Demo: CVE vs Lambda

Zane Billings

25 February, 2020

This document will have a big test of the relationship between lambda and CVE.

```r
library(here)
```

```
## here() starts at /Users/zanebillings/Stuff/Research/Lawson_380/Lawson_380_R_code
```

```r
# Setup and sourcing functions
set.seed(300)
source(here::here("Scripts","Dimensionless_exploration.R"))
```

```
## Loading required package: MASS
```

```r
source(here::here("Scripts", "Helpers.R"))
source(here::here("Scripts", "Least_squares_methods.R"))
```

```r
# Build functions needed that are not in a script yet
random_partition <- function(data, k) {
  # data is a data frame of observations
  # k is the number of partitions to form
  # Returns a list of k random partitions of the data
  num_obs <- nrow(data)
  part_size <- floor(num_obs/k)
  rem <- seq(1, num_obs, 1)
  parts <- list()
  for (i in 1:(k - 1)) {
    choices <- sample(rem, part_size, replace = F)
    parts[[i]] <- data[choices, ]
    rem <- setdiff(rem, choices)
  }
  parts[[k]] <- data[rem, ]
  return(parts)
}
calculate_SSE <- function(true, fits) {
  # Calculate the sum of squared errors for a model.
  SSE <- sum((true - fits)^2)
  return(SSE)
}
```

Setup stuff for cross validation.

```r
# Setup for CV
df <- generate_dimensionless_logistic_data(.1, 0.1, 50, 0.1, FALSE)
lambdas <- seq(from = 0.01, to = 10000, by = 0.01)
k <- 3
parts <- random_partition(df, k)
```

```r
combos <- combn(1:k, k-1, simplify = F)
cves <- numeric(length(lambdas))
```

Actually do the 3-fold CV (UPDATE: this is implemented more elegantly in the HMBGR package now.)

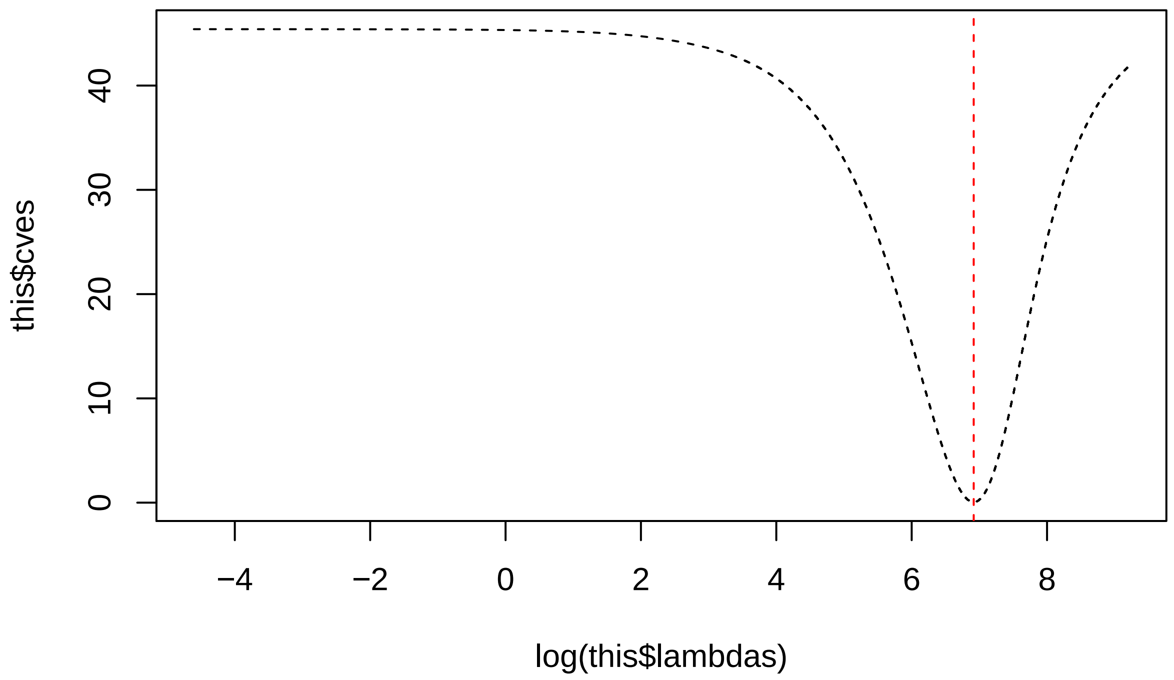
```r
# Actually do CVE
for (a in 1:length(lambdas)) {
  l = lambdas[[a]]
  errors <- numeric(length(parts))
  # Need to do this k number of times (# of parts)
  for (i in 1:k) {
    # Grab the first combo of indices to use
    to_use <- combos[[i]]
    # Make a blank dataframe
    train <- data.frame()
    # This part has to be done k-1 times to build training data.
    for (j in to_use) {
      # Get the next data frame.
      new_dat <- parts[[j]]
      # Add this data frame to the bottom of the rest.
      train <- rbind(train, new_dat)
    }
    # Now train has k-1 of the parts in it.
    # Set test to be the part that wasn't selected.
    not_used <- (1:k)[!(1:k %in% to_use)]
    test <- parts[[not_used]]
    # Prep and model the data, specify time step manually.
    prepped <- prep_data(train, 0.1)
    mod <- model_logistic_data_dimensionless_smoothing(prepped, l)
    # Next need to calculate predicted values using the estimated growth rate from the model.
    #P = (P0e^(rt))/(1+P0(1-e^(rt)))
    t <- test$t
    pred <- (0.1*exp(mod*t))/(1+0.1*(exp(mod*t)-1))
    errors[i] <- calculate_SSE(test$P,pred)
  }

  cves[a] <- mean(errors)
}
```

And let's save and look at the data.

```r
# Get data
this <- data.frame(lambdas,cves)
write.csv(this, "Demo_cve_data.csv")
this <- read.csv("Demo_cve_data.csv")
plot(x = log(this$lambdas), y = this$cves, type = "l", lty = 2)

abline(v = log(this$lambdas[which(this$cves == min(this$cves))]), lty = 2, col = "red")
```


this$cves
0
10
20
30
40
−4
−2
0
2
4
6
8
log(this$lambdas)