to interpret
- ⇑ visually re-presentable
- ⇑ seem to resemble human decision making
- ⇑ handle categorical variables without dummy variables
- ⇓ less predictive accuracy
- ⇓ sensitive to slight changes in the data (not robust)

Tree-based Methods

DIY in R

1. Carry out a regression tree for the prostate cancer data using library(tree)
2. Carry out a classification tree for the SA hearth disease data using library(tree)

Please study the different methods in the ISL book.

Tree-based Methods

DIY in R

1. Carry out a regression tree for the prostate cancer data using library(tree)
2. Carry out a classification tree for the SA hearth disease data using library(tree)

Please study the different methods in the ISL book.

Tree-based Methods

DIY in R

1. Carry out a regression tree for the prostate cancer data using library(tree)
2. Carry out a classification tree for the SA hearth disease data using library(tree)

Please study the different methods in the ISL book.

Tree-based Methods

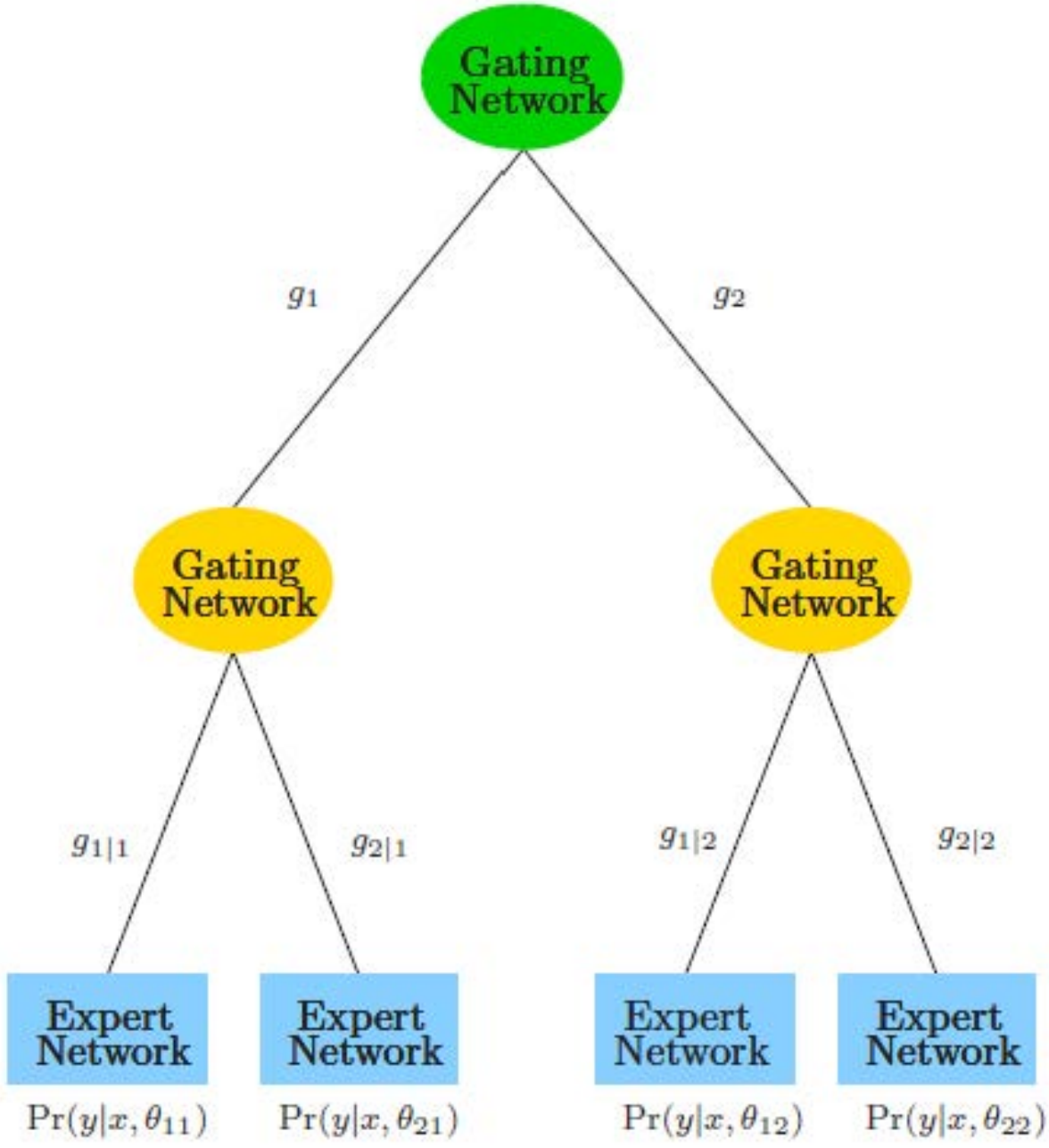
DIY in R

1. Carry out a regression tree for the prostate cancer data using library(tree)
2. Carry out a classification tree for the SA hearth disease data using library(tree)

Please study the different methods in the ISL book.

Hierarchical Mixtures of Experts (HME)

Hierarchical Mixtures of Experts



Hierarchical Mixtures of Experts

- where $g_j(x, \gamma_j)$ is a softmax function in x with parameters γ_j
- and $g_{\ell|j}(x, \gamma_{j\ell})$ is another softmax function in x with parameters $\gamma_{j\ell}$
- at the terminal the output $Y \sim P(y|x, \theta_{j\ell})$
  - Regression: P is normal with its parameters
  - Classification: P is the logistic CDF
- Then the mixture probability of the output is

$$P(y|x, \Psi) = \sum_{j=1}^{K} g_j(x, \gamma_j) \sum_{\ell=1}^{K} g_{\ell|j}(x, \gamma_{j\ell}) P(y|x, \theta_{j\ell})$$

- where $\Psi = (\gamma_j, \gamma_{j\ell}, \theta_{j\ell})$ is estimated using maximum likelihood methods and EM algorithm.

Hierarchical Mixtures of Experts

- where $g_j(x, \gamma_j)$ is a softmax function in x with parameters γ_j
- and $g_{\ell|j}(x, \gamma_{j\ell})$ is another softmax function in x with parameters $\gamma_{j\ell}$
- at the terminal the output $Y \sim P(y|x, \theta_{j\ell})$
  - Regression: P is normal with its parameters
  - Classification: P is the logistic CDF
- Then the mixture probability of the output is

$$P(y|x, \Psi) = \sum_{j=1}^{K} g_j(x, \gamma_j) \sum_{\ell=1}^{K} g_{\ell|j}(x, \gamma_{j\ell}) P(y|x, \theta_{j\ell})$$

- where $\Psi = (\gamma_j, \gamma_{j\ell}, \theta_{j\ell})$ is estimated using maximum likelihood methods and EM algorithm.

Hierarchical Mixtures of Experts

- where $g_j(x, \gamma_j)$ is a softmax function in x with parameters γ_j
- and $g_{\ell|j}(x, \gamma_{j\ell})$ is another softmax function in x with parameters $\gamma_{j\ell}$
- at the terminal the output $Y \sim P(y|x, \theta_{j\ell})$
  - Regression: P is normal with its parameters
  - Classification: P is the logistic CDF
- Then the mixture probability of the output is

$$P(y|x, \Psi) = \sum_{j=1}^{K} g_j(x, \gamma_j) \sum_{\ell=1}^{K} g_{\ell|j}(x, \gamma_{j\ell}) P(y|x, \theta_{j\ell})$$

- where $\Psi = (\gamma_j, \gamma_{j\ell}, \theta_{j\ell})$ is estimated using maximum likelihood methods and EM algorithm.

Hierarchical Mixtures of Experts

- where $g_j(x, \gamma_j)$ is a softmax function in x with parameters γ_j
- and $g_{\ell|j}(x, \gamma_{j\ell})$ is another softmax function in x with parameters $\gamma_{j\ell}$
- at the terminal the output $Y \sim P(y|x, \theta_{j\ell})$
  - Regression: P is normal with its parameters
  - Classification: P is the logistic CDF
- Then the mixture probability of the output is

$$P(y|x, \Psi) = \sum_{j=1}^{K} g_j(x, \gamma_j) \sum_{\ell=1}^{K} g_{\ell|j}(x, \gamma_{j\ell}) P(y|x, \theta_{j\ell})$$

- where $\Psi = (\gamma_j, \gamma_{j\ell}, \theta_{j\ell})$ is estimated using maximum likelihood methods and EM algorithm.

Hierarchical Mixtures of Experts

- where $g_j(x, \gamma_j)$ is a softmax function in x with parameters γ_j
- and $g_{\ell|j}(x, \gamma_{j\ell})$ is another softmax function in x with parameters $\gamma_{j\ell}$
- at the terminal the output $Y \sim P(y|x, \theta_{j\ell})$
  - Regression: P is normal with its parameters
  - Classification: P is the logistic CDF
- Then the mixture probability of the output is

$$P(y|x, \Psi) = \sum_{j=1}^{K} g_j(x, \gamma_j) \sum_{\ell=1}^{K} g_{\ell|j}(x, \gamma_{j\ell}) P(y|x, \theta_{j\ell})$$

- where $\Psi = (\gamma_j, \gamma_{j\ell}, \theta_{j\ell})$ is estimated using maximum likelihood methods and EM algorithm.

Hierarchical Mixtures of Experts

- where $g_j(x, \gamma_j)$ is a softmax function in x with parameters γ_j
- and $g_{\ell|j}(x, \gamma_{j\ell})$ is another softmax function in x with parameters $\gamma_{j\ell}$
- at the terminal the output $Y \sim P(y|x, \theta_{j\ell})$
  - Regression: P is normal with its parameters
  - Classification: P is the logistic CDF
- Then the mixture probability of the output is

$$P(y|x, \Psi) = \sum_{j=1}^{K} g_j(x, \gamma_j) \sum_{\ell=1}^{K} g_{\ell|j}(x, \gamma_{j\ell}) P(y|x, \theta_{j\ell})$$

- where $\Psi = (\gamma_j, \gamma_{j\ell}, \theta_{j\ell})$ is estimated using maximum likelihood methods and EM algorithm.

Hierarchical Mixtures of Experts

- where $g_j(x, \gamma_j)$ is a softmax function in x with parameters γ_j
- and $g_{\ell|j}(x, \gamma_{j\ell})$ is another softmax function in x with parameters $\gamma_{j\ell}$
- at the terminal the output $Y \sim P(y|x, \theta_{j\ell})$
  - Regression: P is normal with its parameters
  - Classification: P is the logistic CDF
- Then the mixture probability of the output is

$$P(y|x, \Psi) = \sum_{j=1}^{K} g_j(x, \gamma_j) \sum_{\ell=1}^{K} g_{\ell|j}(x, \gamma_{j\ell}) P(y|x, \theta_{j\ell})$$

- where $\Psi = (\gamma_j, \gamma_{j\ell}, \theta_{j\ell})$ is estimated using maximum likelihood methods and EM algorithm.

End of Set 6