Linear Regression & Regression Tree Demos

Data Mining

Prof. Dawn Woodard
School of ORIE
Cornell University

Outline

1. Toyota Corollas Data
2. Toyota Corollas Analysis
3. Regression Trees on the CPUs Data

Toyota Corollas Data

- Download the ToyotaCorollas.xls file from Blackboard.

- This dataset is from www.dataminingbook.com, and is analyzed in Chapter 6 of SPB.

- We will improve on the analysis from the book.

Toyota Corollas Data

Motivation:

- When a dealership takes a trade-in vehicle from purchasers of new cars, it wants to know roughly the amount for which it will be able to resell the trade-in.
- The goal is to predict sale amount, using the variables known to the dealership like age and mileage of the vehicle.

Toyota Corollas Data

Variables of interest:

- Price: outcome
- Age_08_04: age as of August 2004
- KM: kilometers on odometer
- Fuel_Type: “Petrol”, “Diesel”, or “CNC”
- HP: horsepower
- Met_Color: metallic color? (binary)
- Doors: number of doors
- Quarterly_Tax
- Weight

Toyota Corollas Analysis

- The dataset will need to be in .csv (comma-separated values) text format before R can read it in.
- Open the Excel file, and navigate to the worksheet that has the data (the other worksheet has the metadata).
- save as a .csv file by going to File->Save & Send->Change file type->CSV and clicking “Save as”.
- It will give a warning and ask if you only want to save the current sheet; click OK.

Toyota Corollas Analysis

- Read the file into R:

  corollas = read.table(file = “C:/temp/ToyotaCorolla.csv”, sep = “,”, header = T)

- “header = T” tells R that the first line of the csv file contains the variable names

- Check the number of rows of the dataset. There are _____ rows in the dataset in R, but only 1436 cars in the original Excel file! When Excel exported the data as a .csv file, it added empty rows at the end. Remove these rows:

  corollas2 = corollas[1:1436,]

Toyota Corollas Analysis

- Check for missing values.

- Check whether the categorical predictors have been read in correctly as “factor” type:

  ```
  is.factor( corollas2[,”Fuel_Type”] )
  ```

  Or equivalently,

  ```
  is.factor( corollas2$Fuel_Type )
  ```

- Does Fuel_Type take the same 3 values indicated in the metadata?

  A. Yes

  B. No

Toyota Corollas Analysis

- Check whether the other categorical predictor (“Met_Color”) has been read in as “factor” type, and if not convert it to factor type.

- Choose the variables to be used in our analysis:

  ```
  corollas3 = corollas2[, c(“Price”, “Age_08_04”, “KM”, “Fuel_Type”, “HP”,
  “Met_Color”, “Doors”, “Quarterly_Tax”, “Weight”) ]
  ```

- Split into train & test datasets (code also on Blackboard):

  ```
  nData = nrow( corollas3 )
  nTrain = floor( .6 * nData )
  trainInd = sample( (1:nData), nTrain )
  trainData = corollas3[ trainInd, ]
  testData = corollas3[ -trainInd, ]
  ```

Toyota Corollas Analysis

- Fit the linear regression model. This gives you roughly the results from SPB (not exactly because we drew the test & train datasets randomly):

  ```
  corollasLM = lm( formula = Price ~ ., data = trainData )
  summary(corollasLM)
  ```

- Predict the price of the cars in the test dataset:

  ```
  preds = predict.lm( object = corollasLM, newdata = testData )
  hist(preds)
  ```

Toyota Corollas Analysis

- Calculate the “residuals” (difference between the actual and predicted values):

```
resids = testData$Price - preds
hist(resids)
```

- How far off were we “on average”? The most common measure is the “Root Mean Squared Error” (RMSE):

```
rmse = sqrt( mean( resids∧2 ) )
```

A. 0-1600
B. 1601-3200
C. 3201-4800
D. 4801-6400

Toyota Corollas Analysis

- Actually there is an issue with this analysis. The assumptions of the model don't really hold. Create pairwise scatterplots of the continuous variables in the dataset to see this:

  ```
  pairs( trainData[, c("Price", "Age_08_04", "KM", "HP", "Doors",
  "Quarterly_Tax", "Weight") ] )
  ```

- Transform several variables so that the assumptions of the linear model are more reasonable. Don't forget to transform the predictor variables in the test data also (code also on Blackboard). Recheck the pairwise scatterplots!

  ```
  trainData2 = trainData
  trainData2$Price = log( trainData$Price )
  trainData2$KM = sqrt( trainData$KM )
  trainData2$Weight = log( trainData$Weight - 950 )
  testData2 = testData
  testData2$KM = sqrt( testData$KM )
  testData2$Weight = log( testData$Weight - 950 )
  ```

Toyota Corollas Analysis

- Now fit the model to trainData2, and test on testData2. Did the error rate for the test data decrease?

  ```
  corollasLM = lm( formula = Price ∼ ., data = trainData2 )
  preds = predict.lm( object = corollasLM, newdata = testData2 )
  resids = testData2$Price - exp( preds )
  rmse = sqrt( mean( resids∧2 ) )
  ```

Regression Trees

- Install the “tree” package in R (packages->install package->choose a U.S. mirror->tree). Then load by calling “library(tree)”
- We will use the “cpus” data in the MASS package.
  - don’t have to install MASS. Just call
  - now can view & use:
    cpus[1:10,]

Regression Trees

This dataset gives the performance of (computer) CPUs on a benchmark mix, along with specifications like the cache size.

The goal is to predict the performance using these design factors. This could be used to:

Relevant variables:

- perf: the measured performance
- syct: the cycle time in nanoseconds
- mmin: minimum main memory in kilobytes
- mmax: maximum main memory in kilobytes
- cach: cache size in kilobytes
- chmin: minimum number of channels
- chmax: maximum number of channels

Regression Trees

- Create a histogram of “perf”; what do you notice?

- This suggests that we should:

- create a scatterplot with log(perf) on the y-axis and syct on the x-axis. Same for log(perf) and mmin. What do you notice?

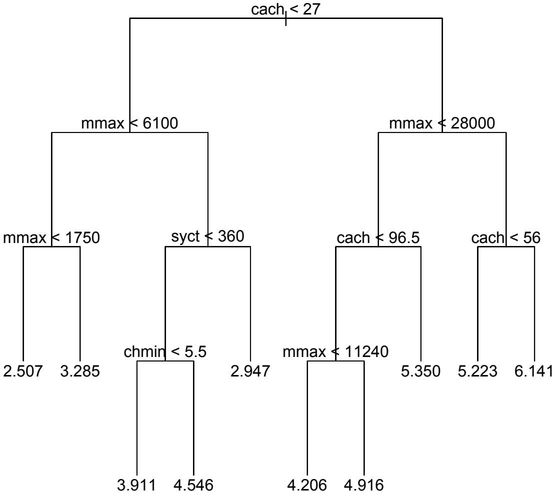
Regression Trees

- Fit the regression tree:
  cpus.tree = tree(formula = log(perf) $\sim$ syct + mmin + mmax + cach + chmin + chmax, data = cpus)

- Plot the tree:

Regression Trees

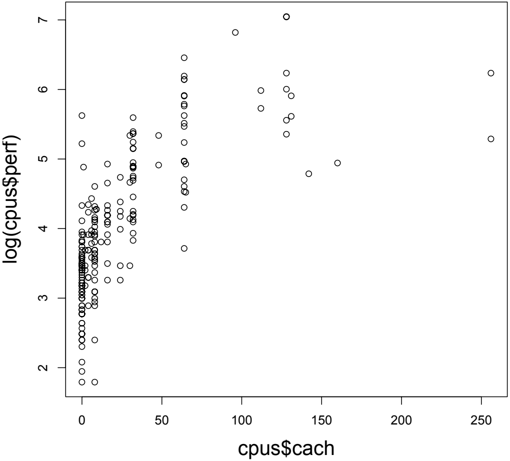
Here the left branch means “yes”



Regression Trees

Why did the “tree” function decide to split first on [cach < 27]?

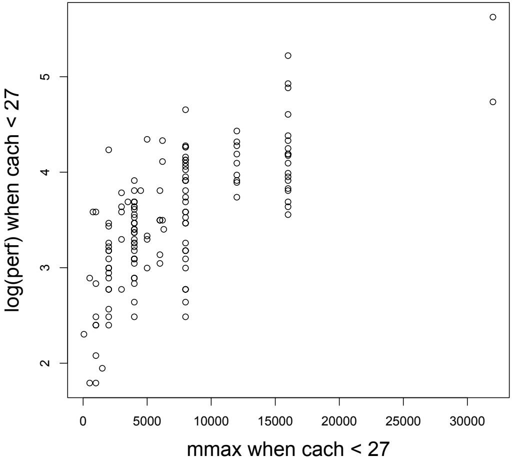
plot(cpus$cach, log(cpus$perf))



Regression Trees

Why did it split next on [mmax < 6100] if cach < 27?

plot(cpus$mmax[cpus$cach < 27], log(cpus$perf[cpus$cach < 27]))



Regression Trees

- So we see that regression trees handle nonlinearities and can be fit in an automated fashion (they provide a good “black box” method)

- Predict the value of perf if cach = 40, mmax = 20000, syct = 300, mmin = 3000, chmin = 8, chmax = 50.

  A. 0-2
  B. 2-4
  C. 4-7
  D. 7-20
  E. 20+

Regression Trees

Check your answer using the “predict” function: