R Camp - Day 4 Homework

Emily Elia

8/22/2019

Disclaimer. . .

I know that this model is not correctly modeling what I want it to. I didn't have time to get it right before the 2:30 deadline. This code will run "correctly" and produce plots, but it is not producing the right distributions.

Problem 1 - Quantitatively Predictive Model

Research Question

I model the likelihood of a corrupt incumbent politican getting voted out of office by the electorate. I base my model on an ongoing project about corruption, party identity, and voting behavior. I tackle the puzzle of why corrupt politicians often get re-elected, especially when surveys and experiments show that the majority of citizens oppose corruption and act electorally to vote it out in lab/survey settings. Why does this not translate into the "real world?" I argue that examining this puzzle often leaves out an important factor in a voter's decision-making process: the opposition. Even if a voter dislikes corruption, an undesirable oppostion may drive them to continute to vote for the incumbent candidate as a "lesser of two evils." I argue that the more idealogically distant an opposition candidate is from the incumbent, the more likley the incumbent will be to remain in office because voters will not have a viable alternative to vote for in place of the corrupt incumbent. However, if the opposition is ideologically closer to the incumbent, then voters have an option that is more likely to satisfy their preferences. Therefore, they'll electorally punish the corrupt incumbent by moving their vote to the oppostion.

Theoretical Variable Ranges

Theoretically, ideology is often measured on a left-right scale that typically ranges from 1 (extreme left) to 10 (extreme right). The ideological distance between an incumbent and an opposition would be the difference between the incumbent's ideological placement and the oppostion's ideological placement. This variable would range from 0 to 9, as the placements could be identical, which would yield a difference of 0, or the placements could be at opposite ends of the scale, which would yield a difference of 9.

Many different scales and measurements can capture how peope feel about corruption in their governments. Corruption perceptions can be measured on surveys with categorical answers, such as "Corruption in government is: uncommon, somewhat uncommon, common, etc." Corruption perceptions can also be measured like an index where someone "ranks" the prevalence of corruption on a scale from 0 to 100. In this exercise, for simplicity of simulating data, I will measure voters' corruption perceptions on a scale from 0 to 100, where 0 indicates that a person believes their government contains absolutely no corruption and 100 indicates that a person believers their government is completely corrupt.

Formally, P_{V} = F ((alpha{V} * delta_{V}) + (beta{V} * -epsilon_{V}))

alpha = parameter, how much a corruption perception matters to a voter

delta = voter's corruption perception

beta = parameter, how much ideological difference matters to a voter

epsilon = ideological distance between incumbent and oppostion

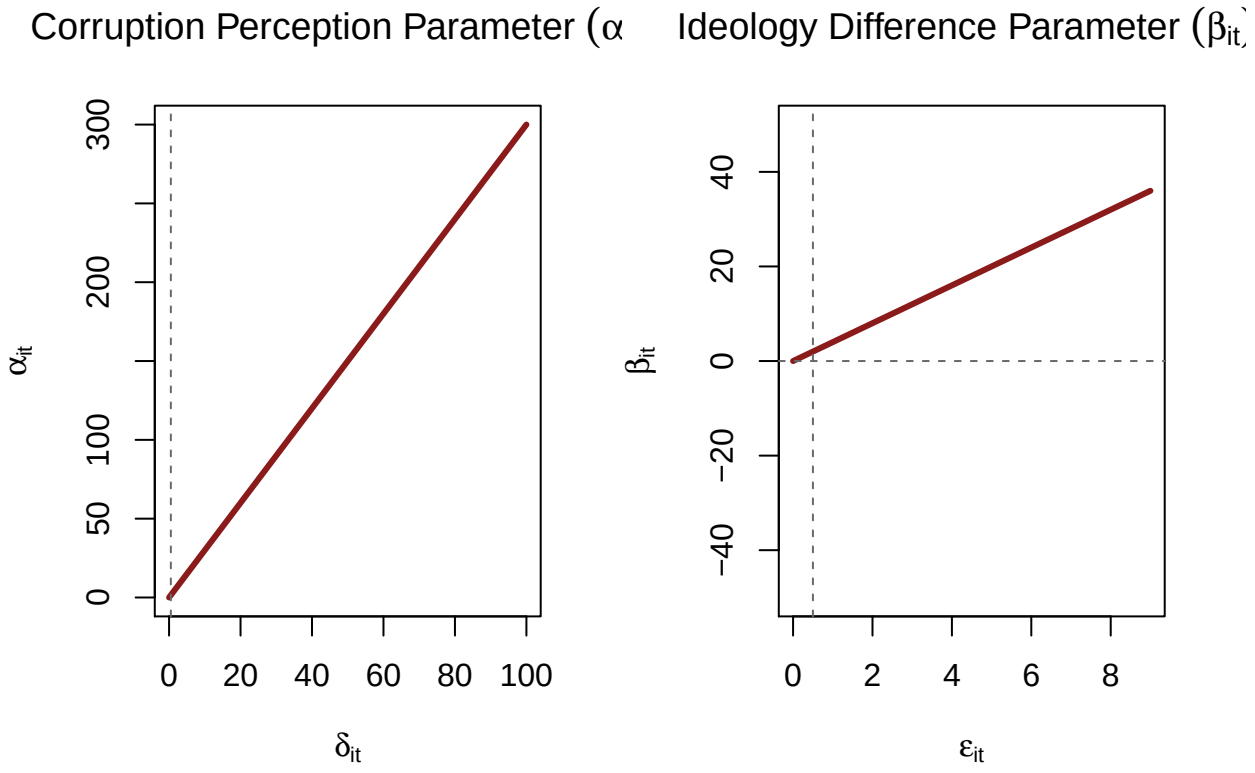
P_{V} = likelihood voter will vote for oppostion over corrupt candidate

F = some cumulative distribution function

H1: Voters have a dislike for corruption, or, in other words, corruption would hurt their utility function, so voters do not want a corrupt politician in office H2: Voters have ideological preferences, and these can take precedence over their dislike for corruption and cause them to vote for the corrupt candidate despite the claim in H1 being true

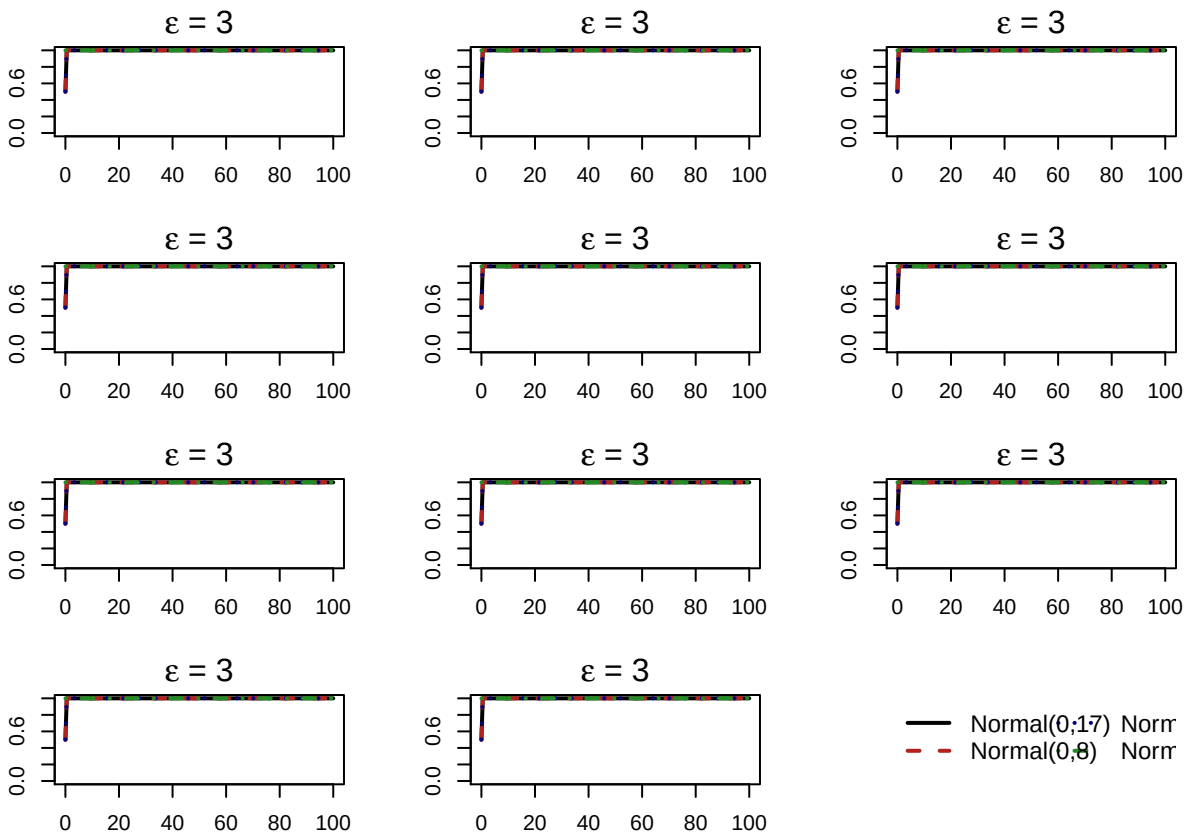
Constructing A Model

```
delta = seq(from=0,to=100,by=0.5)     # corruption perceptions
alpha = 3*delta                    # Weight parameter of perceptions

epsilon = seq(from=0,to=9,by=0.5)      # ideological differences
beta = 4*epsilon               # weight parameter of perceptions

par(mfrow=c(1,2))
plot(alpha ~ delta, type="l", col="firebrick4", lwd=3,
     xlab=expression(delta[it]), ylab=expression(alpha[it]),
     main=expression(paste("Corruption Perception Parameter ", (alpha[it]), sep=" ")))
abline(v=0.5, lty=2, col="dimgray")
plot(beta ~ epsilon, type="l",  col="firebrick4", lwd=3,
     xlab=expression(epsilon[it]), ylab=expression(beta[it]), ylim=c(-50,50),
     main=expression(paste("Ideology Difference Parameter ", (beta[it]), sep=" ")))
abline(v=0.5, lty=2, col="dimgray"); abline(h=0, lty=2, col="dimgray")
```



Looping for Plots

```
# Create similar plots under different conditions
# We use a loop for this purpose

  # Run from here all the way down
  rm(list=ls())
  par(mfrow=c(4,3), mar = c(3, 3, 2, 2)) # Default (5,4,4,2) c(bottom, left, top, right)


for(i in 0:10){                                 # We explore 0 to 10 in epsilon

    epsilon_val = i * 0.5                         # Parameters to be varied
    SD = 5                                       # Parameters to be varied
    SD_small = 2                                  # Parameters to be varied
    SD_large = 8                                 # Parameters to be varied


    epsilon_val = 3.0 # tuning parameter (ideology)

    epsilon = seq(from=0,to=9,by=0.5)    #ideology difference
    delta = seq(from=0,to=100,by=0.5)    # corruption perceptions
    alpha = 3*delta                   # Weight parameter of perceptions
    beta = 4*epsilon_val              # weight parameter of perceptions

    q = (alpha * delta) * (beta * epsilon_val)


    prob  <- pnorm(q=q, mean=0, sd=SD)          # Probability of voting
    prob2 <- pnorm(q=q, mean=0, sd=SD_small)    # Probability of voting
    prob3 <- pnorm(q=q, mean=0, sd=SD_large)    # Probability of voting
    prob4 <- pnorm(q=q, mean=-20, sd=SD_small) # Probability of voting

    plot(prob ~ delta, type="l", lwd=2, ylim=c(0,1), ylab="",
        xlab=expression(Delta), cex.main=1.5,
        main=substitute(paste(epsilon, sep=" = ", v), list(v=epsilon_val)))
    lines(prob2 ~ delta, type="l", lty=2, col="firebrick", lwd=2)
    lines(prob3 ~ delta, type="l", lty=3, col="navy", lwd=2)
    lines(prob4 ~ delta, type="l", lty=4, col="forestgreen", lwd=2)
     }

  # We want to put a legend on this combined graph
  plot(1, type = "n", axes=FALSE, xlab="", ylab="") # No plotting
  legend(x = "topleft",
         legend = c("Normal(0,17)", "Normal(0,8)", "Normal(0,25)",
                    "Normal(-20,8)"),
         col=c("black", "firebrick", "navy", "forestgreen"),
         lty=c(1,2,3,4), ncol=2,
         lwd=2, cex=1.1, horiz = FALSE, text.width=0.2, box.col = "white")
```

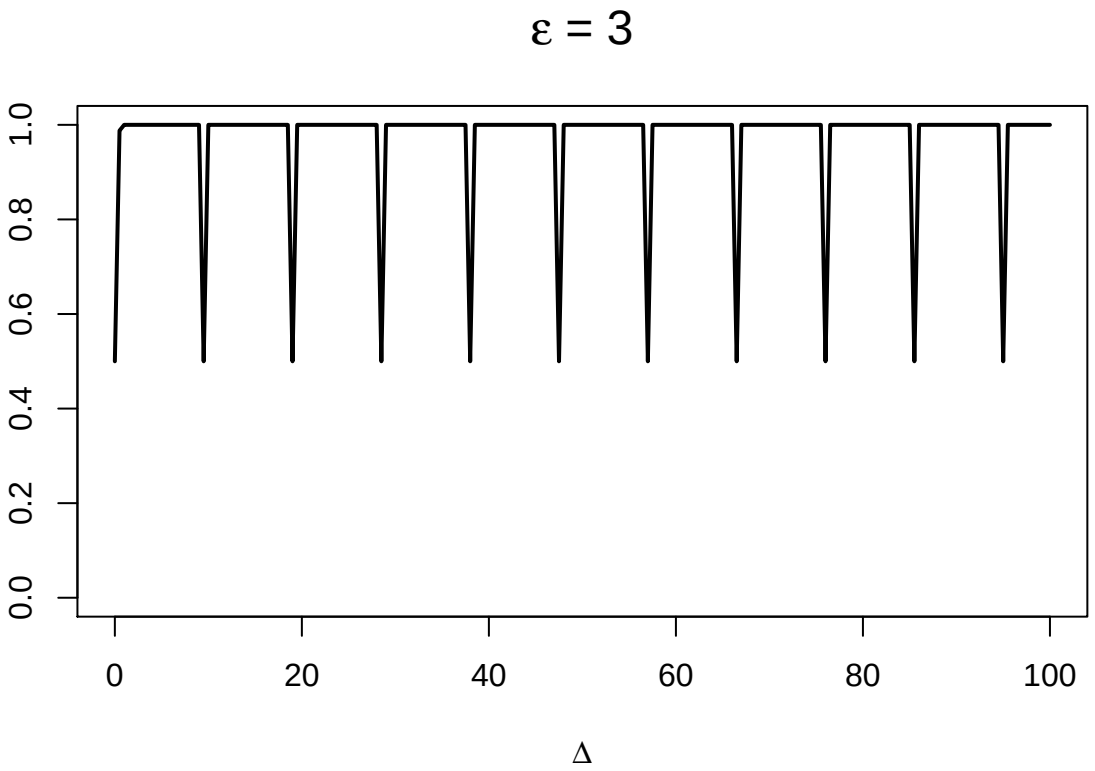


Problem 2 - Building A Function For the Model

```
# Building a function for the model #

PredModel<- function(xvar1, xvar2, setx2, SD){
  #variable 1, variable 2, setx x variable 2, standard deviation)

  alpha #para1
  delta <- xvar1
  beta #para2
  epsilon <- xvar2
  epsilon_val <- setx2
  SD <- SD
  q #dist

  delta = seq(from=0,to=100,by=0.5)     # corruption perceptions
  alpha = 3*delta                    # Weight parameter of perceptions

  epsilon = seq(from=0,to=9,by=0.5)      # ideological differences
  beta = 4*epsilon                # weight parameter of perceptions

  q = (alpha * delta) * (beta * epsilon_val)

  prob  <- pnorm(q=q, mean=0, sd=SD)          # Probability of voting
```

```
  plot(prob ~ delta, type="l", lwd=2, ylim=c(0,1), ylab="",
       xlab=expression(Delta), cex.main=1.5,
       main=substitute(paste(epsilon, sep=" = ", v), list(v=epsilon_val)))

  return(q)

  }

PredModel(delta, epsilon, epsilon_val, 2)
```

```
## Warning in (alpha * delta) * (beta * epsilon_val): longer object length is
## not a multiple of shorter object length
```




```
##   [1]       0.0       4.5      36.0     121.5     288.0     562.5     972.0
##   [8]    1543.5    2304.0    3280.5    4500.0    5989.5    7776.0    9886.5
##  [15]   12348.0   15187.5   18432.0   22108.5   26244.0       0.0    1800.0
##  [22]    3969.0    6534.0    9522.0   12960.0   16875.0   21294.0   26244.0
##  [29]   31752.0   37845.0   44550.0   51894.0   59904.0   68607.0   78030.0
##  [36]   88200.0   99144.0  110889.0       0.0    6844.5   14400.0   22693.5
##  [43]   31752.0   41602.5   52272.0   63787.5   76176.0   89464.5  103680.0
##  [50]  118849.5  135000.0  152158.5  170352.0  189607.5  209952.0  231412.5
##  [57]  254016.0       0.0   15138.0   31329.0   48600.0   66978.0   86490.0
##  [64]  107163.0  129024.0  152100.0  176418.0  202005.0  228888.0  257094.0
##  [71]  286650.0  317583.0  349920.0  383688.0  418914.0  455625.0       0.0
##  [78]   26680.5   54756.0   84253.5  115200.0  147622.5  181548.0  217003.5
##  [85]  254016.0  292612.5  332820.0  374665.5  418176.0  463378.5  510300.0
##  [92]  558967.5  609408.0  661648.5  715716.0       0.0   41472.0   84681.0
##  [99]  129654.0  176418.0  225000.0  275427.0  327726.0  381924.0  438048.0
## [106]  496125.0  556182.0  618246.0  682344.0  748503.0  816750.0  887112.0
## [113]  959616.0 1034289.0       0.0   59512.5  121104.0  184801.5  250632.0
```

```
## [120]  318622.5  388800.0  461191.5  535824.0  612724.5  691920.0  773437.5
## [127]  857304.0  943546.5 1032192.0 1123267.5 1216800.0 1312816.5 1411344.0
## [134]       0.0   80802.0  164025.0  249696.0  337842.0  428490.0  521667.0
## [141]  617400.0  715716.0  816642.0  920205.0 1026432.0 1135350.0 1246986.0
## [148] 1361367.0 1478520.0 1598472.0 1721250.0 1846881.0       0.0  105340.5
## [155]  213444.0  324337.5  438048.0  554602.5  674028.0  796351.5  921600.0
## [162] 1049800.5 1180980.0 1315165.5 1452384.0 1592662.5 1736028.0 1882507.5
## [169] 2032128.0 2184916.5 2340900.0       0.0  133128.0  269361.0  408726.0
## [176]  551250.0  696960.0  845883.0  998046.0 1153476.0 1312200.0 1474245.0
## [183] 1639638.0 1808406.0 1980576.0 2156175.0 2335230.0 2517768.0 2703816.0
## [190] 2893401.0       0.0  164164.5  331776.0  502861.5  677448.0  855562.5
## [197] 1037232.0 1222483.5 1411344.0 1603840.5 1800000.0
```