COMPSCI 589
Lecture 9: KNN Regression, Regression Trees, and Feature Selection

Benjamin M. Marlin

College of Information and Computer Sciences
University of Massachusetts Amherst

Slides by Benjamin M. Marlin (marlin@cs.umass.edu).
Created with support from National Science Foundation Award# IIS-1350522.

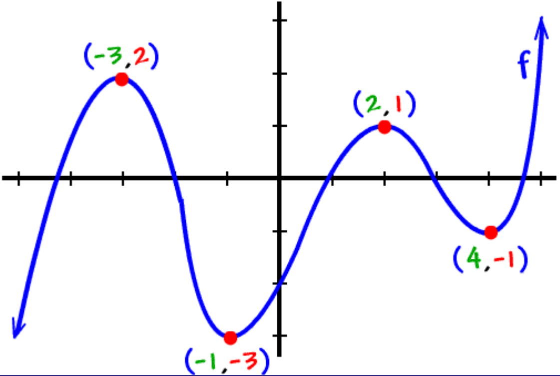
Outline

1. Review
2. KNN Regression
3. Regression Trees
4. Feature Selection

The Regression Task

Definition: The Regression Task

Given a feature vector $\mathbf{x} \in \mathbb{R}^D$, predict it's corresponding output value y.



The Regression Learning Problem

Definition: Regression Learning Problem

Given a data set of example pairs $\mathcal{D} = \{(\mathbf{x}_i, y_i), i = 1 : N\}$ where $\mathbf{x}_i \in \mathbb{R}^D$ is a feature vector and $y_i \in \mathbb{R}$ is the output, learn a function $f : \mathbb{R}^D \to \mathbb{R}$ that accurately predicts y for any feature vector $\mathbf{x}$.

Error Measures: MSE

Definition: Mean Squared Error

Given a data set of example pairs $\mathcal{D} = \{(\mathbf{x}_i, y_i), i = 1 : N\}$ and a function $f : \mathbb{R}^D \rightarrow \mathcal{Y}$, the mean squared error of f on $\mathcal{D}$ is:

$$MSE(f, \mathcal{D}) = \frac{1}{N} \sum_{i=1}^{N} (y_i - f(\mathbf{x}_i))^2$$

Error Measures: MSE

Definition: Mean Squared Error

Given a data set of example pairs $\mathcal{D} = \{(\mathbf{x}_i, y_i), i = 1 : N\}$ and a function $f : \mathbb{R}^D \rightarrow \mathcal{Y}$, the mean squared error of f on $\mathcal{D}$ is:

$$MSE(f, \mathcal{D}) = \frac{1}{N} \sum_{i=1}^{N} (y_i - f(\mathbf{x}_i))^2$$

Related measures include:
Sum of Squared Errors: $SSE(f, \mathcal{D}) = N \cdot MSE(f, \mathcal{D})$
Risidual Sum of Squares: $RSS(f, \mathcal{D}) = N \cdot MSE(f, \mathcal{D})$
Root Mean Squared Error: $RMSE(f, \mathcal{D}) = \sqrt{MSE(f, \mathcal{D})}$

Outline

K Nearest Neighbors Regression

The KNN regression is a non-parametric regression method that simply stores the training data $\mathcal{D}$ and makes a prediction for each new instance $\mathbf{x}$ using an average over it's set of K nearest neighbors $\mathcal{N}_K(\mathbf{x})$ computed using any distance function $d : \mathbb{R}^D \times \mathbb{R}^D \to \mathbb{R}$.

K Nearest Neighbors Regression

The KNN regression is a non-parametric regression method that simply stores the training data $\mathcal{D}$ and makes a prediction for each new instance $\mathbf{x}$ using an average over it's set of K nearest neighbors $\mathcal{N}_K(\mathbf{x})$ computed using any distance function $d : \mathbb{R}^D \times \mathbb{R}^D \rightarrow \mathbb{R}$.

KNN Regression Function

$$f_{KNN}(\mathbf{x}) = \frac{1}{K} \sum_{i \in \mathcal{N}_K(\mathbf{x})} y_i$$

K Nearest Neighbors Regression

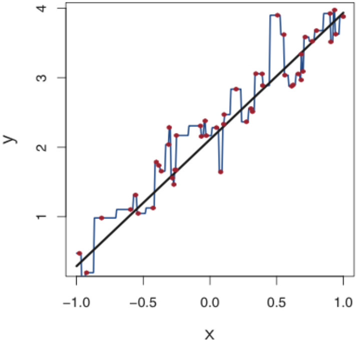
The KNN regression is a non-parametric regression method that simply stores the training data $\mathcal{D}$ and makes a prediction for each new instance $\mathbf{x}$ using an average over it's set of K nearest neighbors $\mathcal{N}_K(\mathbf{x})$ computed using any distance function $d : \mathbb{R}^D \times \mathbb{R}^D \to \mathbb{R}$.

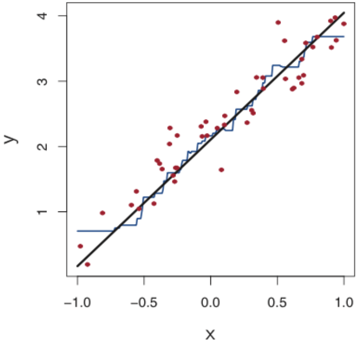
KNN Regression Function

$$f_{KNN}(\mathbf{x}) = \frac{1}{K} \sum_{i \in \mathcal{N}_K(\mathbf{x})} y_i$$

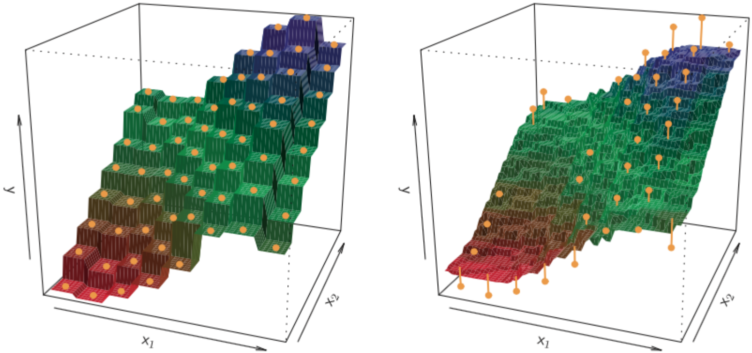
As with classification, use of KNN requires choosing the distance function d and the number of neighbors K.

Example: 1D KNN (K=1 vs K=9)





Example: 2D KNN (K=1 vs K=9)



Weighted KNN Regression

- Instead of giving all of the K neighbors equal weight in the average, a distance-weighted average can be used:

$$f_{KNN}(\mathbf{x}) = \frac{\sum_{i \in \mathcal{N}_K(\mathbf{x})} w_i y_i}{\sum_{i \in \mathcal{N}_K(\mathbf{x})} w_i}$$

$$w_i = \exp(-\alpha d_i)$$

Outline

1. Review
2. KNN Regression
3. Regression Trees
4. Feature Selection

Regression Trees

- A regression tree makes predictions using a conjunction of rules organized into a binary tree structure.

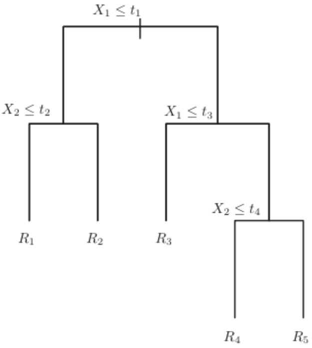
Regression Trees

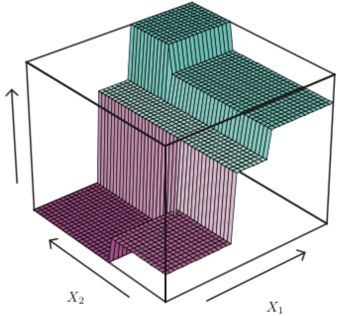
- A regression tree makes predictions using a conjunction of rules organized into a binary tree structure.
- Each internal node in a regression tree contains a rule of the form $(x_d < t)$ or $(x_d = t)$ that tests a single data dimension d against a single threshold value t and assigns the data case to it's left or right sub-tree according to the result.

Regression Trees

- A regression tree makes predictions using a conjunction of rules organized into a binary tree structure.
- Each internal node in a regression tree contains a rule of the form $(x_d < t)$ or $(x_d = t)$ that tests a single data dimension d against a single threshold value t and assigns the data case to it's left or right sub-tree according to the result.
- A data case is routed through the tree from the root to a leaf. Each leaf node is associated with a predicted output, and a data case is assigned the output of the leaf node it is routed to.

Example: 2D Regression Trees





Building Regression Trees

Algorithm 1 $BuildTree(Root, \mathcal{D}, h, minS, maxD)$

$d, t = BestSplit(\mathcal{D})$

Building Regression Trees

Algorithm 2 $BuildTree(Root, \mathcal{D}, h, minS, maxD)$

$d, t = BestSplit(\mathcal{D})$

$\mathcal{D}_1 = \{(y_i, \mathbf{x}_i) | x_{di} \leq t\}, \mathcal{D}_2 = \{(y_i, \mathbf{x}_i) | x_{di} > t\}$

Building Regression Trees

Algorithm 3 $BuildTree(Root, \mathcal{D}, h, minS, maxD)$

$d, t = BestSplit(\mathcal{D})$

$\mathcal{D}_1 = \{(y_i, \mathbf{x}_i) | x_{di} \leq t\}, \mathcal{D}_2 = \{(y_i, \mathbf{x}_i) | x_{di} > t\}$

if $|\mathcal{D}_1| <= minS$ or $h + 1 \geq maxD$ **then**

Building Regression Trees

Algorithm 4 $BuildTree(Root, \mathcal{D}, h, minS, maxD)$

$d, t = BestSplit(\mathcal{D})$

$\mathcal{D}_1 = \{(y_i, \mathbf{x}_i) | x_{di} \leq t\}, \mathcal{D}_2 = \{(y_i, \mathbf{x}_i) | x_{di} > t\}$

if $|\mathcal{D}_1| <= minS$ or $h + 1 \geq maxD$ **then**

$\quad Root.RightChild.Prediction = \frac{1}{|\mathcal{D}_1|} \sum_{y \in \mathcal{D}_1} y$

Building Regression Trees

Algorithm 5 $BuildTree(Root, \mathcal{D}, h, minS, maxD)$

$d, t = BestSplit(\mathcal{D})$
$\mathcal{D}_1 = \{(y_i, \mathbf{x}_i) | x_{di} \leq t\}, \mathcal{D}_2 = \{(y_i, \mathbf{x}_i) | x_{di} > t\}$
if $|\mathcal{D}_1| <= minS$ or $h + 1 \geq maxD$ **then**
    $Root.RightChild.Prediction = \frac{1}{|\mathcal{D}_1|} \sum_{y \in \mathcal{D}_1} y$
else

Building Regression Trees

Algorithm 6 $BuildTree(Root, \mathcal{D}, h, minS, maxD)$

$d, t = BestSplit(\mathcal{D})$

$\mathcal{D}_1 = \{(y_i, \mathbf{x}_i) | x_{di} \leq t\}, \mathcal{D}_2 = \{(y_i, \mathbf{x}_i) | x_{di} > t\}$

if $|\mathcal{D}_1| <= minS$ or $h + 1 \geq maxD$ **then**

  $Root.RightChild.Prediction = \frac{1}{|\mathcal{D}_1|} \sum_{y \in \mathcal{D}_1} y$

else

  $BuildTree(Root.RightChild, \mathcal{D}_1, h + 1, minS, maxD)$

Building Regression Trees

Algorithm 7 $BuildTree(Root, \mathcal{D}, h, minS, maxD)$

$d, t = BestSplit(\mathcal{D})$

$\mathcal{D}_1 = \{(y_i, \mathbf{x}_i) | x_{di} \leq t\}, \mathcal{D}_2 = \{(y_i, \mathbf{x}_i) | x_{di} > t\}$

if $|\mathcal{D}_1| <= minS$ or $h + 1 \geq maxD$ **then**

$\quad Root.RightChild.Prediction = \frac{1}{|\mathcal{D}_1|} \sum_{y \in \mathcal{D}_1} y$

else

$\quad BuildTree(Root.RightChild, \mathcal{D}_1, h + 1, minS, maxD)$

Building Regression Trees

Algorithm 8 $BuildTree(Root, \mathcal{D}, h, minS, maxD)$

$d, t = BestSplit(\mathcal{D})$

$\mathcal{D}_1 = \{(y_i, \mathbf{x}_i) | x_{di} \leq t\}, \mathcal{D}_2 = \{(y_i, \mathbf{x}_i) | x_{di} > t\}$

if $|\mathcal{D}_1| <= minS$ or $h + 1 \geq maxD$ **then**

  $Root.RightChild.Prediction = \frac{1}{|\mathcal{D}_1|} \sum_{y \in \mathcal{D}_1} y$

else

  $BuildTree(Root.RightChild, \mathcal{D}_1, h + 1, minS, maxD)$

if $|\mathcal{D}_2| <= minS$ or $h + 1 \geq maxD$ **then**

Building Regression Trees

Algorithm 9 $BuildTree(Root, \mathcal{D}, h, minS, maxD)$

$d, t = BestSplit(\mathcal{D})$

$\mathcal{D}_1 = \{(y_i, \mathbf{x}_i) | x_{di} \leq t\}, \mathcal{D}_2 = \{(y_i, \mathbf{x}_i) | x_{di} > t\}$

if $|\mathcal{D}_1| <= minS$ or $h + 1 \geq maxD$ **then**

$\quad Root.RightChild.Prediction = \frac{1}{|\mathcal{D}_1|} \sum_{y \in \mathcal{D}_1} y$

else

$\quad BuildTree(Root.RightChild, \mathcal{D}_1, h + 1, minS, maxD)$

if $|\mathcal{D}_2| <= minS$ or $h + 1 \geq maxD$ **then**

$\quad Root.LeftChild.Prediction = \frac{1}{|\mathcal{D}_2|} \sum_{y \in \mathcal{D}_2} y$

Building Regression Trees

Algorithm 10 $BuildTree(Root, \mathcal{D}, h, minS, maxD)$

$d, t = BestSplit(\mathcal{D})$

$\mathcal{D}_1 = \{(y_i, \mathbf{x}_i) | x_{di} \leq t\}, \mathcal{D}_2 = \{(y_i, \mathbf{x}_i) | x_{di} > t\}$

if $|\mathcal{D}_1| <= minS$ or $h + 1 \geq maxD$ **then**

$\quad Root.RightChild.Prediction = \frac{1}{|\mathcal{D}_1|} \sum_{y \in \mathcal{D}_1} y$

else

$\quad BuildTree(Root.RightChild, \mathcal{D}_1, h + 1, minS, maxD)$

if $|\mathcal{D}_2| <= minS$ or $h + 1 \geq maxD$ **then**

$\quad Root.LeftChild.Prediction = \frac{1}{|\mathcal{D}_2|} \sum_{y \in \mathcal{D}_2} y$

else

Building Regression Trees

Algorithm 11 $BuildTree(Root, \mathcal{D}, h, minS, maxD)$

$d, t = BestSplit(\mathcal{D})$

$\mathcal{D}_1 = \{(y_i, \mathbf{x}_i) | x_{di} \leq t\}, \mathcal{D}_2 = \{(y_i, \mathbf{x}_i) | x_{di} > t\}$

if $|\mathcal{D}_1| <= minS$ or $h + 1 \geq maxD$ **then**

  $Root.RightChild.Prediction = \frac{1}{|\mathcal{D}_1|} \sum_{y \in \mathcal{D}_1} y$

else

  $BuildTree(Root.RightChild, \mathcal{D}_1, h + 1, minS, maxD)$

if $|\mathcal{D}_2| <= minS$ or $h + 1 \geq maxD$ **then**

  $Root.LeftChild.Prediction = \frac{1}{|\mathcal{D}_2|} \sum_{y \in \mathcal{D}_2} y$

else

  $BuildTree(Root.LeftChild, \mathcal{D}_2, h + 1, minS, maxD)$

Building Regression Trees

Algorithm 12 $BuildTree(Root, \mathcal{D}, h, minS, maxD)$

$d, t = BestSplit(\mathcal{D})$

$\mathcal{D}_1 = \{(y_i, \mathbf{x}_i) | x_{di} \leq t\}, \mathcal{D}_2 = \{(y_i, \mathbf{x}_i) | x_{di} > t\}$

if $|\mathcal{D}_1| <= minS$ or $h + 1 \geq maxD$ **then**

$\quad Root.RightChild.Prediction = \frac{1}{|\mathcal{D}_1|} \sum_{y \in \mathcal{D}_1} y$

else

$\quad BuildTree(Root.RightChild, \mathcal{D}_1, h + 1, minS, maxD)$

if $|\mathcal{D}_2| <= minS$ or $h + 1 \geq maxD$ **then**

$\quad Root.LeftChild.Prediction = \frac{1}{|\mathcal{D}_2|} \sum_{y \in \mathcal{D}_2} y$

else

$\quad BuildTree(Root.LeftChild, \mathcal{D}_2, h + 1, minS, maxD)$

Building Regression Trees

Algorithm 13 $BuildTree(Root, \mathcal{D}, h, minS, maxD)$

$d, t = BestSplit(\mathcal{D})$

$\mathcal{D}_1 = \{(y_i, \mathbf{x}_i) | x_{di} \leq t\}$, $\mathcal{D}_2 = \{(y_i, \mathbf{x}_i) | x_{di} > t\}$

if $|\mathcal{D}_1| <= minS$ or $h + 1 \geq maxD$ **then**

  $Root.RightChild.Prediction = \frac{1}{|\mathcal{D}_1|} \sum_{y \in \mathcal{D}_1} y$

else

  $BuildTree(Root.RightChild, \mathcal{D}_1, h + 1, minS, maxD)$

if $|\mathcal{D}_2| <= minS$ or $h + 1 \geq maxD$ **then**

  $Root.LeftChild.Prediction = \frac{1}{|\mathcal{D}_2|} \sum_{y \in \mathcal{D}_2} y$

else

  $BuildTree(Root.LeftChild, \mathcal{D}_2, h + 1, minS, maxD)$

$Root.d = d, Root.t = t$

Building Regression Trees

Algorithm 14 $BuildTree(Root, \mathcal{D}, h, minS, maxD)$

$d, t = BestSplit(\mathcal{D})$

$\mathcal{D}_1 = \{(y_i, \mathbf{x}_i) | x_{di} \leq t\}, \mathcal{D}_2 = \{(y_i, \mathbf{x}_i) | x_{di} > t\}$

if $|\mathcal{D}_1| <= minS$ or $h + 1 \geq maxD$ **then**

  $Root.RightChild.Prediction = \frac{1}{|\mathcal{D}_1|} \sum_{y \in \mathcal{D}_1} y$

else

  $BuildTree(Root.RightChild, \mathcal{D}_1, h + 1, minS, maxD)$

if $|\mathcal{D}_2| <= minS$ or $h + 1 \geq maxD$ **then**

  $Root.LeftChild.Prediction = \frac{1}{|\mathcal{D}_2|} \sum_{y \in \mathcal{D}_2} y$

else

  $BuildTree(Root.LeftChild, \mathcal{D}_2, h + 1, minS, maxD)$

$Root.d = d, Root.t = t$

return $Root$

Finding the Best Split

Algorithm 15 *BestSplit*$(\mathcal{D})$

Finding the Best Split

Algorithm 16 *BestSplit*($\mathcal{D}$)

for d from 1 to D **do**

Finding the Best Split

Algorithm 17 $BestSplit(\mathcal{D})$

for d from 1 to D **do**

$\mathbf{s} = sort(\{x_{d1}, ..., x_{dN}\})$

Finding the Best Split

Algorithm 18 *BestSplit*($\mathcal{D}$)

for d from 1 to D **do**
    $\mathbf{s} = sort(\{x_{d1}, ..., x_{dN}\})$
    for t in $\{(s_i + s_{i+1})/2 | i = 1...N-1\}$ **do**

Finding the Best Split

Algorithm 19 $BestSplit(\mathcal{D})$

for d from 1 to D **do**
  $\mathbf{s} = sort(\{x_{d1}, ..., x_{dN}\})$
  for t in $\{(s_i + s_{i+1})/2 | i = 1...N-1\}$ **do**
    $\mathcal{D}_1 = \{(y_i, \mathbf{x}_i) | x_{di} \leq t\}$

Finding the Best Split

Algorithm 20 *BestSplit*$(\mathcal{D})$

for d from 1 to D **do**
  $\mathbf{s} = sort(\{x_{d1}, ..., x_{dN}\})$
  for t in $\{(s_i + s_{i+1})/2 | i = 1...N-1\}$ **do**
    $\mathcal{D}_1 = \{(y_i, \mathbf{x}_i) | x_{di} \leq t\}$
    $\mathcal{D}_2 = \{(y_i, \mathbf{x}_i) | x_{di} > t\}$

Finding the Best Split

Algorithm 21 $BestSplit(\mathcal{D})$

for d from 1 to D **do**
  $\mathbf{s} = sort(\{x_{d1}, ..., x_{dN}\})$
  for t in $\{(s_i + s_{i+1})/2 | i = 1...N-1\}$ **do**
    $\mathcal{D}_1 = \{(y_i, \mathbf{x}_i) | x_{di} \leq t\}$
    $\mathcal{D}_2 = \{(y_i, \mathbf{x}_i) | x_{di} > t\}$
    $\bar{y}_1 = \frac{1}{|\mathcal{D}_1|} \sum_{y \in \mathcal{D}_1} y$

Finding the Best Split

Algorithm 22 *BestSplit*($\mathcal{D}$)

for d from 1 to D **do**
  $\mathbf{s} = sort(\{x_{d1}, ..., x_{dN}\})$
  for t in $\{(s_i + s_{i+1})/2 | i = 1...N-1\}$ **do**
    $\mathcal{D}_1 = \{(y_i, \mathbf{x}_i) | x_{di} \leq t\}$
    $\mathcal{D}_2 = \{(y_i, \mathbf{x}_i) | x_{di} > t\}$
    $\bar{y}_1 = \frac{1}{|\mathcal{D}_1|} \sum_{y \in \mathcal{D}_1} y$
    $\bar{y}_2 = \frac{1}{|\mathcal{D}_2|} \sum_{y \in \mathcal{D}_2} y$

Finding the Best Split

Algorithm 23 $BestSplit(\mathcal{D})$

for d from 1 to D **do**

  $\mathbf{s} = sort(\{x_{d1}, ..., x_{dN}\})$

  for t in $\{(s_i + s_{i+1})/2 | i = 1...N-1\}$ **do**

    $\mathcal{D}_1 = \{(y_i, \mathbf{x}_i) | x_{di} \leq t\}$

    $\mathcal{D}_2 = \{(y_i, \mathbf{x}_i) | x_{di} > t\}$

    $\bar{y}_1 = \frac{1}{|\mathcal{D}_1|} \sum_{y \in \mathcal{D}_1} y$

    $\bar{y}_2 = \frac{1}{|\mathcal{D}_2|} \sum_{y \in \mathcal{D}_2} y$

    $Score(d,t) = \sum_{y \in \mathcal{D}_1} (y - \bar{y}_1)^2 + \sum_{y \in \mathcal{D}_2} (y - \bar{y}_2)^2$

Finding the Best Split

Algorithm 24 $BestSplit(\mathcal{D})$

for d from 1 to D **do**

  $\mathbf{s} = sort(\{x_{d1}, ..., x_{dN}\})$

  for t in $\{(s_i + s_{i+1})/2 | i = 1...N-1\}$ **do**

    $\mathcal{D}_1 = \{(y_i, \mathbf{x}_i) | x_{di} \leq t\}$

    $\mathcal{D}_2 = \{(y_i, \mathbf{x}_i) | x_{di} > t\}$

    $\bar{y}_1 = \frac{1}{|\mathcal{D}_1|} \sum_{y \in \mathcal{D}_1} y$

    $\bar{y}_2 = \frac{1}{|\mathcal{D}_2|} \sum_{y \in \mathcal{D}_2} y$

    $Score(d, t) = \sum_{y \in \mathcal{D}_1} (y - \bar{y}_1)^2 + \sum_{y \in \mathcal{D}_2} (y - \bar{y}_2)^2$

Finding the Best Split

Algorithm 25 *BestSplit*$(\mathcal{D})$

for d from 1 to D **do**

  $\mathbf{s} = sort(\{x_{d1}, ..., x_{dN}\})$

  for t in $\{(s_i + s_{i+1})/2 | i = 1...N-1\}$ **do**

    $\mathcal{D}_1 = \{(y_i, \mathbf{x}_i) | x_{di} \leq t\}$

    $\mathcal{D}_2 = \{(y_i, \mathbf{x}_i) | x_{di} > t\}$

    $\bar{y}_1 = \frac{1}{|\mathcal{D}_1|} \sum_{y \in \mathcal{D}_1} y$

    $\bar{y}_2 = \frac{1}{|\mathcal{D}_2|} \sum_{y \in \mathcal{D}_2} y$

    $Score(d, t) = \sum_{y \in \mathcal{D}_1} (y - \bar{y}_1)^2 + \sum_{y \in \mathcal{D}_2} (y - \bar{y}_2)^2$

Finding the Best Split

Algorithm 26 *BestSplit*$(\mathcal{D})$

for d from 1 to D **do**
  $\mathbf{s} = sort(\{x_{d1}, ..., x_{dN}\})$
  for t in $\{(s_i + s_{i+1})/2 | i = 1...N-1\}$ **do**
    $\mathcal{D}_1 = \{(y_i, \mathbf{x}_i) | x_{di} \leq t\}$
    $\mathcal{D}_2 = \{(y_i, \mathbf{x}_i) | x_{di} > t\}$
    $\bar{y}_1 = \frac{1}{|\mathcal{D}_1|} \sum_{y \in \mathcal{D}_1} y$
    $\bar{y}_2 = \frac{1}{|\mathcal{D}_2|} \sum_{y \in \mathcal{D}_2} y$
    $Score(d,t) = \sum_{y \in \mathcal{D}_1} (y - \bar{y}_1)^2 + \sum_{y \in \mathcal{D}_2} (y - \bar{y}_2)^2$
$d, t = \arg\min_{d',t'} Score(d,t)$

Finding the Best Split

Algorithm 27 $BestSplit(\mathcal{D})$

for d from 1 to D **do**

  $\mathbf{s} = sort(\{x_{d1}, ..., x_{dN}\})$

  for t in $\{(s_i + s_{i+1})/2 | i = 1...N-1\}$ **do**

    $\mathcal{D}_1 = \{(y_i, \mathbf{x}_i) | x_{di} \leq t\}$

    $\mathcal{D}_2 = \{(y_i, \mathbf{x}_i) | x_{di} > t\}$

    $\bar{y}_1 = \frac{1}{|\mathcal{D}_1|} \sum_{y \in \mathcal{D}_1} y$

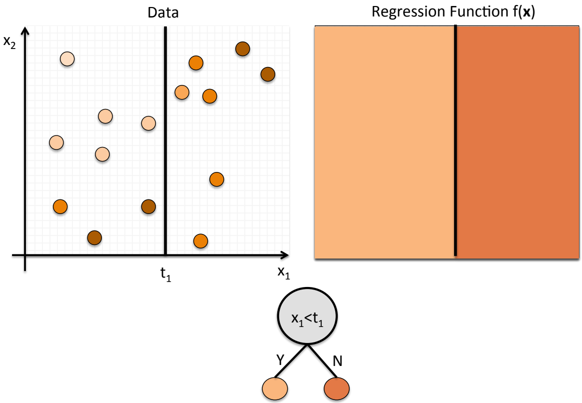
    $\bar{y}_2 = \frac{1}{|\mathcal{D}_2|} \sum_{y \in \mathcal{D}_2} y$

    $Score(d, t) = \sum_{y \in \mathcal{D}_1} (y - \bar{y}_1)^2 + \sum_{y \in \mathcal{D}_2} (y - \bar{y}_2)^2$

$d, t = \arg\min_{d',t'} Score(d, t)$

return (d, t)

Example: Building Regression Trees



Outline

Best Subset Selection

Algorithm 6.1 *Best subset selection*

1. Let $\mathcal{M}_0$ denote the *null model*, which contains no predictors. This model simply predicts the sample mean for each observation.

2. For $k = 1, 2, \ldots p$:

   (a) Fit all $\binom{p}{k}$ models that contain exactly k predictors.

   (b) Pick the best among these $\binom{p}{k}$ models, and call it $\mathcal{M}_k$. Here *best* is defined as having the smallest RSS, or equivalently largest R^2.

3. Select a single best model from among $\mathcal{M}_0, \ldots, \mathcal{M}_p$ using cross-validated prediction error, C_p (AIC), BIC, or adjusted R^2.

Forward Stepwise Selection

Algorithm 6.2 *Forward stepwise selection*

1. Let $\mathcal{M}_0$ denote the *null* model, which contains no predictors.
2. For $k = 0, \ldots, p-1$:
   (a) Consider all $p - k$ models that augment the predictors in $\mathcal{M}_k$ with one additional predictor.
   (b) Choose the *best* among these $p - k$ models, and call it $\mathcal{M}_{k+1}$. Here *best* is defined as having smallest RSS or highest R^2.
3. Select a single best model from among $\mathcal{M}_0, \ldots, \mathcal{M}_p$ using cross-validated prediction error, C_p (AIC), BIC, or adjusted R^2.

Backward Stepwise Selection

Algorithm 6.3 *Backward stepwise selection*

1. Let $\mathcal{M}_p$ denote the *full* model, which contains all p predictors.
2. For $k = p, p-1, \ldots, 1$:
   (a) Consider all k models that contain all but one of the predictors in $\mathcal{M}_k$, for a total of $k-1$ predictors.
   (b) Choose the *best* among these k models, and call it $\mathcal{M}_{k-1}$. Here *best* is defined as having smallest RSS or highest R^2.
3. Select a single best model from among $\mathcal{M}_0, \ldots, \mathcal{M}_p$ using cross-validated prediction error, C_p (AIC), BIC, or adjusted R^2.