Classification and Prediction

CART, Bagging, Random Forests, Boosting

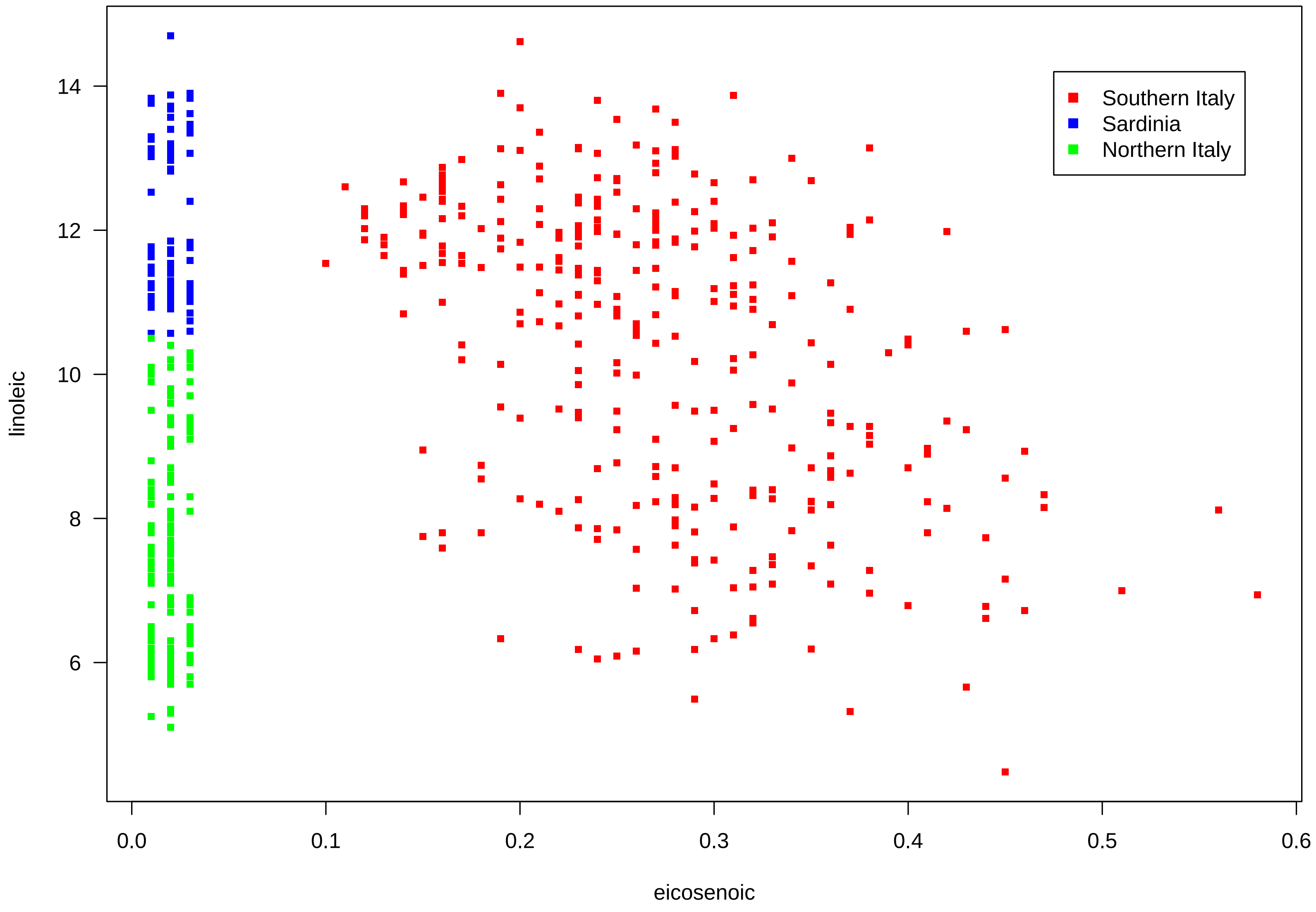
The Olive Data

- 572 olive oils were analyzed for their content of eight fatty acids (palmitic, palmitoleic, stearic, oleic, linoleic, arachidic, linolenic, and eicosenoic).
- There were 9 collection areas, 4 from Southern Italy (North and South Apulia, Calabria, Sicily), two from Sardinia (Inland and Coastal) and 3 from Northern Italy (Umbria, East and West Liguria).
- The concentrations of different fatty acids vary from up to 85% for oleic acid to as low as 0.01% for eicosenoic acid.

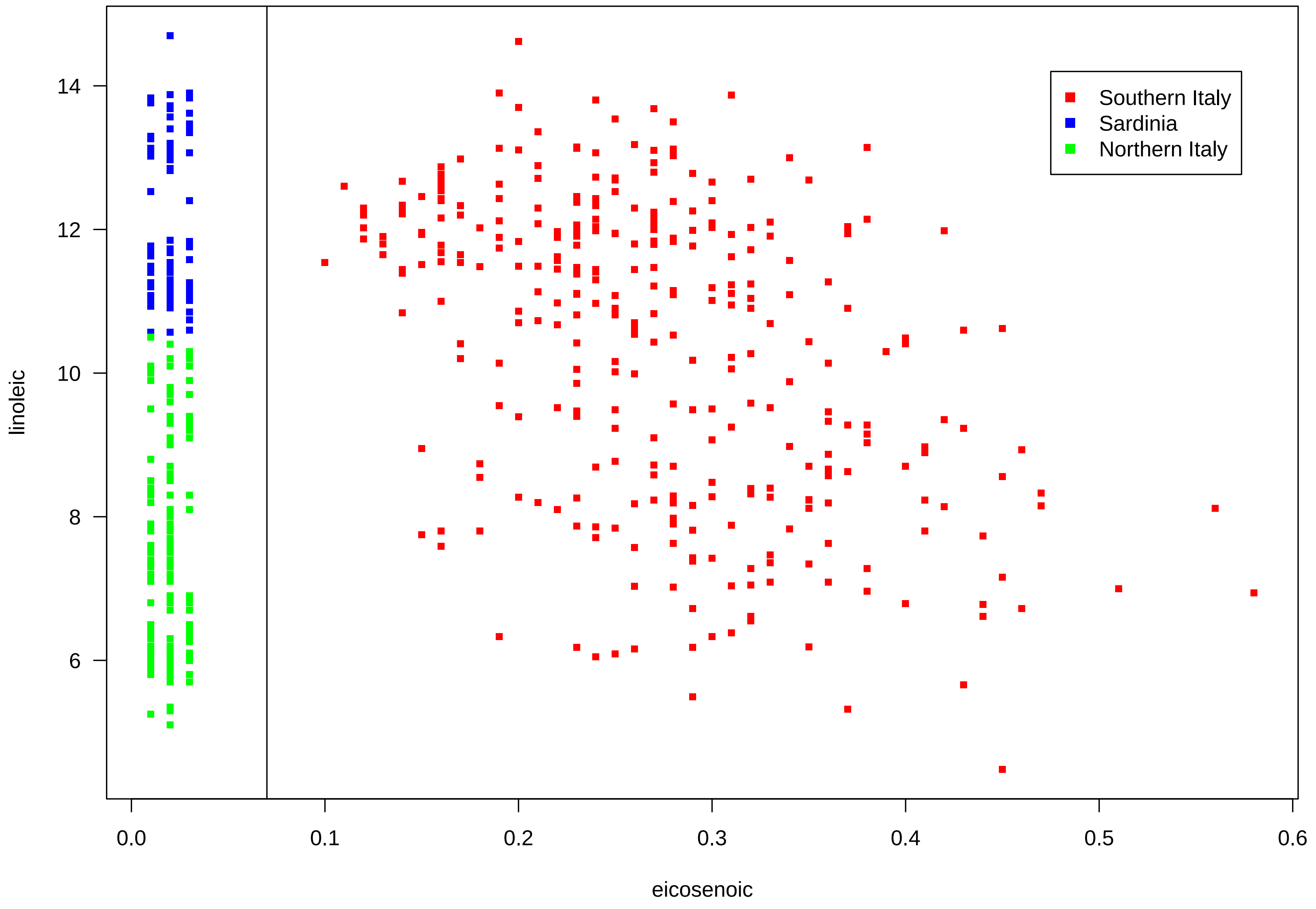
Reference:

Forina M, Armanino C, Lanteri S, and Tiscornia E (1983). *Classification of olive oils from their fatty acid composition*. In Martens H and Russwurm Jr H, editors, Food Research and Data Analysis, pp 189-214. Applied Science Publishers, London.

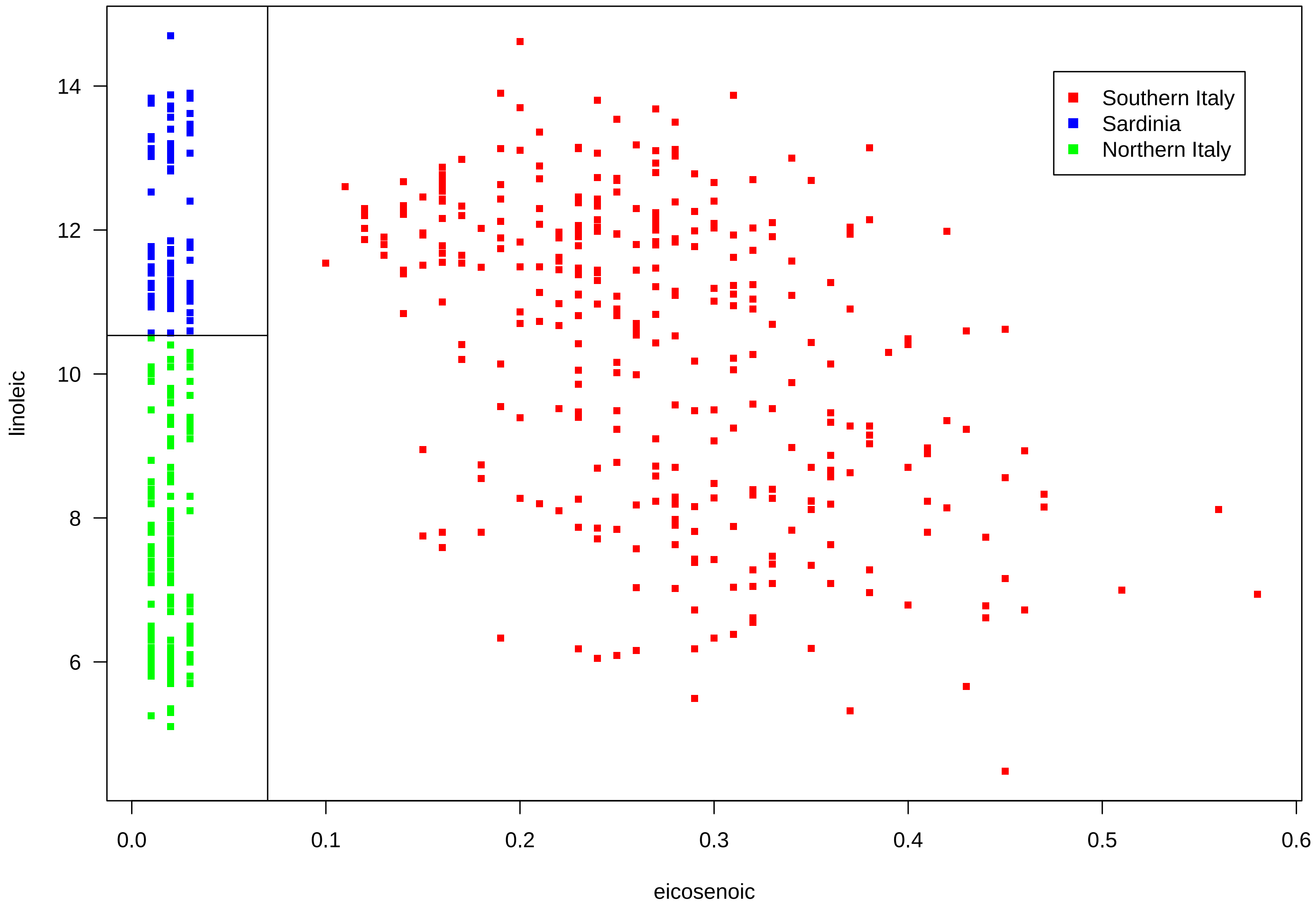
The Olive Data



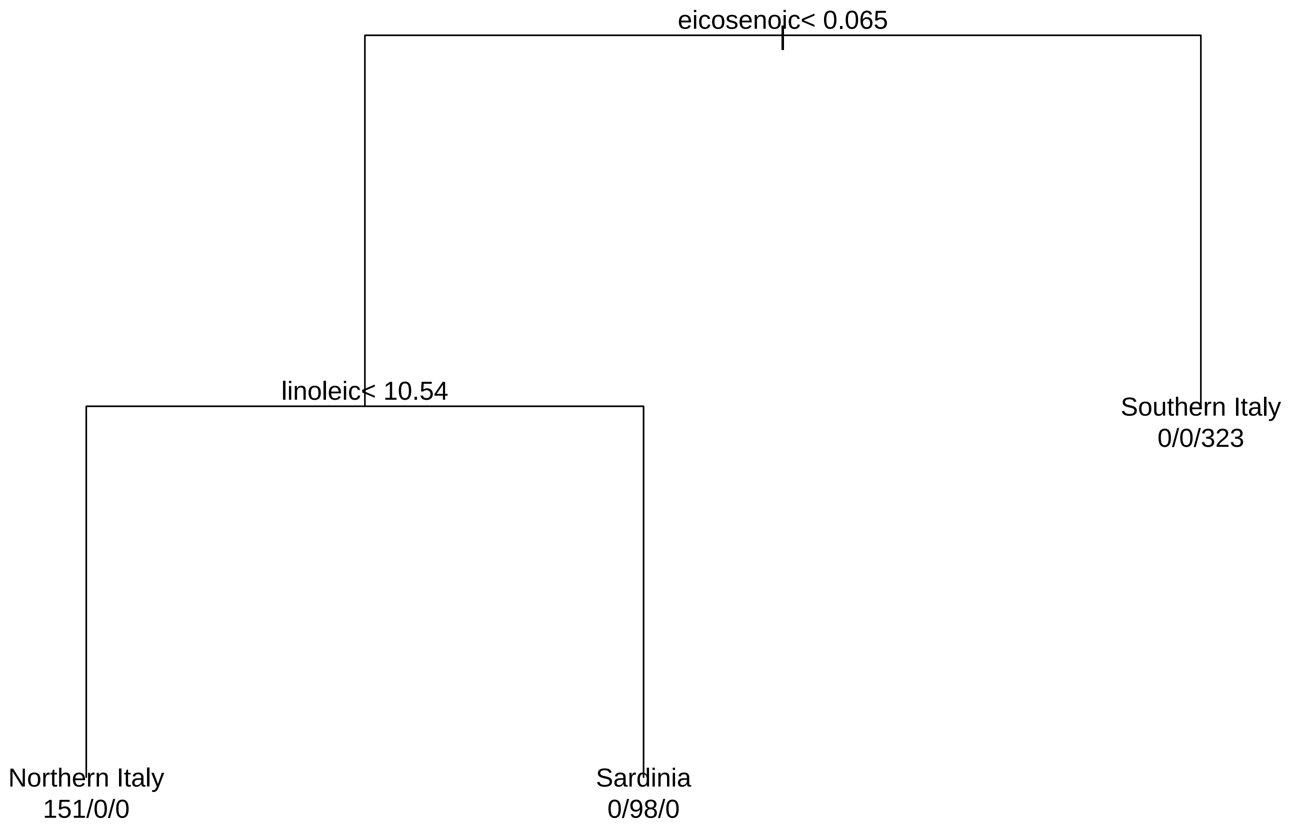
The Olive Data



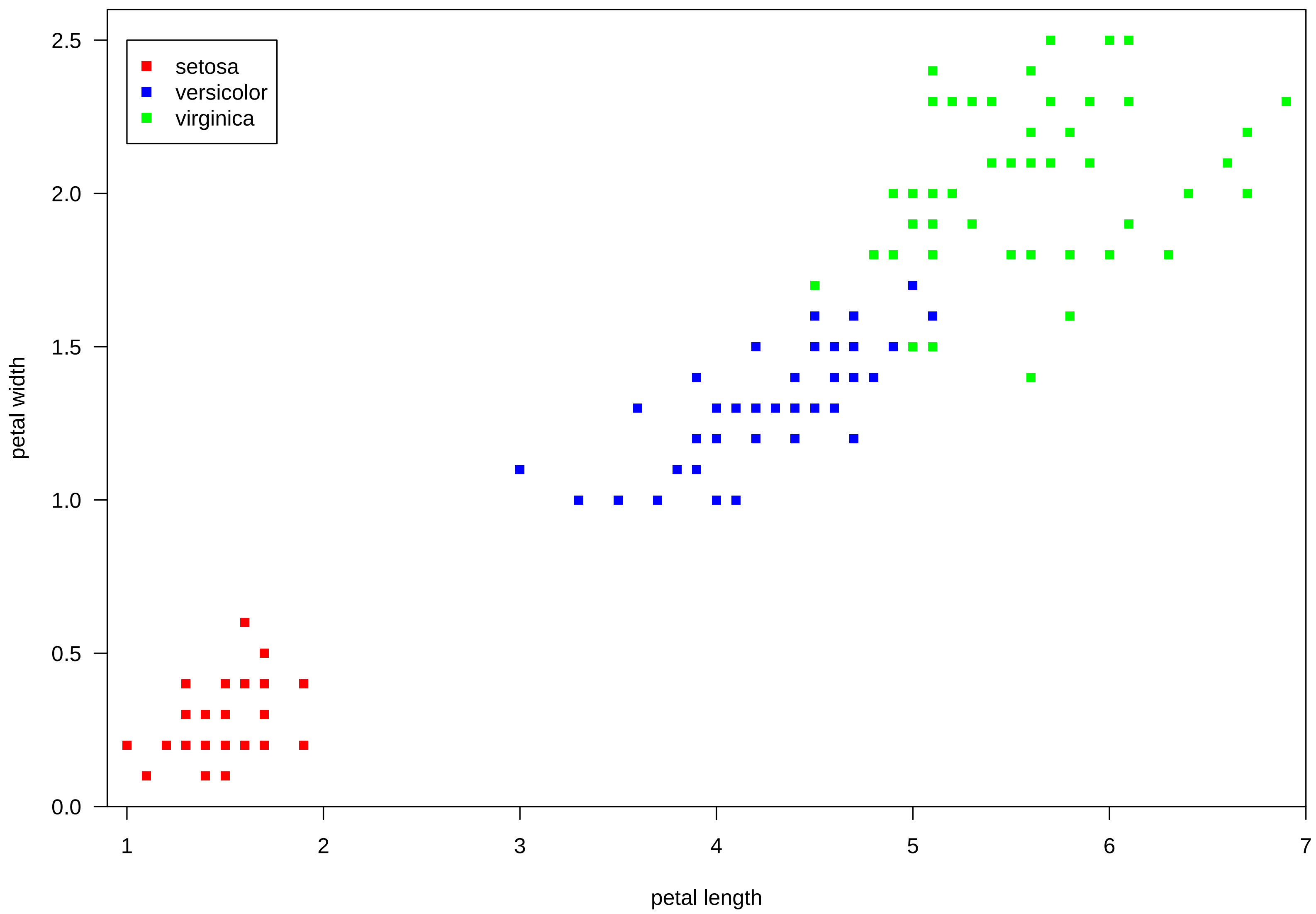
The Olive Data



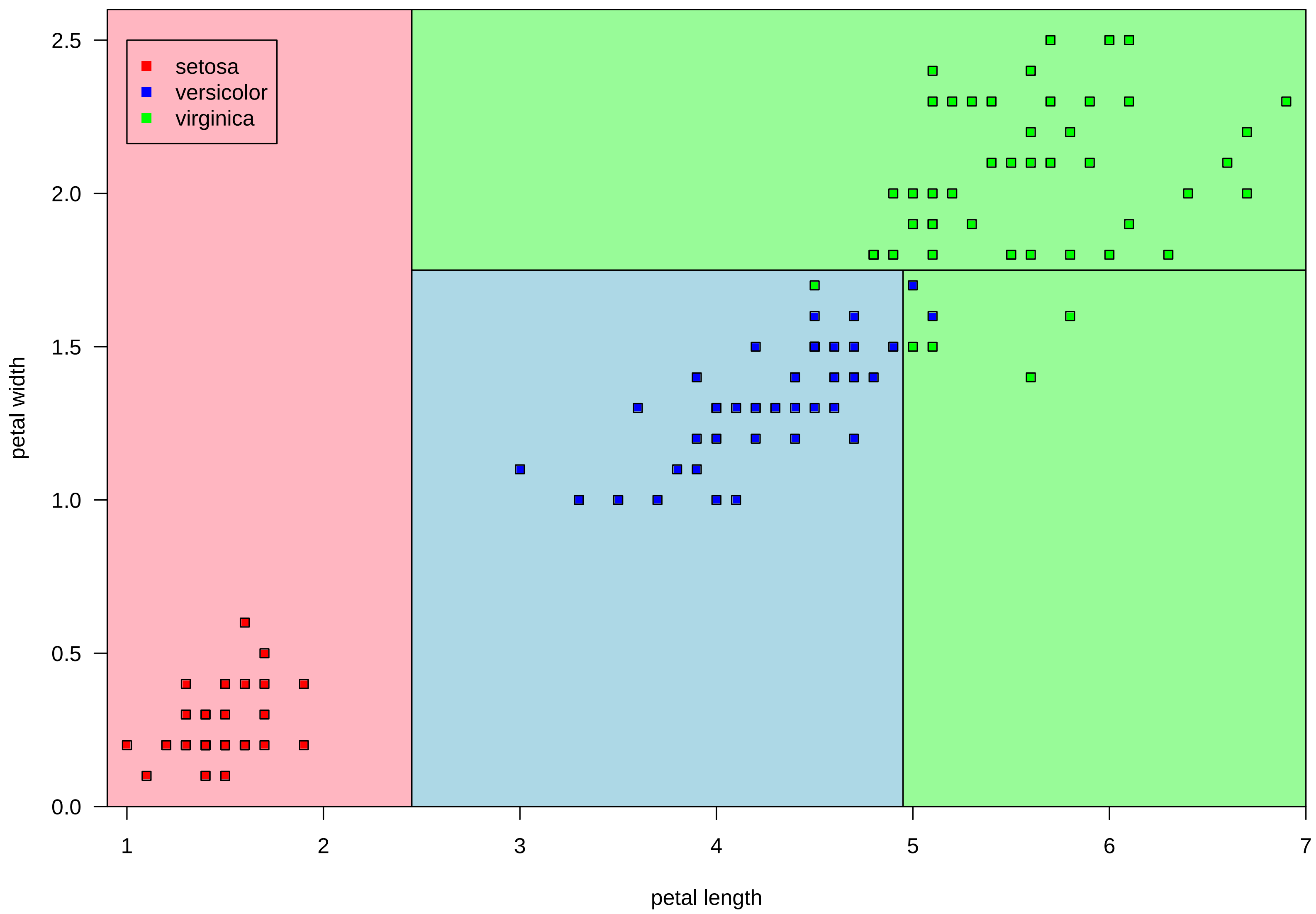
The Olive Data



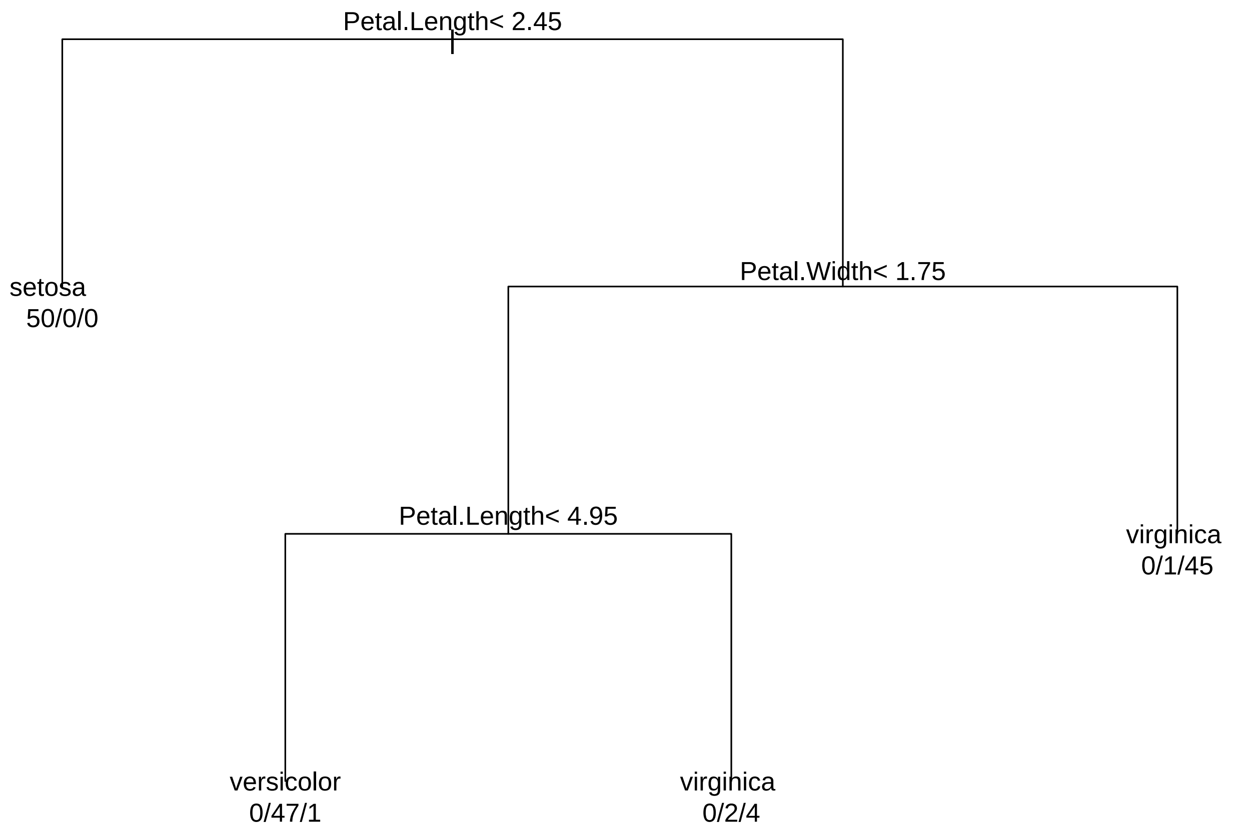
Fisher’s Iris Data



Fisher’s Iris Data

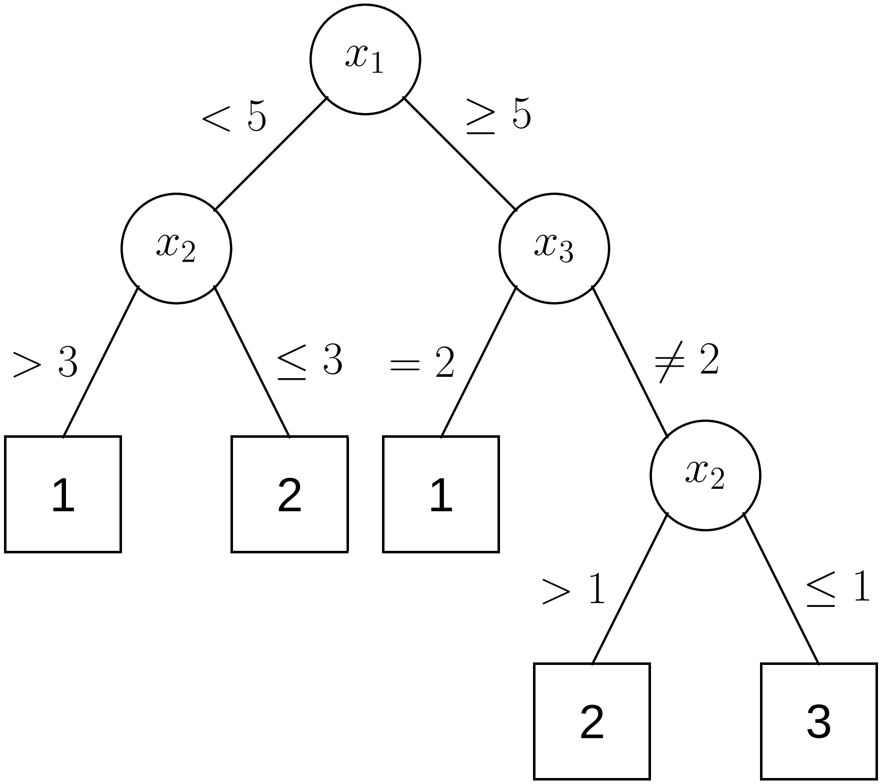


Fisher’s Iris Data



Classification Tree

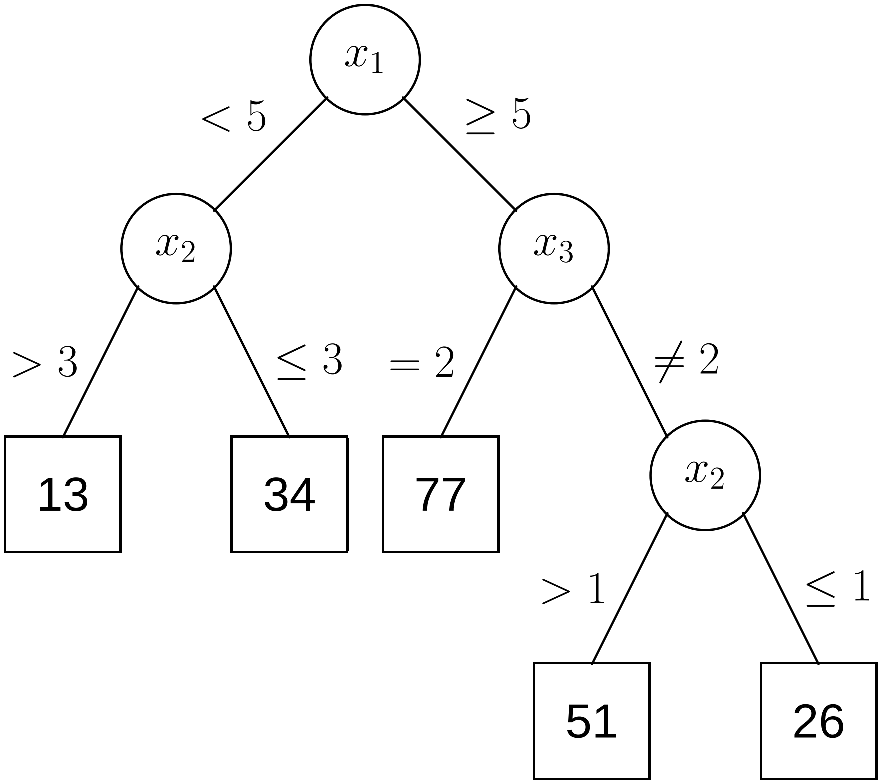
Suppose that we have a scalar outcome, Y, and a p-vector of explanatory variables, X. Assume $Y \in \mathcal{K} = \{1, 2, \ldots, k\}$



A classification tree partitions the X-space and provides a predicted value, perhaps $\arg\max_s \Pr(Y = s | X \in A_k)$ in each region.

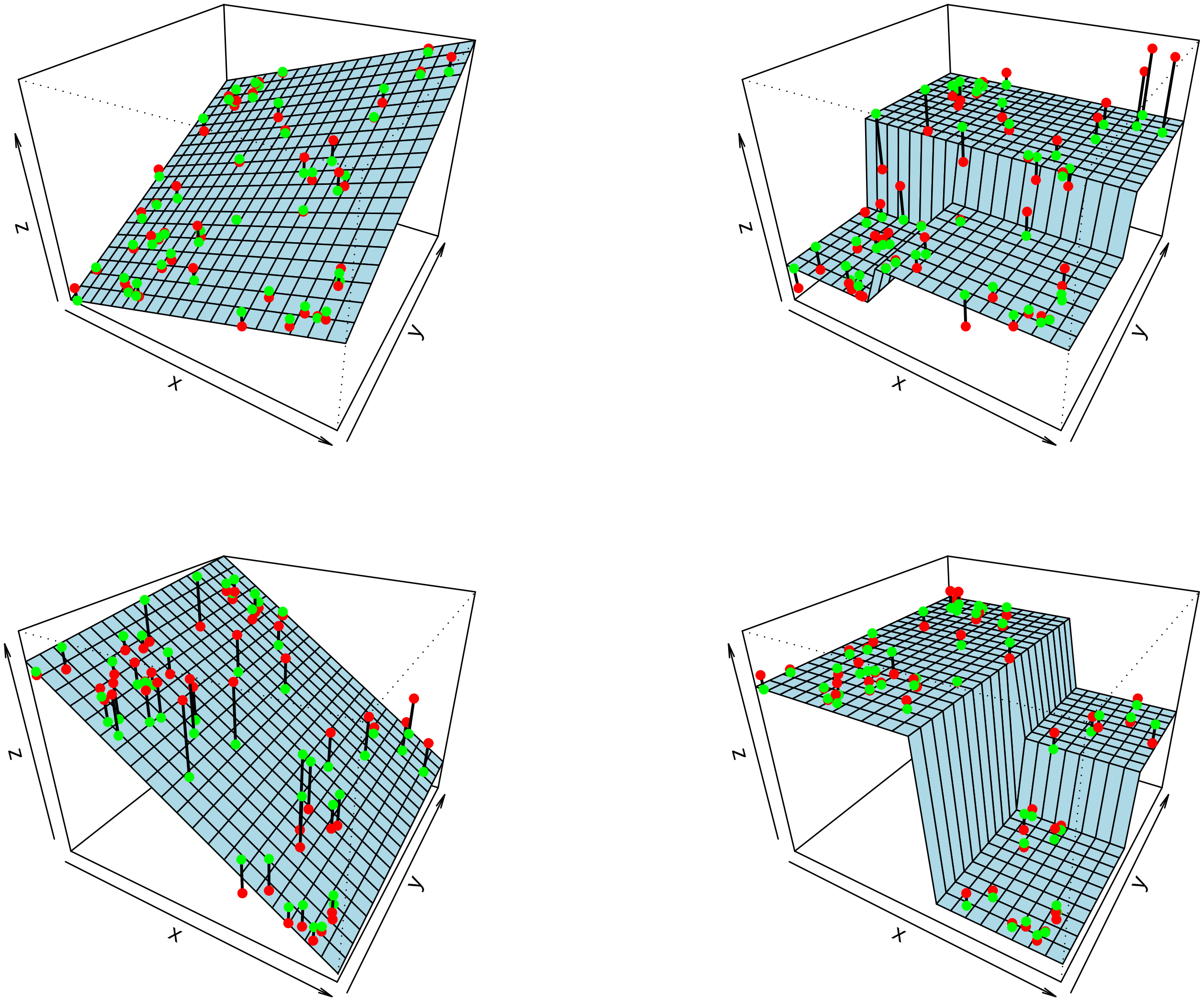
Regression Tree

Again, suppose that we have a scalar outcome, Y, and a p-vector of explanatory variables, X. Now assume $Y \in \mathcal{R}$.



A regression tree partitions the X-space into disjoint regions A_k and provides a fitted value $\mathrm{E}(Y|X \in A_k)$ within each region.

CART versus Linear Model



Tree Search

The search through trees is generally performed as follows:

1. **Grow** an overly large tree using forward selection.

   At each step, find the *best* split.

   Grow until all terminal nodes either

   (a) have $< n$ (perhaps $n = 1$) data points,

   (b) are “pure” (all points in a node have [almost] the same outcome).

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

Note: This suffers from the usual problems of forward selection / greedy searches!

The Predictor Space

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

Each of the X_i can be

a) a numeric variable:
$\longrightarrow$ $n - 1$ possible splits.

b) an ordered factor:
$\longrightarrow$ $k - 1$ possible splits.

b) an unordered factor:
$\longrightarrow$ $2^{k-1} - 1$ possible splits.

We pick the split that results in the greatest decrease in impurity (according to some impurity measure).

A Probabilistic Approach

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

- 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}.

- Given X, the conditional likelihood is then proportional to $\prod_{(\text{leaves } i)} \prod_{(\text{classes } k)} p_{ik}^{n_{ik}}$.

- Define a deviance $D = \sum D_i$, where $D_i = -2 \sum_k n_{ik} \log(p_{ik})$.

- Estimate p_{ik} by $\hat{p}_{ik} = \frac{n_{ik}}{n_{i.}}$.

The Olive Data

Root	$n_{11} = 246$	$n_{12} = 74$	$n_{13} = 116$	$n_1 = 436$	$D = 851.2$
	$\hat{p}_{11} = \frac{246}{436}$	$\hat{p}_{12} = \frac{74}{436}$	$\hat{p}_{13} = \frac{116}{436}$		
Split 1	$n_{11} = 246$	$n_{12} = 0$	$n_{13} = 0$	$n_1 = 246$	$D = 254.0$
	$n_{21} = 0$	$n_{22} = 74$	$n_{23} = 116$	$n_2 = 190$	
	$\hat{p}_{11} = 1$	$\hat{p}_{12} = 0$	$\hat{p}_{13} = 0$		
	$\hat{p}_{21} = 0$	$\hat{p}_{22} = \frac{74}{190}$	$\hat{p}_{23} = \frac{116}{190}$		
Split 2	$n_{11} = 246$	$n_{12} = 0$	$n_{13} = 0$	$n_1 = 246$	$D = 0$
	$n_{21} = 0$	$n_{22} = 74$	$n_{23} = 0$	$n_2 = 74$	
	$n_{31} = 0$	$n_{32} = 0$	$n_{33} = 116$	$n_3 = 116$	

Other Measures of Impurity

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

- the entropy: $\sum p_{ik} \log(p_{ik})$.
- the GINI index: $\sum_{j \neq k} p_{ij} p_{ik} = 1 - \sum_k p_{ik}^2$.

For regression trees we usually define

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

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

Recursive Partitioning

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.

Model Selection

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

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

Model Selection

How to choose α?

- Classification with k classes: $\alpha = 2(k-1)$ is AIC.
- Regression: $\alpha = 2\hat{\sigma}^2$ (based on Mallow's Cp approximation to the AIC criterion).
- Training/test set approach.
- Cross-validation.
- Averaging CV across several splits.

General Points

What’s nice:

- Decision trees are very “natural” constructs, in particular when the explanatory variables are categorical (and even better, when they are binary).

- Trees are very easy to explain to non-statisticians.

- The models are invariant under transformations in the predictor space.

- Multi-factor response is easily dealt with.

- The treatment of missing values is more satisfactory than for most other model classes.

- The models go after interactions immediately, rather than as an afterthought.

- The tree growth is actually more efficient than I have described it.

- There are extensions for survival and longitudinal data, and there is an extension called treed models. There is even a Bayesian version of CART.

General Points

What’s not so nice:

- The tree-space is huge, so we may need a lot of data.
- We might not be able to find the “best” model at all.
- It can be hard to assess uncertainty in inference about trees.
- The results can be quite variable (the tree selection is not very stable).
- Actual additivity becomes a mess in a binary tree.
- Simple trees usually do not have a lot of predictive power.
- There is a selection bias for the splits.

References

- L Breiman.
  Statistical Modeling: The Two Cultures.
  Statistical Science, 16 (3), pp 199-215, 2001.

- L Breiman, JH Friedman, RA Olshen, and CJ Stone.
  Classification and Regression Trees.
  Wadsworth Inc, 1984.

- TM Therneau and EJ Atkinson.
  An Introduction to Recursive Partitioning Using the RPART Routines.
  Technical Report Series No 61, Department of Health Science Research, Mayo Clinic, Rochester, Minnesota, 2000.

- WN Venables and BD Ripley.
  Modern Applied Statistics with S.
  Springer NY, 4th edition, 2002.

Bagging

- Bagging predictors is a method for generating multiple versions of a predictor and using these to get an aggregated predictor.
- The aggregation averages over the versions when predicting a numerical outcome and does a plurality vote when predicting a class.
- The multiple versions are formed by making bootstrap replicates of the learning set and using these as new learning sets.
- The vital element is the instability of the prediction method. If perturbing the learning set can cause significant changes in the predictor constructed, then bagging can improve accuracy.

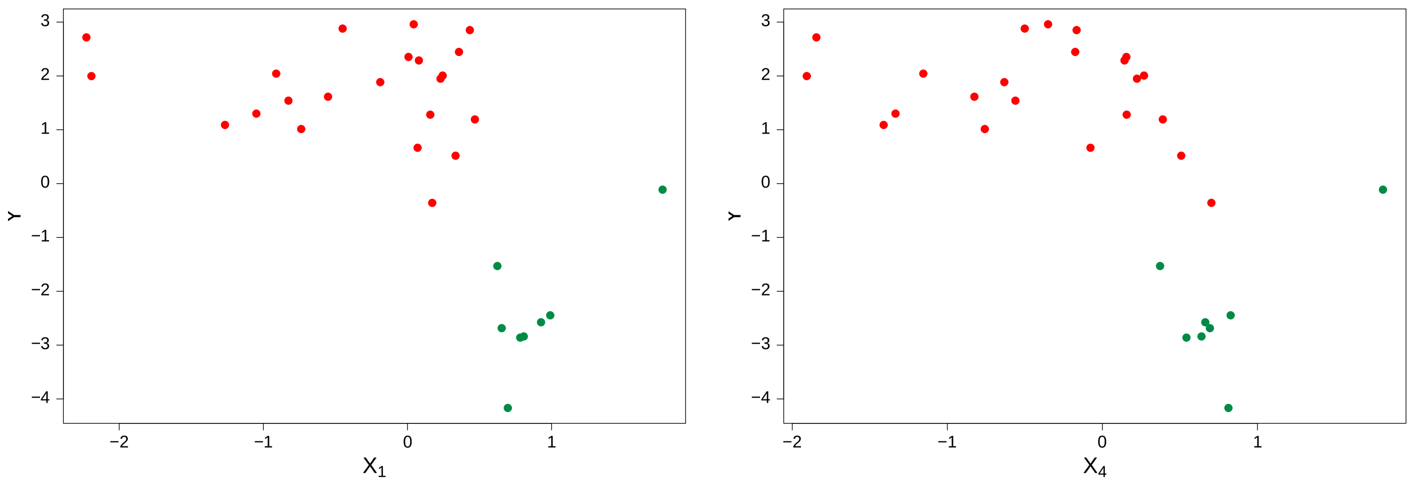
Bagging = Bootstrap aggregating

Reference: Breiman L (1996): *Bagging Predictors*, Machine Learning, Vol 24 (2), pp 123-140.

Bagging

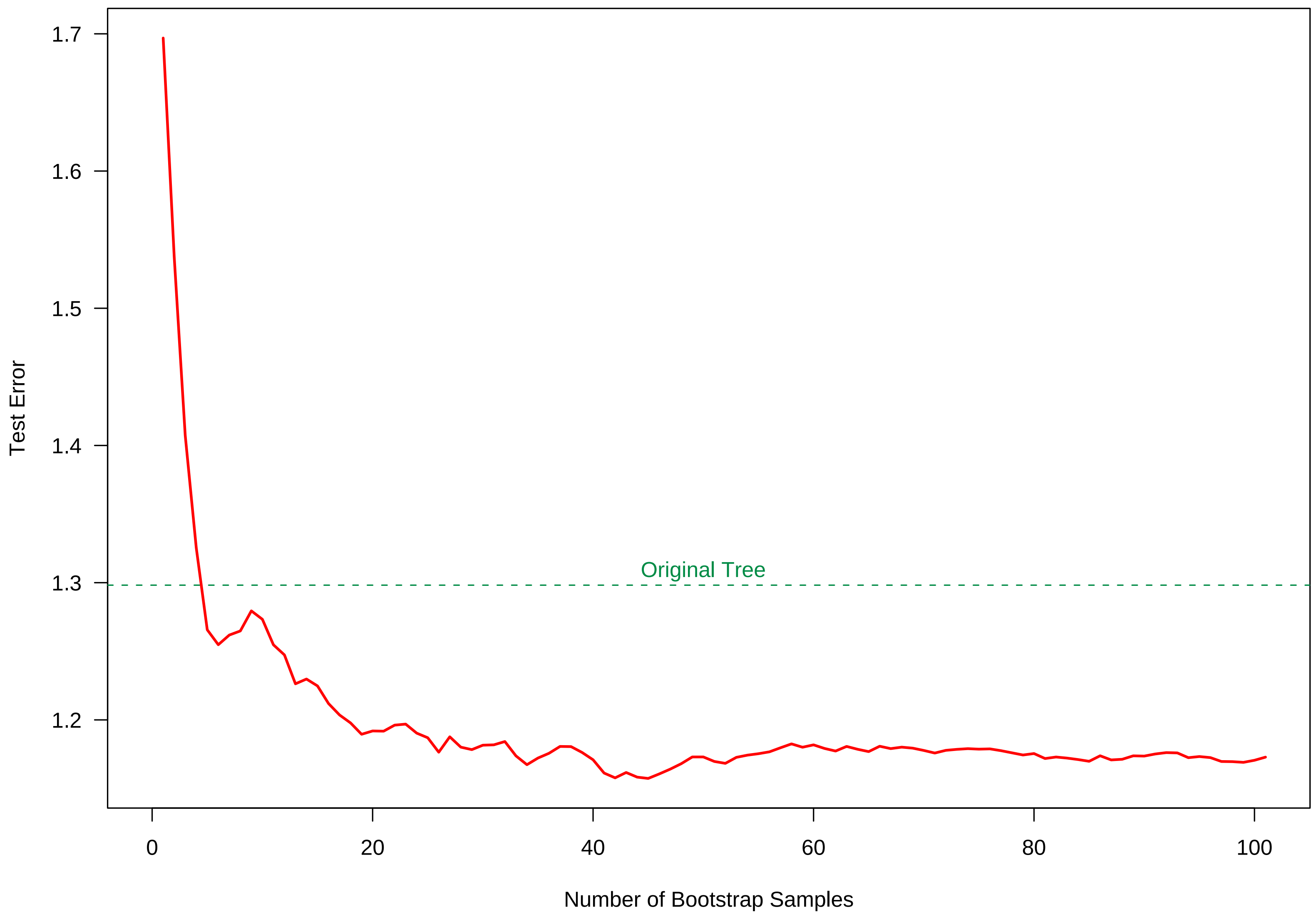
- Generate a sample of size $N = 30$ with two classes and $p = 5$ features, each having a standard Gaussian distribution with pairwise correlation 0.95.

- The response was generated as $Y \sim N\left(\mu = 2 - 4 \times I_{[X_1 > 0.5]}\ ,\ \sigma^2 = 1\right)$



- A test sample of size 2000 was also generated from the same population.

Bagging


Test Error
1.7
1.6
1.5
1.4
1.3
1.2
Original Tree
0
20
40
60
80
100
Number of Bootstrap Samples

Bagging

Note:

- Bagging can dramatically reduce the variance of unstable procedures such as trees, leading to improved prediction.
- A simple argument can show why bagging helps under squared error loss: averaging reduces variance and leaves bias unchanged.

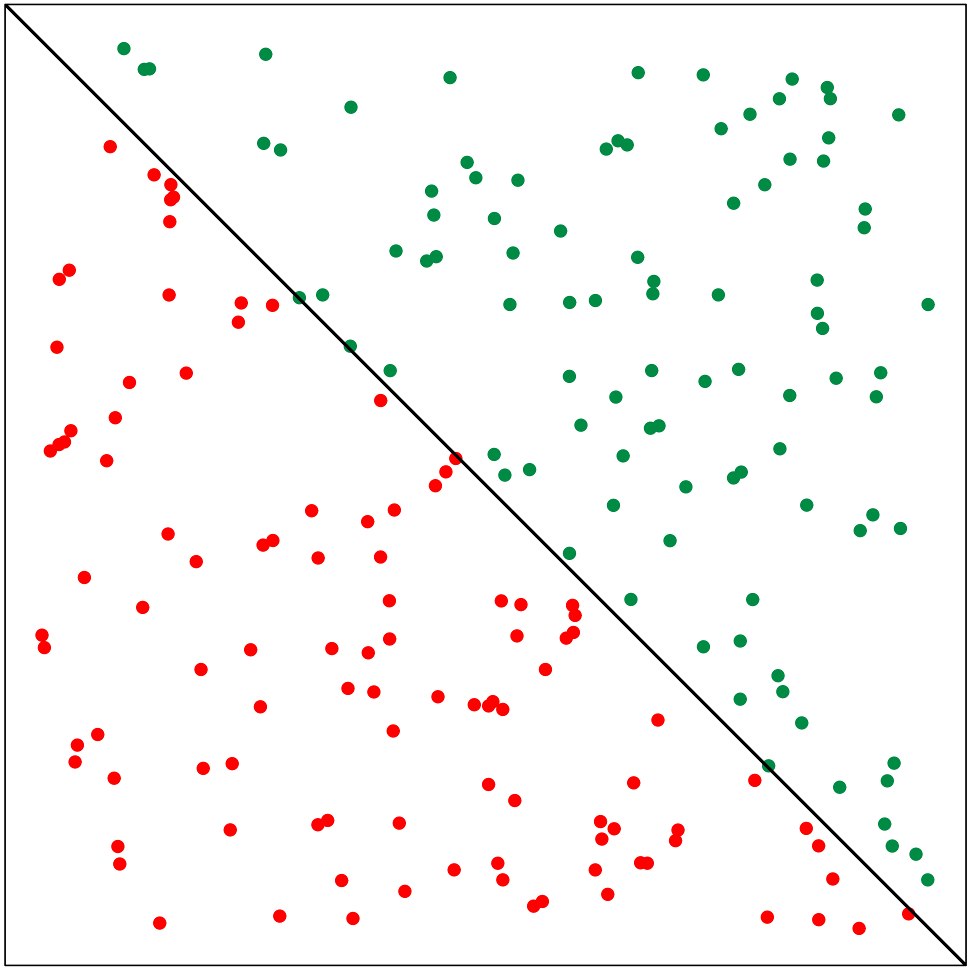
Reference: Hastie T, Tibshirani R, and Friedman J (2001): *The Elements of Statistical Learning*, Springer, NY.

However:

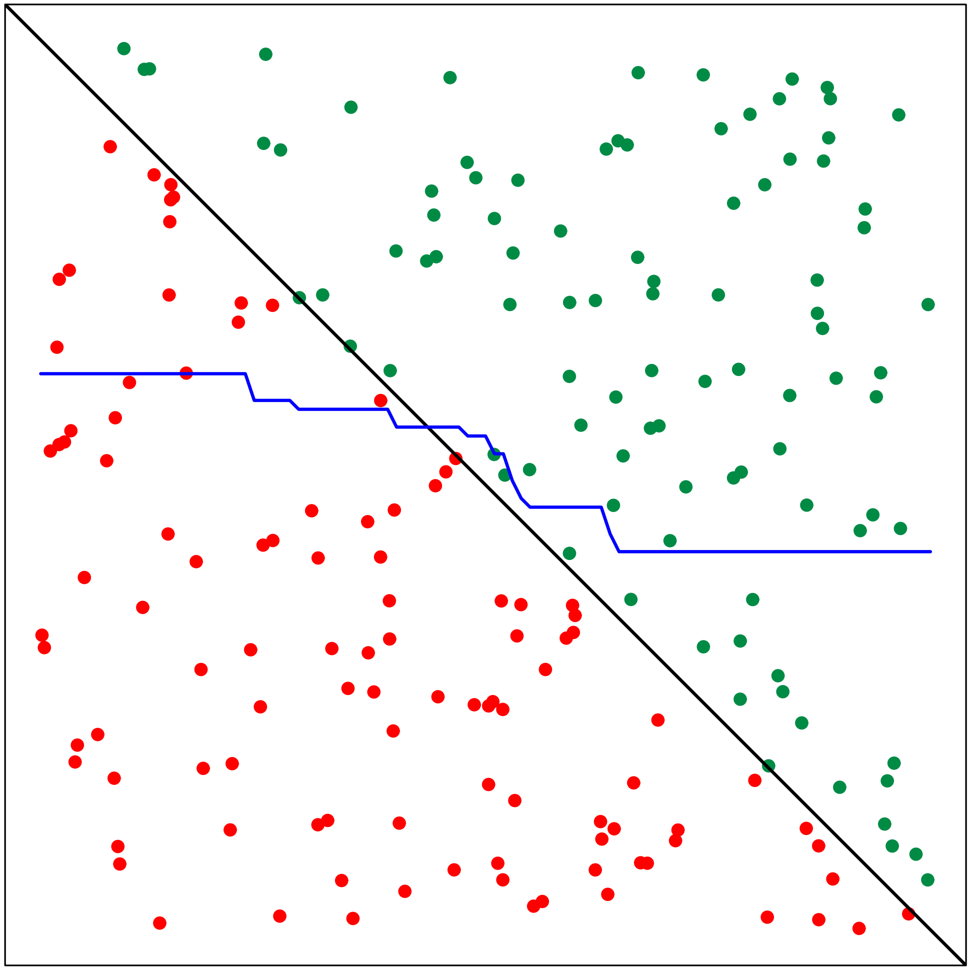
- The above argument breaks down for classification under 0-1 loss.
- Other tree-based classifiers such as random split selection perform consistently better.

Reference: Dietterich T (2000): *An Experimental Comparison of Three Methods for Constructing Ensembles of Decision Trees: Bagging, Boosting, and Randomization*, Machine Learning 40:139-157.

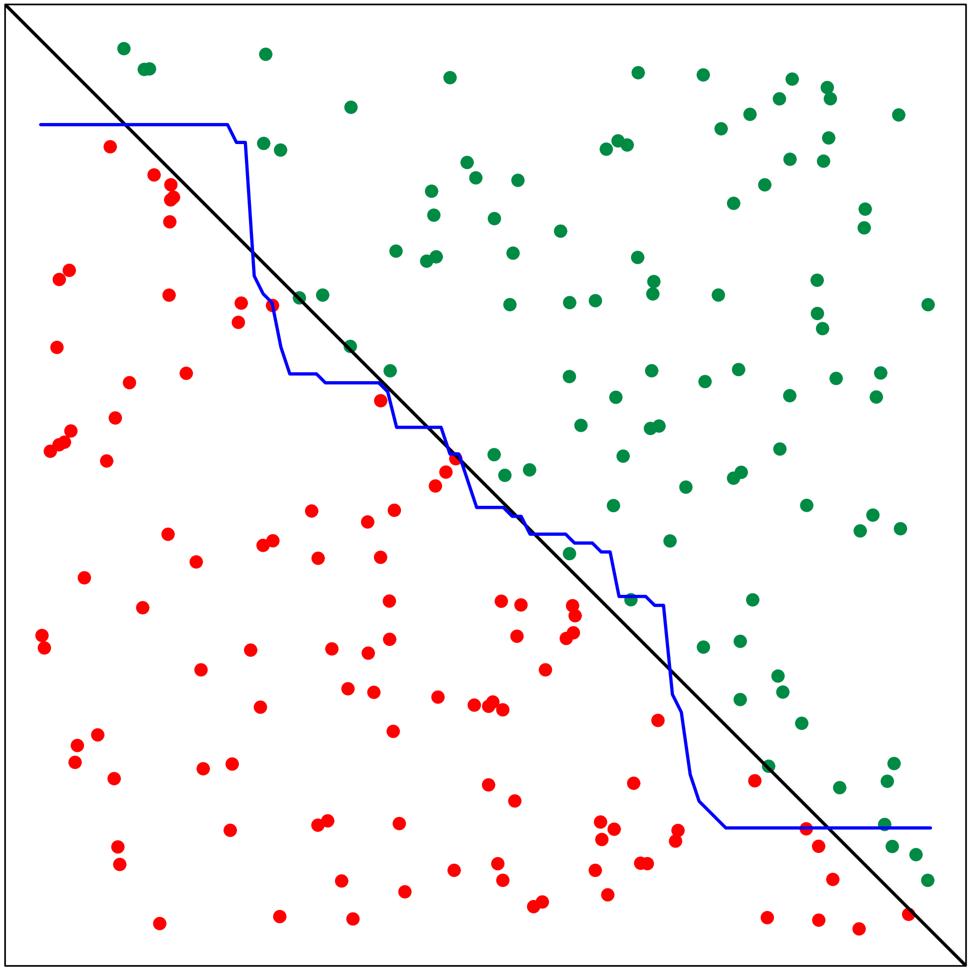
Bagging



Bagging



Bagging



Random Forests

- Grow many classification trees using a probabilistic scheme.
  ⟶ A random forest of trees!

- Classify a new object from an input vector by putting the input vector down each of the trees in the forest.

- Each tree gives a classification (i. e. the tree votes for a class).

- The forest chooses the classification having the most votes over all the trees in the forest.

Adapted from: *Random Forests* by Leo Breiman and Adele Cutler. http://www.math.usu.edu/∼adele/forests/

Random Forests

Each tree is grown as follows:

1. If the number of cases in the training set is N, sample N cases at random - but with replacement, from the original data. This sample will be the training set for growing the tree.

2. If there are M input variables, a number m $<<$ M is specified such that at each node, m variables are selected at random out of the M and the best split on these m is used to split the node. The value of m is held constant during the forest growing.

3. Each tree is grown to the largest extent possible. There is no pruning.

Two very nice properties of Random Forests:

- You can use the out of bag data to get an unbiased estimate of the classification error.
- It is easy to calculate a measure of “variable importance”.

Random Forests

The forest error rate depends on two things:

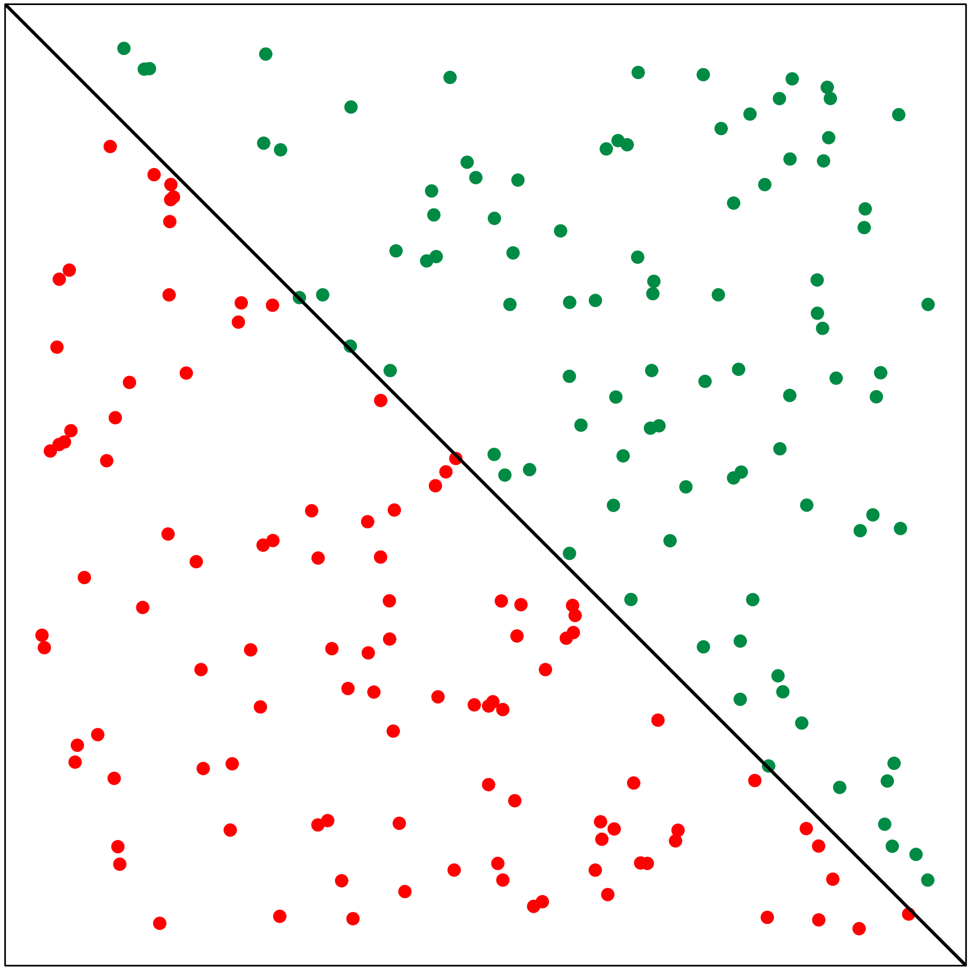
1. The correlation between any two trees in the forest. Increasing the correlation increases the forest error rate.

2. The strength of each individual tree in the forest. A tree with a low error rate is a strong classifier. Increasing the strength of the individual trees decreases the forest error rate.

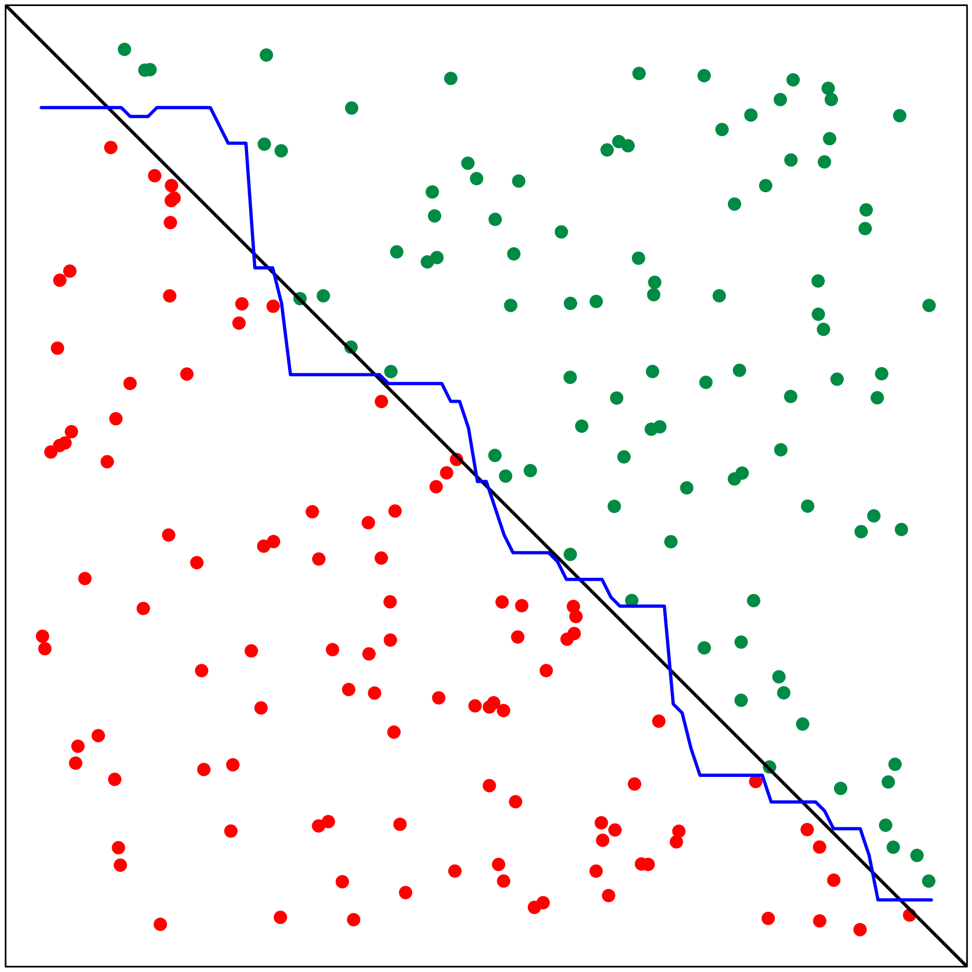
⟶ Reducing m reduces both the correlation and the strength. Increasing it increases both. Somewhere in between is an ”optimal” range of m - usually quite wide. This is the only adjustable parameter to which random forests is somewhat sensitive.

Reference: Breiman L. *Random Forests*. Machine Learning, 45(1):5-32, 2001.

Random Forests



Random Forests



Boosting

Idea: Take a series of weak learners and assemble them into a strong classifier.

Base classifier: $G(X) \rightarrow \{-1, +1\}$

Training data: $(x_i, y_i), \quad i = 1, \ldots, N$.

The most popular version is Adaboost.

$\longrightarrow$ Create a sequence of classifiers, giving higher influence to more accurate classifiers. During the iteration, mis-classified observations get a larger weight in the construction of the next classifier.

Reference: Freund Y and Schapire RE (1996): *Experiments with a New Boosting Algorithm*, Machine Learning: Proceedings of the Thirteenth International Conference, pp 148-156.

Boosting

Adaboost:

1. Initialize the observation weights $w_i = 1/N, \;\; i = 1, \ldots, N$.

2. For $m = 1, \ldots, M$

   (a) Fit a classifier $G_m(x)$ to the training data using the weights w_i.

   (b) Compute

$$\epsilon_m = \frac{\sum_i w_i \times I_{[y_i \neq G_m(x_i)]}}{\sum_i w_i}.$$

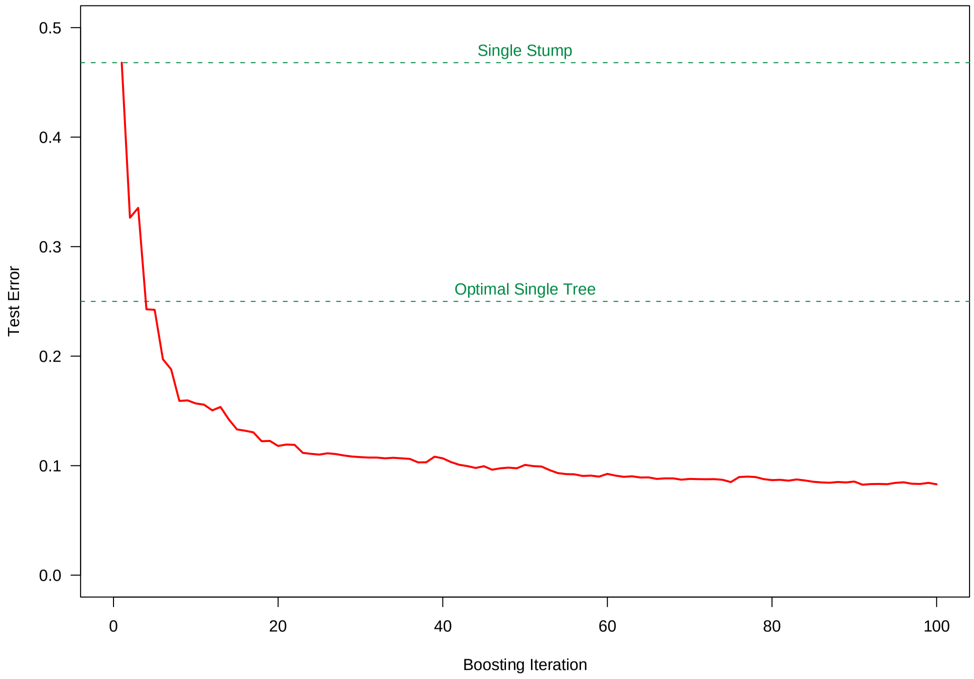
   (c) Compute $\alpha_m = \log\left(\frac{1-\epsilon_m}{\epsilon_m}\right)$.

   (d) Set $w_i \leftarrow w_i \times \exp\left\{\alpha_m I_{[y_i \neq G_m(x_i)]}\right\}, \;\; i = 1, \ldots, N$.

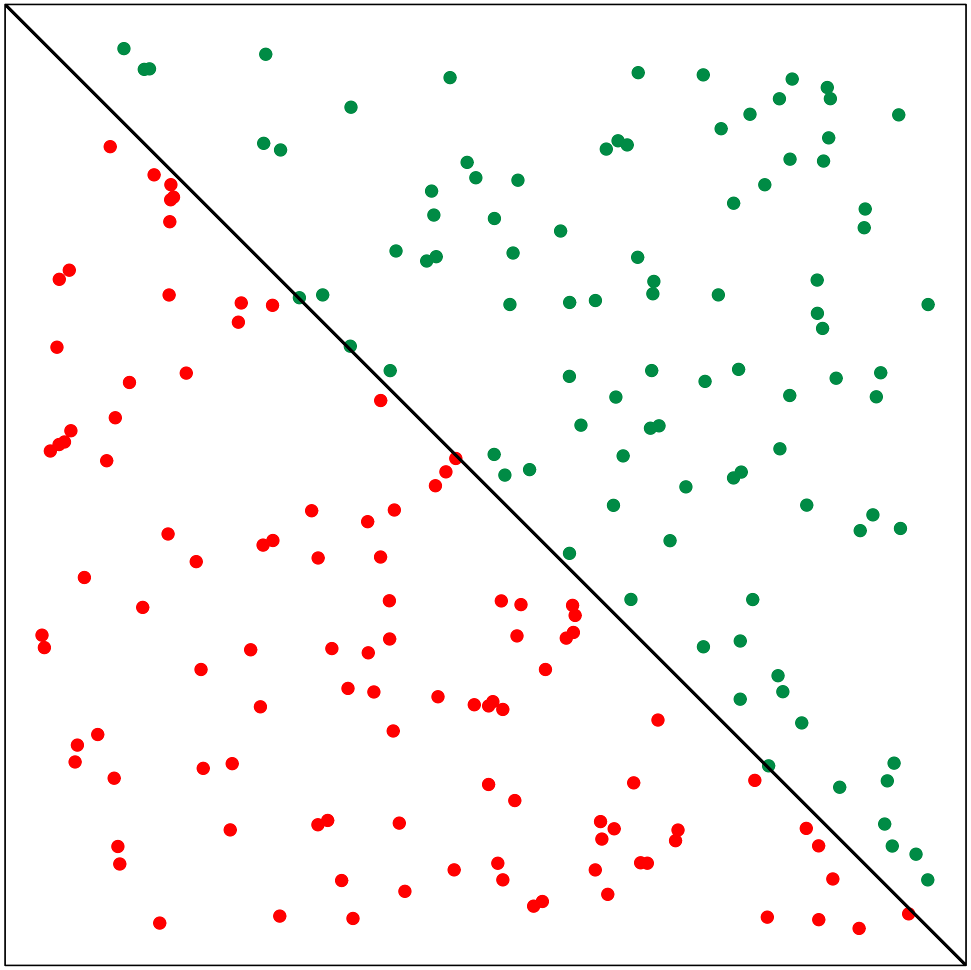
3. Output $G(x) = \text{sign}\left[\sum_m \alpha_m G_m(x)\right]$.

Boosting

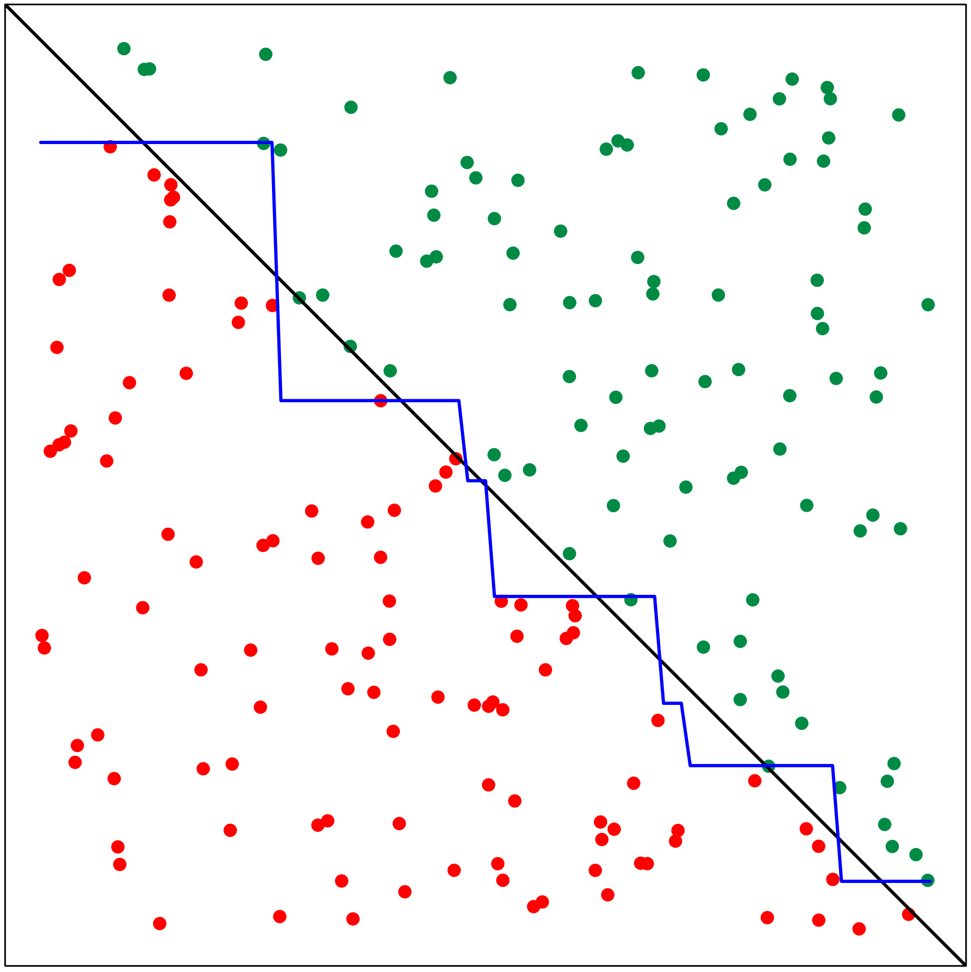
- Generate the features $X_1, \ldots, X_{10}$ as standard independent Gaussian.
- The target Y is defined as 1 if $\sum X_j^2 > \chi^2_{10}(0.5)$, and -1 otherwise.
- There are 2000 training cases with approximately 1000 cases in each class, and 10,000 test observations.



Boosting



Boosting



Miscellaneous

- There are many flavors of boosting - even many flavors of Adaboost!

- What we talked about today also goes under the name Arcing: Adaptive Reweighting (or Resampling) and Combining.

- There are R packages on CRAN for Random Forests (randomForest) and boosting (gbm).

- Find more details about the issues discussed in Hastie T, Tibshirani R, and Friedman J (2001), *The Elements of Statistical Learning*.