



Predicting with trees

Jeffrey Leek, Assistant Professor of Biostatistics
Johns Hopkins Bloomberg School of Public Health

Key ideas

- Iteratively split variables into groups
- Split where maximally predictive
- Evaluate "homogeneity" within each branch
- Fitting multiple trees often works better (forests)

Pros:

- Easy to implement
- Easy to interpret
- Better performance in nonlinear settings

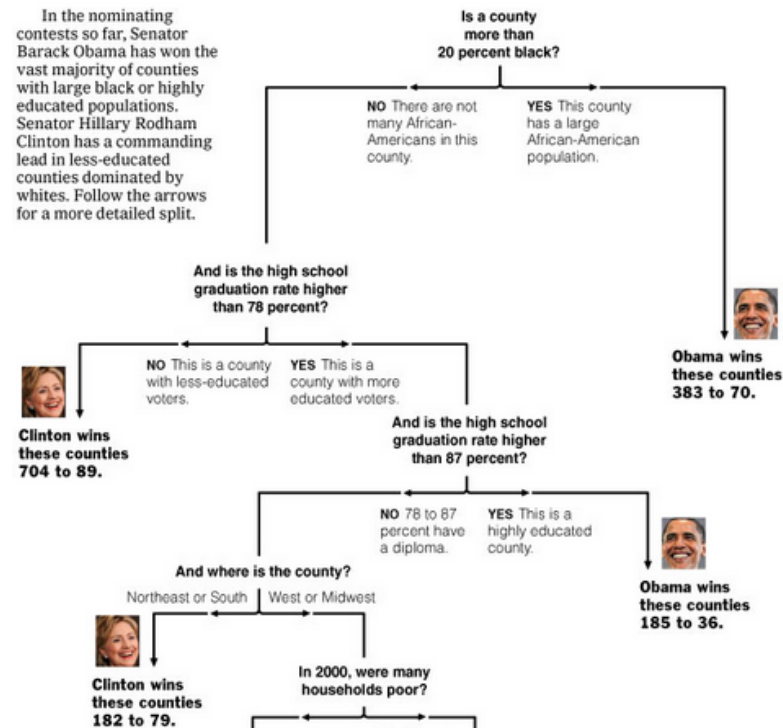
Cons:

- Without pruning/cross-validation can lead to overfitting
- Harder to estimate uncertainty
- Results may be variable

Example Tree

Decision Tree: The Obama-Clinton Divide

In the nominating contests so far, Senator Barack Obama has won the vast majority of counties with large black or highly educated populations. Senator Hillary Rodham Clinton has a commanding lead in less-educated counties dominated by whites. Follow the arrows for a more detailed split.



<http://graphics8.nytimes.com/images/2008/04/16/us/0416-nat-subOBAMA.jpg>

Basic algorithm

1. Start with all variables in one group
2. Find the variable/split that best separates the outcomes
3. Divide the data into two groups ("leaves") on that split ("node")
4. Within each split, find the best variable/split that separates the outcomes
5. Continue until the groups are too small or sufficiently "pure"

Measures of impurity

$$\hat{p}_{mk} = \frac{1}{N_m} \sum_{x_i \text{ in Leaf } m} \mathbb{1}(y_i = k)$$

Misclassification Error:

$$1 - \hat{p}_{mk(m)}$$

Gini index:

$$\sum_{k \neq k'} \hat{p}_{mk} \times \hat{p}_{mk'} = \sum_{k=1}^K \hat{p}_{mk} (1 - \hat{p}_{mk})$$

Cross-entropy or deviance:

$$-\sum_{k=1}^K \hat{p}_{mk} \ln \hat{p}_{mk}$$

Example: Iris Data

```
data(iris)
names(iris)
```

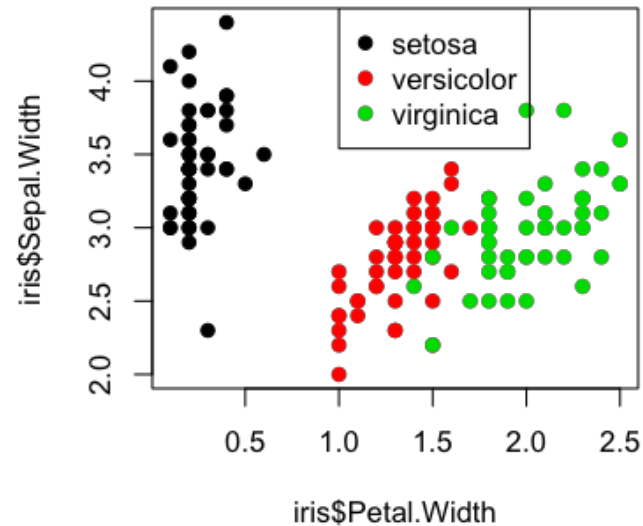
```
[1] "Sepal.Length" "Sepal.Width"  "Petal.Length" "Petal.Width"  "Species"
```

```
table(iris$Species)
```

setosa	versicolor	virginica
50	50	50

Iris petal widths/sepal width

```
plot(iris$Petal.Width,iris$Sepal.Width,pch=19,col=as.numeric(iris$Species))  
legend(1,4.5,legend=unique(iris$Species),col=unique(as.numeric(iris$Species)),pch=19)
```



Iris petal widths/sepal width

```
# An alternative is library(rpart)
library(tree)
tree1 <- tree(Species ~ Sepal.Width + Petal.Width, data=iris)
summary(tree1)
```

Classification tree:

```
tree(formula = Species ~ Sepal.Width + Petal.Width, data = iris)
```

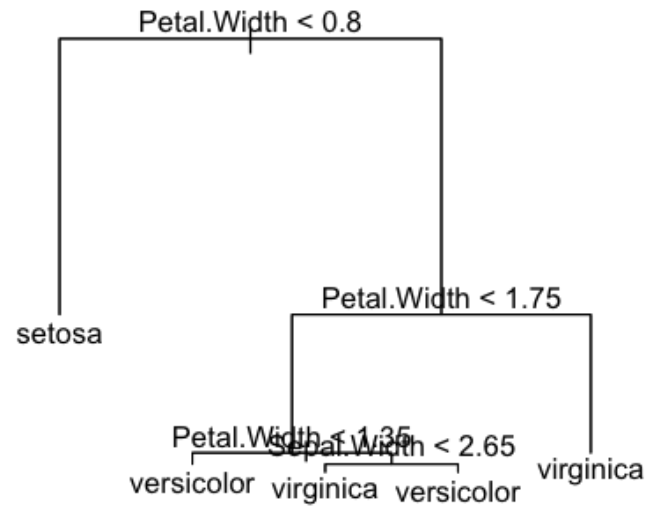
Number of terminal nodes: 5

Residual mean deviance: 0.204 = 29.6 / 145

Misclassification error rate: 0.0333 = 5 / 150

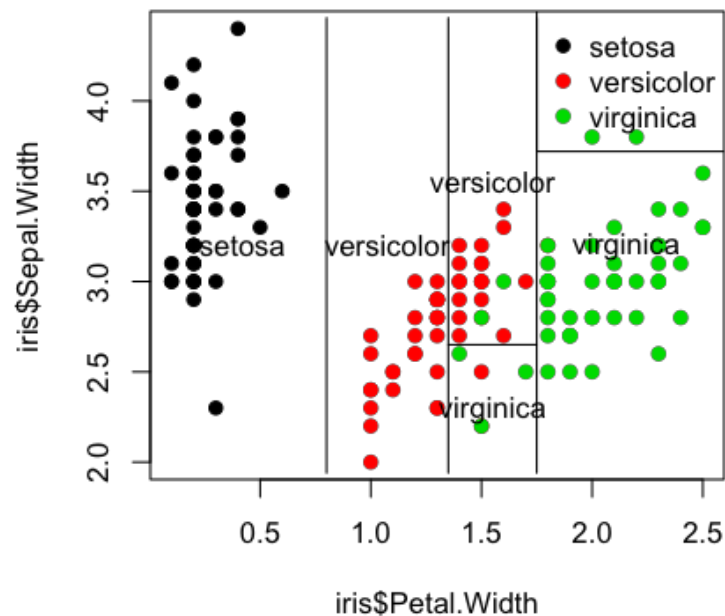
Plot tree

```
plot(tree1)  
text(tree1)
```



Another way of looking at a CART model

```
plot(iris$Petal.Width,iris$Sepal.Width,pch=19,col=as.numeric(iris$Species))  
partition.tree(tree1,label="Species",add=TRUE)  
legend(1.75,4.5,legend=unique(iris$Species),col=unique(as.numeric(iris$Species)),pch=19)
```



Predicting new values

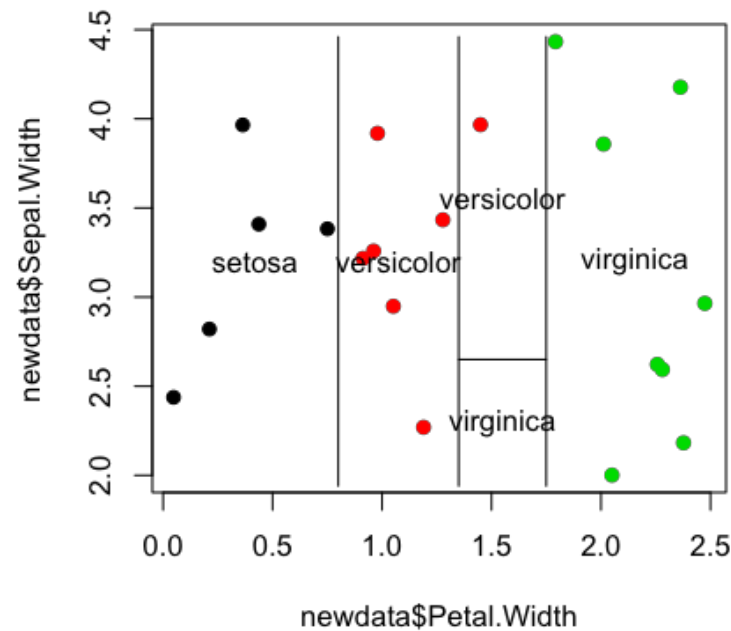
```
set.seed(32313)
newdata <- data.frame(Petal.Width = runif(20,0,2.5),Sepal.Width = runif(20,2,4.5))
pred1 <- predict(tree1,newdata)
pred1
```

	setosa	versicolor	virginica
1	0	0.02174	0.97826
2	0	0.02174	0.97826
3	1	0.00000	0.00000
4	0	1.00000	0.00000
5	0	0.02174	0.97826
6	0	0.02174	0.97826
7	0	0.02174	0.97826
8	0	0.90476	0.09524
9	0	1.00000	0.00000
10	0	0.02174	0.97826
11	0	1.00000	0.00000
12	1	0.00000	0.00000
13	1	0.00000	0.00000
14	1	0.00000	0.00000
15	0	0.02174	0.97826
16	0	0.02174	0.97826
17	0	1.00000	0.00000

11/18

Overlaying new values

```
pred1 <- predict(tree1,newdata,type="class")  
plot(newdata$Petal.Width,newdata$Sepal.Width,col=as.numeric(pred1),pch=19)  
partition.tree(tree1,"Species",add=TRUE)
```



Pruning trees example: Cars

```
data(Cars93, package="MASS")
head(Cars93)
```

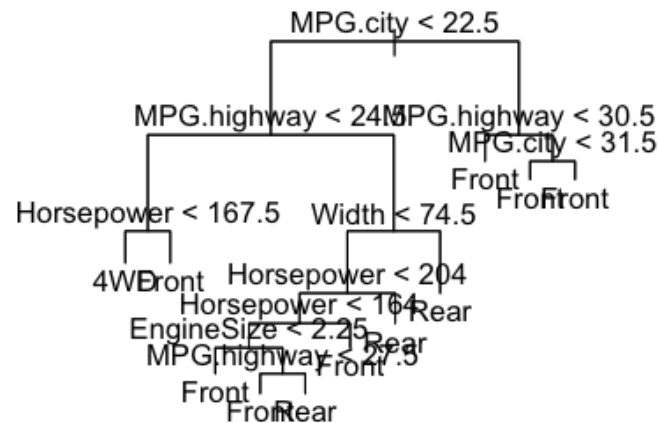
	Manufacturer	Model	Type	Min.Price	Price	Max.Price	MPG.city	MPG.highway	AirBags
1	Acura	Integra	Small	12.9	15.9	18.8	25	31	None
2	Acura	Legend	Midsize	29.2	33.9	38.7	18	25	Driver & Passenger
3	Audi	90	Compact	25.9	29.1	32.3	20	26	Driver only
4	Audi	100	Midsize	30.8	37.7	44.6	19	26	Driver & Passenger
5	BMW	535i	Midsize	23.7	30.0	36.2	22	30	Driver only
6	Buick	Century	Midsize	14.2	15.7	17.3	22	31	Driver only
	DriveTrain	Cylinders	EngineSize	Horsepower	RPM	Rev.per.mile	Man.trans.avail	Fuel.tank.capacity	
1	Front	4	1.8	140	6300	2890	Yes	13.2	
2	Front	6	3.2	200	5500	2335	Yes	18.0	
3	Front	6	2.8	172	5500	2280	Yes	16.9	
4	Front	6	2.8	172	5500	2535	Yes	21.1	
5	Rear	4	3.5	208	5700	2545	Yes	21.1	
6	Front	4	2.2	110	5200	2565	No	16.4	
	Passengers	Length	Wheelbase	Width	Turn.circle	Rear.seat.room	Luggage.room	Weight	Origin
1	5	177	102	68	37	26.5	11	2705	non-USA
2	5	195	115	71	38	30.0	15	3560	non-USA
3	5	180	102	67	37	28.0	14	3375	non-USA
4	6	193	106	70	37	31.0	17	3405	non-USA
5	4	186	109	69	39	27.0	13	3640	non-USA

13/18

Build a tree

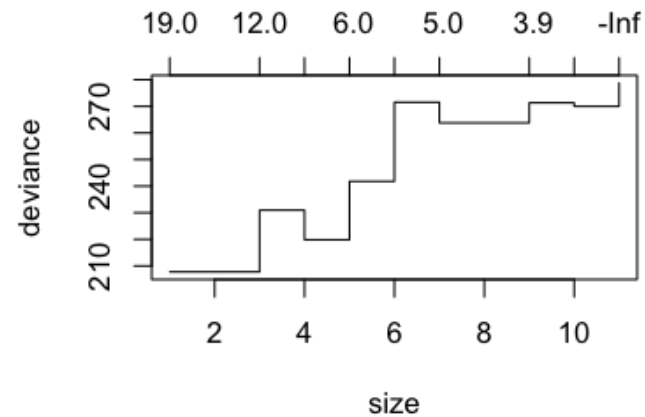
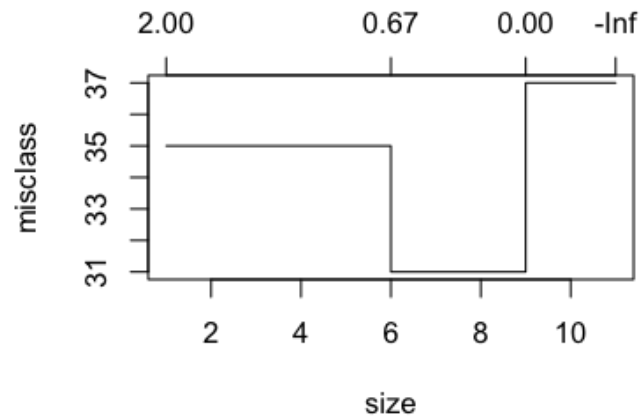
```
treeCars <- tree(DriveTrain ~ MPG.city + MPG.highway + AirBags +
  EngineSize + Width + Length + Weight + Price + Cylinders +
  Horsepower + Wheelbase,data=Cars93)

plot(treeCars)
text(treeCars)
```



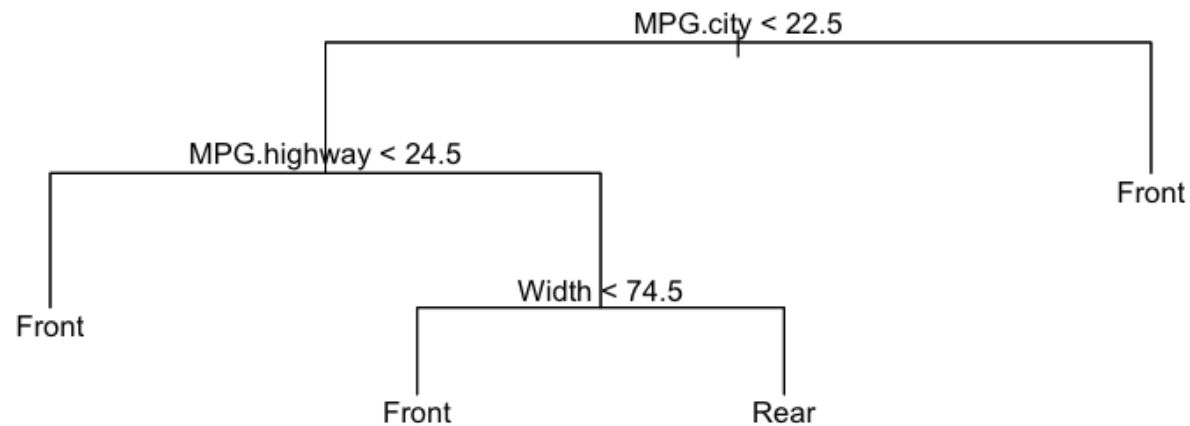
Plot errors

```
par(mfrow=c(1,2))  
plot(cv.tree(treeCars,FUN=prune.tree,method="misclass"))  
plot(cv.tree(treeCars))
```



Prune the tree

```
pruneTree <- prune.tree(treeCars,best=4)  
plot(pruneTree)  
text(pruneTree)
```



Show resubstitution error *

```
table(Cars93$DriveTrain, predict(pruneTree, type="class"))  
table(Cars93$DriveTrain, predict(treeCars, type="class"))
```

	4WD	Front	Rear
4WD	5	5	0
Front	2	61	4
Rear	0	3	13

- Note that cross validation error is a better measure of test set accuracy

Notes and further resources

- [Hector Corrada Bravo's Notes](#), [code](#)
- [Cosma Shalizi's notes](#)
- [Elements of Statistical Learning](#)
- [Classification and regression trees](#)
- [Random forests](#)