S&DS 265 / 565

Introductory Machine Learning

Trees

Tuesday, October 5

Yale

Plan for this and next week

- Assn 2 due tonight
- Keep in mind reminders from last week (select pages in Gradescope, rerun notebook, length of output, line wrap)
- Assn 3 out today — decision trees
- Next week: Quiz 2 (Tuesday); SGD, bias-variance, CV, trees
- Midterm exam: Tuesday October 19 (in class); practice exam released next week
- Questions?

You are here

3	Sept 14, 16	Linear regression and classification	Covid trends (revisited) Classification examples	Tue: Assn1 out	Sept 14: Regression concepts Notes on regression Sept 16: Classification Notes on classification
4	Sept 21, 23	Stochastic gradient descent	SGD examples	Tue: Quiz 1 Thu: Assn 1 in Assn2 out	Sept 21: Classification (continued) Sept 23: Stochastic gradient descent
5	Sept 28, 30	Bias and variance, cross-validation	Bias-variance tradeoff Covid trends (revisited) California housing		Sept 28: Bias and variance Sept 30: Cross-validation
6	Oct 5, 7	Tree-based methods	Trees and forests	Tue: Assn 2 in; Assn 3 out	
7	Oct 12, 14	PCA and dimension reduction	PCA examples	Tue: Quiz 2 Thu: Assn 3 in; Assn 4 out	
8	Oct 19	Midterm exam (in class)			

Classification and Regression Trees (CART)

Trees provide ways of modeling nonlinear relationships by carving out *rectangular regions* in the feature space.

Classification and Regression Trees (CART)

Trees provide ways of modeling nonlinear relationships by carving out *rectangular regions* in the feature space.

- Response variables can be categorical or quantitative.

Classification and Regression Trees (CART)

Trees provide ways of modeling nonlinear relationships by carving out *rectangular regions* in the feature space.

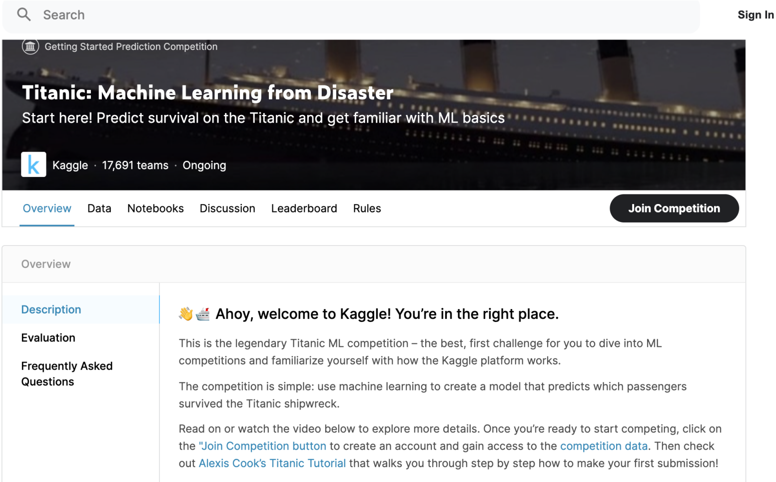
- Response variables can be categorical or quantitative.
- Yields a set of interpretable decision rules.

Classification and Regression Trees (CART)

Trees provide ways of modeling nonlinear relationships by carving out *rectangular regions* in the feature space.

- Response variables can be categorical or quantitative.
- Yields a set of interpretable decision rules.
- Predictive ability is mediocre, *but* can be improved by combining multiple trees (resampling, ensemble methods)

Titanic data



Titanic data

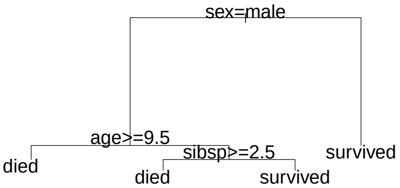
- **Survived**: Outcome of survival (0 = No; 1 = Yes)
- **Pclass**: Socio-economic class (1 = Upper class; 2 = Middle class; 3 = Lower class)
- **Name**: Name of passenger
- **Sex**: Sex of the passenger
- **Age**: Age of the passenger (Some entries contain `NaN`)
- **SibSp**: Number of siblings and spouses of the passenger aboard
- **Parch**: Number of parents and children of the passenger aboard
- **Ticket**: Ticket number of the passenger
- **Fare**: Fare paid by the passenger
- **Cabin** Cabin number of the passenger (Some entries contain `NaN`)
- **Embarked**: Port of embarkation of the passenger (C = Cherbourg; Q = Queenstown; S = Southampton)

Trees

Trees

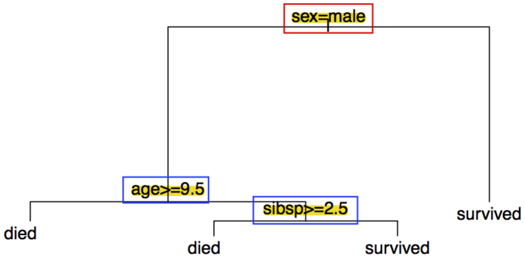
Trees

Modeling Titanic survival:



Tree terminology

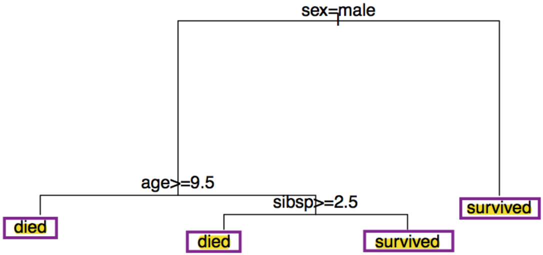
Internal nodes are points where the predictor space is split.



The internal node at the top is the **root** of the tree.

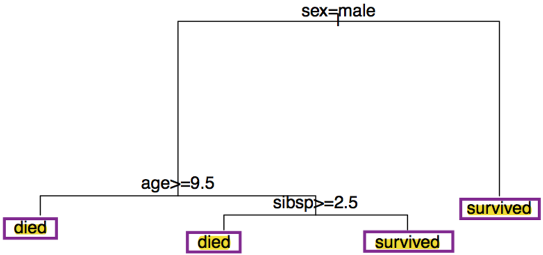
Tree terminology

Terminal nodes (or **leaves**) are the ends of the tree where no further splitting occurs.



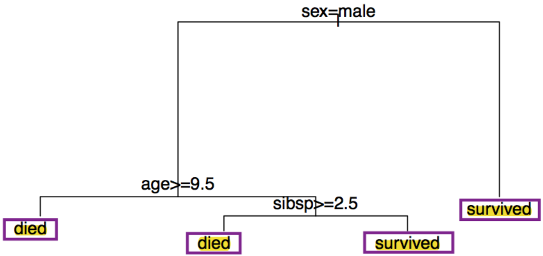
Tree terminology

Terminal nodes (or **leaves**) are the ends of the tree where no further splitting occurs.



Denote these J regions as $R_1, \ldots, R_J$.

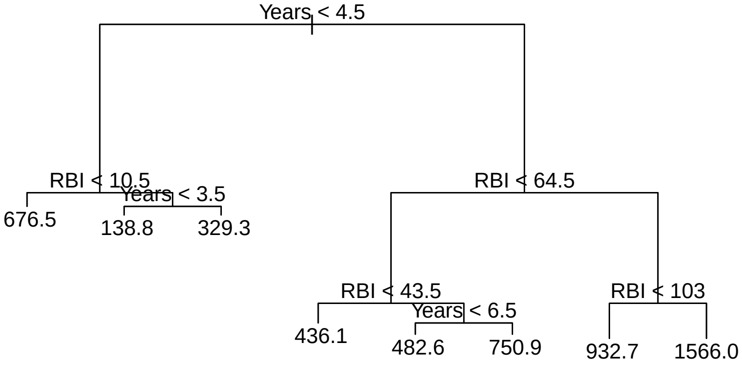
Tree terminology



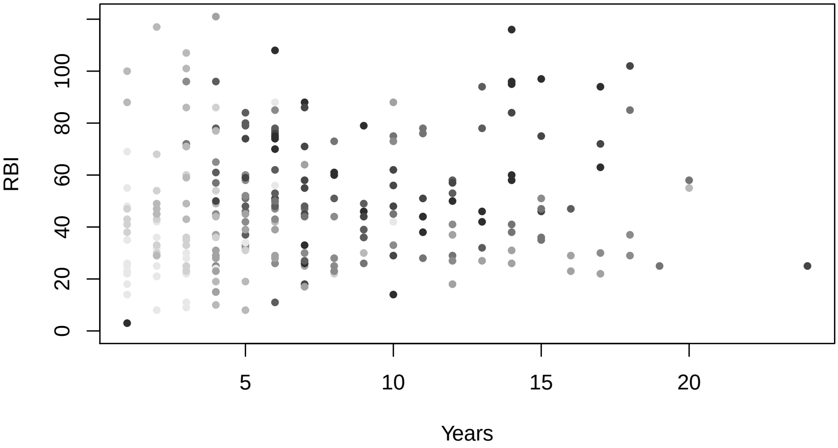
- $R_1 = \{i : \text{sex}_i = \text{male} \cap \text{age}_i \geq 9.5\}$
- $R_2 = \{i : \text{sex}_i = \text{male} \cap \text{age}_i < 9.5 \cap \text{sibsp}_i \geq 2.5\}$
- $R_3 = \{i : \text{sex}_i = \text{male} \cap \text{age}_i < 9.5 \cap \text{sibsp}_i < 2.5\}$
- $R_4 = \{i : \text{sex}_i \neq \text{male}\}$

Regression tree example

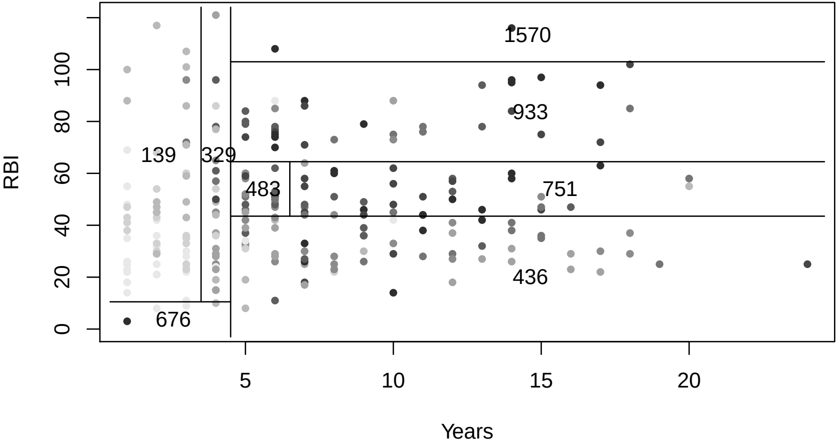
Baseball hitter salaries (in $10,000s):



Regression tree example



Regression tree example



Prediction using trees

Trace each test observation into a leaf R_j based on the sequence of conditions. Predict $\widehat{y}_{R_j}$ for all observations in R_j.

Prediction using trees

Trace each test observation into a leaf R_j based on the sequence of conditions. Predict $\widehat{y}_{R_j}$ for all observations in R_j.

$\widehat{y}_{R_j}$ is a function of all training observations i in R_j.

Prediction using trees

Trace each test observation into a leaf R_j based on the sequence of conditions. Predict $\widehat{y}_{R_j}$ for all observations in R_j.

$\widehat{y}_{R_j}$ is a function of all training observations i in R_j.

- Regression: $\widehat{y}_{R_j} = \bar{y}_{i:i \in R_j}$

Prediction using trees

Trace each test observation into a leaf R_j based on the sequence of conditions. Predict $\widehat{y}_{R_j}$ for all observations in R_j.

$\widehat{y}_{R_j}$ is a function of all training observations i in R_j.

- Regression: $\widehat{y}_{R_j} = \bar{y}_{i:i \in R_j}$
- Classification: $\widehat{y}_{R_j} =$ most frequently occurring class y_i for $i \in R_j$

Prediction using trees

Trace each test observation into a leaf R_j based on the sequence of conditions. Predict $\widehat{y}_{R_j}$ for all observations in R_j.

$\widehat{y}_{R_j}$ is a function of all training observations i in R_j.

- Regression: $\widehat{y}_{R_j} = \bar{y}_{i:i \in R_j}$
- Classification: $\widehat{y}_{R_j} =$ most frequently occurring class y_i for $i \in R_j$

Prediction using trees

Trace each test observation into a leaf R_j based on the sequence of conditions. Predict $\widehat{y}_{R_j}$ for all observations in R_j.

$\widehat{y}_{R_j}$ is a function of all training observations i in R_j.

- Regression: $\widehat{y}_{R_j} = \bar{y}_{i:i \in R_j}$
- Classification: $\widehat{y}_{R_j} =$ most frequently occurring class y_i for $i \in R_j$

Fitting a tree boils down to identifying the appropriate set of regions $R_1, \ldots, R_J$ that "best" describes the relationship between X and y.

Tree building

We want to choose $R_1, \ldots, R_J$ to minimize error:

$$RSS = \sum_{j=1}^{J} \sum_{i \in R_j} (y_i - \bar{y}_{R_j})^2$$

Tree building

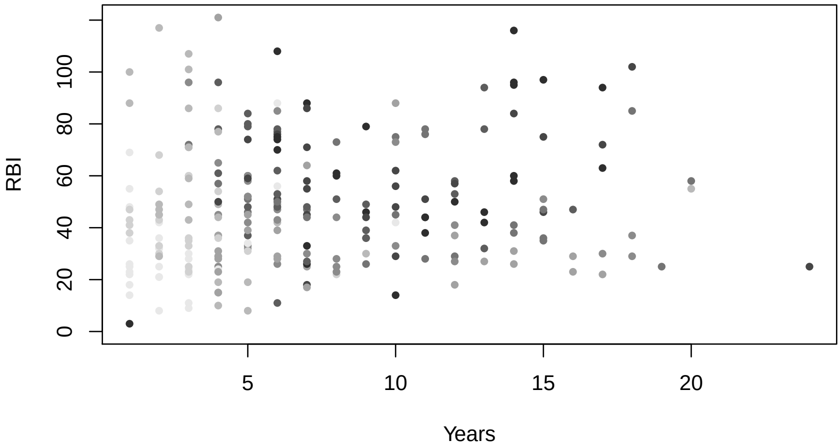
We want to choose $R_1, \ldots, R_J$ to minimize error:

$$RSS = \sum_{j=1}^{J} \sum_{i \in R_j} (y_i - \bar{y}_{R_j})^2$$

Tree building takes a *greedy* approach.

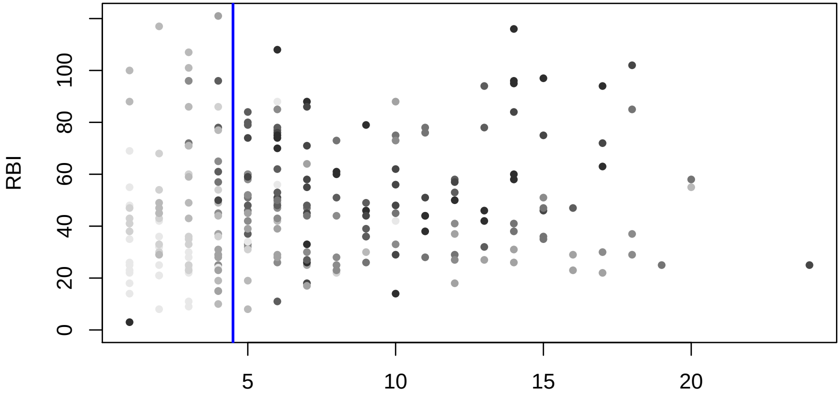
- Grow the tree by recursive binary splitting
- Prune back the tree

Tree growth

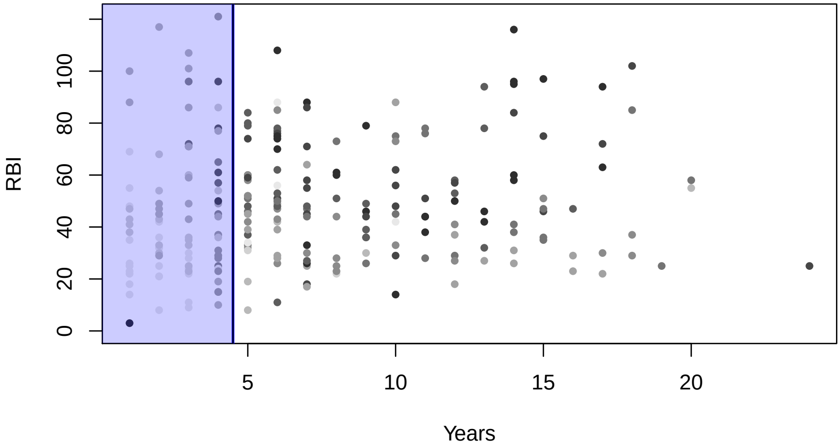


Tree growth

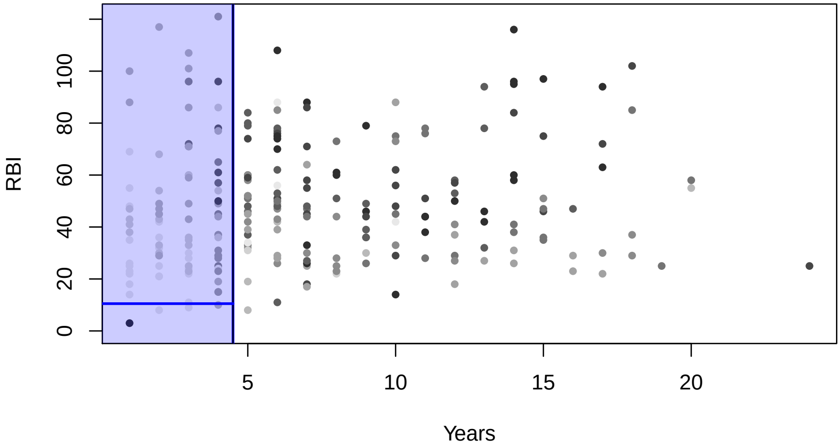
Where can we draw a horizontal or vertical line that best splits the data into two homogeneous parts?



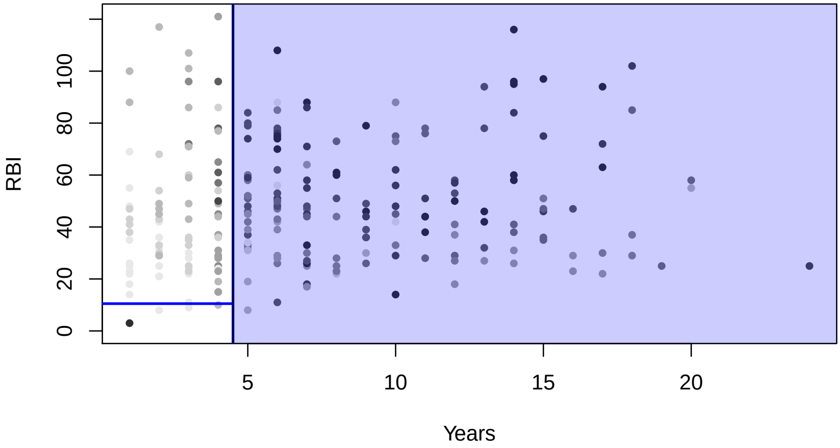
Tree growth



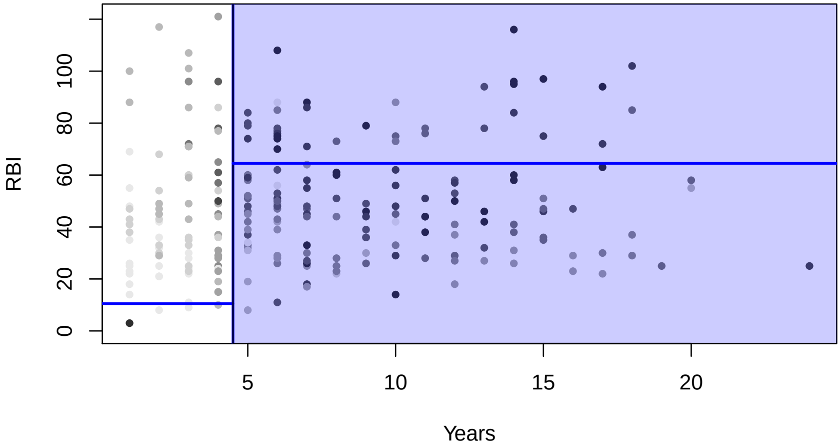
Tree growth



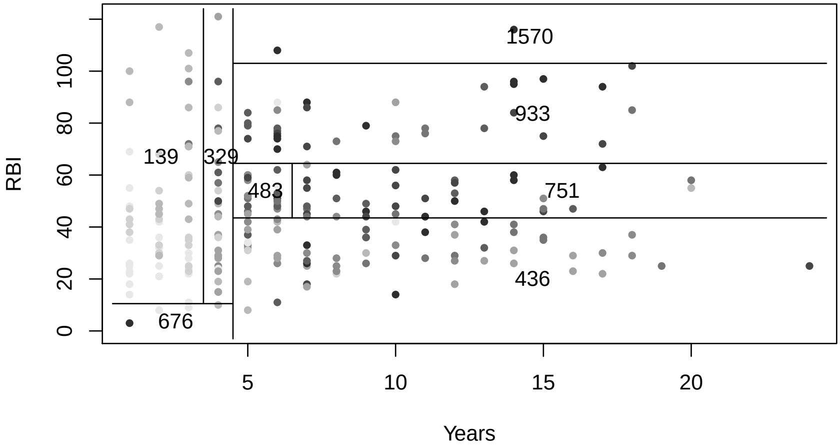
Tree growth



Tree growth



Tree growth


1570
933
139
329
483
751
436
676
RBI
0
20
40
60
80
100
5
10
15
20
Years

Tree growth (Regression)

1. Cycle through predictors X_k for $k = 1, \ldots, p$. For each X_k,

Tree growth (Regression)

1. Cycle through predictors X_k for $k = 1, \ldots, p$. For each X_k,
   - (Quantitative X_k) Consider cutpoints s (unique values of X_k) that divide up the region into two parts:

$$R_1(k, s) = \{i | X_{ik} < s\} \quad \text{and} \quad R_2(k, s) = \{i | X_{ik} \geq s\}$$

Tree growth (Regression)

1. Cycle through predictors X_k for $k = 1, \ldots, p$. For each X_k,
   - (Quantitative X_k) Consider cutpoints s (unique values of X_k) that divide up the region into two parts:

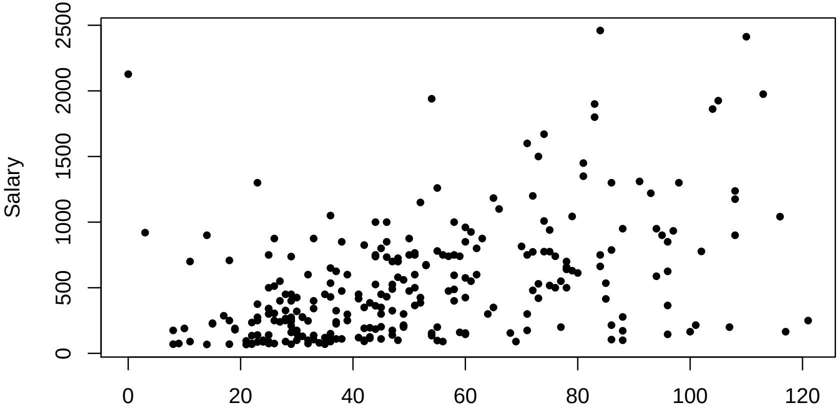
     $$R_1(k,s) = \{i | X_{ik} < s\} \quad \text{and} \quad R_2(k,s) = \{i | X_{ik} \geq s\}$$

   - Evaluate (for regression trees):

     $$Q_k(s) = \sum_{i:i \in R_1(k,s)} (y_i - \bar{y}_{R_1})^2 + \sum_{i:i \in R_2(k,s)} (y_i - \bar{y}_{R_2})^2$$

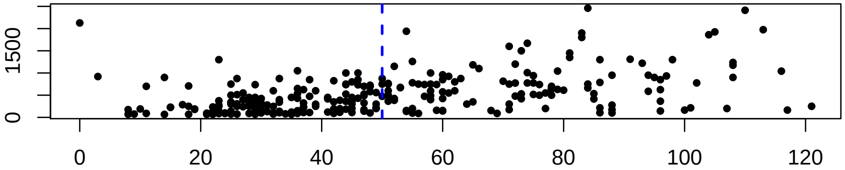
Example: Evaluating Q_{RBI}

$$Q_k(s) = \sum_{i:i \in R_1(k,s)} (y_i - \bar{y}_{R_1})^2 + \sum_{i:i \in R_2(k,s)} (y_i - \bar{y}_{R_2})^2$$

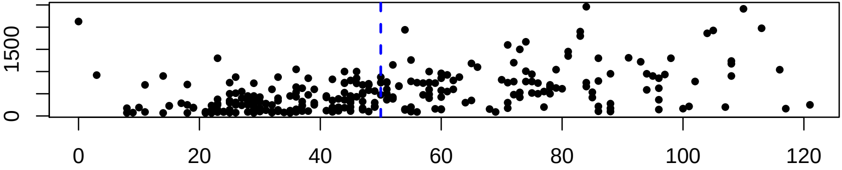


Example: Evaluating Q_{RBI}

$$Q_k(s) = \sum_{i:i \in R_1(k,s)} (y_i - \bar{y}_{R_1})^2 + \sum_{i:i \in R_2(k,s)} (y_i - \bar{y}_{R_2})^2$$

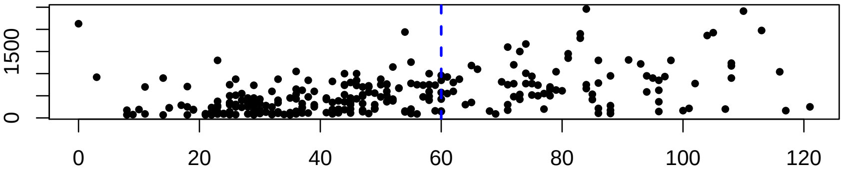


Example: Evaluating Q_{RBI}



- $R_1(RBI, 50)$: $\bar{y}_{R_1} = 359$
- $R_2(RBI, 50)$: $\bar{y}_{R_2} = 753$
- $$\begin{aligned} Q_{RBI}(50) &= \sum_{i:i \in R_1} (y_i - 359)^2 + \sum_{i:i \in R_2} (y_i - 753)^2 \\ &= 13015000 + 30186039 \\ &= 43201039 \end{aligned}$$

Example: Evaluating Q_{RBI}



- $R_1(RBI, 60)$: $\bar{y}_{R_1} = 405$
- $R_2(RBI, 60)$: $\bar{y}_{R_2} = 802$
- $$Q_{RBI}(60) = \sum_{i:i \in R_1} (y_i - 405)^2 + \sum_{i:i \in R_2} (y_i - 805)^2$$
  $$= 19186489 + 24943383$$
  $$= 44129872$$

Compute $Q_{RBI}(s)$ for all distinct values of RBI s.

Categorical predictors

If a predictor X_k is categorical, then cut points are modified.

Categorical predictors

If a predictor X_k is categorical, then cut points are modified.

Suppose X_k has unique values in $\{A, B, C, D, E\}$. Then, the possible splits include:

- $\{D\}$ vs. $\{A, B, C, E\}$
- $\{D, B\}$ vs. $\{A, C, E\}$
- $\ldots$

Categorical predictors

If a predictor X_k is categorical, then cut points are modified.

Suppose X_k has unique values in $\{A, B, C, D, E\}$. Then, the possible splits include:

- $\{D\}$ vs. $\{A, B, C, E\}$
- $\{D, B\}$ vs. $\{A, C, E\}$
- $\ldots$

Every possible partition of the set of unique values into 2 subsets is considered, and again we identify the split producing the lowest resulting RSS.

Tree growth

1. Cycle through predictors X_k for $k = 1, \ldots, p$. For each X_k:
   - (Quantitative X_k) Consider cutpoints s (unique values of X_k) that divide up the region into two parts:

     $$R_1(k, s) = \{i | X_{ik} < s\} \quad \text{and} \quad R_2(k, s) = \{i | X_{ik} \geq s\}$$

   - Evaluate (for regression trees):

     $$Q_k(s) = \sum_{i:i \in R_1(k,s)} (y_i - \bar{y}_{R_1})^2 + \sum_{i:i \in R_2(k,s)} (y_i - \bar{y}_{R_2})^2$$

   - Find the value of s that minimizes $Q_k(s)$. Call this s_k.

Tree growth

1. Cycle through predictors X_k for $k = 1, \ldots, p$. For each X_k:
   - (Quantitative X_k) Consider cutpoints s (unique values of X_k) that divide up the region into two parts:

$$R_1(k, s) = \{i | X_{ik} < s\} \quad \text{and} \quad R_2(k, s) = \{i | X_{ik} \geq s\}$$

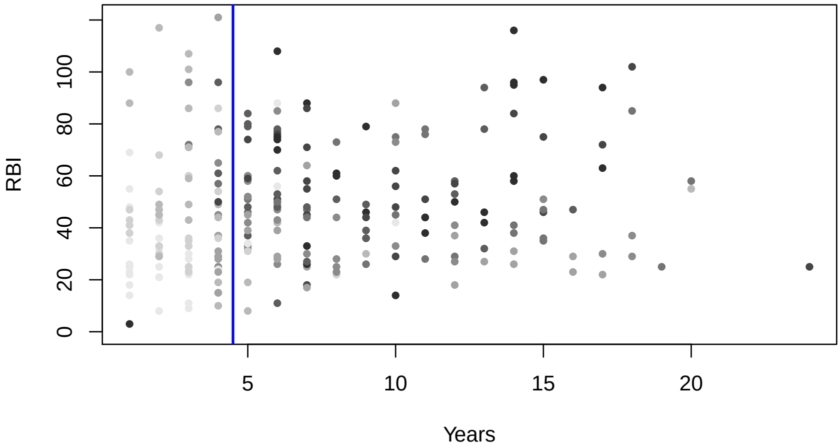
   - Evaluate (for regression trees):

$$Q_k(s) = \sum_{i:i \in R_1(k,s)} (y_i - \bar{y}_{R_1})^2 + \sum_{i:i \in R_2(k,s)} (y_i - \bar{y}_{R_2})^2$$

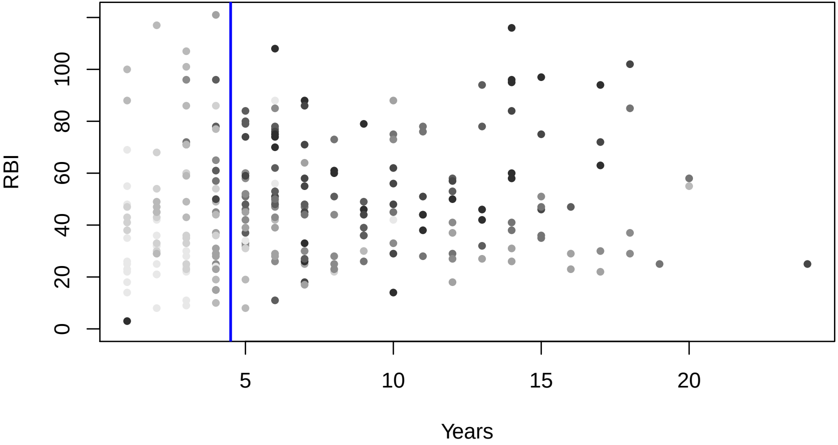
   - Find the value of s that minimizes $Q_k(s)$. Call this s_k.

2. Find the predictor X_k with the minimum $Q_1(s_1), Q_2(s_2), \ldots, Q_p(s_p)$. Make the first binary partition along predictor X_k at cut point s_k.

Tree growth



Tree growth



Essentially repeat the previous 2 steps on each of the resulting regions separately, iteratively. (Hence **recursive** binary partitioning.)

Bias-variance

Bias-variance

- As tree is grown deeper, bias decreases
- But the variance increases
- How to choose the right size of tree?

Stopping criterion

Once we stop, we relabel the terminal nodes to be $R_1, \ldots, R_J$ and compute $\bar{y}_{R_j}$ (means within each region) to serve as $\widehat{y}$ values.

But when do we stop?

Stopping criterion

Once we stop, we relabel the terminal nodes to be $R_1, \ldots, R_J$ and compute $\bar{y}_{R_j}$ (means within each region) to serve as $\widehat{y}$ values.

But when do we stop?

Some possibilities:

- number of observations in a node has reached a minimum

Stopping criterion

Once we stop, we relabel the terminal nodes to be $R_1, \ldots, R_J$ and compute $\bar{y}_{R_j}$ (means within each region) to serve as $\widehat{y}$ values.

But when do we stop?

Some possibilities:

- number of observations in a node has reached a minimum
- depth of tree has reached a maximum

Stopping criterion

Once we stop, we relabel the terminal nodes to be $R_1, \ldots, R_J$ and compute $\bar{y}_{R_j}$ (means within each region) to serve as $\widehat{y}$ values.

But when do we stop?

Some possibilities:

- number of observations in a node has reached a minimum
- depth of tree has reached a maximum
- grow until no further splits can reduce RSS by some amount

Stopping criterion

Once we stop, we relabel the terminal nodes to be $R_1, \ldots, R_J$ and compute $\bar{y}_{R_j}$ (means within each region) to serve as $\widehat{y}$ values.

But when do we stop?

Some possibilities:

- number of observations in a node has reached a minimum
- depth of tree has reached a maximum
- grow until no further splits can reduce RSS by some amount

Stopping criterion

Once we stop, we relabel the terminal nodes to be $R_1, \ldots, R_J$ and compute $\bar{y}_{R_j}$ (means within each region) to serve as $\widehat{y}$ values.

But when do we stop?

Some possibilities:

- number of observations in a node has reached a minimum
- depth of tree has reached a maximum
- grow until no further splits can reduce RSS by some amount

Many options – resulting in tuning parameters that are hard to deal with.

Tree pruning

Another way to get around the overfitting problem is to grow a large tree and then **prune** it back.

Tree pruning

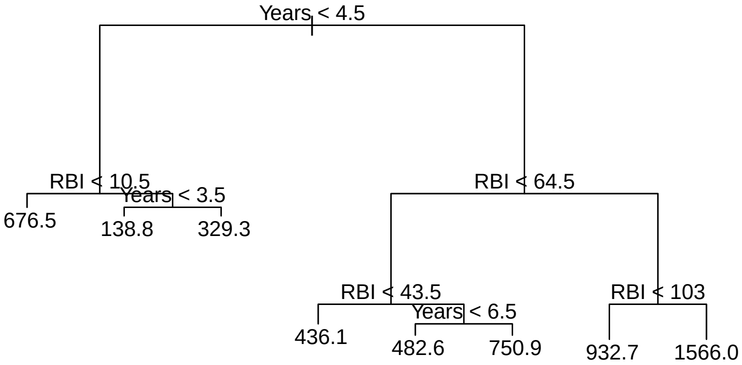
Another way to get around the overfitting problem is to grow a large tree and then **prune** it back.

Typically, pruning involves looking at subtrees of the fully-grown tree, and comparing how well the subtrees perform.

Tree pruning

Another way to get around the overfitting problem is to grow a large tree and then **prune** it back.

Typically, pruning involves looking at subtrees of the fully-grown tree, and comparing how well the subtrees perform.



Tree pruning

How do we prune?

Tree pruning

How do we prune?

- cross validation

Tree pruning

How do we prune?

- cross validation
- cost-complexity pruning

Cost-complexity pruning

$$C(T) = \sum_{m=1}^{|T|} \sum_{i \in R_m} (y_i - \widehat{y}_{R_m})^2 + \alpha |T|$$

α is a tuning parameter that controls for the complexity of the model.

Cost-complexity pruning

$$C(T) = \sum_{m=1}^{|T|} \sum_{i \in R_m} (y_i - \widehat{y}_{R_m})^2 + \alpha |T|$$

α is a tuning parameter that controls for the complexity of the model.

- $\alpha = 0$ implies the full tree

Cost-complexity pruning

$$C(T) = \sum_{m=1}^{|T|} \sum_{i \in R_m} (y_i - \widehat{y}_{R_m})^2 + \alpha |T|$$

α is a tuning parameter that controls for the complexity of the model.

- $\alpha = 0$ implies the full tree
- Larger α implies higher penalty for complexity of model

Cost-complexity pruning

$$C(T) = \sum_{m=1}^{|T|} \sum_{i \in R_m} (y_i - \widehat{y}_{R_m})^2 + \alpha |T|$$

α is a tuning parameter that controls for the complexity of the model.

- $\alpha = 0$ implies the full tree
- Larger α implies higher penalty for complexity of model

Cost-complexity pruning

$$C(T) = \sum_{m=1}^{|T|} \sum_{i \in R_m} (y_i - \widehat{y}_{R_m})^2 + \alpha |T|$$

α is a tuning parameter that controls for the complexity of the model.

- $\alpha = 0$ implies the full tree
- Larger α implies higher penalty for complexity of model

It is possible to efficiently identify a sequence of nested subtrees that are optimal for a sequence of increasing α.

Tree pruning

1. Grow a big tree on a *training set*.
2. Obtain a nested set of subtrees $T_L \subset \cdots \subset T_2 \subset T_1 \subset T_0$ corresponding to a sequence of α values.
3. Use K-fold cross-validation to identify the subtree/α that does best.

Issues with trees

- Instability. Trees can have high variance. As data change, tree topology can change dramatically, making interpretation difficult
- Lack of smoothness. The splits lead to a “jagged” decision boundary. More of a problem for regression than classification
- Difficulty capturing additive structure, where the regression function is a sum of terms

Demos

Some nice demonstrations of decision trees:

`http://www.r2d3.us/visual-intro-to-machine-learning-part-1/`

`http://www.r2d3.us/visual-intro-to-machine-learning-part-2`

Summary from today

- Trees give interpretable, nonlinear prediction rules
- Deep trees have low bias, high variance
- Shallow trees have high bias, low variance
- Trees are grown greedily to be full, then pruned back