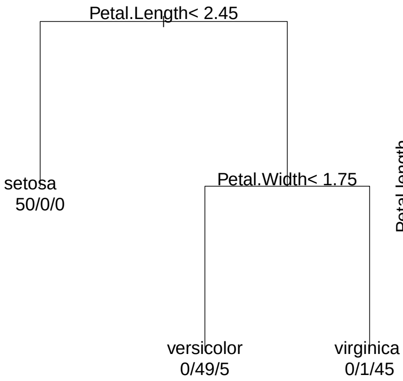
Classification and Regression Trees (CART)

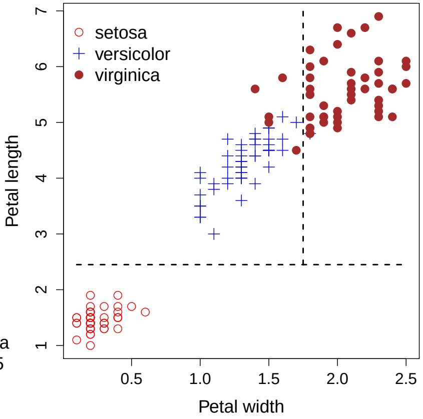
Bin Li

IIT Lecture Series

Classification tree in iris example

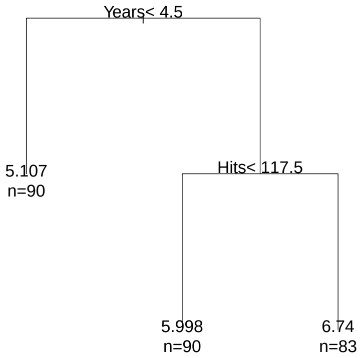
The famous iris data set gives the measurements in centimeters of the variables sepal length and width and petal length and width, respectively, for 50 flowers from each of 3 species of iris. The species are Iris *setosa*, *versicolor*, and *virginica*.

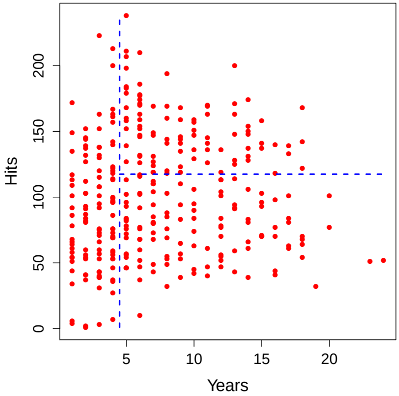




Regression tree in baseball players' salary example

- Data: 322 major league baseball players on 20 variables
- Response: `Salary` is measured in 1000's, and log-transformed to make it more bell-shaped.
- Variable: `Years` (number of years played in the major leagues) and `Hits` (the number of hits made in the previous year).

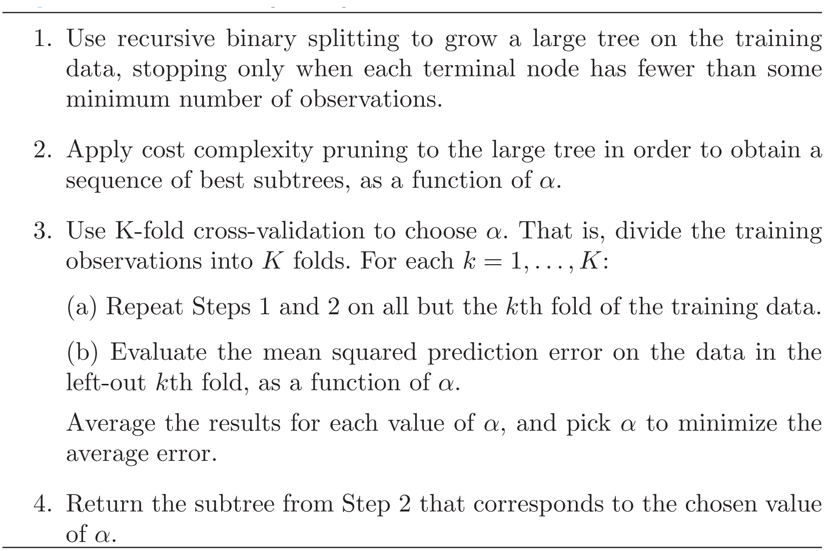




CART

- Some tree-related terms: root, leaf, internal node, tree size, tree depth.
- Tree partitions the feature space into a set of rectangles, and fit a simple model in each leaf (terminal node).
  - a constant in regression
  - a class in classification
- A tree is constructed in two steps:
  - growing: binary split on each region repeatedly.
  - pruning: weakest link pruning (collapse internal node).
- A key advantage of the CART is its interpretability. The feature space partition is fully described by a single tree (a nice graphical representation).
- `CART` is emplemented in `rpart` library in `R`.
- A major competitor of `CART` is `C5.0`

Algorithm of building a regression tree



1. Use recursive binary splitting to grow a large tree on the training data, stopping only when each terminal node has fewer than some minimum number of observations.

2. Apply cost complexity pruning to the large tree in order to obtain a sequence of best subtrees, as a function of α.

3. Use K-fold cross-validation to choose α. That is, divide the training observations into K folds. For each $k = 1, \ldots, K$:

   (a) Repeat Steps 1 and 2 on all but the kth fold of the training data.

   (b) Evaluate the mean squared prediction error on the data in the left-out kth fold, as a function of α.

   Average the results for each value of α, and pick α to minimize the average error.

4. Return the subtree from Step 2 that corresponds to the chosen value of α.

Figure from ISLR 2013.

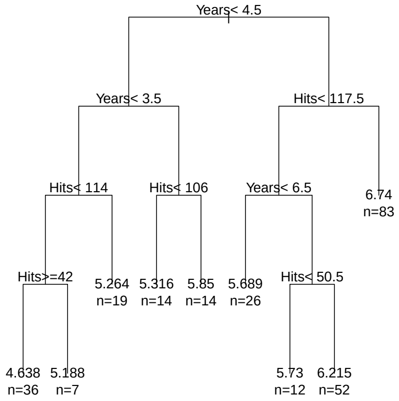
How to grow a regression tree in Step 1?

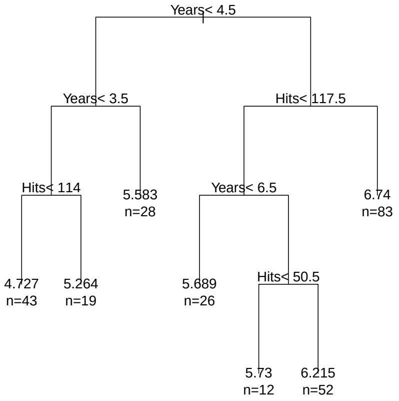
- How to split? Binary split (multiway splits fragment the data too fast, leaving insufficient data at next level down).
- Where to split? Find the variable j and splitting point s to minimize

$$\sum_{x_i \in R_1(j,s)} (y_i - \hat{c}_1)^2 + \sum_{x_i \in R_2(j,s)} (y_i - \hat{c}_2)^2$$

- What constant should be used for each region? Use the average of y_i in the region R_m to minimize SSE.
- Repeat the splitting process on each of the two resulted regions till some stopping criterion is met.
- We can control the tree structure by the following options in `rpart.control` in **R**.
  - `minsplit`: the minimum size for a node to split.
  - `minbucket`: the minimum size for a terminal node (leaf size).
  - `maxdepth`: maximum depth of any node of the final tree, with the root node counted as depth 0.

Baseball player example





```
fit1 <- rpart(log(Salary) ~ Hits + Years, data=Hitters)
fit2 <- rpart(log(Salary) ~ Hits + Years, data=Hitters,
              control = rpart.control(minbucket=10, minsplit=30))
```

How to prune a regression tree in Step 2?

- A very large tree often overfit the data, while a small tree might not capture the important structure.
- Tree *size* $|T|$ is a tuning parameter governing the model complexity and should be adaptively chosen from the data.
- Cost complexity criterion:

$$C_\alpha(T) = \sum_{m=1}^{|T|} \left[\sum_{x_i \in R_m} (y_i - \hat{c}_m)^2 \right] + \alpha |T|$$

- For each α, find a *subtree* T_α (collapse any number of internal nodes) to minimize $C_\alpha(T)$
  - large (small) α result in small (large) trees.
- Estimate the value of α through cross-validation.

Revisit baseball player example

- First, randomly divided the data set in half, yielding 132/131 obs. in training/test sets.
- Built a large regression tree on the training data using nine of the features.
- Varied α in order to create subtrees with different size of trees.
- Performed six-fold cross-validation in order to estimate the cross-validated MSE of the trees as a function of α.
- We can see the CV error is a reasonable approximation of the test error (on slide 12).
- The unpruned regression tree is shown on next slide.
- The optimal tree that minimizes both CV and test errors is shown on Slide 3 with only three leafs.

Unpruned tree in baseball player example

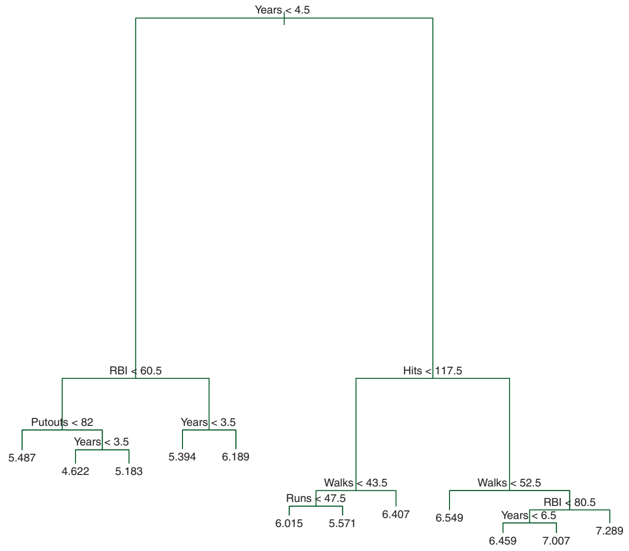


Figure from ISLR 2013.

CV errors vs. tree size

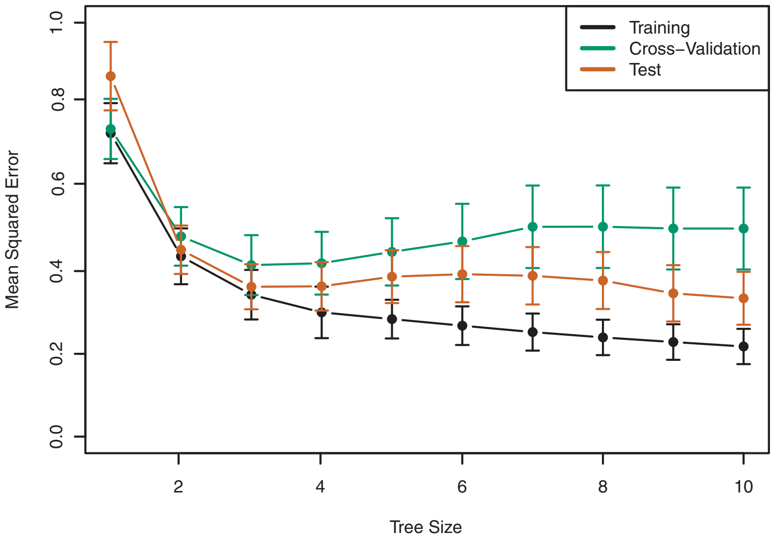


Figure from ISLR 2013.

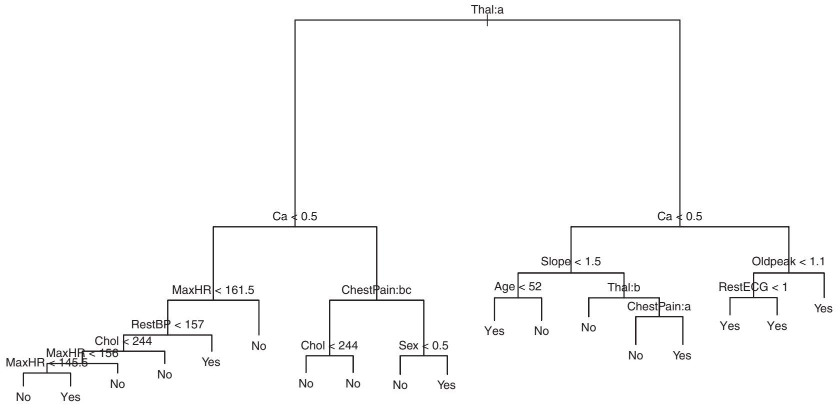
Classification tree

- Response is a categorical variable with $K \geq 2$ classes.
- The only changes in the tree algorithm is the criteria for splitting nodes and pruning the tree
- CART uses the majority class (of a node) to classify the new observation falls into this node.
- Three commonly used measures of node impurity in classification trees: misclassification error, Gini index $(\sum_k p(1-p))$ and cross-entropy $(-\sum_k p \log p)$.
- For two classes, three measures are $1 - \max(p, 1-p)$, $2p(1-p)$ and $-p \log p - (1-p)\log(1-p)$
- Gini and entropy are differentiable and easier to optimize. They are used to grow trees.
- Misclassification rate is used to guide pruning.

Heart disease example

- Data consist of 303 patients who presented with chest pain.
- A binary outcome with `Yes` (the presence of heart disease based on an angiographic test) and `No` (no heart disease).
- There are 13 predictors including `Age`, `Sex`, `Chol` (a cholesterol measurement), and other heart and lung function measurements. Three of them are qualitative: `Sex`, `Thal` (Thalium stress test) and `ChestPain`. Rest are all numeric.
- An unpruned tree is shown on next slide.
- Cross-validation results in a tree with six terminal nodes (see slide 17).

Unpruned tree



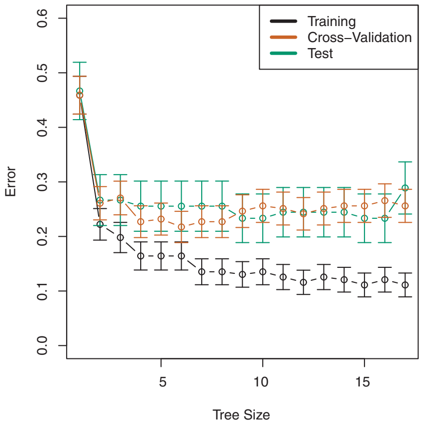
`Thal:a` indicates the the 1st value of the `Thal` variable (i.e. normal vs. fixed or reversible defects).
`ChestPain:bc` indicates the 2nd and 3rd values of the `ChestPain` variable. Possible values are typical angina, atypical angina, non-anginal pain, and asymptomatic.

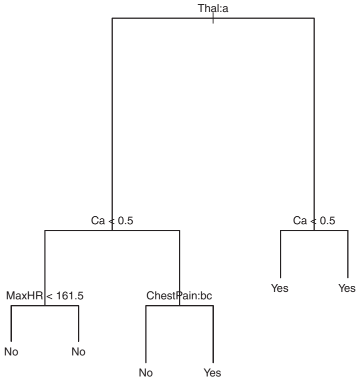
Figure from ISLR 2013.

A surprising phenomenon

- Some splits yield two leafs with the same predicted value (e.g. the `RestECG<1` split near the bottom right of the unpruned tree.)
- The split is performed to *increased node purity*. All 9 patients corresponding to the right-hand leaf have a response value of `Yes`. 7 out of 11 patients corresponding to the left-hand leaf have a response value of `Yes`.
- Does node purity matter? For a new patient belongs to the right-hand leaf region, we can be pretty certain the response value is `Yes`. In contrast, if a patient belongs to the region given by the left-hand leaf, then the response value is probably `Yes`, but we are much less certain.
- Although the split `RestECG<1` does not reduce the classification error, it improves the Gini index and the cross-entropy, which are more sensitive to node purity.

CV plot and pruned tree





Left: CV, training, and test errors, for different sizes of the pruned tree.
Right: The pruned tree corresponding to the minimal CV error.
Figure from ISLR 2013.

Trees vs. linear models

- CART has a very different flavor from the classical linear models for regression and classification.
  - Linear regression assumes a model of the form $f(x) = \beta_0 + \sum_j \beta_j x_j$
  - Regression trees assume a model of the form $f(x) = \sum_k c_k \mathbf{1}_{x \in R_k}$
- Which is better? It depends on the problem at hand.
  - If the relationship between the predictors and the response is approximately linear and additive, linear model or GAM wins.
  - If instead there is a highly non-linear and complex relationship between X's and Y, tree may win.

Trees vs. linear models

- Top row: the true decision boundary is linear. A linear boundary (left) outperforms a decision tree (right) that performs splits parallel to the axes.
- Bottom row: the true decision boundary is nonlinear. A linear model is unable to capture the true decision boundary (left), whereas a decision tree (right) is successful.

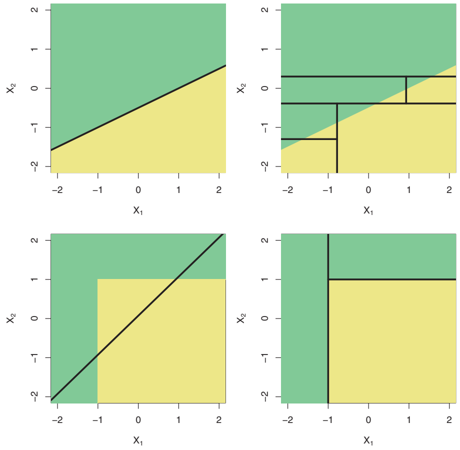
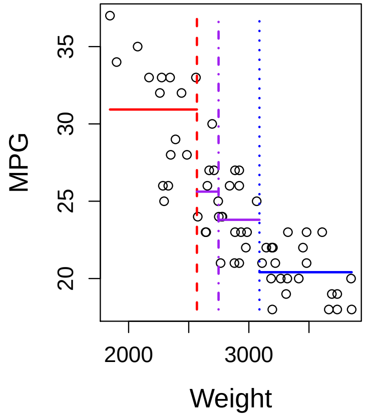
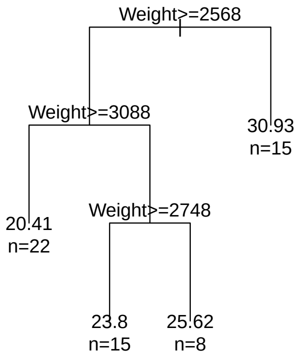


Figure from ISLR 2013.

Regression tree in automobile example

- Data: 60 makes of cars based on April, 1990 issue of Consumer Reports
- Response: mileage per gallon
- Variable: weight





Automobile example (cont.)

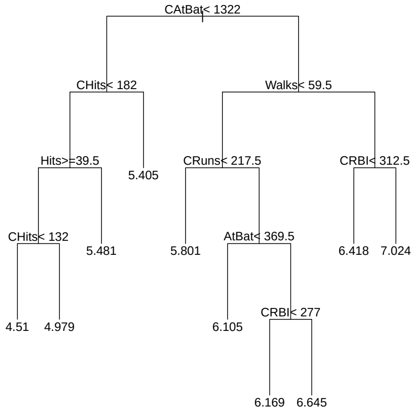
```
> attach(car.test.frame)
> fit1 <- rpart(Mileage ~ Weight, car.test.frame)
> pred1 <- predict(fit1,car.test.frame)
> 1-sum((pred1 - Mileage)^2)/sum((Mileage-mean(Mileage))^2)
[1] 0.7427057
>
> fit.1 <- lm(Mileage ~ Weight, car.test.frame)
> summary(fit.1)

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept) 48.3493466  1.9794137   24.43   <2e-16 ***
Weight      -0.0081928  0.0006728  -12.18   <2e-16 ***
---
Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 0.1   1

Residual standard error: 2.562 on 58 degrees of freedom
Multiple R-squared: 0.7189,     Adjusted R-squared: 0.714
F-statistic: 148.3 on 1 and 58 DF,  p-value: < 2.2e-16
```

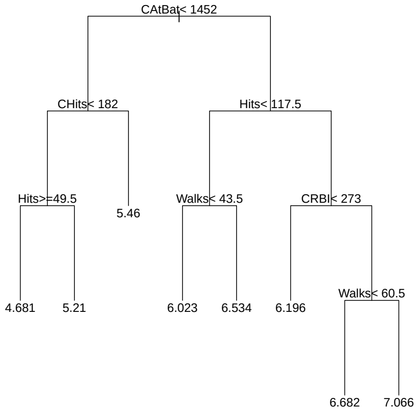
Issues about CART

- When splitting a predictor having q unordered values, there are $2^{q-1} - 1$ possible partitions into two groups. CART tends to favor categorical predictors with many levels (large q) and lead to severe overfitting. Such variables should be avoided or collapsed to fewer levels.
- CART has two ways to handle missing values in predictor variables.
  - Categorical predictors: add a new category for "missing"
  - Surrogate variables: for example, assume that the split ('age$<$40', 'age$\geq$40') has been chosen. The surrogate variables are found by reapplying the partitioning algorithm (without recursion) to predict the two categories 'age$<$40' vs. 'age$\geq$40' using the other predictor variables.
- Trees have high variance. Often a small change in the data can result in a very different tree. This instability is due to the hierarchical nature of the process.

Revisit baseball player example

- Both trees are generated using about 90% (290 obs.) of the whole data.
- We can see the tree structure, splitting variables and splitting points differ in two trees.



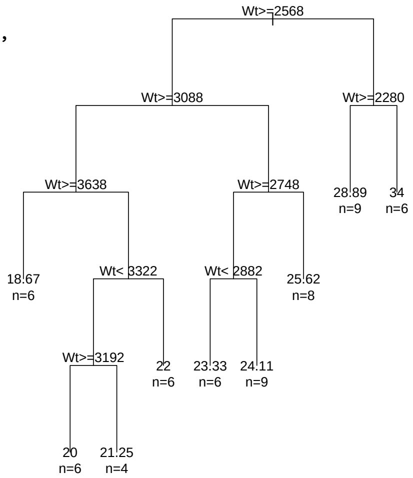


Pros and Cons of trees

- Very easy to explain (IF/AND/THEN) to people (even easier to explain than linear models!)
- Some believe that decision trees mirror human decision-making.
- Trees can be displayed graphically, and are easily interpreted even by a non-expert (especially if they are small).
- Easily handle qualitative predictors without the need to create dummy variables.
- Can naturally handle the missing values in the predictors.
- Trees generally do not have the same level of predictive accuracy as some of the other regression and classification approaches.

Revisit the automobile example in R

```
> set.seed(1)
> fit1<-rpart(Mileage ~ Wt, car.test.frame,
+        control = rpart.control(xval=10,
+        minbucket=4, minsplit=10, cp=0))
> fit1
 1) root 60 1354.58 24.58
   2) Wt>=2567.5 45  361.20 22.47
     4) Wt>=3087.5 22   61.32 20.41
       8) Wt>=3637.5 6    3.33 18.67 *
       9) Wt< 3637.5 16   32.94 21.06
        18) Wt< 3322.5 10   16.50 20.50
          36) Wt>=3192.5 6   10.00 20.00 *
          37) Wt< 3192.5 4    2.75 21.25 *
        19) Wt>=3322.5 6    8.00 22.00 *
     5) Wt< 3087.5 23  117.65 24.43
      10) Wt>=2747.5 15   60.40 23.80
        20) Wt< 2882.5 6   19.33 23.33 *
        21) Wt>=2882.5 9   38.89 24.11 *
      11) Wt< 2747.5 8   39.87 25.62 *
   3) Wt< 2567.5 15  186.93 30.93
     6) Wt>=2280 9   76.88 28.88 *
     7) Wt< 2280 6   16.00 34.00 *
```



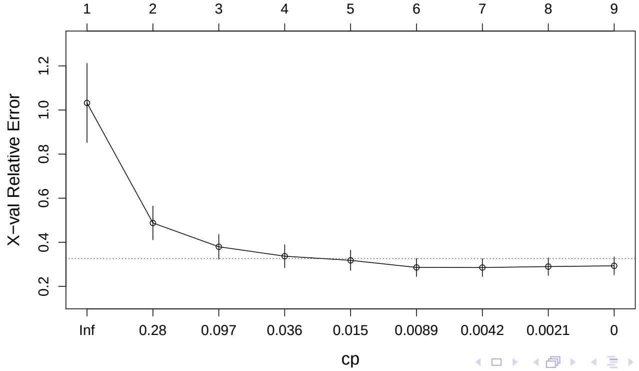
cptable output in R

- We can output the cross-validation results for a grown tree with different cp values.
- The last two columns are important.
  - `xerror`: cross-validation error (10-fold CV in automobile example).
  - `xstd`: standard error of validation errors.

```
> print(fit1$cptable)
           CP nsplit rel error    xerror       xstd
1 0.595349123      0 1.0000000 1.0319924 0.17931937
2 0.134528190      1 0.4046509 0.4875114 0.07613003
3 0.069426843      2 0.2701227 0.3795295 0.05610012
4 0.018490814      3 0.2006958 0.3367415 0.05225040
5 0.012828427      4 0.1822050 0.3179277 0.04552389
6 0.006228853      5 0.1693766 0.2857965 0.04105127
7 0.002768379      6 0.1631477 0.2852686 0.04046187
8 0.001607710      7 0.1603794 0.2895803 0.03988263
9 0.000000000      8 0.1587717 0.2930557 0.04001814
```

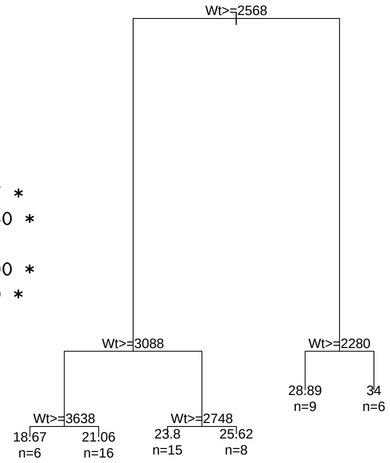
Select optimal cp value

- Ideally, we should choose the cp value to minimize CV error. In practice, we may use instead the 1-SE rule. Any risk within 1-SE of the achieved minimum is considered as being equivalent to the minimum (i.e. considered to be part of the at plateau).
- In automible example: the best tree has 4 splits (5 leaves) using 1-SE rule (0.2853+0.0405=0.3258).



Pruned tree in automobile example

```
> fit2 <- prune(fit1, cp=.0128)
> fit2
n= 60
node), split, n, deviance, yval
      * denotes terminal node
 1) root 60 1354.583000 24.58333
   2) Wt>=2567.5 45  361.200000 22.46667
     4) Wt>=3087.5 22   61.318180 20.40909
       8) Wt>=3637.5 6     3.333333 18.66667 *
       9) Wt< 3637.5 16   32.937500 21.06250 *
     5) Wt< 3087.5 23  117.652200 24.43478
      10) Wt>=2747.5 15   60.400000 23.80000 *
      11) Wt< 2747.5 8   39.875000 25.62500 *
   3) Wt< 2567.5 15  186.933300 30.93333
     6) Wt>=2280 9   76.888890 28.88889 *
     7) Wt< 2280 6   16.000000 34.00000 *
> par(mar=c(1,1,1,1),xpd=NA)
> plot(fit2,uniform=F)
> text(fit2, use.n=TRUE)
```



Note: The depth of the branches is proportional to the reduction in error due to the split.

Reference

- Classification and Regression Trees by Leo Breiman, Jerome Friedman, Charles J. Stone and R.A. Olshen.
- An Introduction to Statistical Learning with Applications in `R` by Gareth James, Daniela Witten, Trevor Hastie and Robert Tibshirani.
- The Elements of Statistical Learning: Data Mining, Inference, and Prediction (2nd Ed.) by Trevor Hastie, Robert Tibshirani, and Jerome Friedman.