Tree-based Methods

Héctor Corrada Bravo

University of Maryland, College Park, USA

CMSC 643: 2018-10-08

We saw in previous units the limitation of using linear methods for regression classification.

We saw in previous units the limitation of using linear methods for regression classification.

In this unit, we look at tree-based methods.

We saw in previous units the limitation of using linear methods for regression classification.

In this unit, we look at tree-based methods.

These are elegant and versatile methods that allow modeling of predictor space with regions that take complex, non-linear, shapes

We saw in previous units the limitation of using linear methods for regression classification.

In this unit, we look at tree-based methods.

These are elegant and versatile methods that allow modeling of predictor space with regions that take complex, non-linear, shapes

But still produce models that are interpretable.

We saw in previous units the limitation of using linear methods for regression classification.

In this unit, we look at tree-based methods.

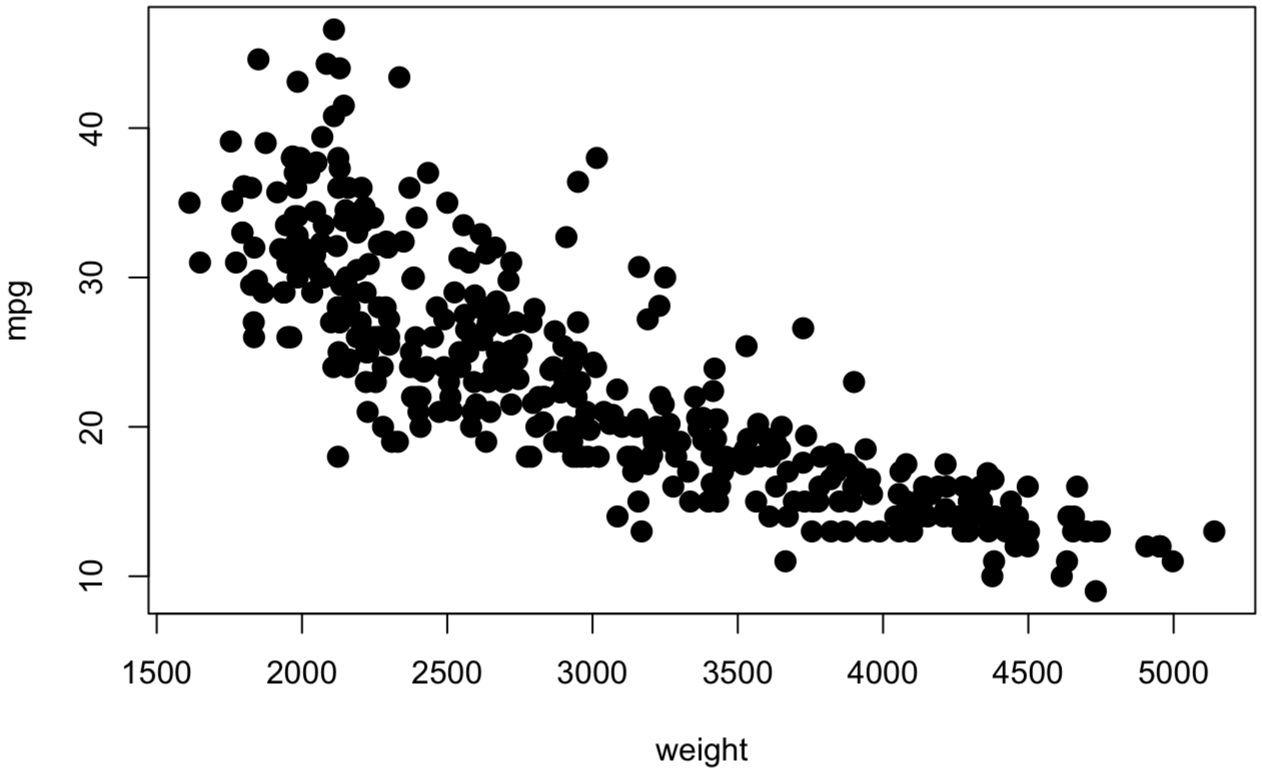
These are elegant and versatile methods that allow modeling of predictor space with regions that take complex, non-linear, shapes

But still produce models that are interpretable.

We will concentrate on Regression and Decision Trees and their extension to Random Forests.

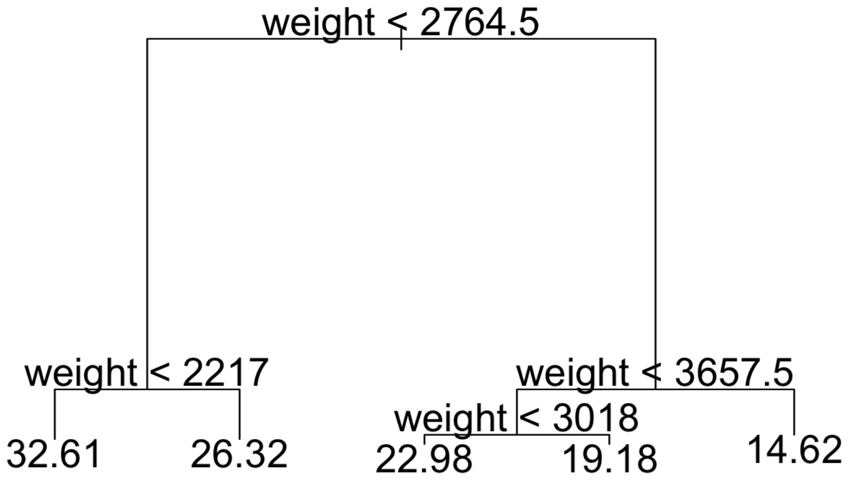
Regression Trees

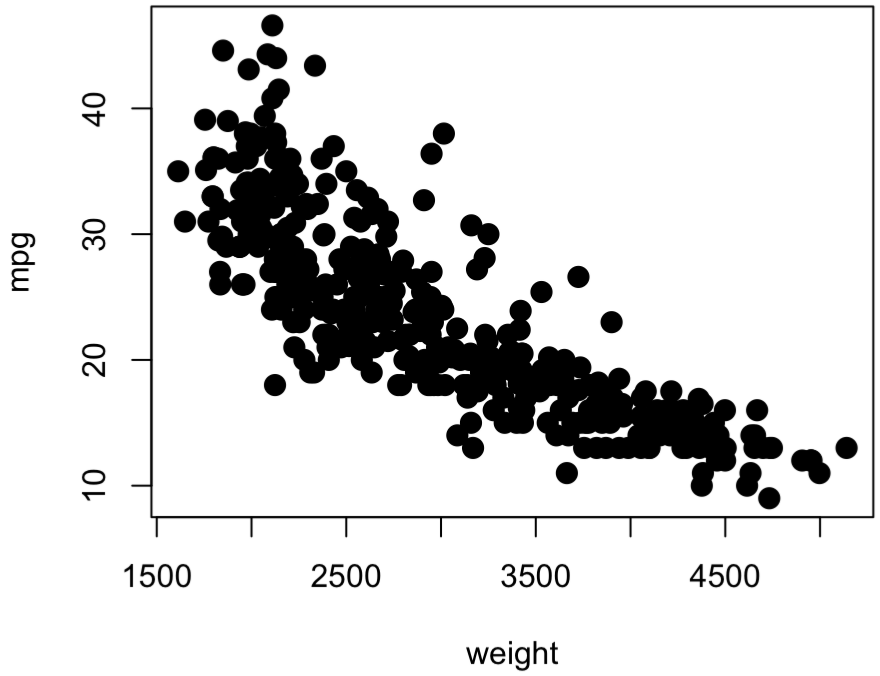
Consider a task where we are trying to predict a car's fuel consumption in miles per gallon based on the car's weight. A linear model in this case is not a good fit.



Regression Trees

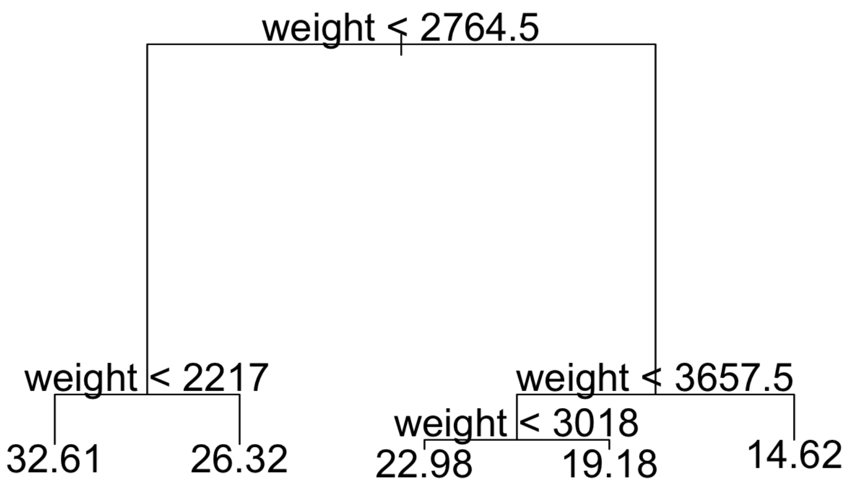
Let's take a look at what a regression tree estimates in this case.

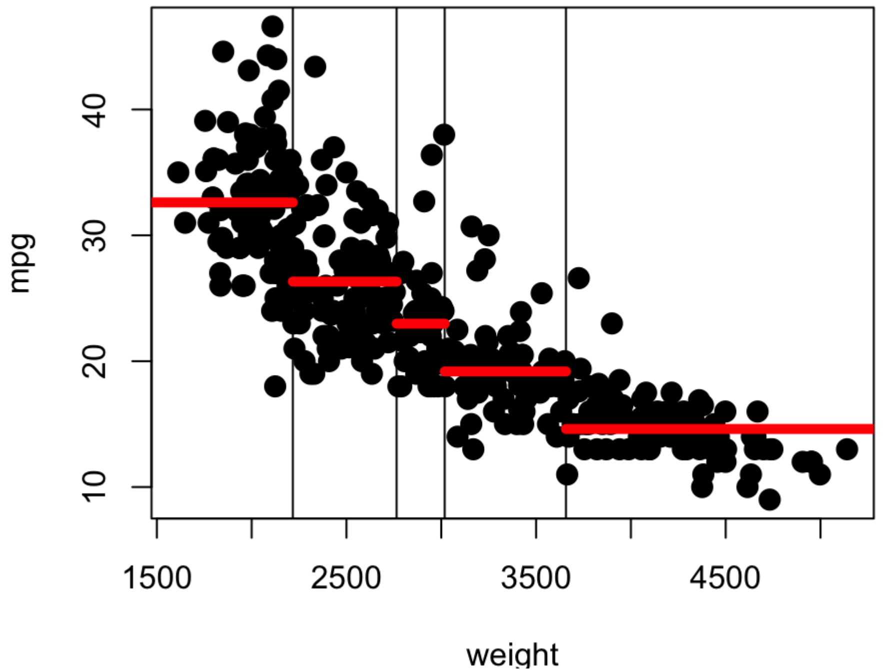




Regression trees

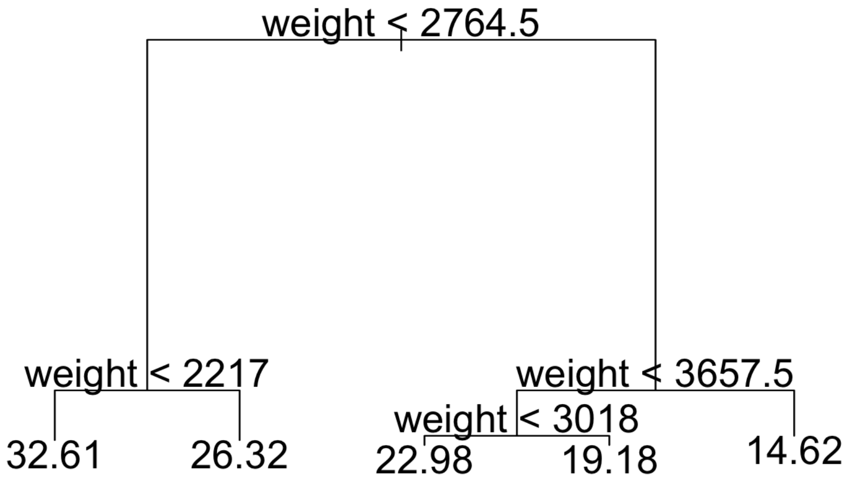
The decision trees partitions the `weight` predictor into regions based on its value.

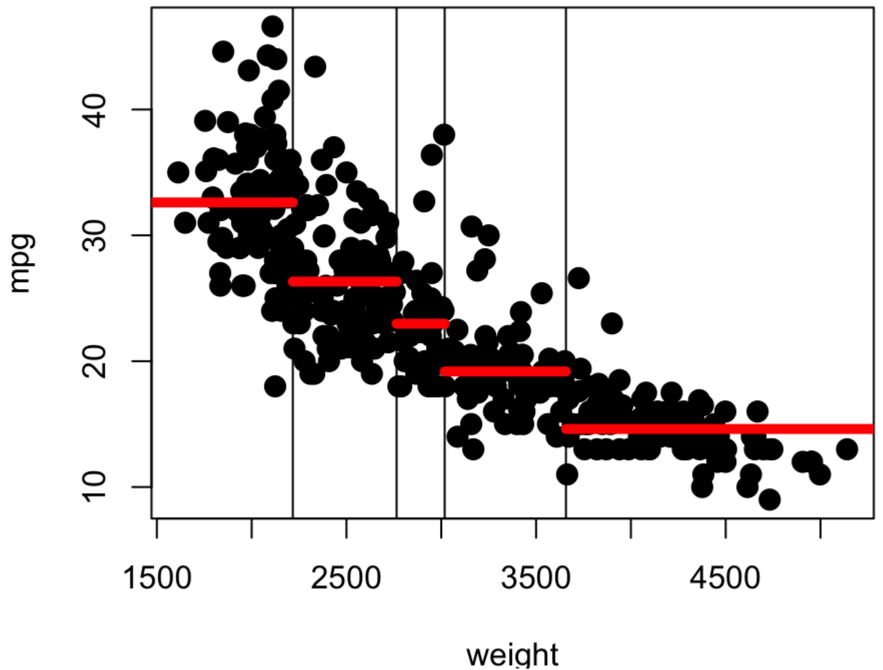




Regression Trees

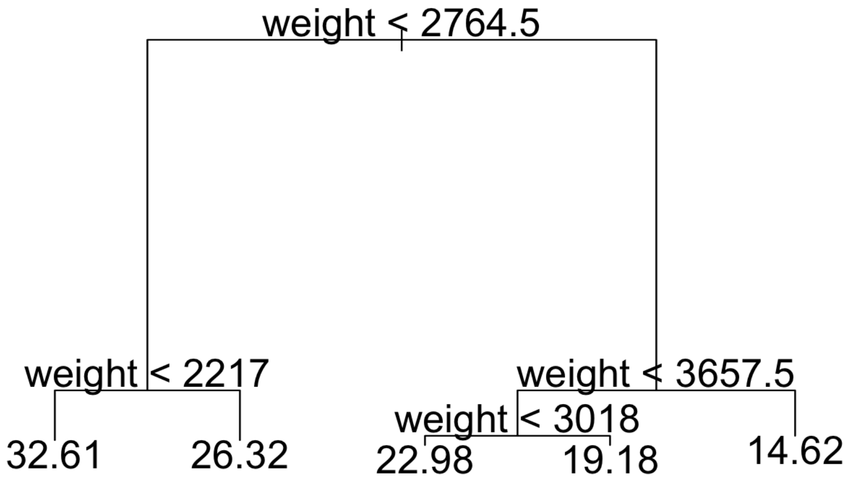
Outcome (`mpg` in this case) is predicted to be the mean *within each of the data partitions*.

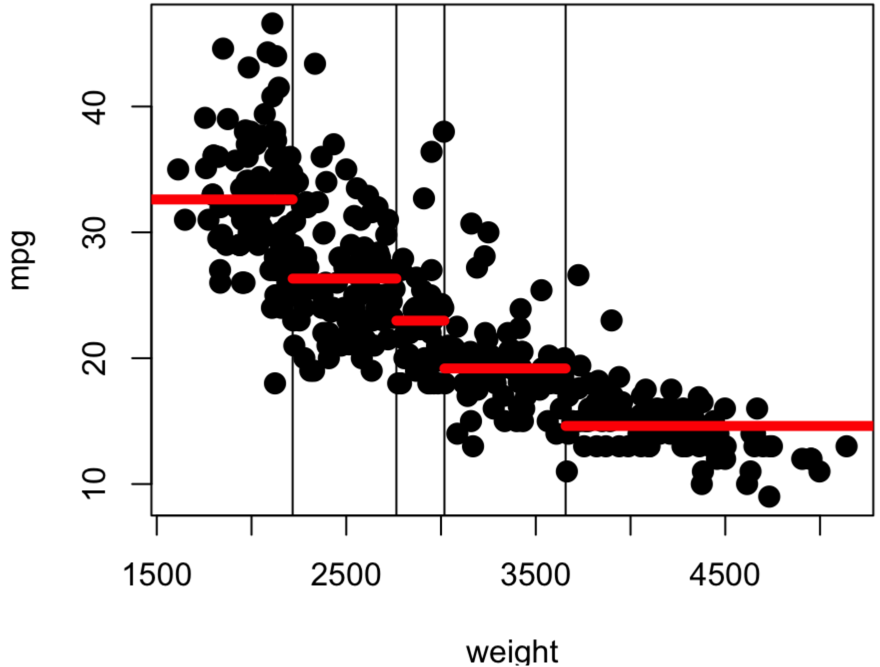




Regression Trees

Thus provides an empirical estimate of where conditioning is given by this region partitioning.





Tree models

Regression and decision trees operate by prediction an outcome variable Y by partitioning feature (predictor) space.

Tree models

Regression and decision trees operate by prediction an outcome variable Y by partitioning feature (predictor) space.

In general, the regression tree model:

1. Partitions space into J non-overlapping regions, $R_1, R_2, \ldots, R_J$.
2. For every observation that falls within region R_j, predict response as mean of response for training observations in R_j.

Tree models

Regression and decision trees operate by prediction an outcome variable Y by partitioning feature (predictor) space.

In general, the regression tree model:

1. Partitions space into J non-overlapping regions, $R_1, R_2, \ldots, R_J$.
2. For every observation that falls within region R_j, predict response as mean of response for training observations in R_j.

The important observation is that **Regression Trees create partitions recursively**

Tree Models

For example, consider finding a good predictor j to partition space along its axis. A recursive algorithm would look like this:

- Find predictor j and value s that minimize RSS:

$$\sum_{i:\, x_i \in R_1(j,s))} (y_i - \hat{y}_{R_1})^2 + \sum_{i:\, x_i \in R_2(j,s))} (y_i - \hat{y}_{R_2})^2$$

Where R_1 and R_2 are regions resulting from splitting observations on predictor j and value s:

$$R_1(j,s) = X|X_j < s \text{ and } R_2(j,s) X|X_j \geq s$$

Tree Models

For example, consider finding a good predictor j to partition space along its axis. A recursive algorithm would look like this:

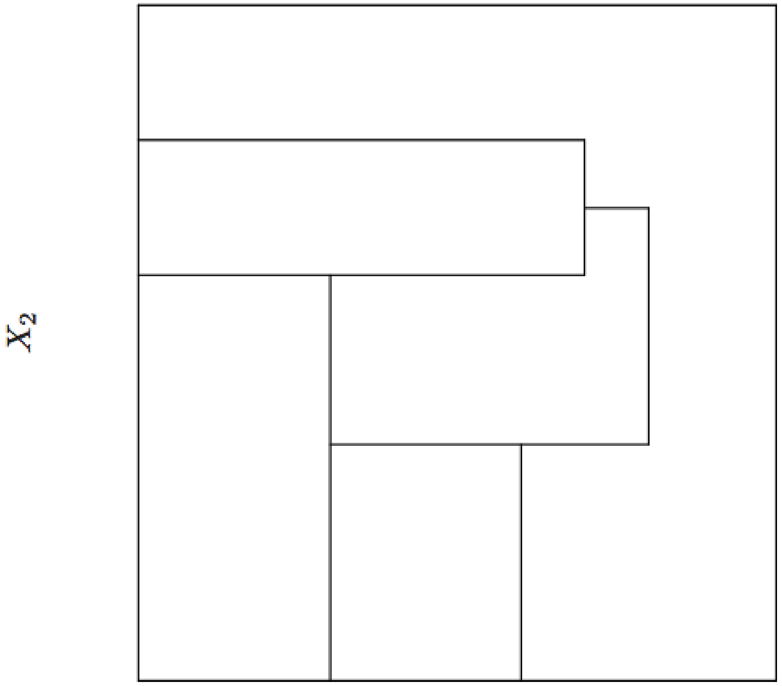
- Find predictor j and value s that minimize RSS:

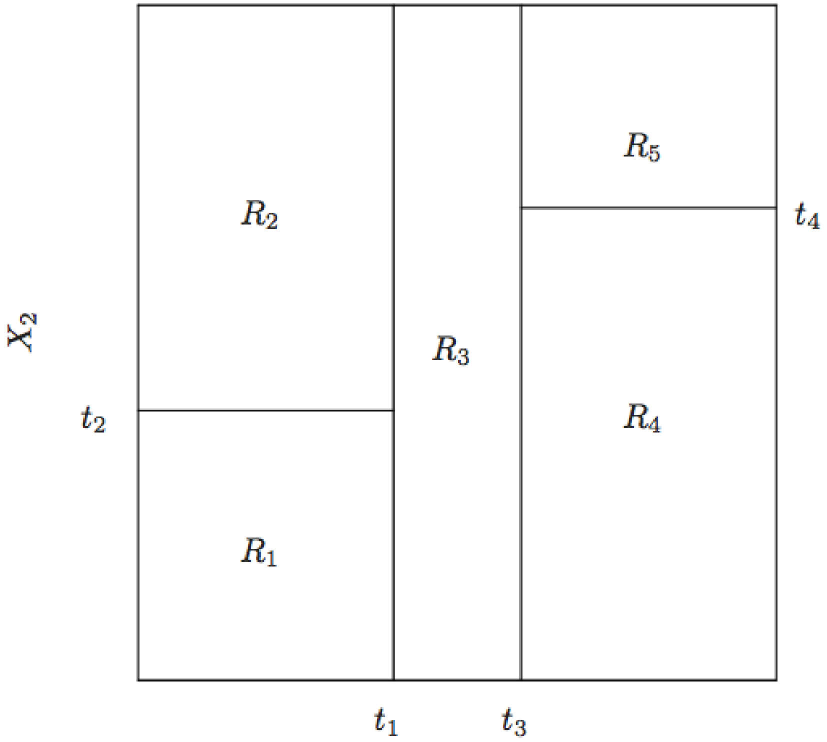
$$\sum_{i:\ x_i \in R_1(j,s))} (y_i - \hat{y}_{R_1})^2 + \sum_{i:\ x_i \in R_2(j,s))} (y_i - \hat{y}_{R_2})^2$$

- Apply recursively to regions R_1 and R_2.

Tree Models

Within each region a prediction $\hat{y}_{R_j}$ is made as the mean of the response Y of observations in R_j.

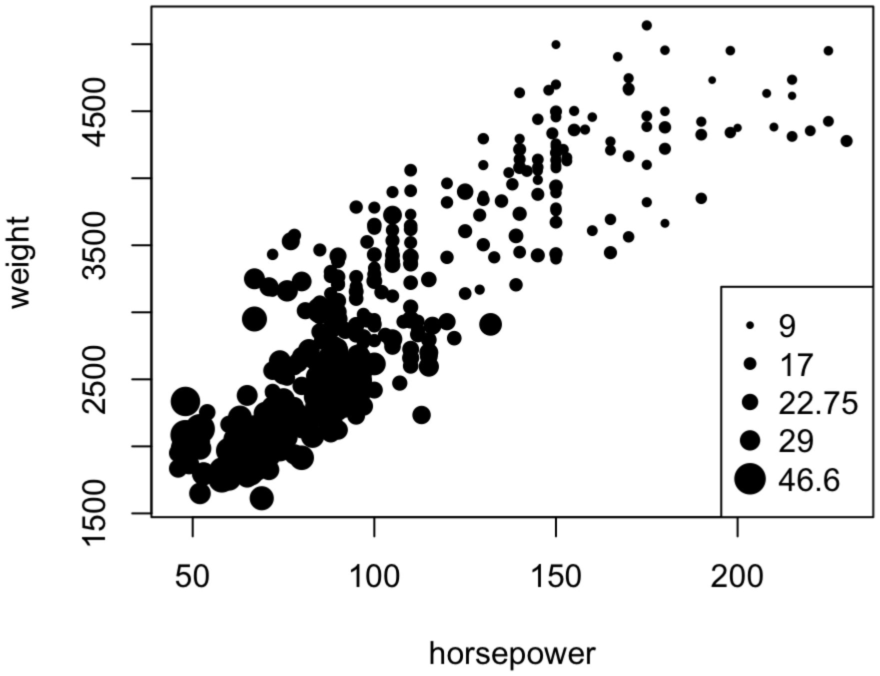




Regression Trees

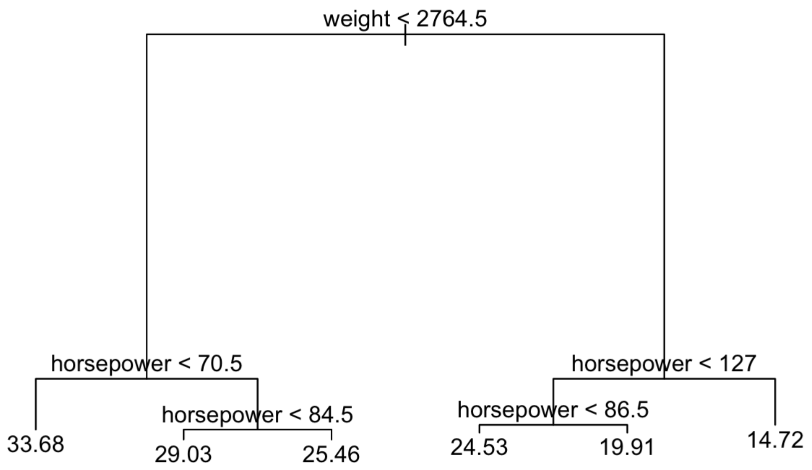
Consider building a model that used both `horsepower` and `weight`.

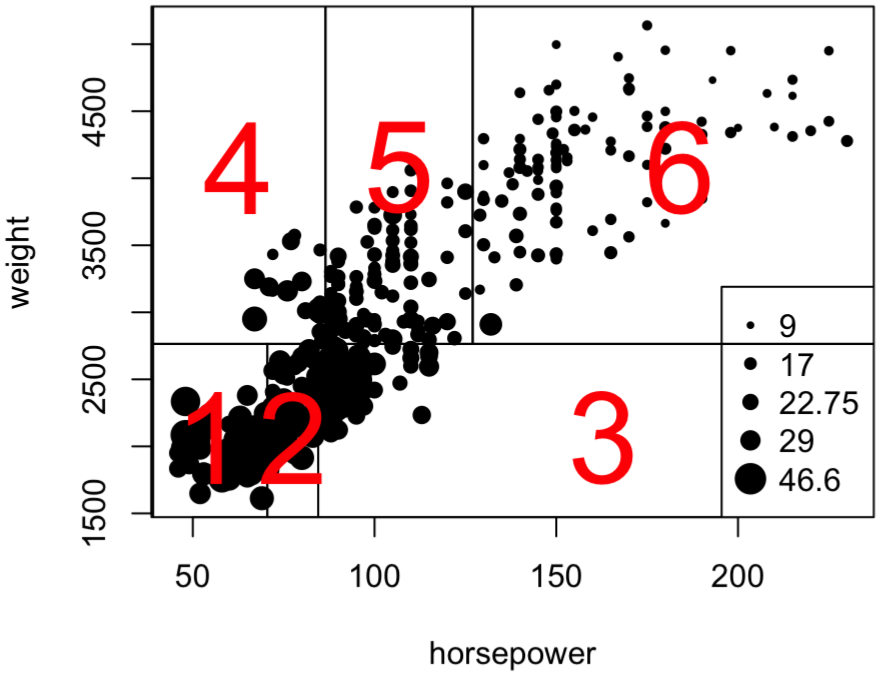
Here, value of the response Y is indicated by the size of the point.



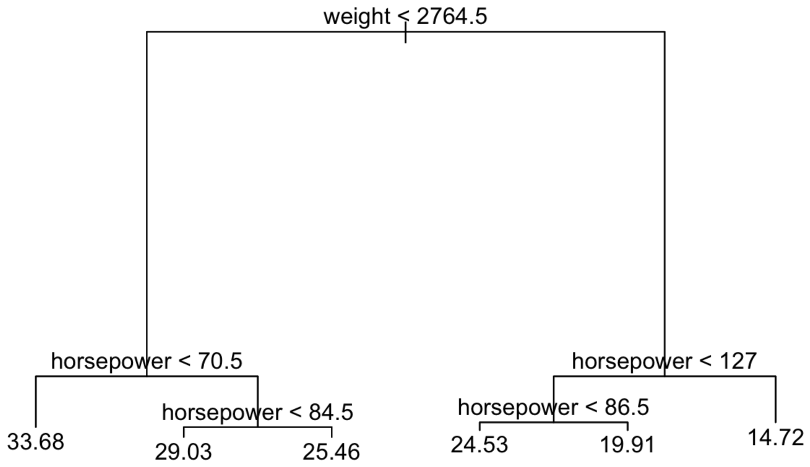
Regression Trees

This is what a decision tree would look like for these two predictors:





Regression Trees



Quiz What would this tree predict as `mpg` for an instance with variable values

- `horsepower=85`
- `weight=2800`

Classification (Decision) Trees

Classification, or decision trees, are used in classification problems, where the outcome is categorical.

Classification (Decision) Trees

Classification, or decision trees, are used in classification problems, where the outcome is categorical.

The same partitioning principle holds, but now, each region predicts the majority class for training observations within region.

Classification (Decision) Trees

Classification, or decision trees, are used in classification problems, where the outcome is categorical.

The same partitioning principle holds, but now, each region predicts the majority class for training observations within region.

The recursive partitioning method requires a score function to choose predictors (and values) to partition with.

Classification (Decision) Trees

Classification, or decision trees, are used in classification problems, where the outcome is categorical.

The same partitioning principle holds, but now, each region predicts the majority class for training observations within region.

The recursive partitioning method requires a score function to choose predictors (and values) to partition with.

A naive approach would looking for partitions that minimize training error.

Classification (Decision) Trees

Classification, or decision trees, are used in classification problems, where the outcome is categorical.

The same partitioning principle holds, but now, each region predicts the majority class for training observations within region.

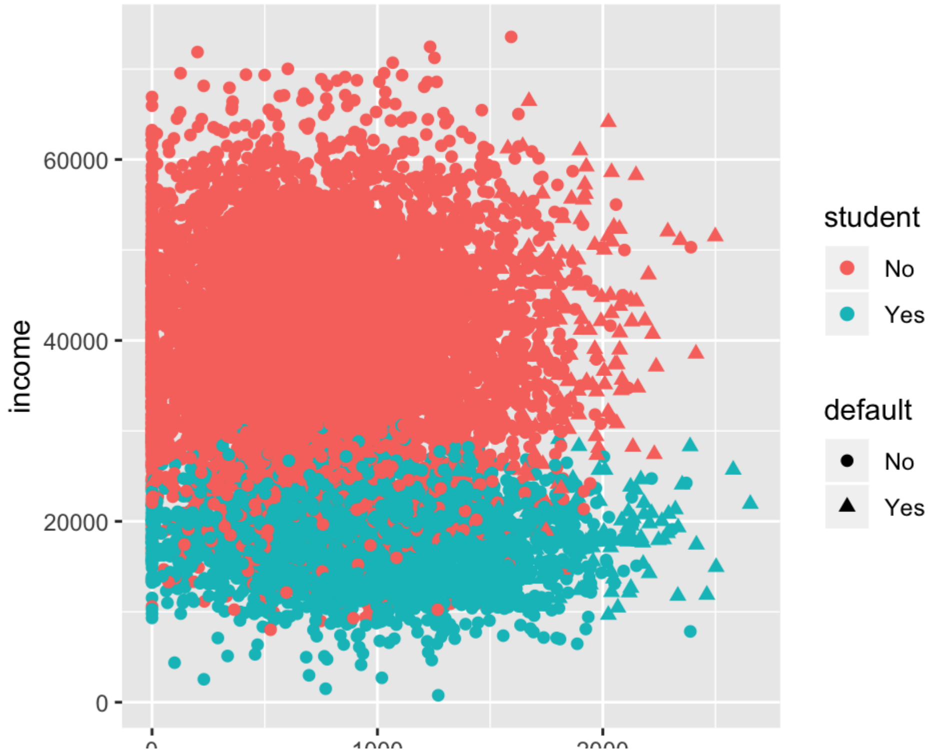
The recursive partitioning method requires a score function to choose predictors (and values) to partition with.

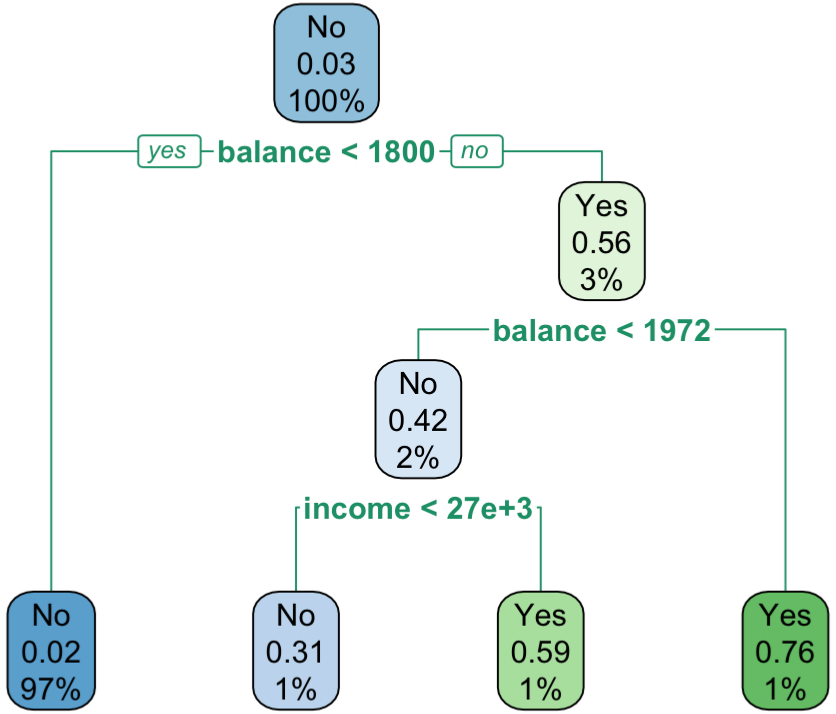
A naive approach would looking for partitions that minimize training error.

Better performing approaches use more sophisticated metrics.

Decision Trees

Let's look at how a classification tree performs on a credit card default dataset.





Specifics of the partitioning algorithm

The predictor space

Suppose we have p explanatory variables $X_1, \ldots, X_p$ and N observations.

Specifics of the partitioning algorithm

The predictor space

Suppose we have p explanatory variables $X_1, \ldots, X_p$ and N observations.

Each of the X_i can be

a) a numeric variable: there are $n - 1$ possible splits
b) an ordered factor (categorical variable): there are $k - 1$ possible splits
c) an unordered factor: $2^{k-1} - 1$ possible splits.

Specifics of the partitioning algorithm

Learning Strategy

The general procedure for tree learning is the following:

Grow: an overly large tree using forward selection as follows: at each step, find the *best* split among all attributes. Grow until all terminal nodes either

(a) have $< m$ (perhaps $m = 1$) data points
(b) are "pure" (all points in a node have [almost] the same outcome).

Specifics of the partitioning algorithm

Learning Strategy

The general procedure for tree learning is the following:

Grow: an overly large tree using forward selection

Prune: the tree back, creating a nested sequence of trees, decreasing in *complexity*

Specifics of the partitioning algorithm

Tree Growing

The recursive partitioning algorithm is as follows:

INITIALIZE All cases in the root node

REPEAT Find optimal allowed split; Partition leaf according to split

STOP Stop when pre-defined criterion is met

Specifics of the partitioning algorithm

Tree Growing

An important issue in tree construction is how to use the training data to determine the binary splits of dataset X

Specifics of the partitioning algorithm

Tree Growing

An important issue in tree construction is how to use the training data to determine the binary splits of dataset $\mathcal{X}$

The fundamental idea is to select each split of a subset so that the data in each of the descendent subsets are "purer" than the data in the parent subset.

Specifics of the partitioning algorithm

Deviance as a measure of impurity

A simple approach is to assume a multinomial model and then use deviance as a definition of impurity.

Specifics of the partitioning algorithm

Deviance as a measure of impurity

Assume $Y \in G = \{1, 2, \ldots, k\}$.

- At each node i of a classification tree we have a probability distribution p_{ik} over the k classes.
- We observe a random sample n_{ik} from the multinomial distribution specified by the probabilities p_{ik}.

Specifics of the partitioning algorithm

Deviance as a measure of impurity

Assume $Y \in G = \{1, 2, \ldots, k\}$.

- Given X, the conditional likelihood is then proportional to $\prod_{(\text{leaves } i)} \prod_{(\text{classes } k)} p_{ik}^{n_{ik}}$.
- Estimate p_{ik} by $\hat{p}_{ik} = \frac{n_{ik}}{n_i}$.
- Define deviance $D = \sum D_i$, where $D_i = -2 \sum_k n_{ik} \log(p_{ik})$.

Specifics of the partitioning algorithm

Deviance as a measure of impurity

Assume $Y \in G = \{1, 2, \ldots, k\}$.

- Given X, the conditional likelihood is then proportional to $\prod_{(\text{leaves } i)} \prod_{(\text{classes } k)} p_{ik}^{n_{ik}}$.
- Estimate p_{ik} by $\hat{p}_{ik} = \frac{n_{ik}}{n_i}$.
- Define deviance $D = \sum D_i$, where $D_i = -2 \sum_k n_{ik} \log(p_{ik})$.

Select splits that improve deviance D

Specifics of the partitioning algorithm

Deviance as a measure of impurity

Quiz Compute deviance for the following cases

a) $n_{i1} = 6,\ n_{i2} = 1,\ n_{i3} = 1$

b) $n_{i1} = 9,\ n_{i2} = 1,\ n_{i3} = 0$

c) $n_{i1} = 90,\ n_{i2} = 10,\ n_{i3} = 0$

Specifics of the partitioning algorithm

Other measures of impurity

Other commonly used measures of impurity at a node i of a classification tree are

missclasification rate: $\frac{1}{n_i}\sum_{j\in A_i} I(y_j \neq k_i) = 1 - \hat{p}_{ik_i}$
entropy: $\sum p_{ik}\log(p_{ik})$
GINI index: $\sum_{j\neq k} p_{ij}p_{ik} = 1 - \sum_k p_{ik}^2$

where k_i is the most frequent class in node i.

Specifics of the partitioning algorithm

Other measures of impurity

Other commonly used measures of impurity at a node i of a classification tree are

missclasification rate: $\frac{1}{n_i} \sum_{j \in A_i} I(y_j \neq k_i) = 1 - \hat{p}_{ik_i}$
entropy: $\sum p_{ik} \log(p_{ik})$
GINI index: $\sum_{j \neq k} p_{ij} p_{ik} = 1 - \sum_k p_{ik}^2$

where k_i is the most frequent class in node i.

In practice, the GINI index is preferred

Specifics of the partitioning algorithm

For regression trees we use the residual sum of squares:

$$D = \sum_{\text{cases } j} (y_j - \mu_{[j]})^2$$

where $\mu_{[j]}$ is the mean values in the node that case j belongs to.

Specifics of the partitioning algorithm

Tree Pruning

- Grow a big tree T
- Consider snipping off terminal subtrees (resulting in so-called rooted subtrees)
- Let D_i be a measure of impurity at leaf i in a tree. Define $D = \sum_i D_i$
- Define size as the number leaves in a tree
- Let $D_\alpha = D + \alpha \times \text{size}$

Specifics of the partitioning algorithm

Tree Pruning

- Grow a big tree T
- Consider snipping off terminal subtrees (resulting in so-called rooted subtrees)
- Let D_i be a measure of impurity at leaf i in a tree. Define $D = \sum_i D_i$
- Define size as the number leaves in a tree
- Let $D_\alpha = D + \alpha \times \text{size}$

The set of rooted subtrees of T that minimize D_α is nested.

Specifics of the partitioning algorithm

Tree Pruning

We can prune the tree sequentially

Specifics of the partitioning algorithm

Tree Pruning

We can prune the tree sequentially

Given tree T,

- for every node R_j in tree, compute D_α after removing subtree rooted at R_j
- select node R_j that minimizes D_α
- Remote subtree rooted at R_j from T
- Continue until D_α increases

Properties of Tree Method

Good properties of Regression and Classification trees include:

- Decision trees are very "natural" constructs, in particular when the explanatory variables are catgorical (and even better when they are binary)

Properties of Tree Method

Good properties of Regression and Classification trees include:

- Decision trees are very "natural" constructs, in particular when the explanatory variables are catgorical (and even better when they are binary)

- Trees are easy to explain to non-data analysts

Properties of Tree Method

Good properties of Regression and Classification trees include:

- Decision trees are very "natural" constructs, in particular when the explanatory variables are catgorical (and even better when they are binary)
- Trees are easy to explain to non-data analysts
- The models are invariant under transformations in the predictor space

Properties of Tree Method

Good properties of Regression and Classification trees include:

- Multi-factor responses are easily dealt with

Properties of Tree Method

Good properties of Regression and Classification trees include:

- Multi-factor responses are easily dealt with
- The treatment of missing values is more satisfactory than for most other models

Properties of Tree Method

Good properties of Regression and Classification trees include:

- Multi-factor responses are easily dealt with
- The treatment of missing values is more satisfactory than for most other models
- The models go after interactions immediately, rather than as an afterthought

Properties of Tree Method

Good properties of Regression and Classification trees include:

- Multi-factor responses are easily dealt with
- The treatment of missing values is more satisfactory than for most other models
- The models go after interactions immediately, rather than as an afterthought
- Tree growth is much more efficient than described here

Properties of Tree Method

However, they do have important issues to address

- Tree space is huge, so we may need lots of data

Properties of Tree Method

However, they do have important issues to address

- Tree space is huge, so we may need lots of data
- We might not be able to find the *best* model at all as it is a greedy algorithm

Properties of Tree Method

However, they do have important issues to address

- Tree space is huge, so we may need lots of data
- We might not be able to find the *best* model at all as it is a greedy algorithm
- It can be hard to assess uncertainty in inference about trees

Properties of Tree Method

However, they do have important issues to address

- Results can be quite variable (tree selection is not very stable)

Properties of Tree Method

However, they do have important issues to address

- Results can be quite variable (tree selection is not very stable)
- Simple trees usually don't have a lot of predictive power

Random Forests

Random Forests are a **very popular** approach that addresses these shortcomings via resampling of the training data.

Random Forests

Random Forests are a **very popular** approach that addresses these shortcomings via resampling of the training data.

Their goal is to improve prediction performance and reduce instability by *averaging* multiple decision trees (a forest constructed with randomness).

Random Forests

It uses two ideas to accomplish this. The first idea is *Bagging* (bootstrap aggregation)

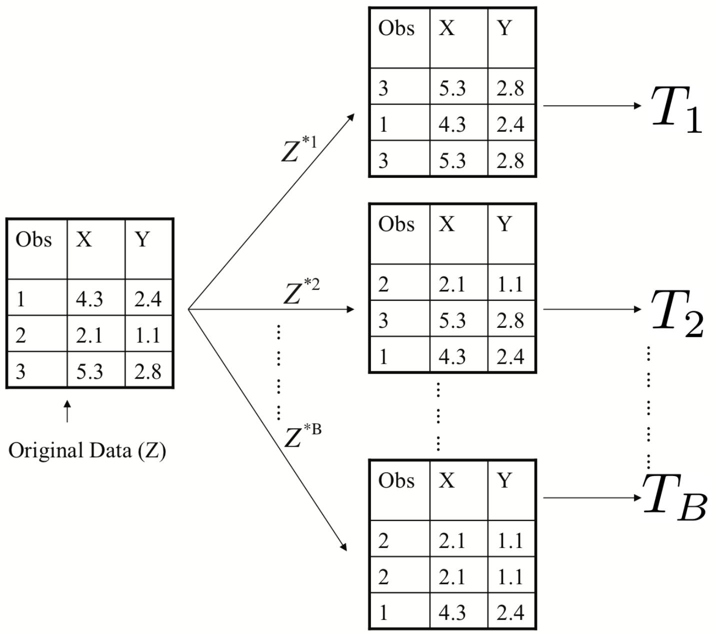
General scheme:

1. Build many decision trees $T_1, T_2, \dots, T_B$ from training set
2. Given a new observation, let each T_j predict $\hat{y}_j$
3. For regression: predict average $\frac{1}{B}\sum_{j=1}^{B}\hat{y}_j$, for classification: predict with majority vote (most frequent class)

Random Forests

How do we get many decision trees from a single training set?

Use the *bootstrap* resampling technique.

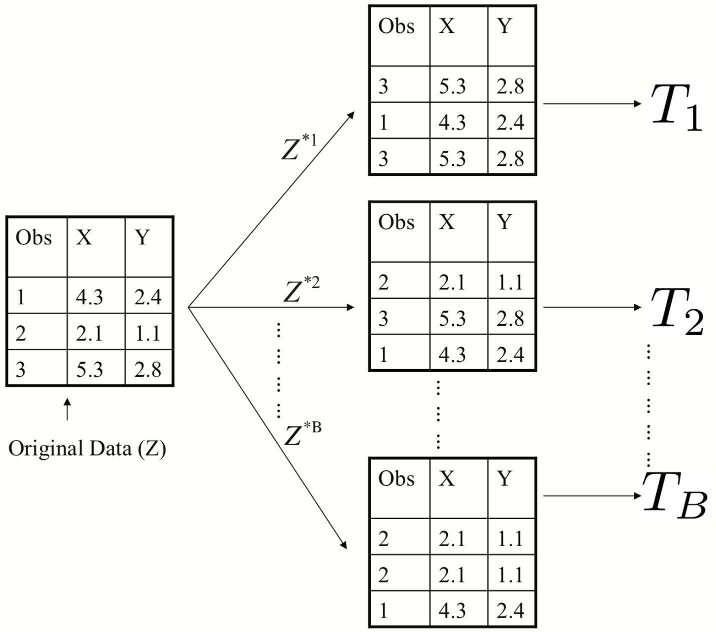


Random Forests

How do we get many decision trees from a single training set?

To create $T_j,\ j = 1, \dots, B$ from training set of size n:

a) create a bootstrap training set by sampling n observations from training set **with replacement**

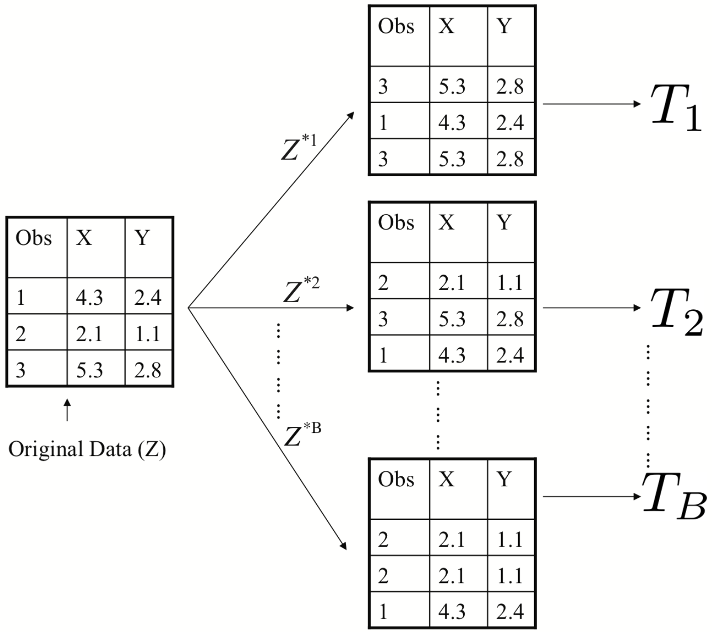


Random Forests

How do we get many decision trees from a single training set?

To create $T_j, j = 1, \ldots, B$ from training set of size n:

b) build a decision tree from bootstrap training set



Random Forests

The second idea used in Random Forests is to use a random selection of features to split when deciding partitions.

Random Forests

The second idea used in Random Forests is to use a random selection of features to split when deciding partitions.

Specifically, when building each tree T_j, at each recursive partition:

Random Forests

The second idea used in Random Forests is to use a random selection of features to split when deciding partitions.

Specifically, when building each tree T_j, at each recursive partition:

only consider a randomly selected subset of predictors to find best split.

Random Forests

The second idea used in Random Forests is to use a random selection of features to split when deciding partitions.

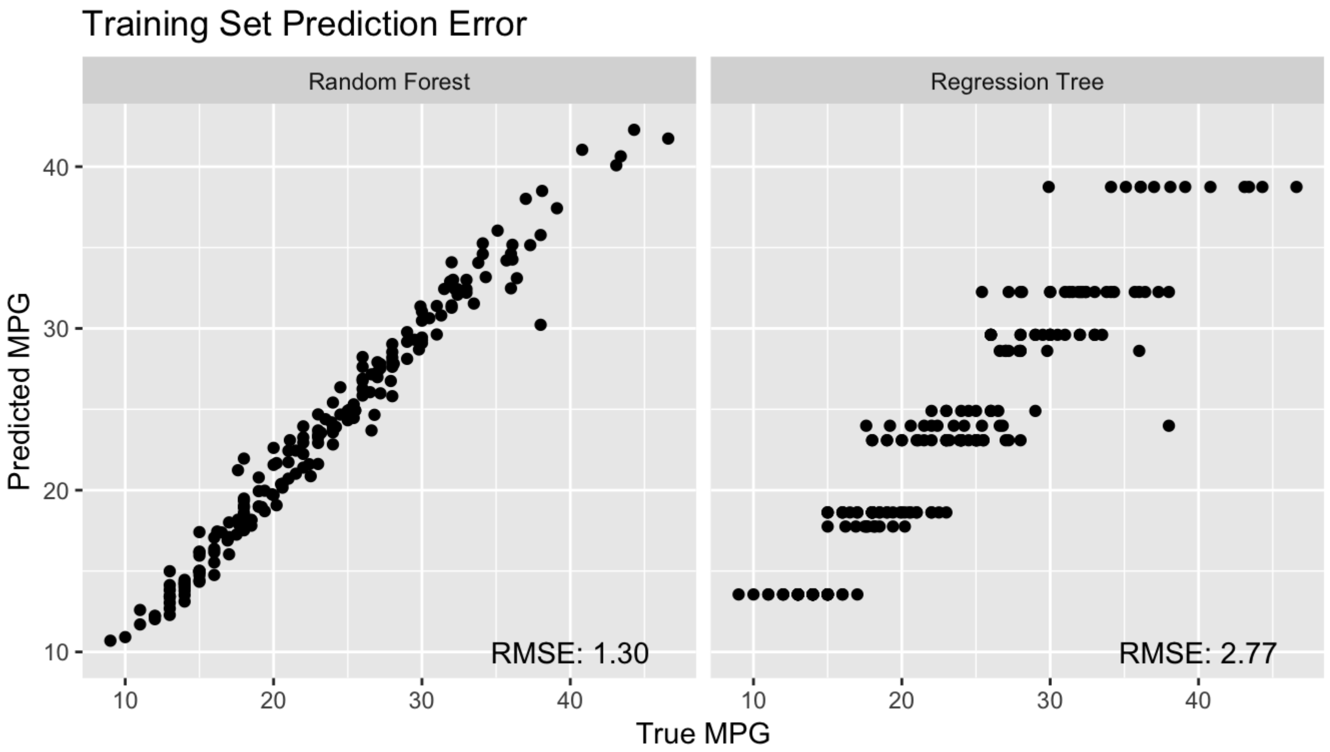
Specifically, when building each tree T_j, at each recursive partition:

only consider a randomly selected subset of predictors to find best split.

This reduces correlation between trees in forest, improving prediction accuracy.

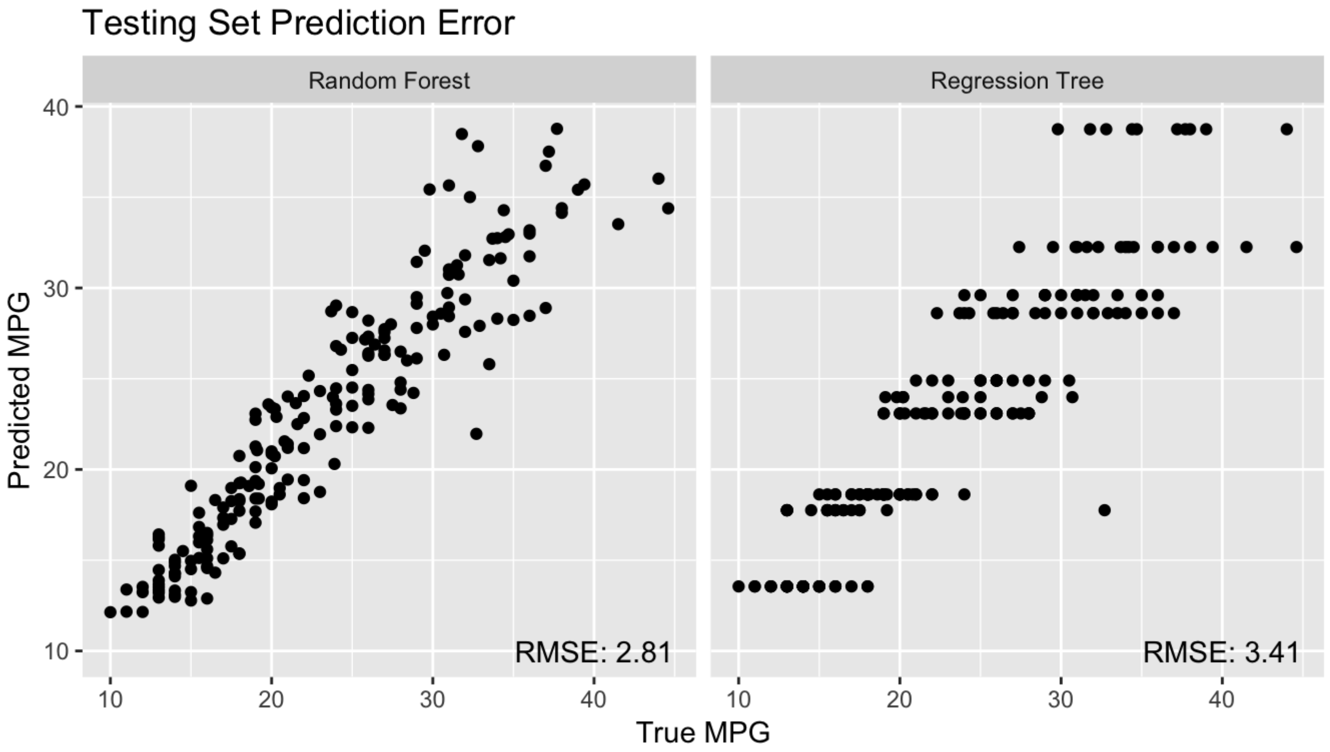
Random Forests

Let's look at the same car dataset again and plot predicted vs. true miles per gallon given by a random forest and a regression tree.



Random Forests

Now let's look at the same plot on a *testing* dataset.



Random Forests

A disadvantage of random forests is that we lose interpretability.

Random Forests

A disadvantage of random forests is that we lose interpretability.

However, we can use the fact that a bootstrap sample was used to construct trees to measure *variable importance* from the random forest.

Random Forests

A disadvantage of random forests is that we lose interpretability.

However, we can use the fact that a bootstrap sample was used to construct trees to measure *variable importance* from the random forest.

Since we used bootstrap samples we can get out-of-bag (OOB) samples for each tree in the random forest.

Random Forests

When the bth tree is constructed, use the OOB samples as follows

1. Compute error rate for the OOB samples

2. For each predictor j:

    a. permute its values in the OOB samples and recompute error rate
    b. calculate increase in error rate

Report increase in error rate over all bootstrap samples

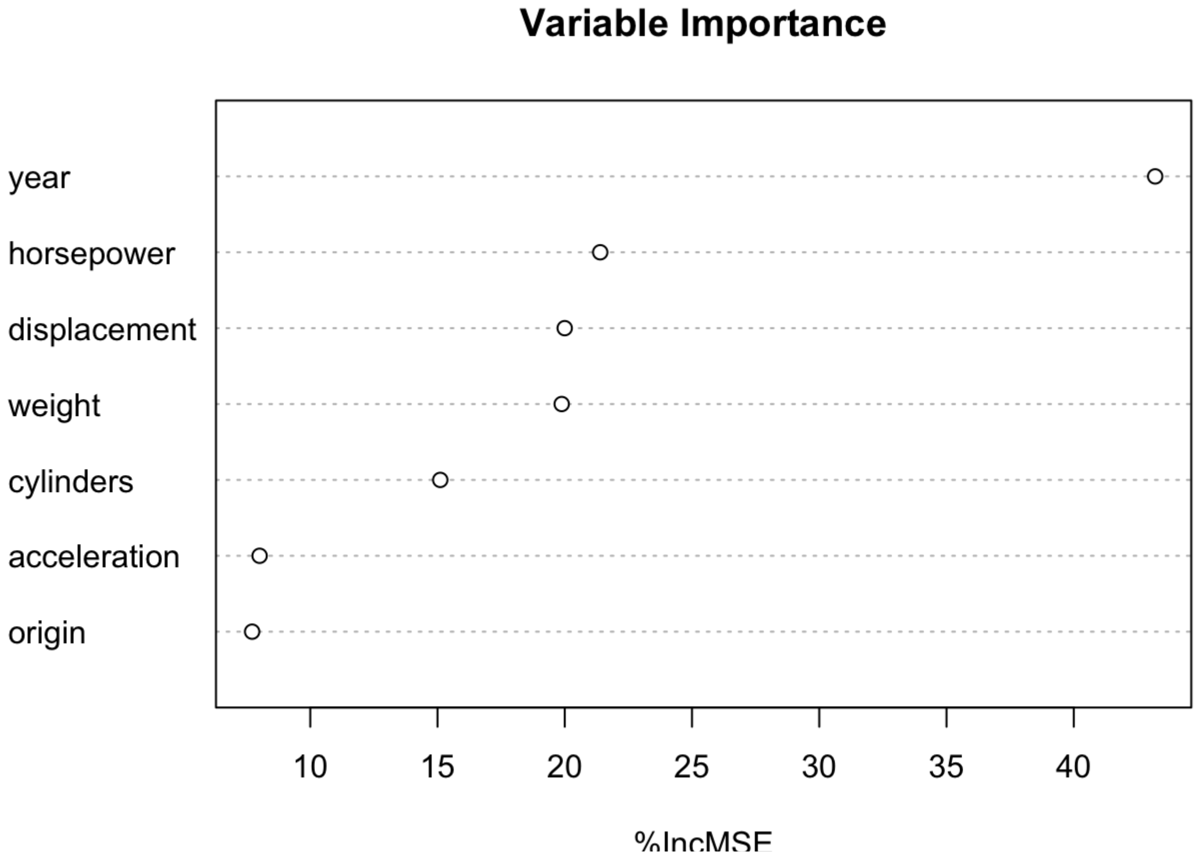
Random Forests

Here is a table of *variable importance* for the random forest we just constructed.

	%IncMSE	IncNodePurity
cylinders	15.11	2328.05
displacement	20.00	2480.60
horsepower	21.39	2779.68
weight	19.88	2325.81
acceleration	8.01	377.69
year	43.20	1341.62

Random Forests

And a plot of variable importance



Tree-based methods summary

Tree-based methods are highly interpretable *prediction* models.

Tree-based methods summary

Tree-based methods are highly interpretable *prediction* models.

Some inferential tasks are possible (e.g., variable importance in random forests), but are much more limited than linear models.

Tree-based methods summary

Tree-based methods are highly interpretable *prediction* models.

Some inferential tasks are possible (e.g., variable importance in random forests), but are much more limited than linear models.

These methods are very commonly used across many application domains

Tree-based methods summary

Tree-based methods are highly interpretable *prediction* models.

Some inferential tasks are possible (e.g., variable importance in random forests), but are much more limited than linear models.

These methods are very commonly used across many application domains

Random Forests often perform at state-of-the-art for many tasks.