So far { Nearest neighbor
neural network
polynominal regression

A different approach to estimate none-linear function

S&DS 365 / 565

Data Mining and Machine Learning

(Decision) Trees

Yale

Parametric vs Nonparametric Models

The linear regression, logistic regression (including version where we include polynomials of the features) fall under the category of **parametric** models. We have parameters that are learned.

The kNN algorithm is a **nonparametric** method, and can be thought as *local averaging*. No explicit parameters.

Classification and Regression Trees (CART)

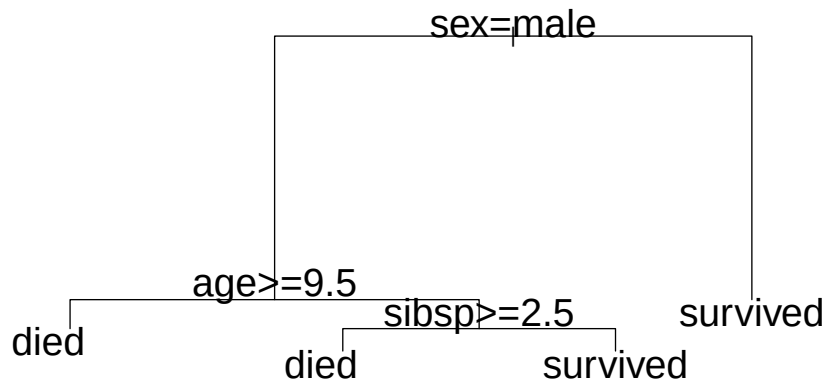
Trees provide alternative ways of modeling nonlinear relationships by carving out *rectangular regions* in the covariate space.

- Response variables can be categorical or quantitative.
- Yields a set of interpretable decision rules. if BMI > 18 then ~
- Predictive ability is mediocre, *but* can be improved with ideas of resampling.

Trees

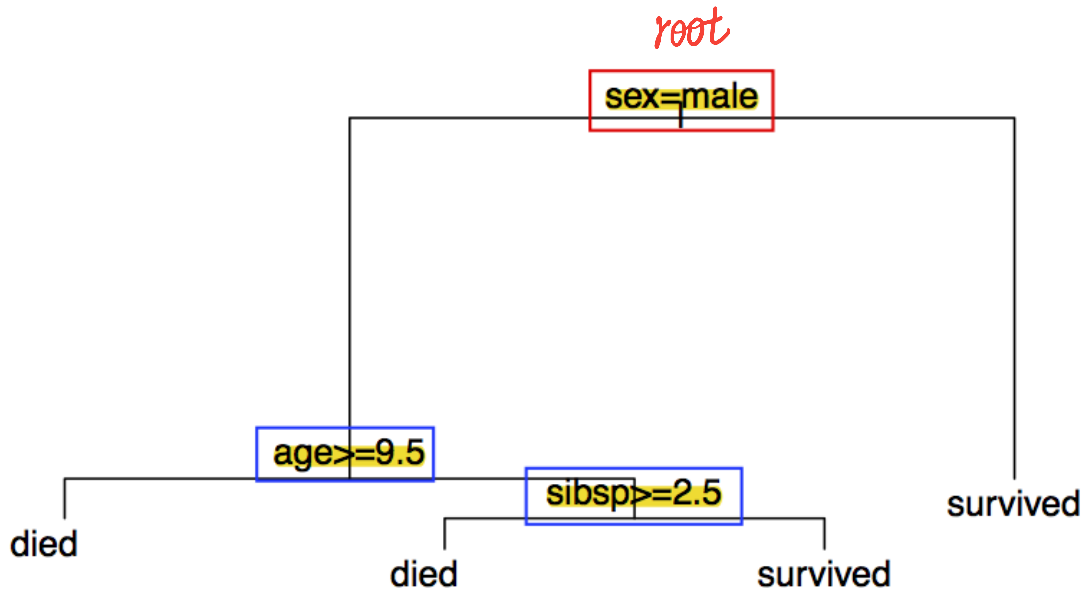
Trees

Modeling Titanic survival:



Tree terminology

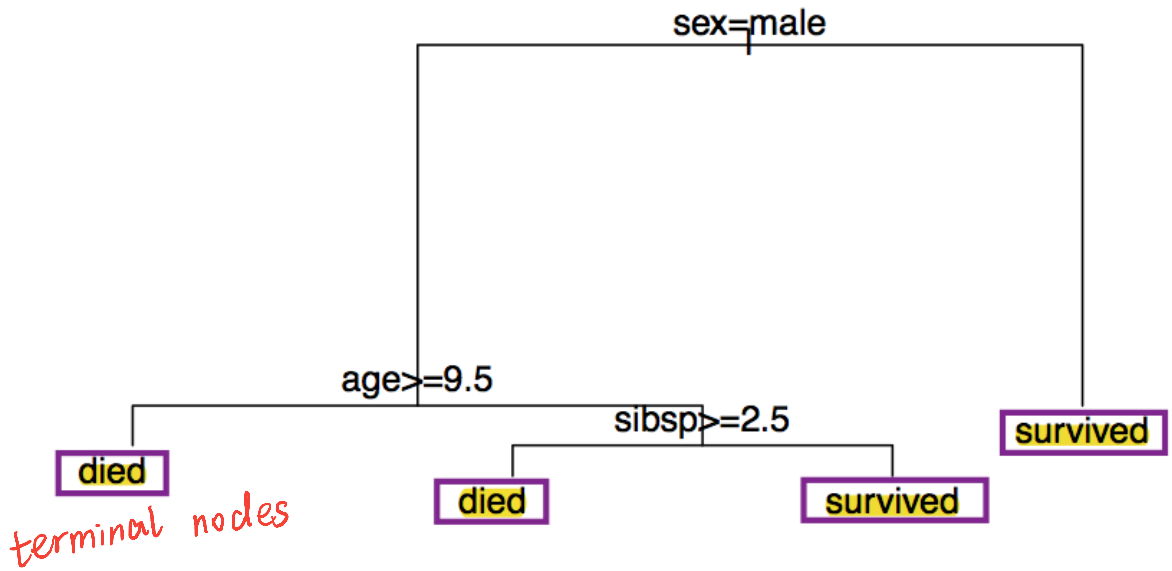
Internal nodes are points where the predictor space is split.



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

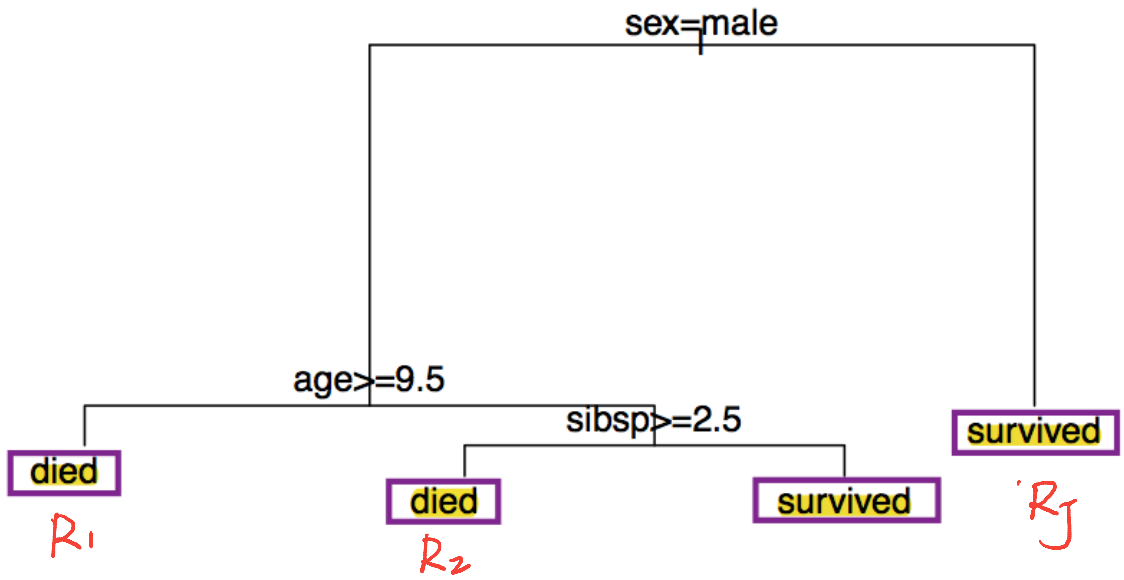
Tree terminology

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



Tree terminology

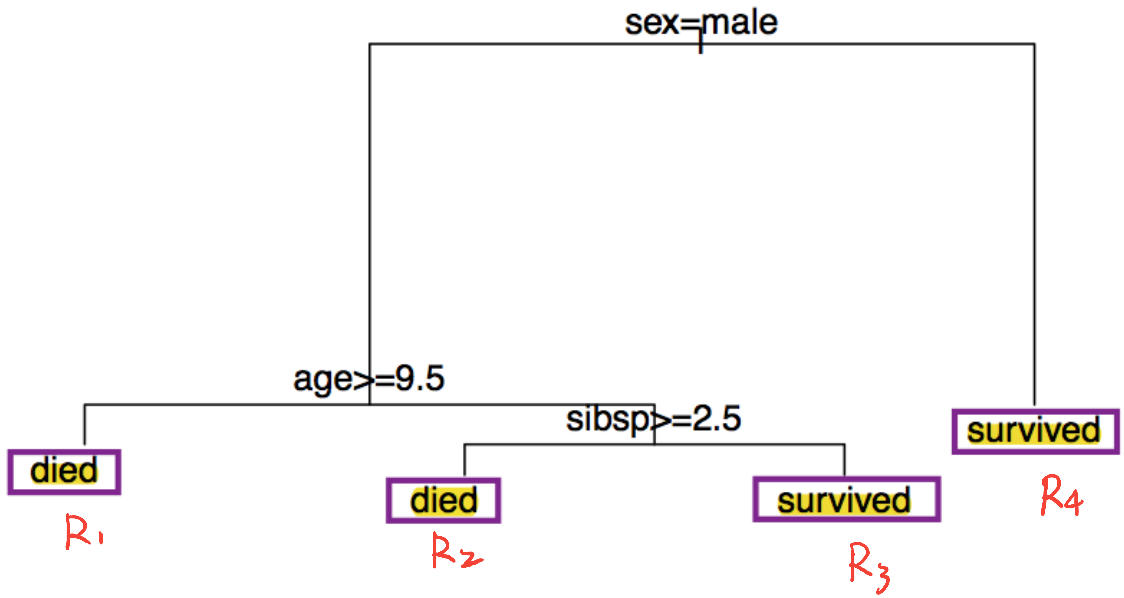
Terminal nodes (or **leaves**) are the ends of the tree where no further splitting occurs.



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

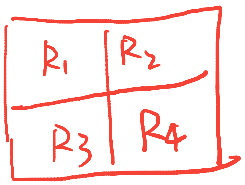
不相交
Disjoint region (rectangle)

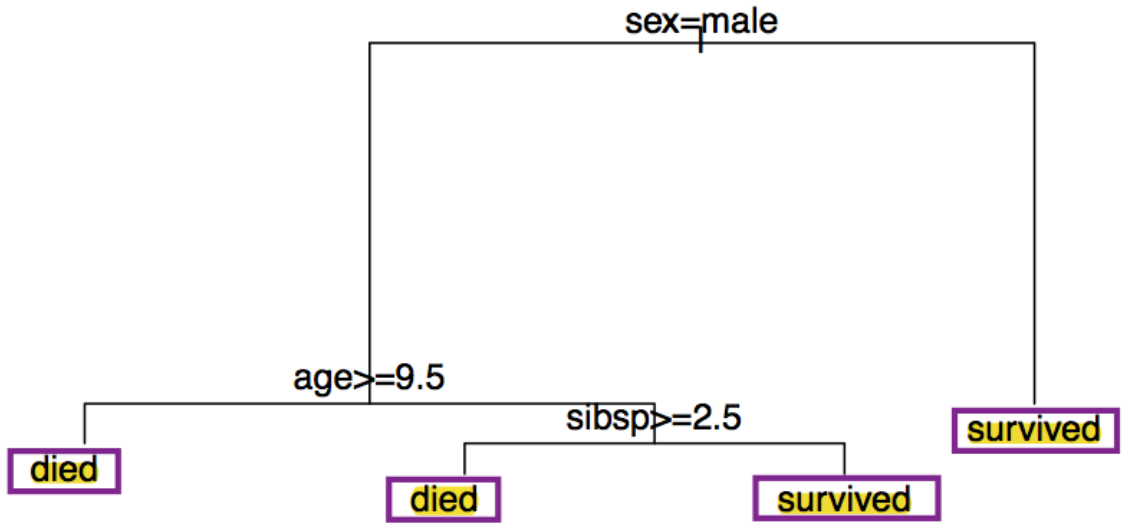
Tree terminology



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

Tree terminology



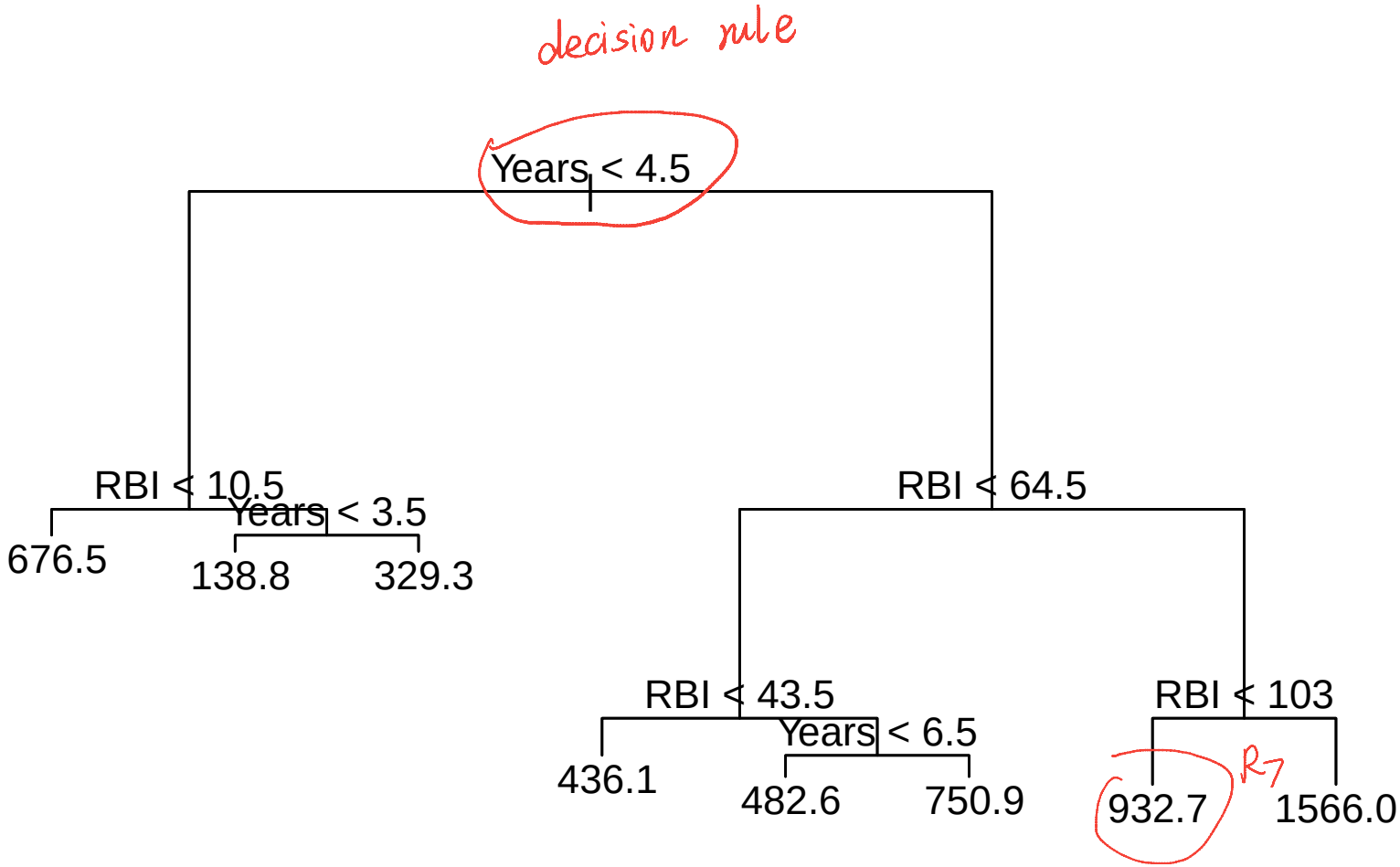


- $R_1 = \{i : \text{sex}_i = \text{male} \cap \text{age}_i \geq 9.5\}$
- $R_2 = \{i : \text{sex}_i = \text{male} \cap \text{age}_i < 9.5 \cap \text{sibsp}_i \geq 2.5\}$
- $R_3 = \{i : \text{sex}_i = \text{male} \cap \text{age}_i < 9.5 \cap \text{sibsp}_i < 2.5\}$
- $R_4 = \{i : \text{sex}_i \neq \text{male}\}$

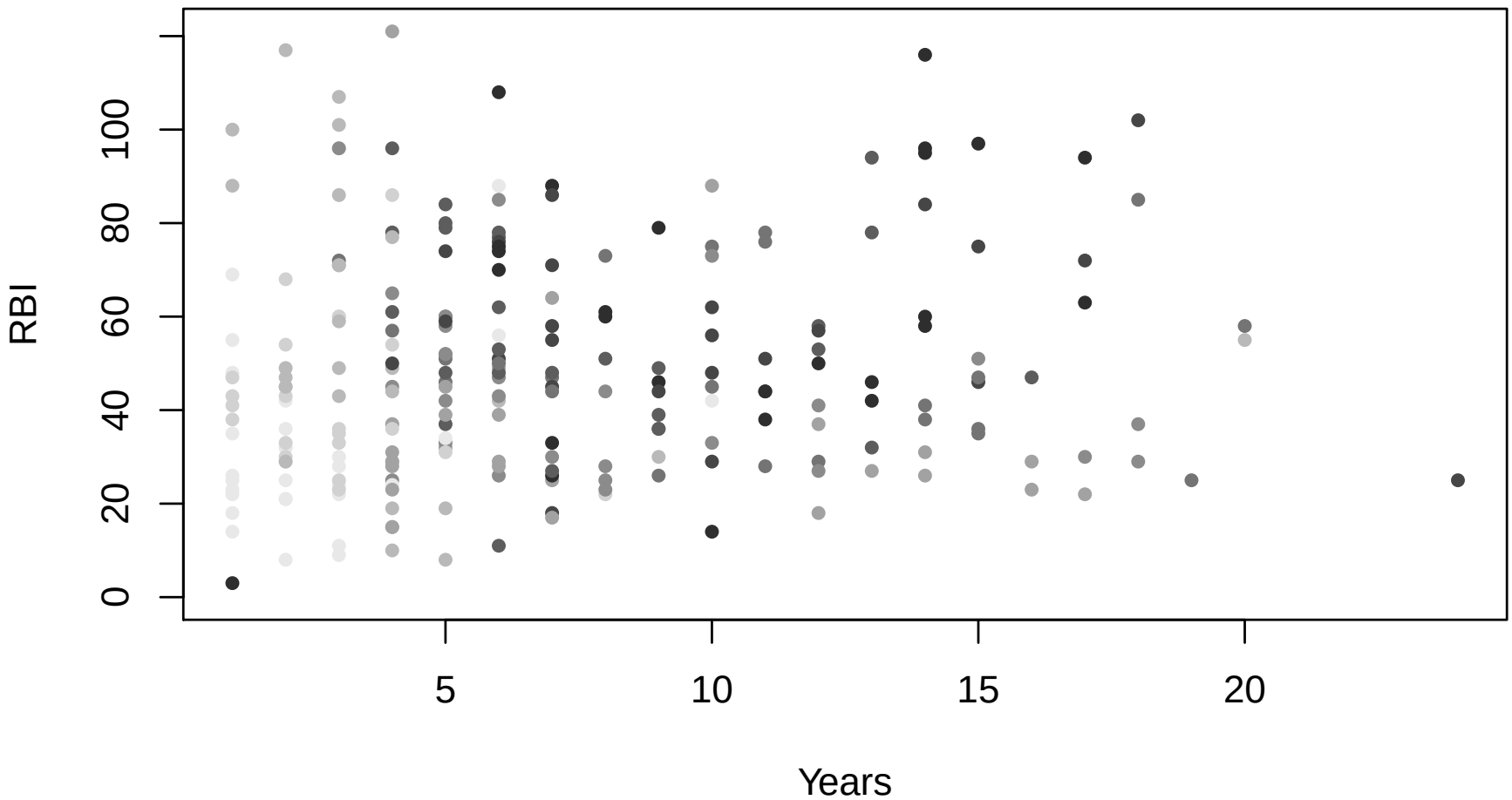
Regions are disjoint and cover the whole space.

Regression tree example

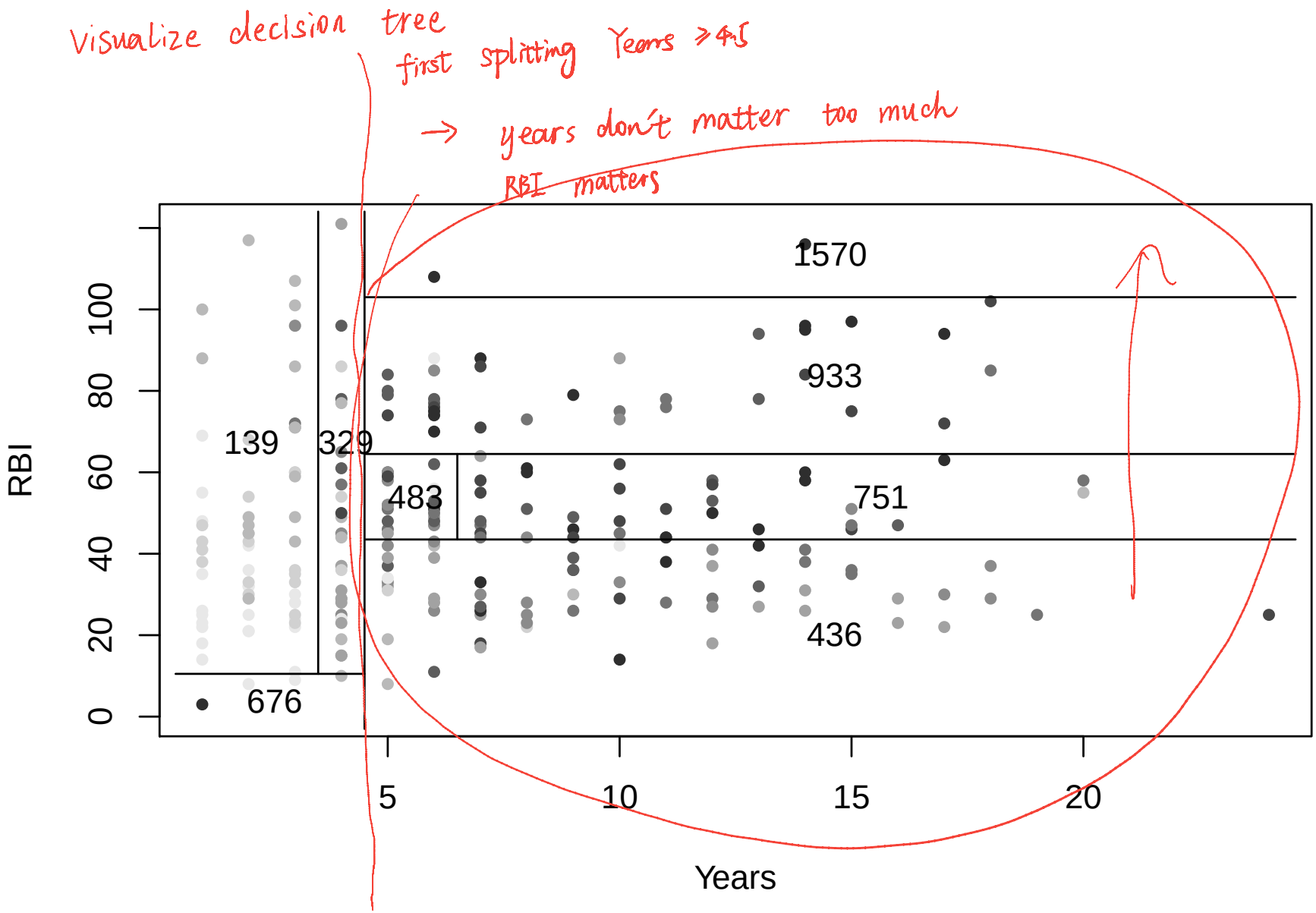
Baseball hitter salaries (1987) (in $1,000s):



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.

assign a test data point to a region

if x assign to Region R_7, then $\hat{y}$ is { mean (Loss: MSE); median (Loss: Mean absolute Deviation)

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

Hamming loss

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.

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.

Given Region, we know

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

重点是

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.

What is best?

How to find the best region?

Tree building

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

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

→ we have Jth regions

↑ data point in Region J

Tree building

Intuitively, 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.

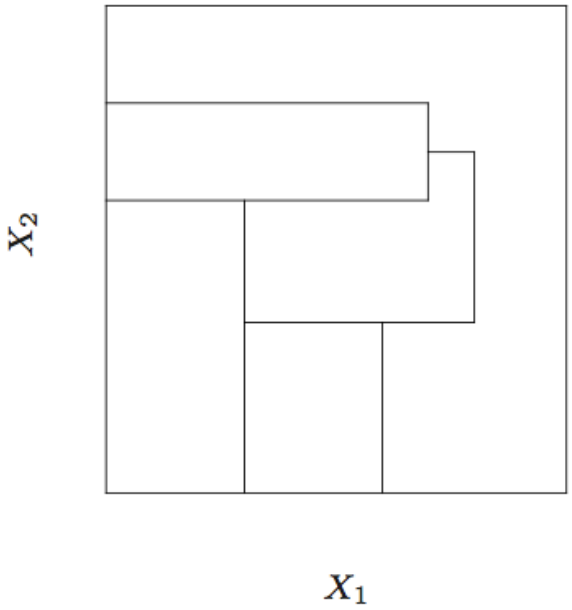
- Grow the tree by recursive binary splitting
- Prune back the tree

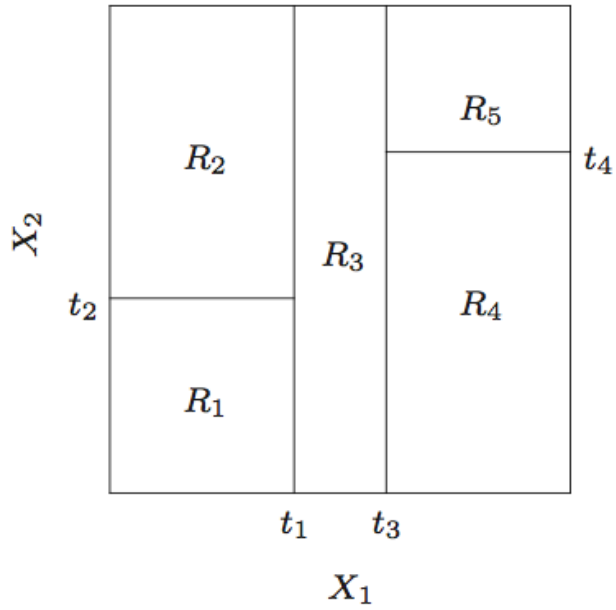
修剪枝叶 i.e. Tree pruning

some leaves may be too much

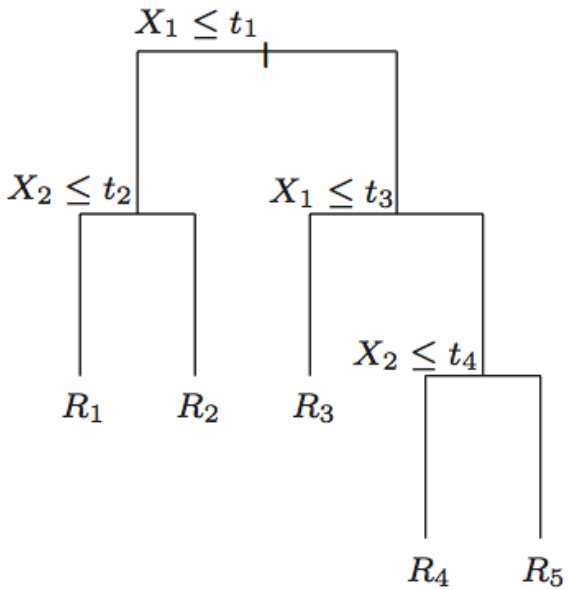
more leaves → more flexibility / complexibility

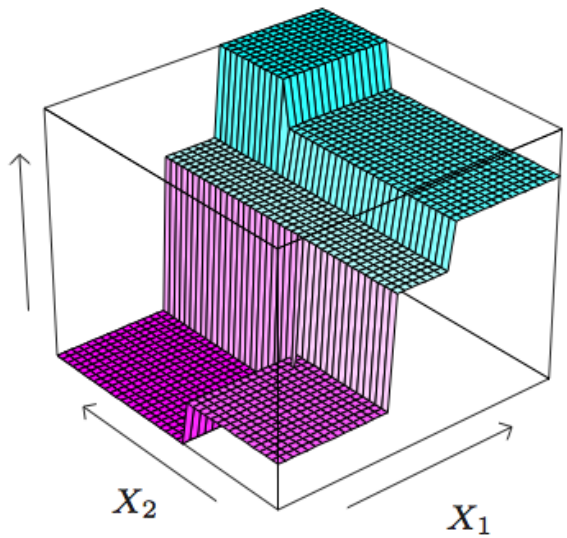
less bias



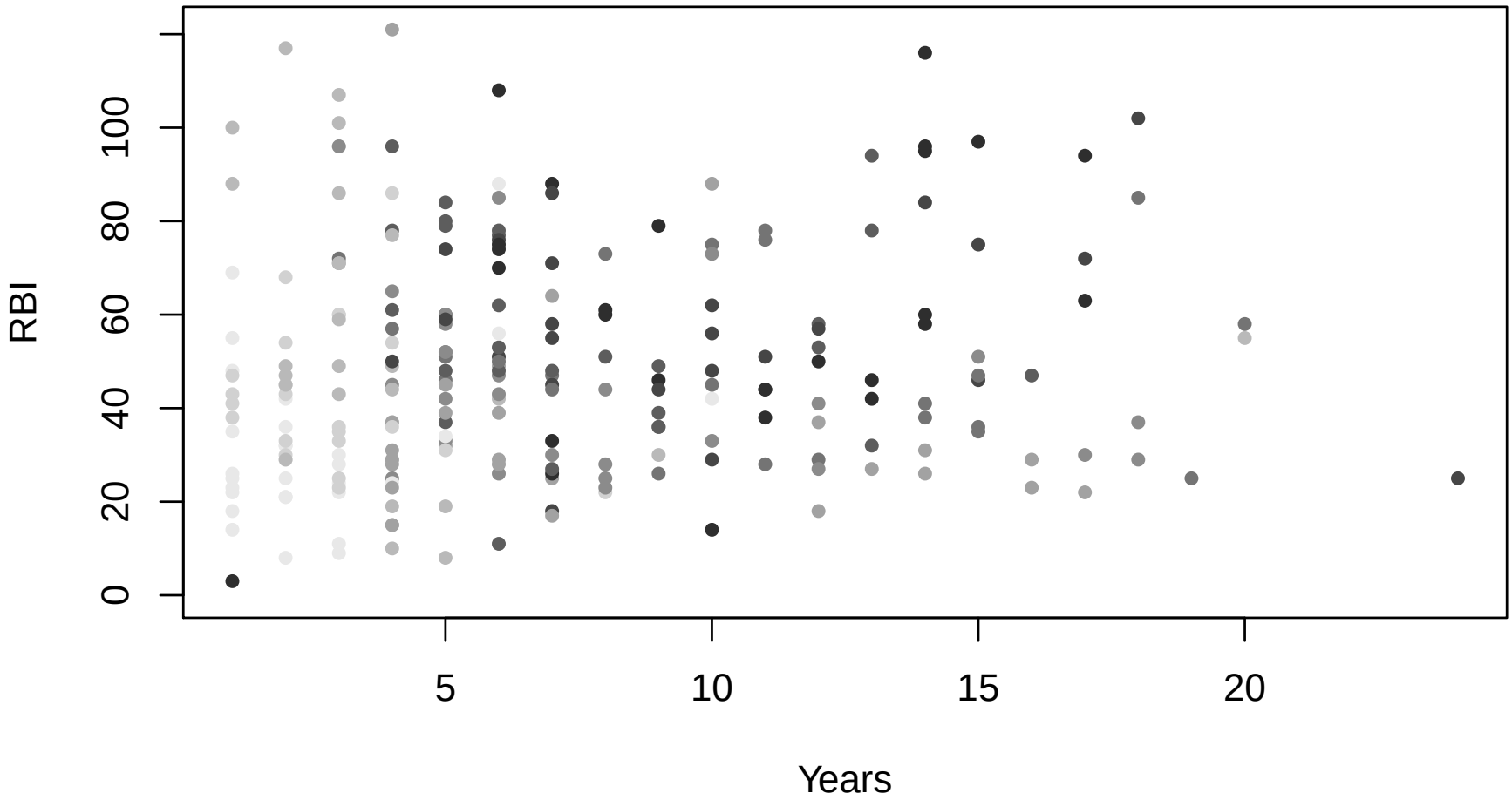


Step function



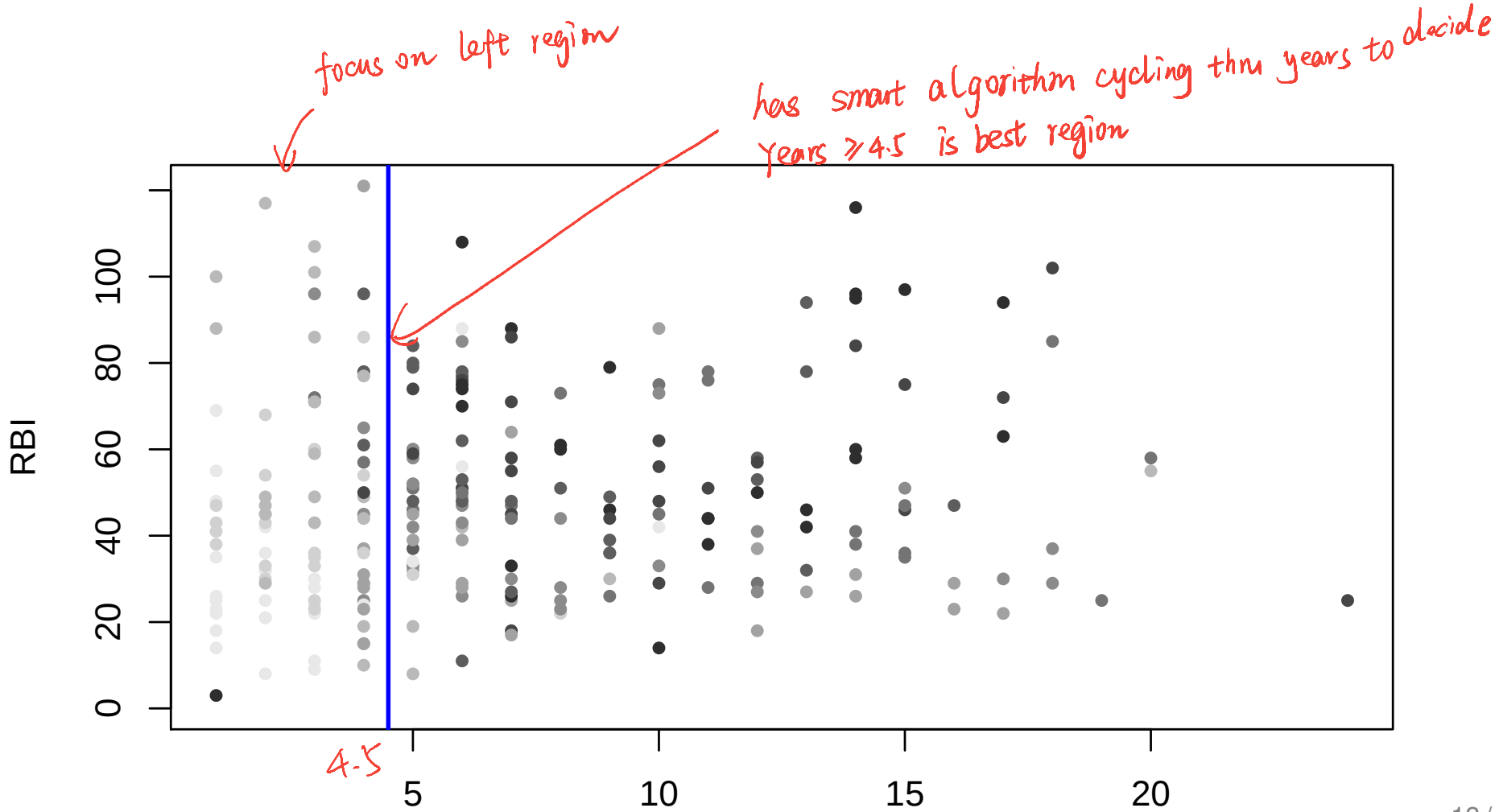


Tree growth

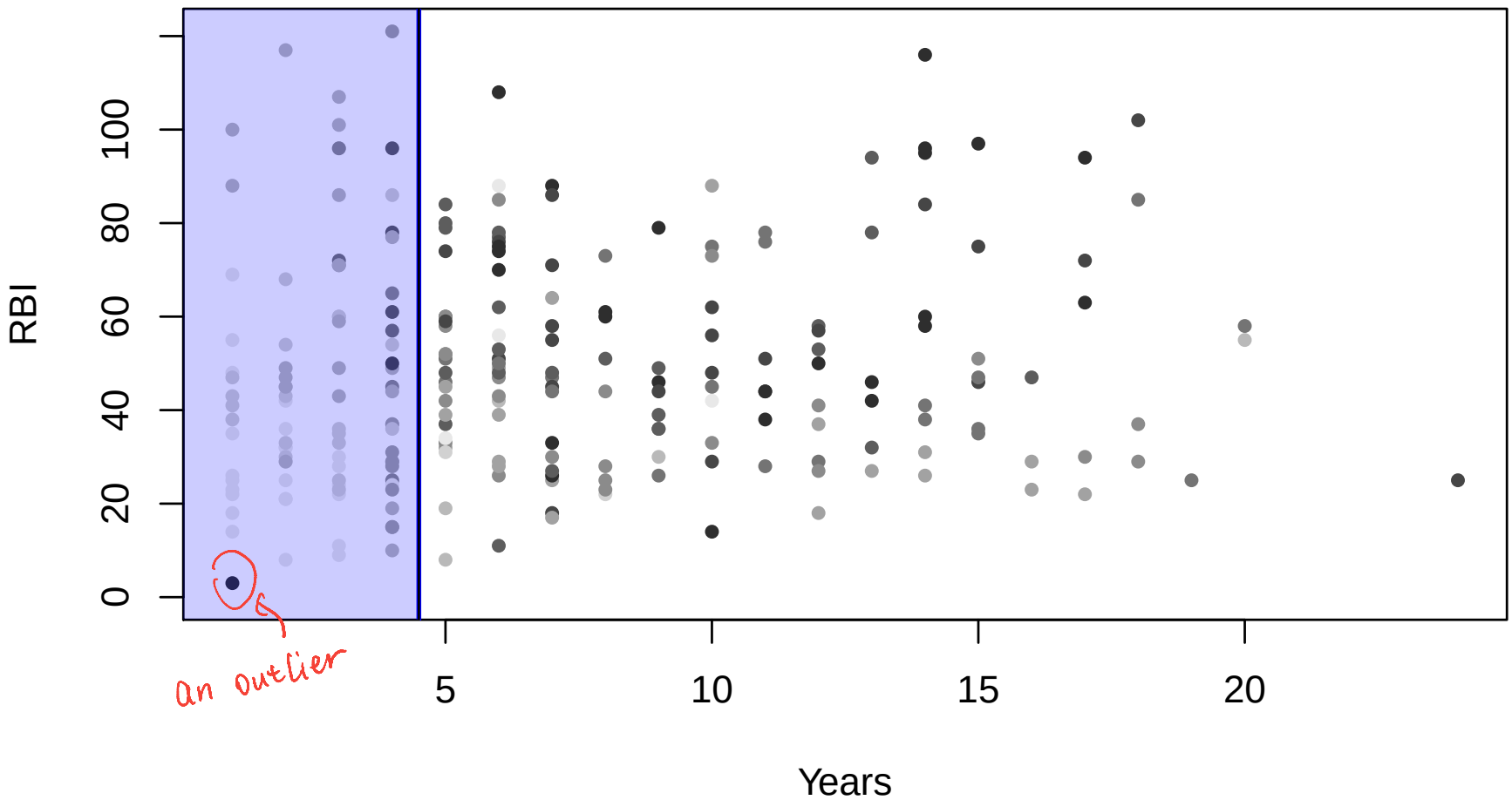


Tree growth

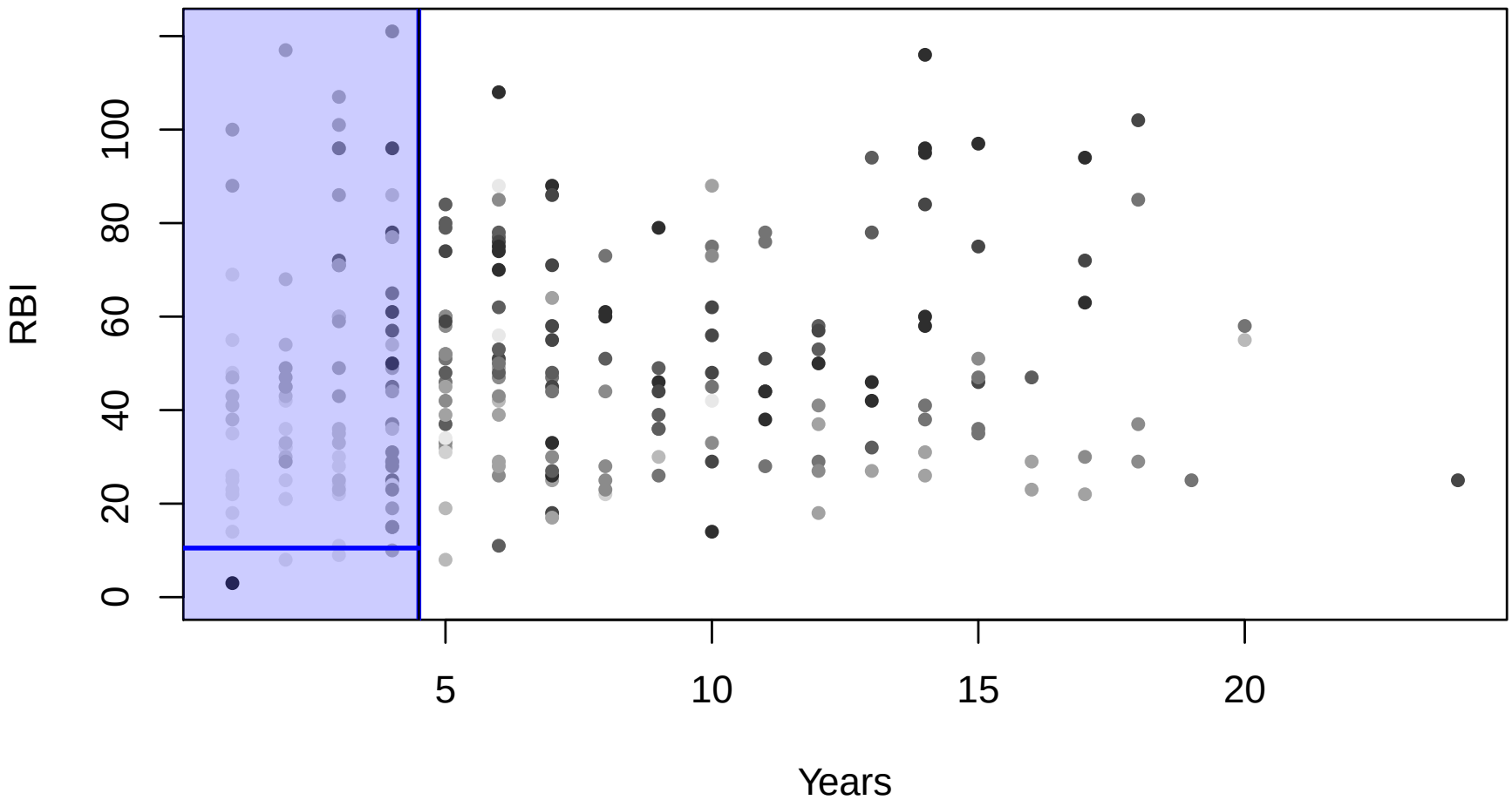
Where can we draw a horizontal or vertical line that best splits the data into two homogeneous parts?



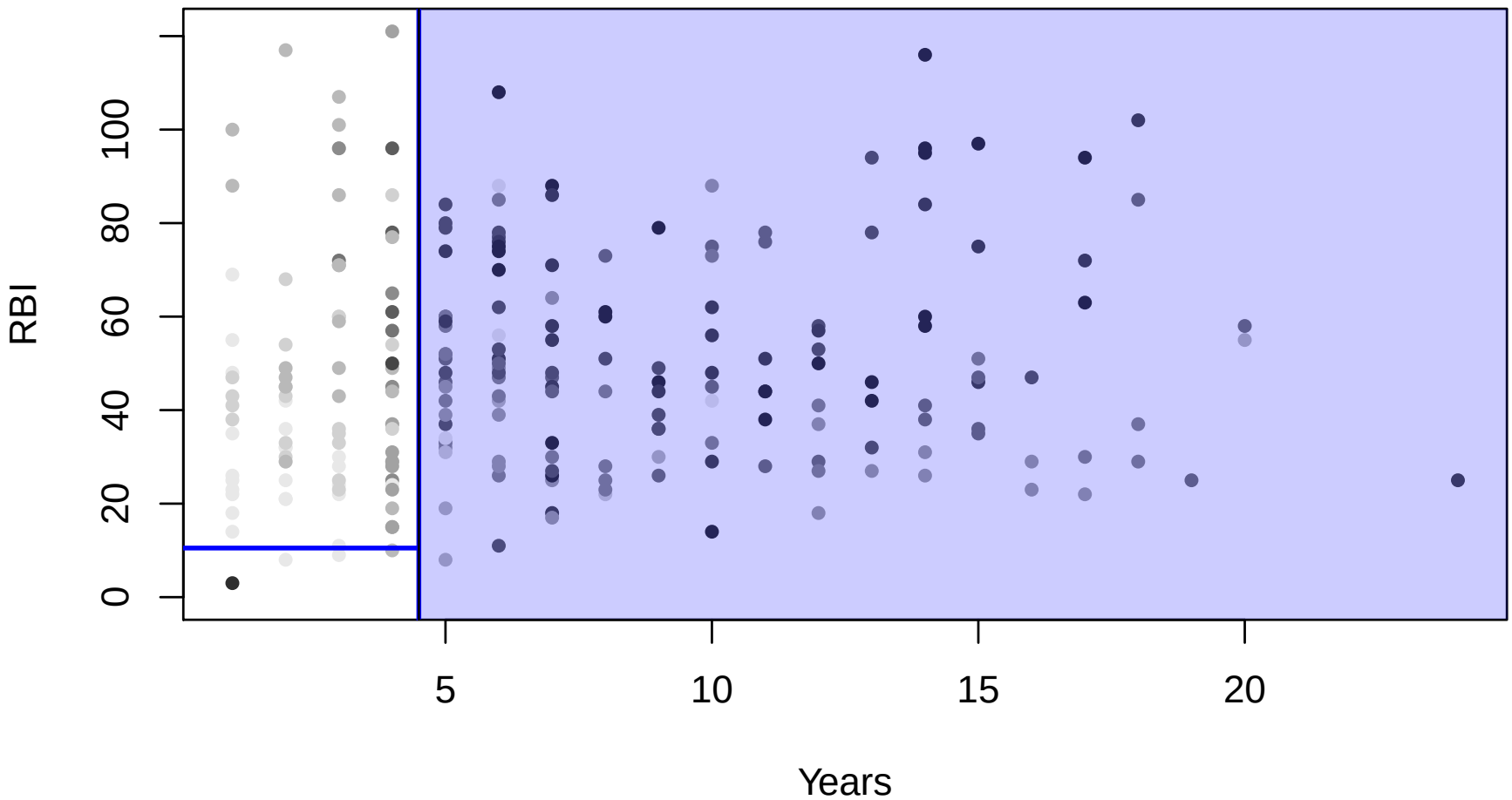
Tree growth



Tree growth

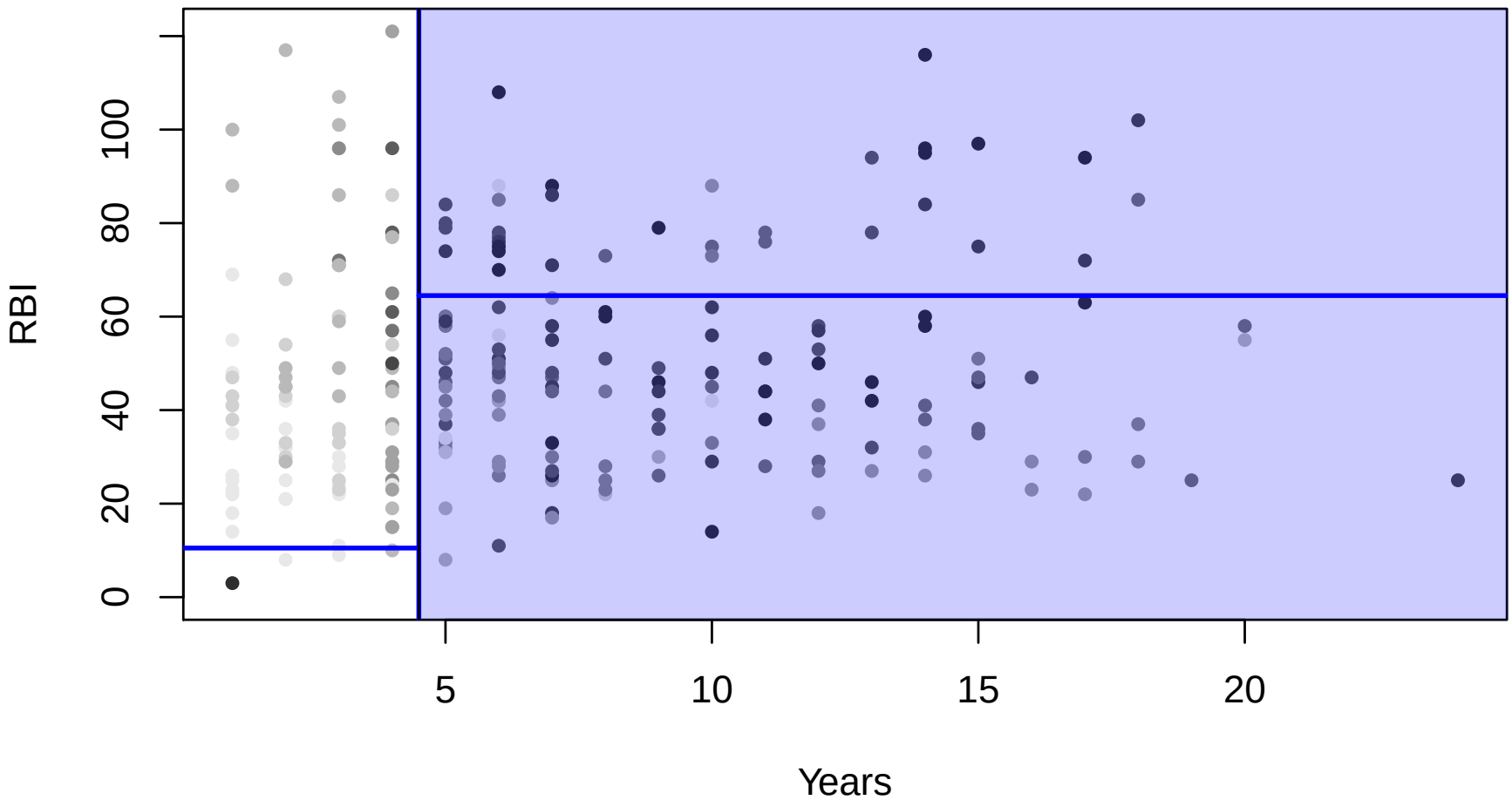


Tree growth

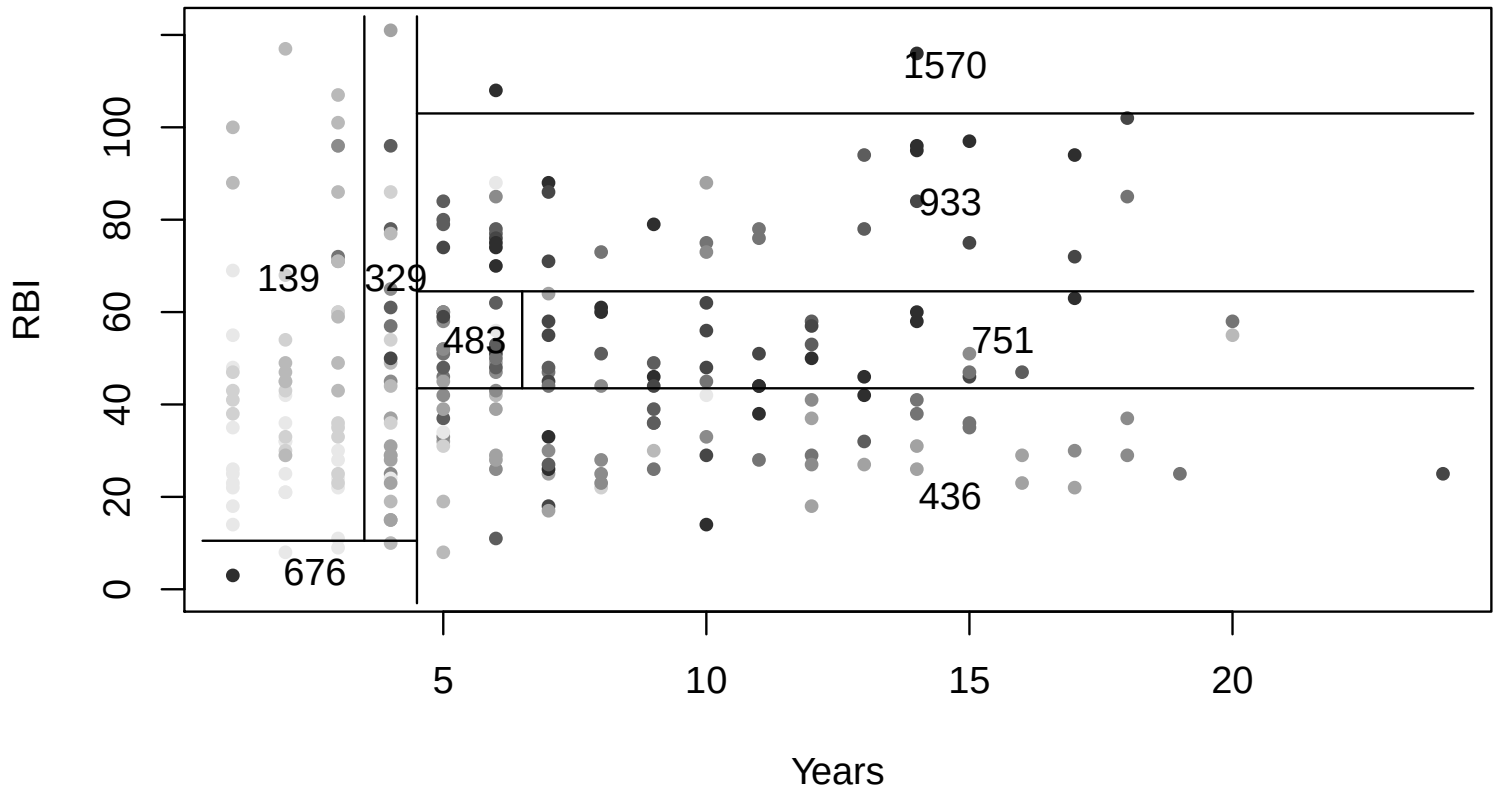


Tree growth

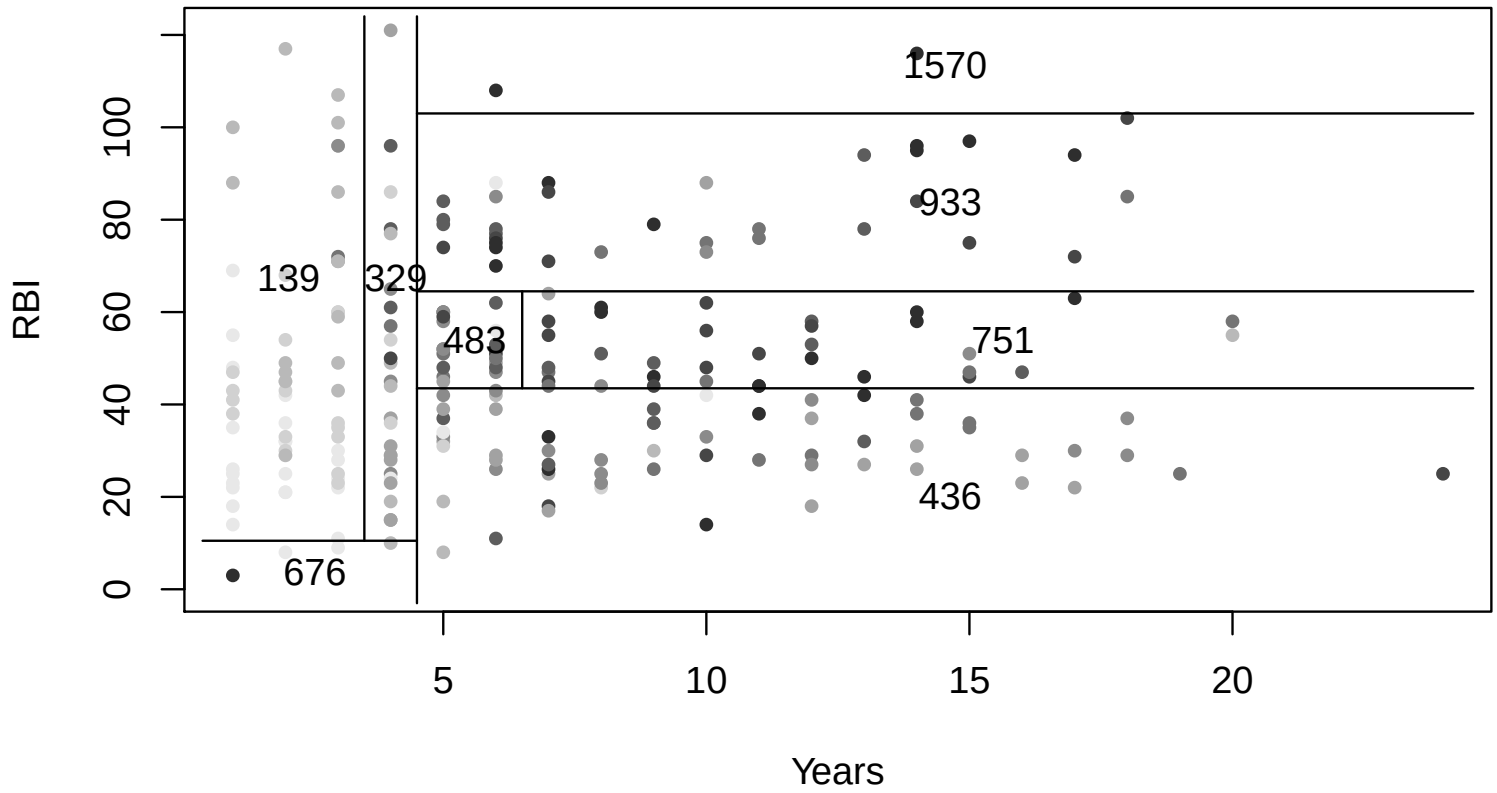
After every splitting, can treat data points in one Region as a own data set, independent of everything else



Tree growth


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

Tree growth



How do we actually pick these split-points, when X is *continuous*?

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:

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

Tree growth (Regression)

In Baseball eg. we only have 2 predictors $p=2$ X_1 = RBI X_2 = Years

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:

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

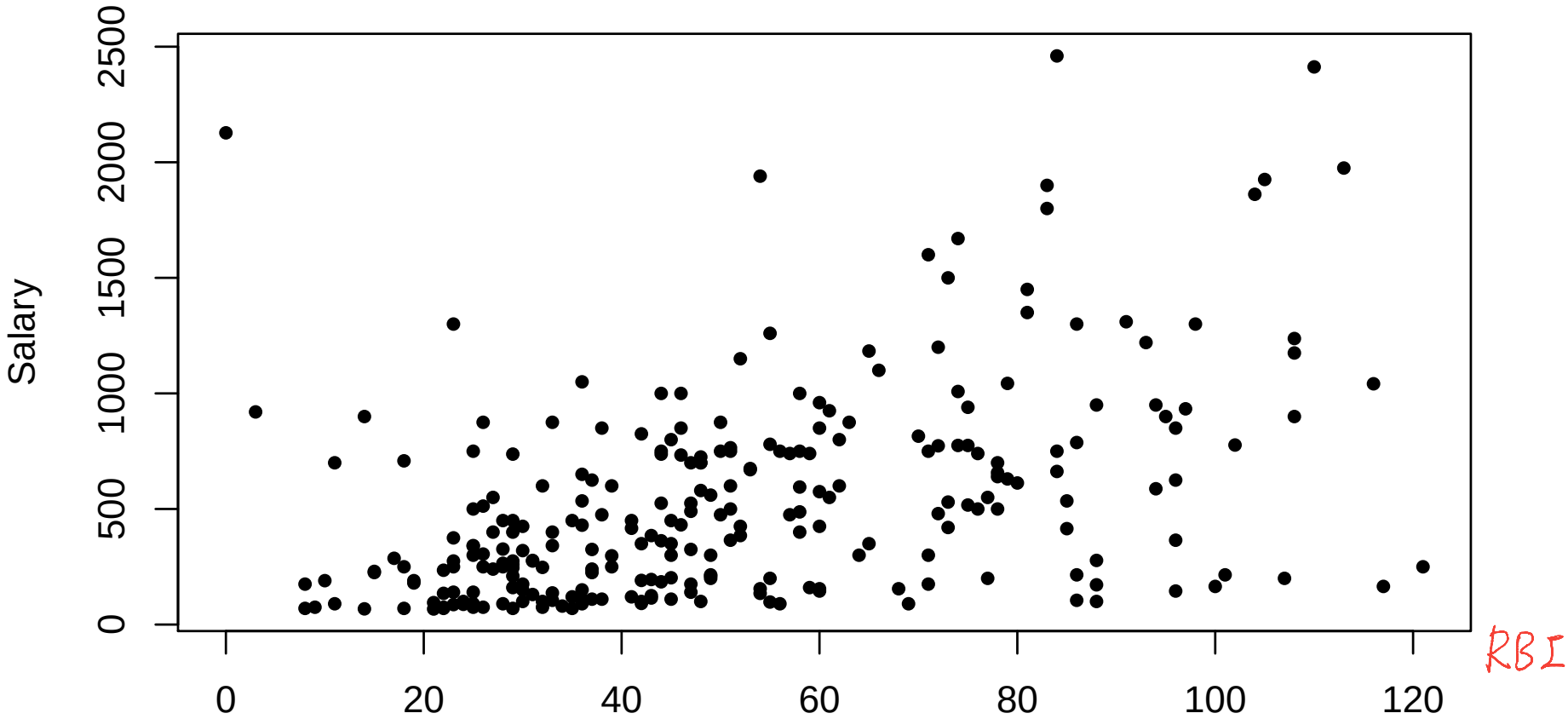
   - Evaluate (for regression trees):

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

Example: Evaluating Q_{RBI}

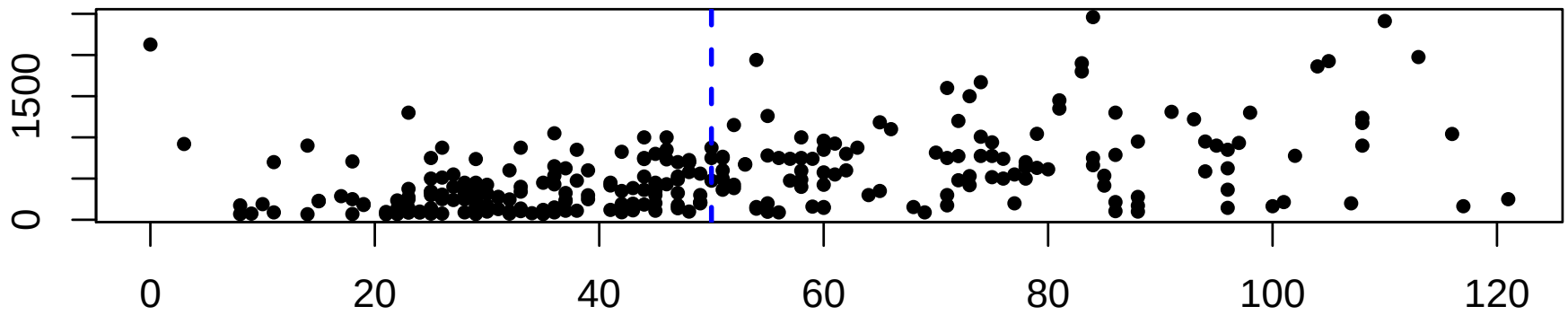
$$Q_k(s) = \sum_{i:i\in R_1(k,s)} (y_i - \bar{y}_{R_1})^2 + \sum_{i:i\in R_2(k,s)} (y_i - \bar{y}_{R_2})^2$$

Plot salary vs. RBI to find best cut point s

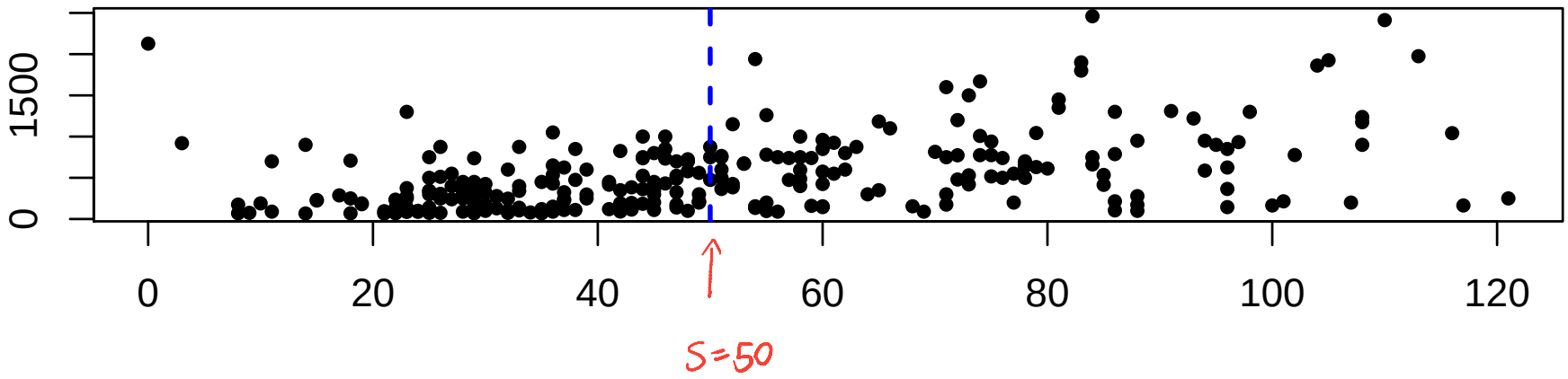


Example: Evaluating Q_{RBI}

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



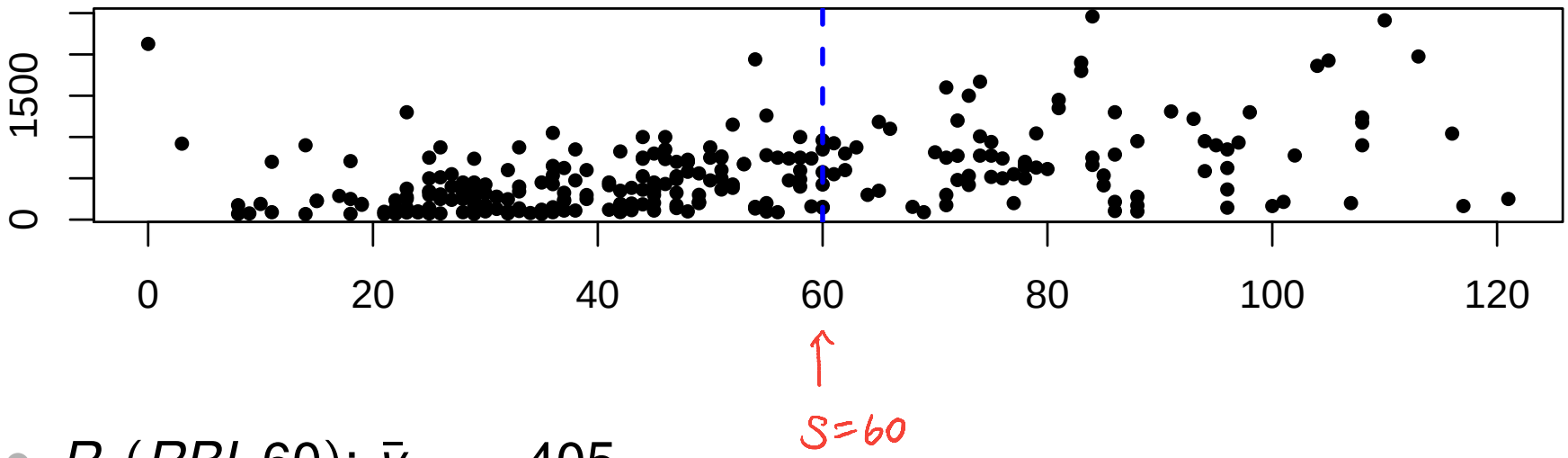
Example: Evaluating Q_{RBI}



- $R_1(RBI, 50)$: $\bar{y}_{R_1} = 359$
- $R_2(RBI, 50)$: $\bar{y}_{R_2} = 753$
-

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

Example: Evaluating Q_{RBI}



- $R_1(RBI, 60)$: $\bar{y}_{R_1} = 405$
- $R_2(RBI, 60)$: $\bar{y}_{R_2} = 802$
-

$$\begin{aligned} Q_{RBI}(60) &= \sum_{i:i\in R_1} (y_i - 405)^2 + \sum_{i:i\in R_2} (y_i - 805)^2 \\ &= 19186489 + 24943383 \\ &= 44129872 \end{aligned}$$

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

Categorical predictors

For continuous predictors, we perform a binary split by taking values below and above a cutoff. This doesn't work if a predictor X_k is categorical.

How should we deal with categorical ones?

eg. cat, dog

see detail in text book

Categorical predictors

For continuous predictors, we perform a binary split by taking values below and above a cutoff. This doesn't work if a predictor X_k is categorical.

How should we deal with categorical ones?

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

multiple combination

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

Categorical predictors

For continuous predictors, we perform a binary split by taking values below and above a cutoff. This doesn't work if a predictor X_k is categorical.

How should we deal with categorical ones?

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

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

Every possible partition of the set of unique values into 2 subsets are 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:

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

   - 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:

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

   - Evaluate (for regression trees):

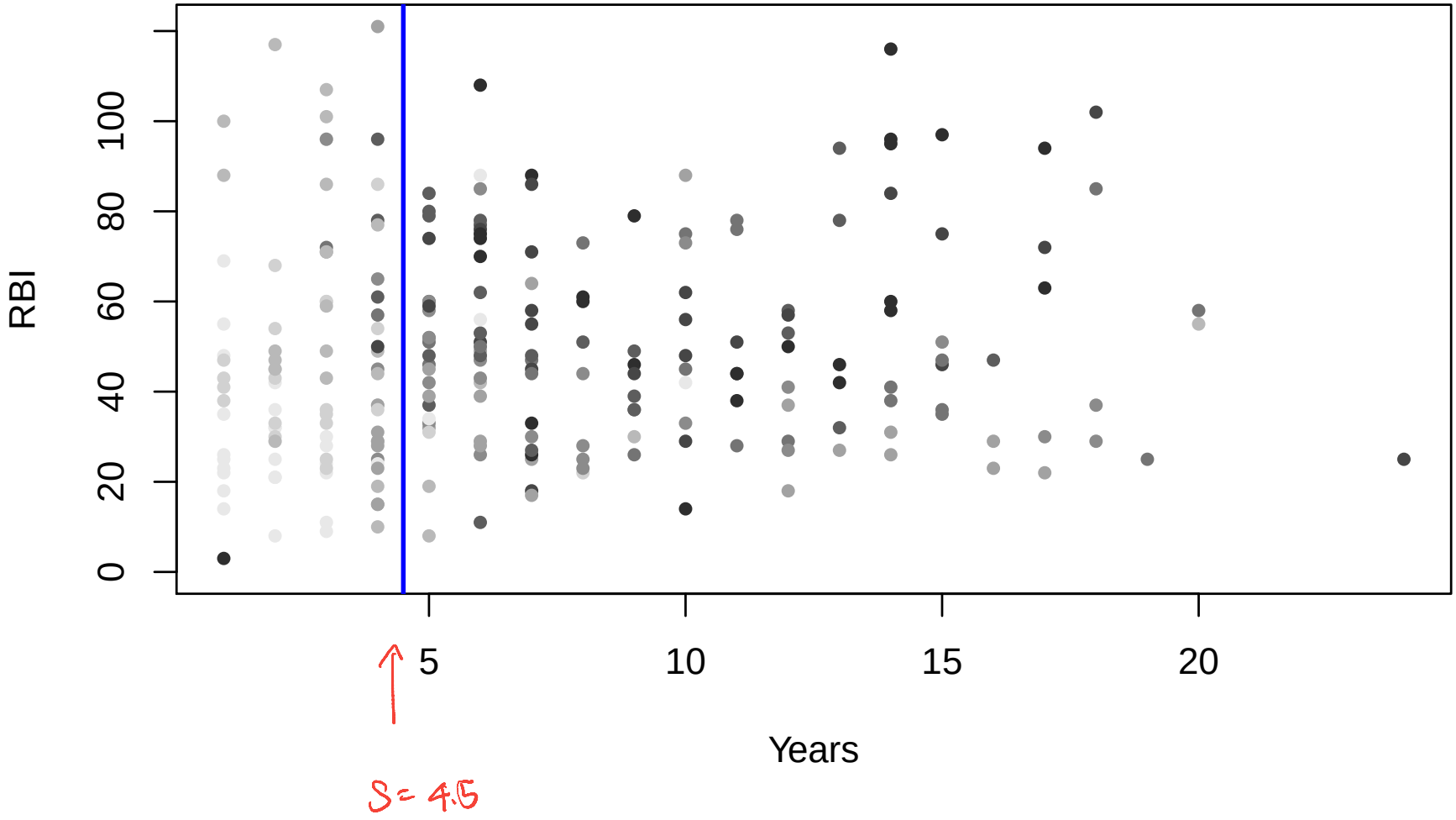
$$Q_k(s) = \sum_{i: i \in R_1(k,s)} (y_i - \bar{y}_{R_1})^2 + \sum_{i: i \in R_2(k,s)} (y_i - \bar{y}_{R_2})^2$$

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

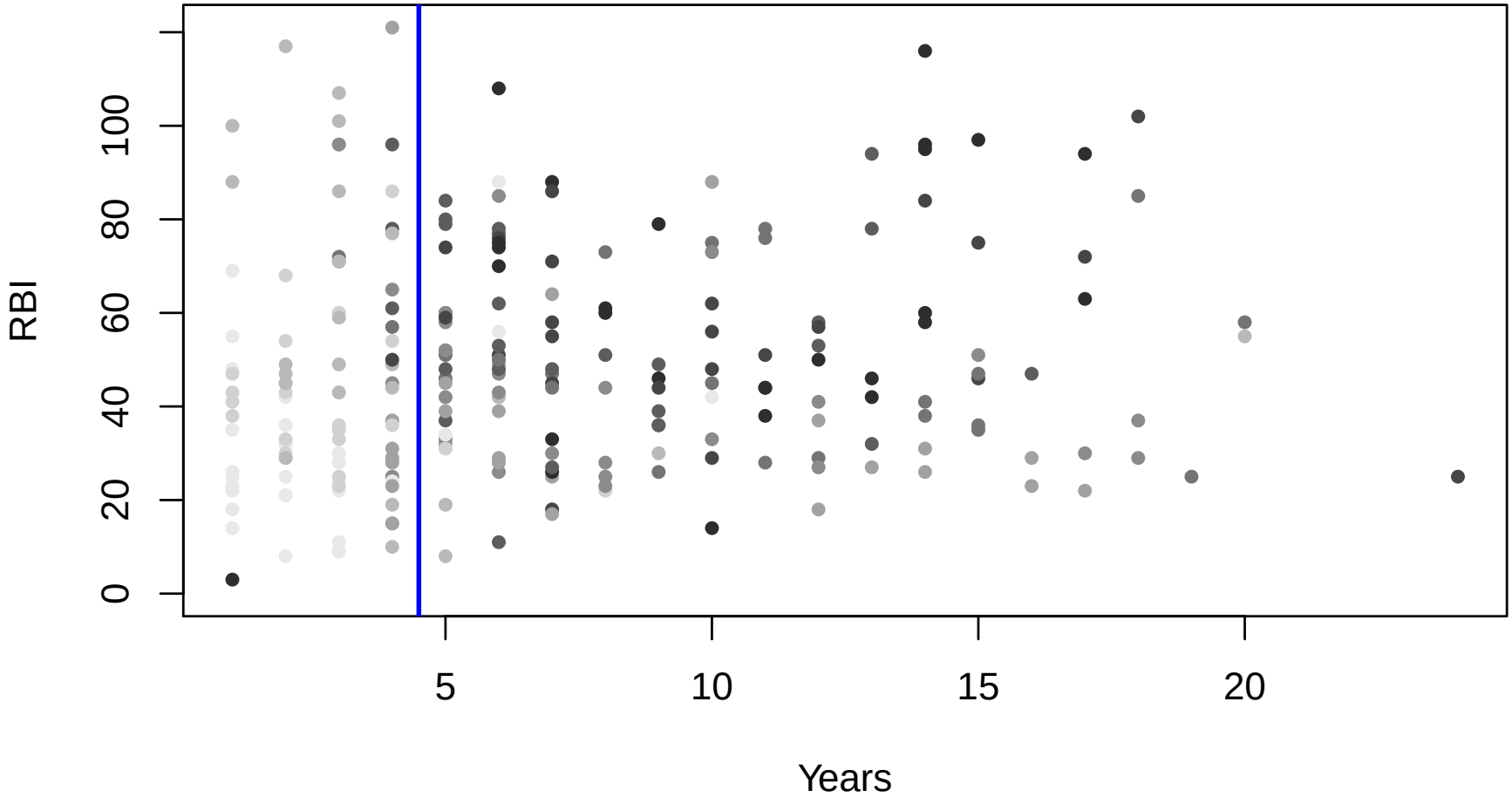
2. Find the predictor X_k with the minimum $Q_1(s_1), Q_2(s_2), \ldots, Q_p(s_p)$. Make the first binary partition along predictor X_k at cut point s_k.

$Q_{min} = Q_k(s)$

Tree growth

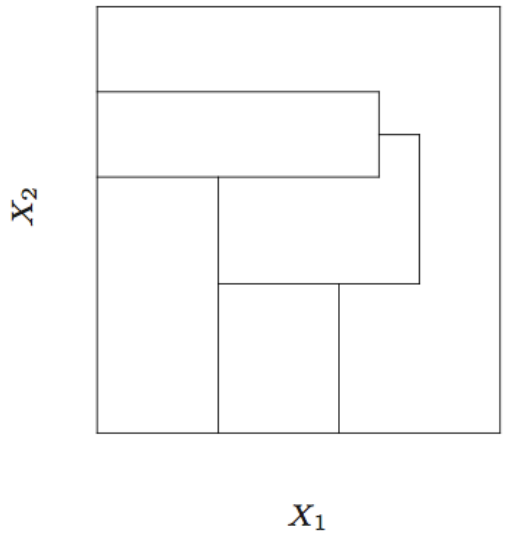


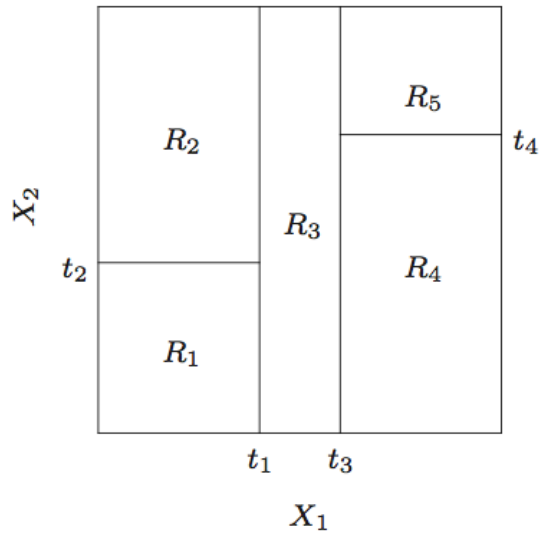
Tree growth

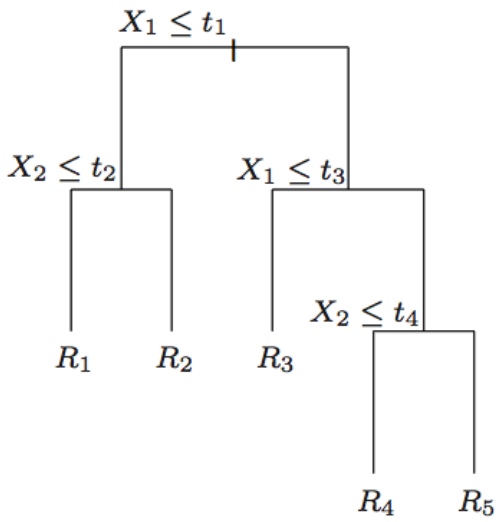


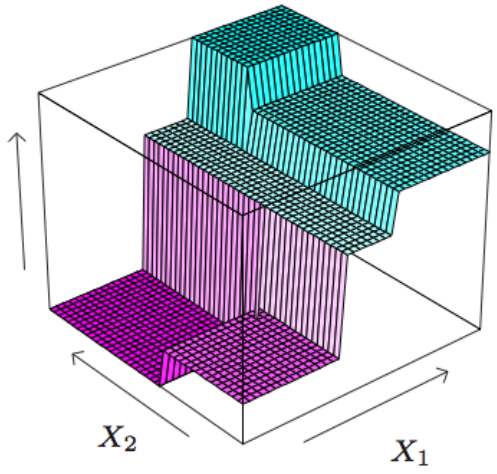
Essentially repeat the previous 2 steps on each of the resulting regions separately, iteratively. (Hence **recursive** binary partitioning.)

What if we want more?









Bias-variance Tradeoff

- 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 e.g. ≤ 5
- depth of tree has reached a maximum e.g. ≤ 100
- grow until no further splits can reduce RSS by some amount

common use in Boosting

threshold

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.

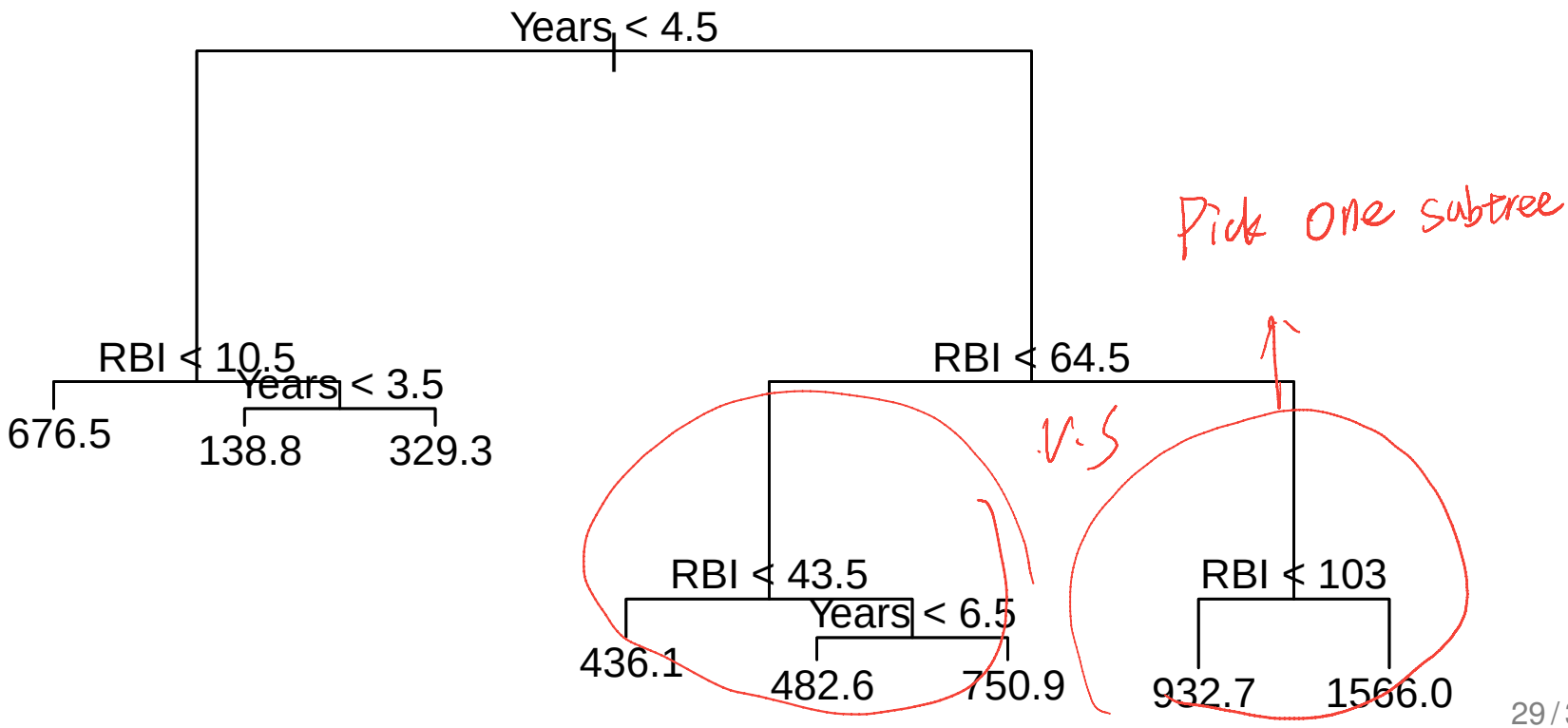
Every leaf node has only 1 data point

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 (weakest link pruning)

Cost-complexity pruning

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

Error

调整参数

α is a tuning parameter that controls for the complexity of the model, and $|T|$ is the number of regions, or the number of leaves of the 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, and $|T|$ is the number of regions, or the number of leaves of the tree.

- $\alpha = 0$ 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, and $|T|$ is the number of regions, or the number of leaves of the tree.

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

It can be shown that the set of trees corresponding to the various α form a sequence of nested subtrees, and it is computationally efficient to determine this sequence.

Tree pruning

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

Classification Trees

- $\widehat{y}_i$ for all $i \in R_j$ is *most commonly occurring class* of training observations in R_j.

$$\widehat{y}_{R_j} = \arg\max_k \widehat{p}_{jk},$$

  where $\widehat{p}_{jk}$ is the proportion of training observations in the R_j.
- No longer want to minimize RSS, but instead minimize...
  - classification error rate

$$E = \sum_{j=1}^{J} |R_j|(1 - \max_k(\widehat{p}_{jk}))$$

Classification Trees

- $\widehat{y}_i$ for all $i \in R_j$ is *most commonly occurring class* of training observations in R_j.

$$\widehat{y}_{R_j} = \arg\max_k \widehat{p}_{jk},$$

  where $\widehat{p}_{jk}$ is the proportion of training observations in the R_j.
- No longer want to minimize RSS, but instead minimize...
  - classification error rate

$$E = \sum_{j=1}^{J} |R_j|(1 - \max_k(\widehat{p}_{jk}))$$

  - Gini index

$$G = \sum_{j=1}^{J} |R_j| \sum_{k=1}^{K} \widehat{p}_{jk}(1 - \widehat{p}_{jk})$$

    Encourages higher **node purity**.

Impurity measures

Define node proportion of class k

$$\begin{aligned} \widehat{p}_{mk} &= \frac{1}{N_m} \sum_{x_i \in R_m} I(y_i = k) \\ k(m) &= \arg\max_k \widehat{p}_{mk} \end{aligned}$$

- Misclassification error: $1 - \widehat{p}_{mk(m)}$
- Gini index: $\sum_{k=1}^{K} \widehat{p}_{mk}(1 - \widehat{p}_{mk})$

Classification Trees

Suppose the dataset consisted of 800 observations split into 2 classes: (400, 400)

Classification Trees

Suppose the dataset consisted of 800 observations split into 2 classes: (400, 400)

2 possible splits:

A. Left Node: (300, 100)

   Right Node: (100, 300)

B. Left Node: (200, 400)

   Right Node: (200, 0)

Which is better?

Classification Trees

Suppose the dataset consisted of 800 observations split into 2 classes: (400, 400)

2 possible splits:

A. Left Node: (300, 100)

   Right Node: (100, 300)

B. Left Node: (200, 400)

   Right Node: (200, 0)

Misclassification error rate:

A. $400 \times \frac{100}{400} + 400 \times \frac{100}{400} = 200$

B. $600 \times \frac{200}{600} + 200 \times \frac{0}{400} = 200$

Classification Trees

Suppose the dataset consisted of 800 observations split into 2 classes: (400, 400)

2 possible splits:

A. Left Node: (300, 100)

   Right Node: (100, 300)

B. Left Node: (200, 400)

   Right Node: (200, 0)

Gini index:

A. $400((0.75)(0.25) + (0.25)(0.75)) + 400((0.25)(0.75) + (0.75)(0.25)) = 300$

B. $600((0.3\bar{3})(0.6\bar{6}) + (0.3\bar{3})(0.6\bar{6})) + 200((1)(0) + 0(1)) = 267$

Nice properties of trees

- Interpretable.
- Inbuilt feature selection. Irrelevant covariates won't be used as often as splits.
- Simple, fast implementation. Performs well with large datasets.

Issues with trees

- Instability. Trees can have high variance. As data change, tree topology can change dramatically, making interpretation difficult.
- Greedy algorithm. As such, cannot guarantee to find the globally optimal decision tree.
- Lack of smoothness. The splits lead to a “jagged” decision boundary.
- Difficulty capturing additive structure. If actual model is additive, this may not be captured by the tree with limited data.