Classification and Regression Trees

Jason Bryer, Ph.D.

2024-03-26

Classification and Regression Trees

The goal of CART methods is to find best predictor in X of some outcome, y. CART methods do this recursively using the following procedures:

- Find the best predictor in X for y.
- Split the data into two based upon that predictor.
- Repeat 1 and 2 with the split data sets until a stopping criteria has been reached.

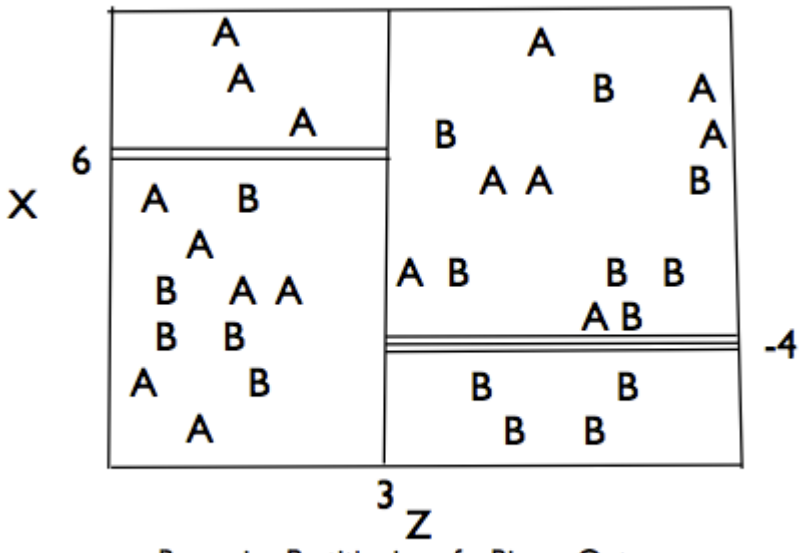
There are a number of possible stopping criteria including: Only one data point remains.

- All data points have the same outcome value.
- No predictor can be found that sufficiently splits the data.

Recursive Partitioning Logic of CART

Consider the scatter plot to the right with the following characteristics:

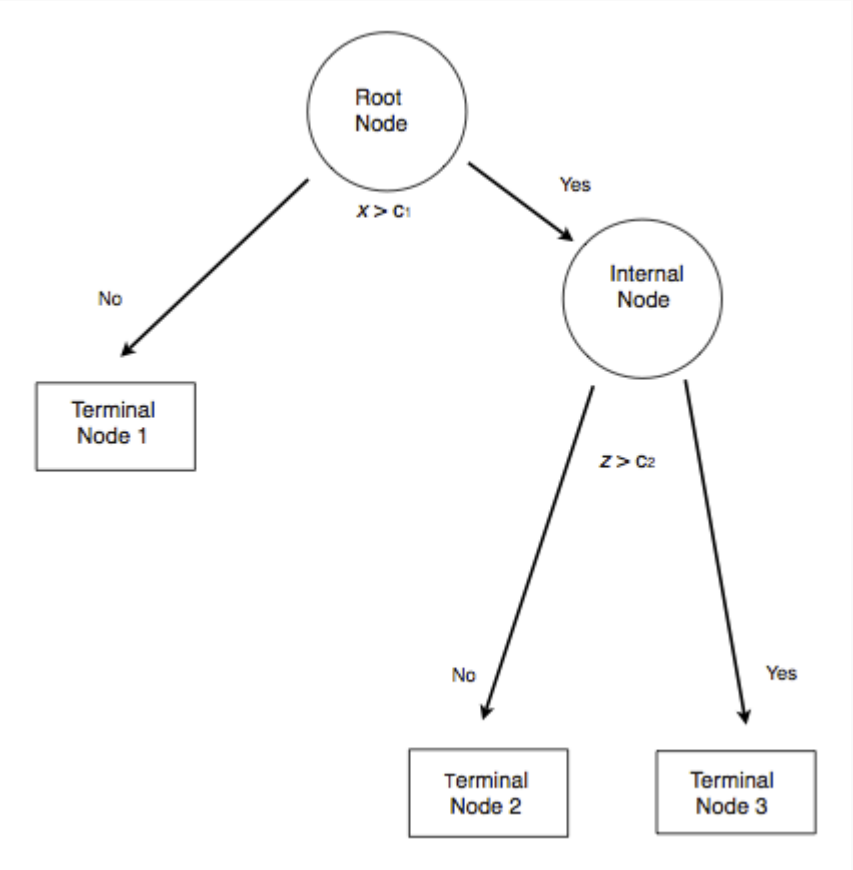
- Binary outcome, G, coded “A” or “B”.
- Two predictors, x and z
- The vertical line at z = 3 creates the first partition.
- The double horizontal line at x = -4 creates the second partition.
- The triple horizontal line at x = 6 creates the third partition.



Recursive Partitioning of a Binary Outcome
(where G = A or B and predictors are Z and X)

Tree Structure

- The root node contains the full data set.
- The data are split into two mutually exclusive pieces. Cases where x > ci go to the right, cases where x <= ci go to the left.
- Those that go to the left reach a terminal node.
- Those on the right are split into two mutually exclusive pieces. Cases where z > c2 go to the right and terminal node 3; cases where z <= c2 go to the left and terminal node 2.



Sum of Squared Errors

The sum of squared errors for a tree *T* is:

$$S = \sum_{c \in leaves(T)} \sum_{i \in c} (y - m_c)^2$$

Where, $m_c = \frac{1}{n} \sum_{i \in c} y_i$, the prediction for leaf \textit{c}.

Or, alternatively written as:

$$S = \sum_{c \in leaves(T)} n_c V_c$$

Where V_c is the within-leave variance of leaf \textit{c}.

Our goal then is to find splits that minimize S.

Advantages of CART Methods

- Making predictions is fast.
- It is easy to understand what variables are important in making predictions.
- Trees can be grown with data containing missingness. For rows where we cannot reach a leaf node, we can still make a prediction by averaging the leaves in the sub-tree we do reach.
- The resulting model will inherently include interaction effects. There are many reliable algorithms available.

Regression Trees

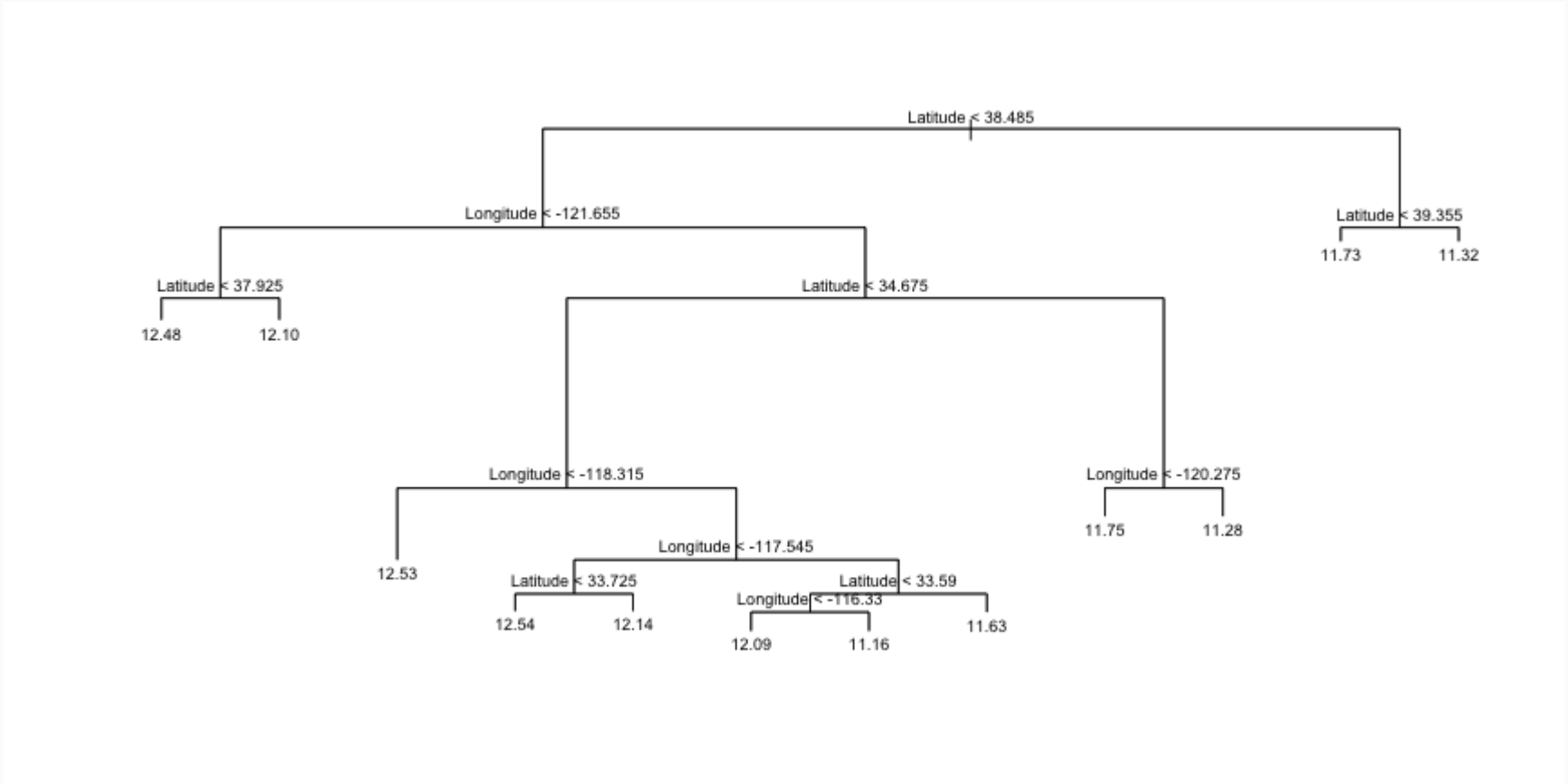
In this example we will predict the median California house price from the house's longitude and latitude.

```
str(calif)
```

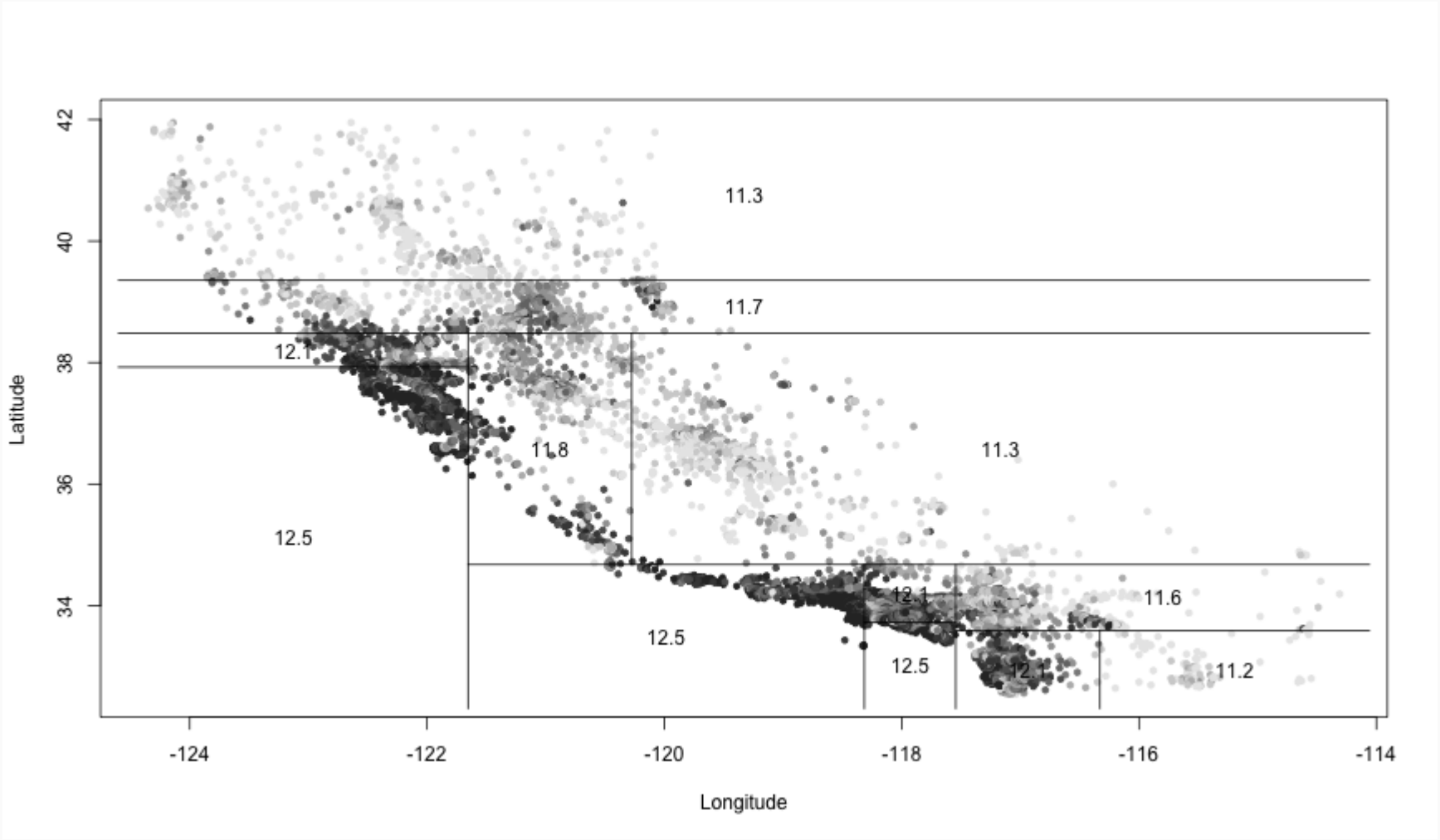
```
## 'data.frame':    20640 obs. of  10 variables:
##  $ MedianHouseValue: num  452600 358500 352100 341300 342200 ...
##  $ MedianIncome    : num  8.33 8.3 7.26 5.64 3.85 ...
##  $ MedianHouseAge  : num  41 21 52 52 52 52 52 52 42 52 ...
##  $ TotalRooms      : num  880 7099 1467 1274 1627 ...
##  $ TotalBedrooms   : num  129 1106 190 235 280 ...
##  $ Population      : num  322 2401 496 558 565 ...
##  $ Households      : num  126 1138 177 219 259 ...
##  $ Latitude        : num  37.9 37.9 37.9 37.9 37.9 ...
##  $ Longitude       : num  -122 -122 -122 -122 -122 ...
##  $ cut.prices      : Factor w/ 4 levels "[1.5e+04,1.2e+05]",..: 4 4 4 4 4 4 4 3 3 3 ...
```

Tree 1

```
treefit <- tree(log(MedianHouseValue) ~ Longitude + Latitude, data=calif)
plot(treefit); text(treefit, cex=0.75)
```



Tree 1


11.3
11.7
12.1
11.8
11.3
12.5
12.1
11.6
12.5
12.5
12.1
11.2
42
40
38
36
34
Latitude
-124
-122
-120
-118
-116
-114
Longitude

Tree 1

```
summary(treefit)
```

```
##
## Regression tree:
## tree(formula = log(MedianHouseValue) ~ Longitude + Latitude,
##     data = calif)
## Number of terminal nodes:  12
## Residual mean deviance:  0.1662 = 3429 / 20630
## Distribution of residuals:
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## -2.75900 -0.26080 -0.01359  0.00000  0.26310  1.84100
```

Here “deviance” is the mean squared error, or root-mean-square error of $\sqrt{.166} = 0.41$.

Tree 2, Reduce Minimum Deviance

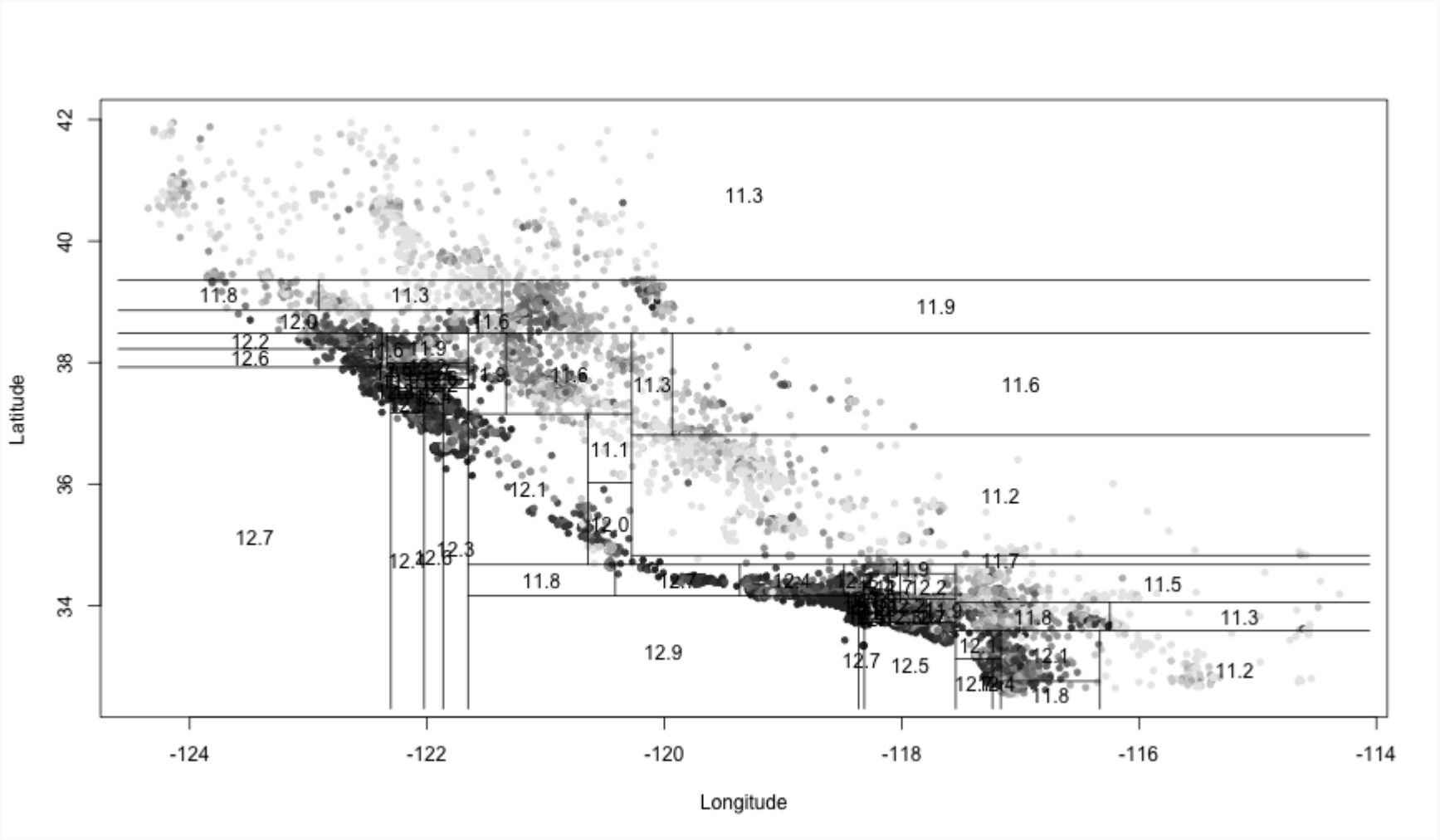
We can increase the fit but changing the stopping criteria with the mindev parameter.

```
treefit2 <- tree(log(MedianHouseValue) ~ Longitude + Latitude, data=calif, mindev=.001)
summary(treefit2)
```

```
##
## Regression tree:
## tree(formula = log(MedianHouseValue) ~ Longitude + Latitude,
##     data = calif, mindev = 0.001)
## Number of terminal nodes:  68
## Residual mean deviance:  0.1052 = 2164 / 20570
## Distribution of residuals:
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## -2.94700 -0.19790 -0.01872  0.00000  0.19970  1.60600
```

With the larger tree we now have a root-mean-square error of 0.32.

Tree 2, Reduce Minimum Deviance



Tree 3, Include All Variables

However, we can get a better fitting model by including the other variables.

```
treefit3 <- tree(log(MedianHouseValue) ~ ., data=calif)
summary(treefit3)
```

```
##
## Regression tree:
## tree(formula = log(MedianHouseValue) ~ ., data = calif)
## Variables actually used in tree construction:
## [1] "cut.prices"
## Number of terminal nodes:  4
## Residual mean deviance:  0.03608 = 744.5 / 20640
## Distribution of residuals:
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## -1.718000 -0.127300  0.009245  0.000000  0.130000  0.358600
```

With all the available variables, the root-mean-square error is 0.11.

Classification Trees

- `pclass` : Passenger class (1 = 1st; 2 = 2nd; 3 = 3rd)
- `survival` : A Boolean indicating whether the passenger survived or not (0 = No; 1 = Yes); this is our target
- `name` : A field rich in information as it contains title and family names
- `sex` : male/female
- `age` : Age, a significant portion of values are missing
- `sibsp` : Number of siblings/spouses aboard
- `parch` : Number of parents/children aboard
- `ticket` : Ticket number.
- `fare` : Passenger fare (British Pound).
- `cabin` : Does the location of the cabin influence chances of survival?
- `embarked` : Port of embarkation (C = Cherbourg; Q = Queenstown; S = Southampton)
- `boat` : Lifeboat, many missing values
- `body` : Body Identification Number
- `home.dest` : Home/destination

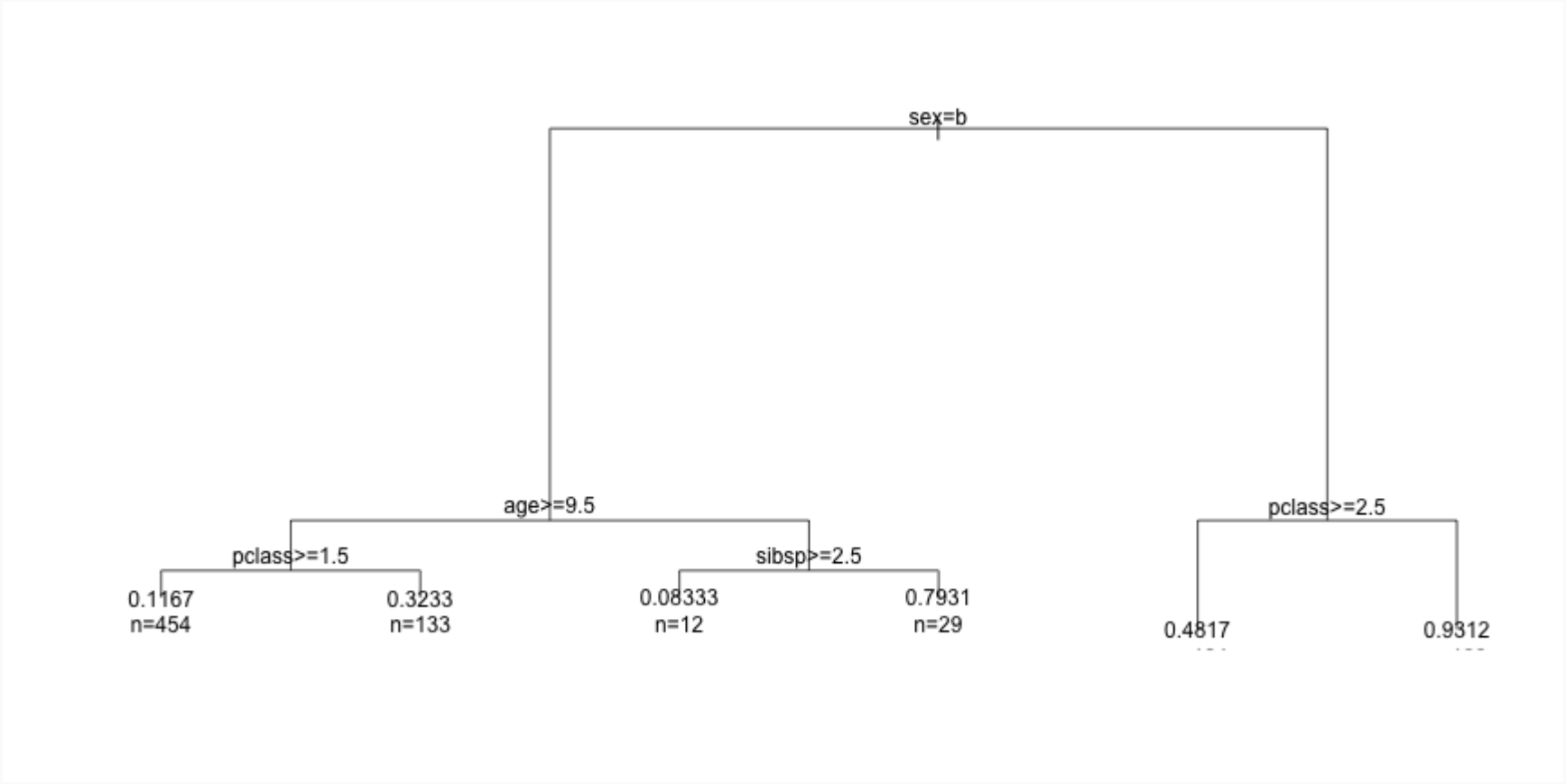
Classification using `rpart`

```
(titanic.rpart <- rpart(survived ~ pclass + sex + age + sibsp,
   data=titanic.train))
```

```
## n= 981
##
## node), split, n, deviance, yval
##       * denotes terminal node
##
##  1) root 981 231.6514000 0.38226300
##    2) sex=male 628  97.0700600 0.19108280
##      4) age>=9.5 587  80.2998300 0.16354340
##        8) pclass>=1.5 454  46.8127800 0.11674010 *
##        9) pclass< 1.5 133  29.0977400 0.32330830 *
##      5) age< 9.5 41   9.9512200 0.58536590
##       10) sibsp>=2.5 12   0.9166667 0.08333333 *
##       11) sibsp< 2.5 29   4.7586210 0.79310340 *
##    3) sex=female 353  70.7932000 0.72237960
##      6) pclass>=2.5 164  40.9451200 0.48170730 *
##      7) pclass< 2.5 189  12.1058200 0.93121690 *
```

Classification using `rpart`

```
plot(titanic.rpart); text(titanic.rpart, use.n=TRUE, cex=1)
```



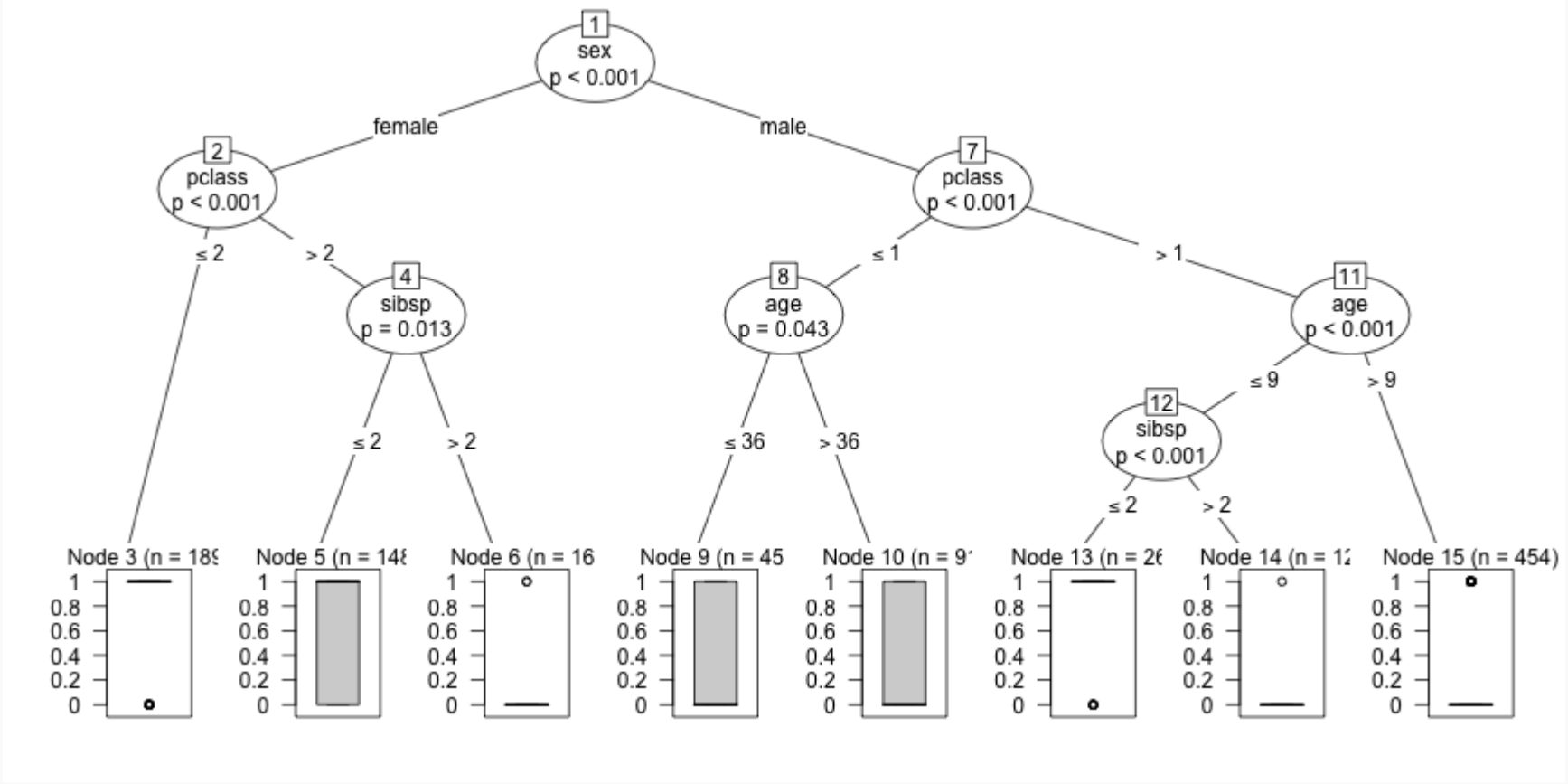
Classification using `ctree`

```
(titanic.ctree <- ctree(survived ~ pclass + sex + age + sibsp, data=titanic.train))
```

```
##
##      Conditional inference tree with 8 terminal nodes
##
## Response:  survived
## Inputs:  pclass, sex, age, sibsp
## Number of observations:  981
##
## 1) sex == {female}; criterion = 1, statistic = 269.855
##   2) pclass <= 2; criterion = 1, statistic = 79.362
##     3)*  weights = 189
##   2) pclass > 2
##     4) sibsp <= 2; criterion = 0.987, statistic = 8.635
##       5)*  weights = 148
##     4) sibsp > 2
##       6)*  weights = 16
## 1) sex == {male}
##   7) pclass <= 1; criterion = 1, statistic = 21.767
##     8) age <= 36; criterion = 0.957, statistic = 6.482
##       9)*  weights = 45
##     8) age > 36
##       10)*  weights = 91
```

Classification using `ctree`

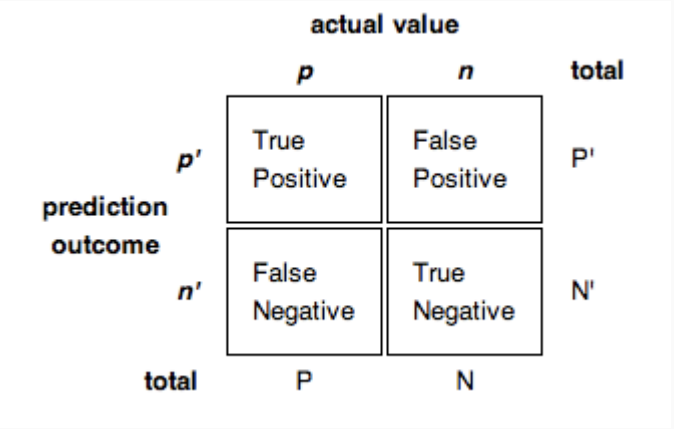
```
plot(titanic.ctree)
```



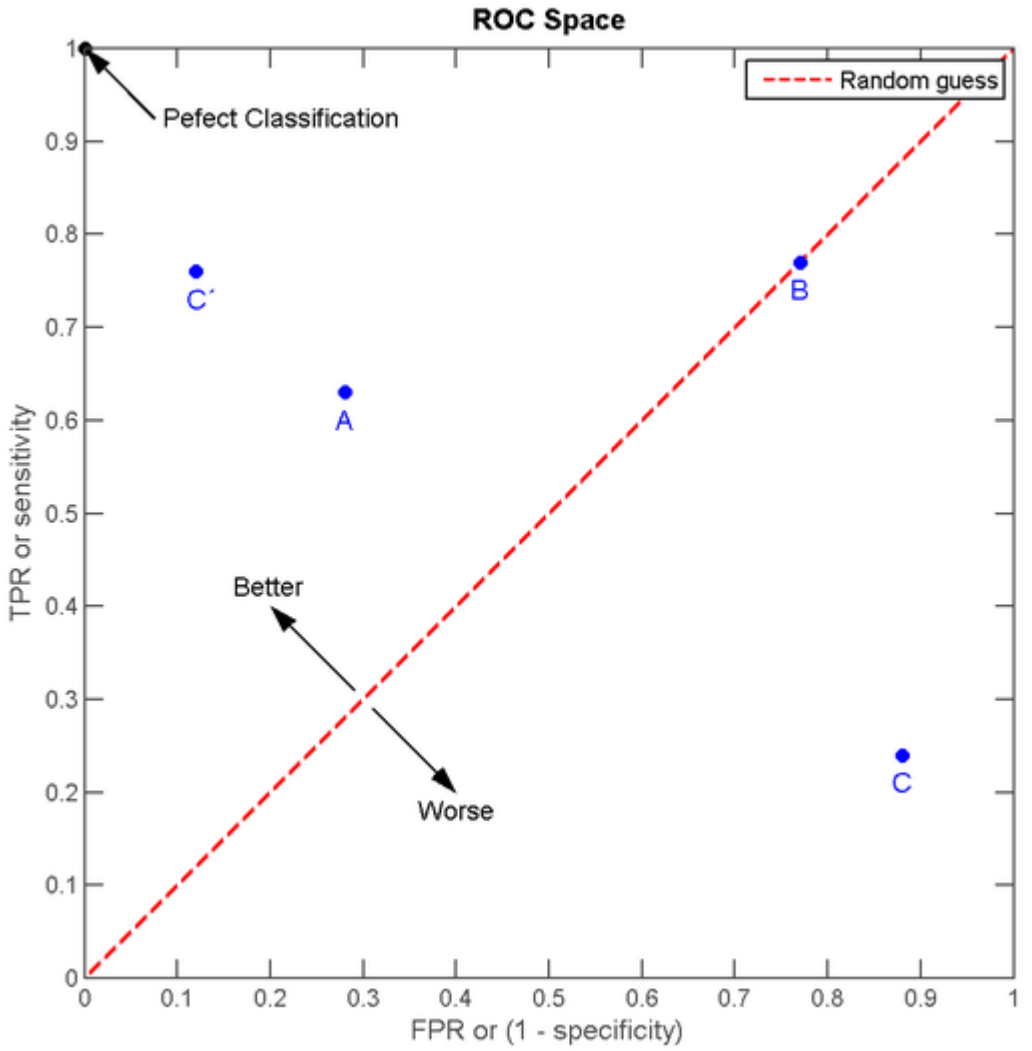
Receiver Operating Characteristic (ROC) Graphs

In a classification model, outcomes are either as positive (p) or negative (n). There are then four possible outcomes:

- **true positive** (TP) The outcome from a prediction is p and the actual value is also p.
- **false positive** (FP) The actual value is n.
- **true negative** (TN) Both the prediction outcome and the actual value are n.
- **false negative** (FN) The prediction outcome is n while the actual value is p.



ROC Curve


ROC Space
Random guess
Pefect Classification
C´
B
A
Better
Worse
C
TPR or sensitivity
FPR or (1 - specificity)
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1

Ensemble Methods

Ensemble methods use multiple models that are combined by weighting, or averaging, each individual model to provide an overall estimate. Each model is a random sample of the sample. Common ensemble methods include:

- *Boosting* - Each successive trees give extra weight to points incorrectly predicted by earlier trees. After all trees have been estimated, the prediction is determined by a weighted "vote" of all predictions (i.e. results of each individual tree model).
- *Bagging* - Each tree is estimated independent of other trees. A simple "majority vote" is take for the prediction.
- *Random Forests* - In addition to randomly sampling the data for each model, each split is selected from a random subset of all predictors.
- *Super Learner* - An ensemble of ensembles. See https://cran.r-project.org/web/packages/SuperLearner/vignettes/Guide-to-SuperLearner.html

Random Forests

The random forest algorithm works as follows:

1. Draw n_{tree} bootstrap samples from the original data.

2. For each bootstrap sample, grow an unpruned tree. At each node, randomly sample m_{try} predictors and choose the best split among those predictors selectedBagging is a special case of random forests where $m_{try} = p$ where p is the number of predictors.

3. Predict new data by aggregating the predictions of the ntree trees (majority votes for classification, average for regression).

Error rates are obtained as follows:

1. At each bootstrap iteration predict data not in the bootstrap sample (what Breiman calls “out-of-bag”, or OOB, data) using the tree grown with the bootstrap sample.

2. Aggregate the OOB predictions. On average, each data point would be out-of-bag 36% of the times, so aggregate these predictions. The calculated error rate is called the OOB estimate of the error rate.

Random Forests: Titanic

```
titanic.rf <- randomForest(factor(survived) ~ pclass + sex + age + sibsp,
                           data = titanic.train,
                           ntree = 5000,
                           importance = TRUE)
```

```
importance(titanic.rf)
```

```
##                0         1 MeanDecreaseAccuracy MeanDecreaseGini
## pclass  82.30573 118.42435            128.33974         45.60656
## sex    228.00111 316.68440            308.33992        120.58263
## age     65.59106  35.33482             79.41983         54.62580
## sibsp   75.13762 -20.40026             56.81944         16.04071
```

Random Forests: Titanic (cont.)

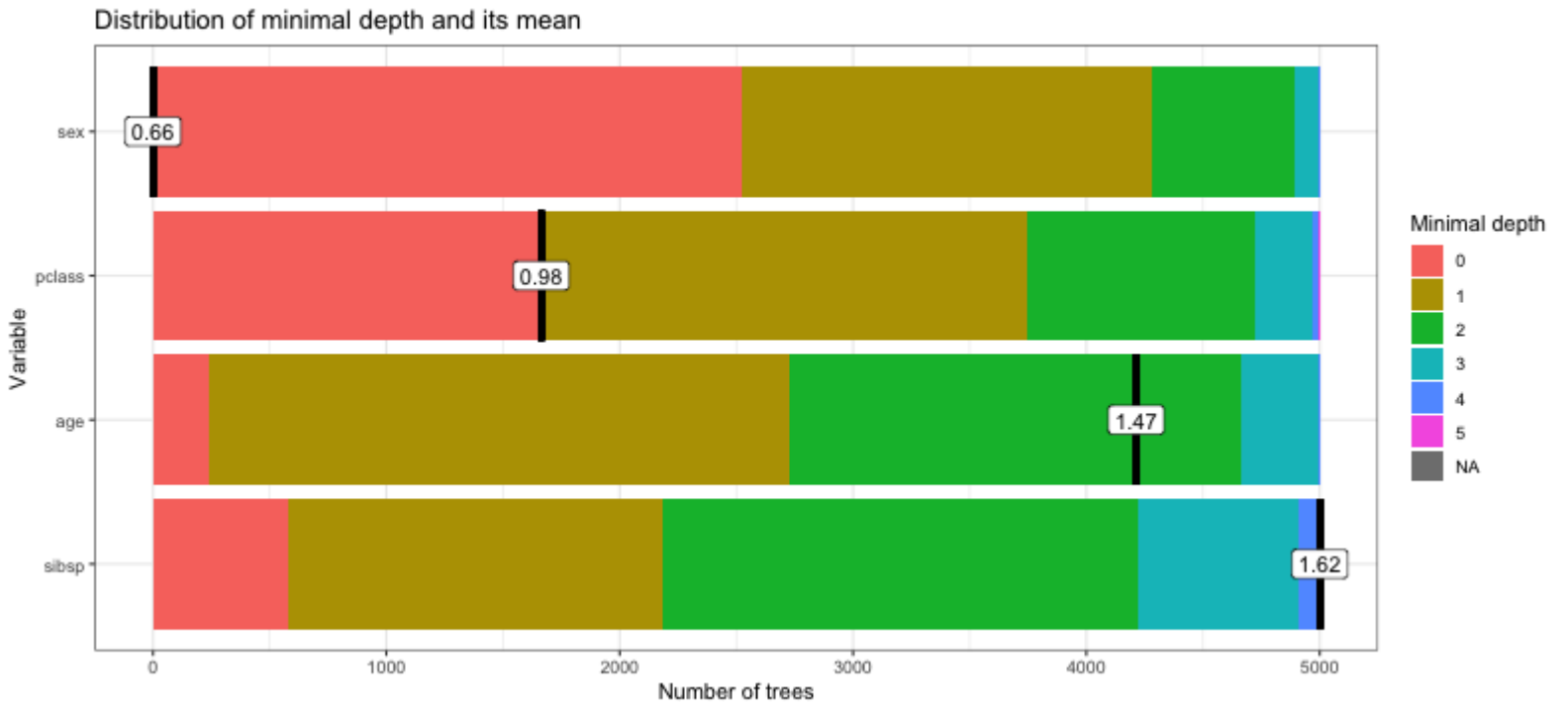
```
importance(titanic.rf)
```

```
##                0         1 MeanDecreaseAccuracy MeanDecreaseGini
## pclass  82.30573 118.42435             128.33974         45.60656
## sex    228.00111 316.68440             308.33992        120.58263
## age     65.59106  35.33482              79.41983         54.62580
## sibsp   75.13762 -20.40026              56.81944         16.04071
```

Random Forests: Titanic

```
min_depth_frame <- min_depth_distribution(titanic.rf)
```

```
plot_min_depth_distribution(min_depth_frame)
```



One Minute Paper

1. What was the most important thing you learned during this class?
2. What important question remains unanswered for you?

https://forms.gle/CD5Qxkq3xtdxSheW8