Pset 2

Maria Neely

11/5/2019

```
library(haven)
library(foreign)
```

```
## Warning: package 'foreign' was built under R version 3.5.2
```

```
data <- read.dta("~/Desktop/macs-405/problem-set-2/PSET 2 Files/conf06.dta")

conf06 <- subset(data, data$nominee!="ALITO")
vars <- c("vote", "nominee", "sameprty", "qual", "lackqual", "EuclDist2", "strngprs") # vector of vars
conf <- conf06[vars] # retain only key vars from above object
conf$numvote <- as.numeric(conf$vote)-1 # from 1/2 to 0/1
conf$numstrngprs <- as.numeric(conf$strngprs)-1 # same as above
```

```
logit <- glm(numvote ~ EuclDist2 + qual + lackqual + sameprty + numstrngprs,
             data = conf,
             family = binomial); summary(logit)
```

```
##
## Call:
## glm(formula = numvote ~ EuclDist2 + qual + lackqual + sameprty +
##     numstrngprs, family = binomial, data = conf)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.3041   0.0828   0.1860   0.3870   2.0300
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)  2.103e+07  4.593e+06   4.578 4.69e-06 ***
## EuclDist2   -4.031e+00  2.867e-01 -14.061  < 2e-16 ***
## qual        -2.103e+07  4.593e+06  -4.578 4.69e-06 ***
## lackqual    -2.103e+07  4.593e+06  -4.578 4.69e-06 ***
## sameprty     1.437e+00  1.542e-01   9.320  < 2e-16 ***
## numstrngprs  1.681e+00  1.491e-01  11.273  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2717.5  on 3708  degrees of freedom
## Residual deviance: 1661.6  on 3703  degrees of freedom
## AIC: 1673.6
##
## Number of Fisher Scoring iterations: 6
```

```
#classify logit
logit.probs <- predict(logit,
                       type = "response")
```

```
head(logit.probs)
```

```
##         1         2         3         4         5         6
## 0.8808255 0.8685701 0.8799734 0.1032824 0.8791534 0.8380614
```

```
#based on our probabilities we need to clasify so that if observation is 0.5, make it a 0- else, make i
logit.pred <- ifelse(logit.probs > 0.5, 1 ,0)

# confusion matrix- columns are actual, and rows are predicted
table<- table(logit.pred, conf$numvote)

false_pos_rate <- (table[1]/table[4])*100

false_neg_rate <- (table[3]/table[2])*100

mean(logit.pred == conf$numvote)
```

```
## [1] 0.9161499
```

Based on our regression, we see that all our independent variables, the squared distance between the senator's ideal point and the nominee's inferred ideal point (EuclDist2), the perceived qualifications of a nominee (qual), the lack of qualification of the nominee (lackqual), a dummy variable indicating that the president is strong (strngprs), and a dummy variable indicating that the senator shares the president's party affiliation (sameprty), are all extremely statistically significant, meaning all had p-values less than .05. We therefore can reject our null hypothesis, and say that there is a relationship between our variables and Senator's voting "yes". This means that very rarely (less than 1% of the time), if there was no relationship, would we get similar results due to random sampling error. This is further substantiated by our confusion matrix. We see true positives to be most common with 3, 1988 instances (i.e. the predicted value is accurately predicted 3,198 times). Compared to the true positives, false positives are very rare, occurring only 244 times. Our false positive rate is 6.2 %. Similarly, we see that true negatives (200 occurrences) are much more common than false negatives (67), with a false negative rate at 27.4%.

```
library(MASS)
lda <- lda(numvote ~ EuclDist2 + qual + lackqual + sameprty + numstrngprs,
           data=conf)
```

```
## Warning in lda.default(x, grouping, ...): variables are collinear
```

```
# inspect the model
lda
```

```
## Call:
## lda(numvote ~ EuclDist2 + qual + lackqual + sameprty + numstrngprs,
##     data = conf)
##
## Prior probabilities of groups:
##         0         1
## 0.1197088 0.8802912
##
## Group means:
##   EuclDist2      qual  lackqual  sameprty numstrngprs
## 0 0.3871892 0.5411149 0.4588851 0.1846847   0.2747748
## 1 0.1545379 0.8102910 0.1897090 0.6052067   0.6125574
##
## Coefficients of linear discriminants:
##                    LD1
## EuclDist2   -2.9852821
```

```
## qual         1.3906733
## lackqual    -1.3906733
## sameprty     0.5915900
## numstrngprs  0.6746973
```

```
numvote <- test$numvote

lda.pred <- predict(lda, newdata=test)

data.frame(lda.pred)[1:5,]
```

```
##    class posterior.0 posterior.1        LD1
## 9      1   0.1012798   0.8987202 -0.6322452
## 11     1   0.1126519   0.8873481 -0.6943067
## 14     1   0.1156756   0.8843244 -0.7098809
## 15     0   0.5674207   0.4325793 -1.9106696
## 18     1   0.1558826   0.8441174 -0.8894992
```

```
# confusion matrix
table <- table(lda.pred$class, numvote)

table
```

```
##    numvote
##       0   1
##   0  40  16
##   1  54 632
```

```
false_pos_rate <- (table[1]/(table[4] + table[1])*100)

false_neg_rate <- (table[3]/(table[2] + table[3])*100)

# check the classification rate
mean(lda.pred$class == numvote)
```

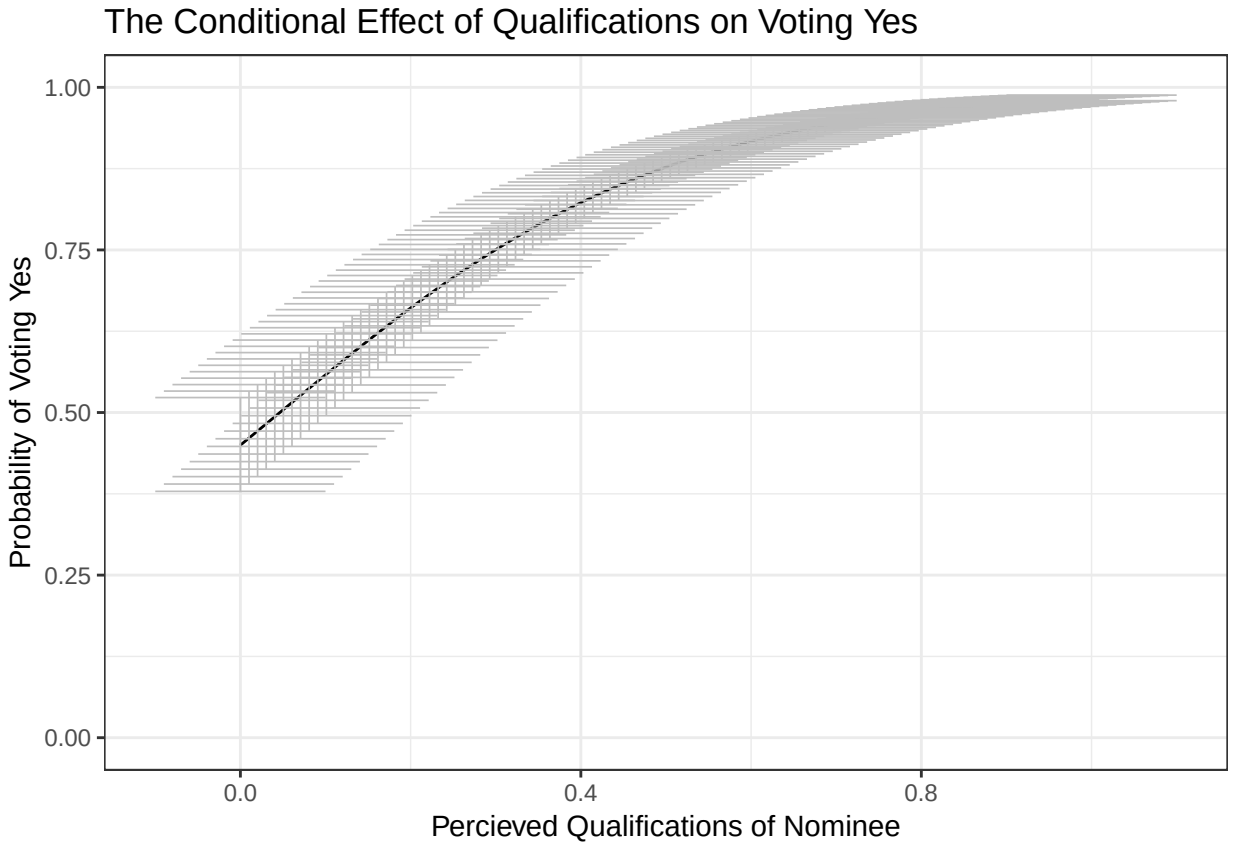
```
## [1] 0.9056604
```

After performing our lda, we see that 12% of your training data corresponds to "no" votes (0) and 88% of your training data corresponds to "yes" votes (1). Looking at the group means, we see that the qualifications of the nominee has the most influence on the "yes" votes and the "no" votes. This is reassuring. Finally, when comparing the coefficients of linear discriminants, we see that the best predictor of yes and no votes is EuclDist2, because it has the coefficient furthest from 0. As EuclDist2 increases, the likelihood of a yes vote decreases. Looking at the confusion matrix, we see a relatively low false positive rate (7.3%) and high false negative rate (30%). However, our classification rate is high, 91.9%, meaning our results are accurate.

```
library(ggplot2)
# A look beyond classification - conditional predicted probabilities
logitmod3 <- glm(numvote ~ EuclDist2 + qual + sameprty + numstrngprs,
                 family = binomial(link=logit),
                 data = conf)

# CIs for predicted probabilities. This is creating out synthetic data: let liberal2 range from 20-80,
newdata2 <- with(conf, data.frame(qual = rep(seq(from = 0, to = 1, length.out = 100)),
                                  numstrngprs = mean(numstrngprs),
                                  sameprty = mean(sameprty),
                                  EuclDist2 = mean(EuclDist2)))
```

```r
newdata3 <- cbind(newdata2, predict(logitmod3,
                                    newdata = newdata2,
                                    type = "link",
                                    se = TRUE))
# Add CIs
newdata3 <- within(newdata3, {
  PredictedProb <- plogis(fit)
  LL <- plogis(fit - (1.96 * se.fit))
  UL <- plogis(fit + (1.96 * se.fit))
})


# Plot predictions with CIs
ggplot(newdata3, aes(x = qual, y = PredictedProb)) +
  geom_line() +
  geom_errorbar(aes(ymin = LL, ymax = UL),
                color="gray",
                size=.3,
                width=.2,
                position = position_dodge(.9)) +
  labs(x = "Percieved Qualifications of Nominee",
       y = "Probability of Voting Yes") +
  scale_fill_hue(breaks = c("No", "Yes"),
                 labels = c("No",  "Yes")) +
  ggtitle("The Conditional Effect of Qualifications on Voting Yes") +
  theme_bw() +
  theme(legend.justification = c(.7,1),
        legend.position = c(.9,.3)) + ylim(0,1)
```

```
## Warning: position_dodge requires non-overlapping x intervals
```



Our graph of "The Conditional Effect of Qualifications on Voting Yes" show us that as the perceived qualifications of the nominee increase, the probability a Senator votes yes also increases. Notably, this happens in a non-linear way, so once the perceived qualifications of the nominee are above around 0.5, the probability of voting yes does not increase by that much. From this graph we also see our model has more errors when perceived qualifications of the nominee are low.

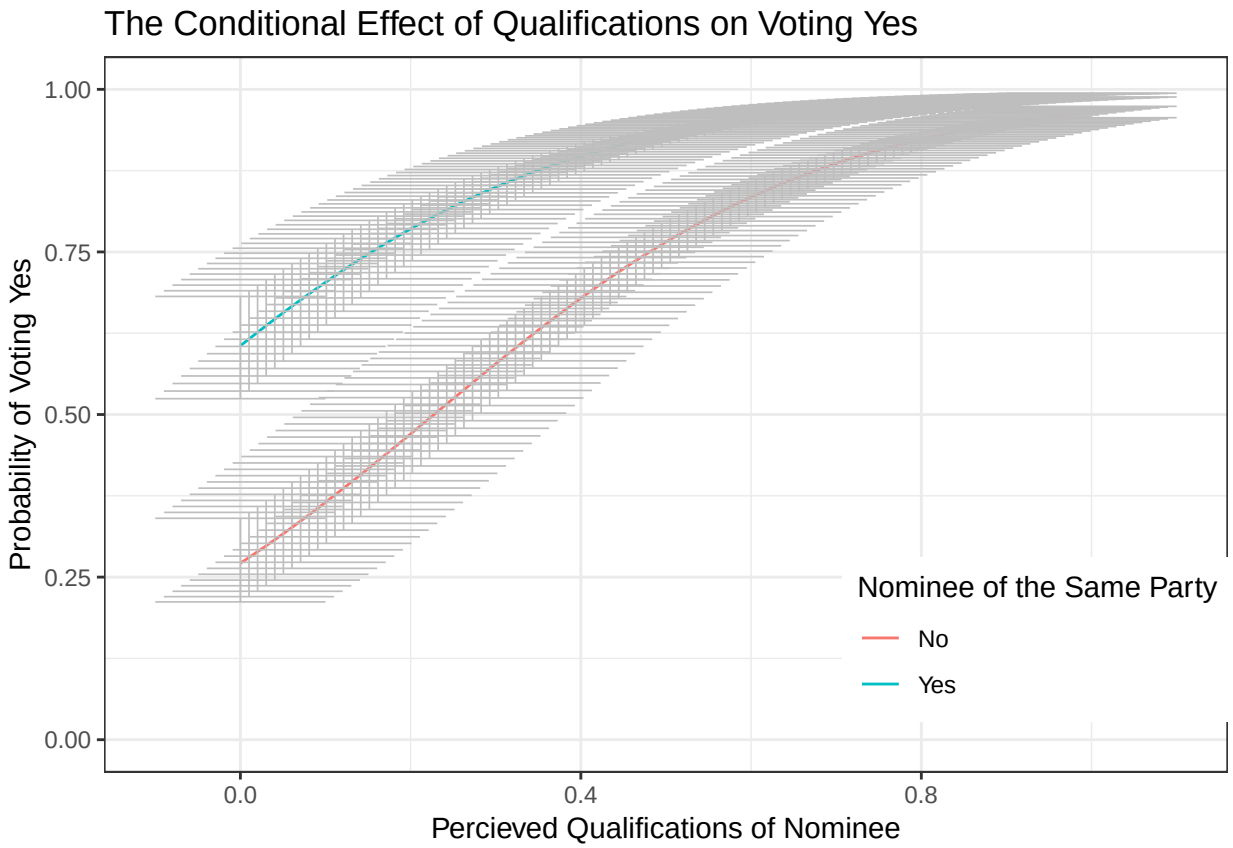
5. Reassuringly, we see in both our LDA and regression models that the squared distance between the senator's ideal point and the nominee's inferred ideal point (EuclDist2) is the variable most related to predicting the number of votes a SCOTUS nominee will receive. While there are other variables that impact predicting the number of votes a SCOTUS nominee will receive, our models find that ultimately Senator's take into account their own ideals, and how these match with the nominee's, impact their votes on a SCOTUS nominee.

Interestingly, although empirically we know that the Supreme Court process has become extremely politicized, our data does not adequately reflect such extreme politicization. For example, in our regression model, the dummy variable representing if the president is the same party as the Senator, we see the smallest relationship between sameprty and the number of votes for the nominee, with the slope between numvote and sameprty being 1.437e+00. However, we would expect to see very high relationship between sameprty and numvote given our empirical knowledge. Similarly, in our LDA classifier, the slope between numvote and sameprty is 0.5915900, a small slope compared to other variables' relationships with numvote. This suggests that despite that our model was deemed accurate through error checks, there perhaps is something missing from our models. Perhaps, there is an intervening variable between sameprty and numvote that masks the true relationship between the two variables. This is an area of further investigation when assessing the validity of our models.

```
library(ggplot2)
# A look beyond classification - conditional predicted probabilities
logitmod3 <- glm(numvote ~ EuclDist2 + qual + sameprty + numstrngprs,
```

```
                family = binomial(link=logit),
                data = conf)

# CIs for predicted probabilities. This is creating out synthetic data: let liberal2 range from 20-80,
newdata2 <- with(conf, data.frame(qual = rep(seq(from = 0, to = 1, length.out = 100),2),
                                  numstrngprs = mean(numstrngprs),
                                  sameprty = rep(0:1, each = 100),
                                  EuclDist2 = mean(EuclDist2)))

newdata3 <- cbind(newdata2, predict(logitmod3,
                                    newdata = newdata2,
                                    type = "link",
                                    se = TRUE))
# Add CIs
newdata3 <- within(newdata3, {
  PredictedProb <- plogis(fit)
  LL <- plogis(fit - (1.96 * se.fit))
  UL <- plogis(fit + (1.96 * se.fit))
})

newdata3$sameprty <- factor(newdata3$sameprty, labels = c("No", "Yes"))


# Plot predictions with CIs
ggplot(newdata3, aes(x = qual, y = PredictedProb, color = sameprty)) +
  geom_line() +
  geom_errorbar(aes(ymin = LL, ymax = UL),
                color="gray",
                size=.3,
                width=.2,
                position = position_dodge(.9)) +
  labs(x = "Percieved Qualifications of Nominee",
       y = "Probability of Voting Yes",
       color = "Nominee of the Same Party") +
  scale_fill_hue(breaks = c("No", "Yes"),
                 labels = c("No",  "Yes")) +
  ggtitle("The Conditional Effect of Qualifications on Voting Yes") +
  theme_bw() +
  theme(legend.justification = c(.7,1),
        legend.position = c(.9,.3)) + ylim(0,1)
```

```
## Warning: position_dodge requires non-overlapping x intervals
```



From this graph, we once again see that for both those of the same party as the nominee and different party than the nominee, as perceived qualifications of the nominee increases the probability the Senator votes yes also increases. However, we see that when there is a nominee of the same party as the voting Senator, the probability of voting yes is already quite high (over 0.6), and therefore perceived qualifications of the nominee has less of an effect on probability of voting yes. For Senator's not in the same party as the nominee, perceived qualifications of the nominee have a much higher effect on the probability to vote yes. Interestingly, once perceived qualifications of a nominee are high (above 0.8), the probability to vote yes converges regardless of if the Senator is in the same party as the nominee or not.

```
library(wnominate)
```

```
## Loading required package: pscl
```

```
## Classes and Methods for R developed in the
## Political Science Computational Laboratory
## Department of Political Science
## Stanford University
## Simon Jackman
## hurdle and zeroinfl functions by Achim Zeileis
```

```
##
## ## W-NOMINATE Ideal Point Package
```

```
## ## Copyright 2006 -2019
```

```
## ## Keith Poole, Jeffrey Lewis, James Lo, and Royce Carroll
```

```
## ## Support provided by the U.S. National Science Foundation
```

```
## ## NSF Grant SES-0611974
```

```
library(pscl)


house113 <- readKH(
"/Users/owner/Desktop/macs-405/problem-set-2/PSET 2 FILES/hou113kh.ord",
yea=c(1,2,3),
nay=c(4,5,6),
missing=c(7,8,9),
notInLegis=0,
desc="113th_House_Roll_Call_Data",
debug=FALSE
)
```

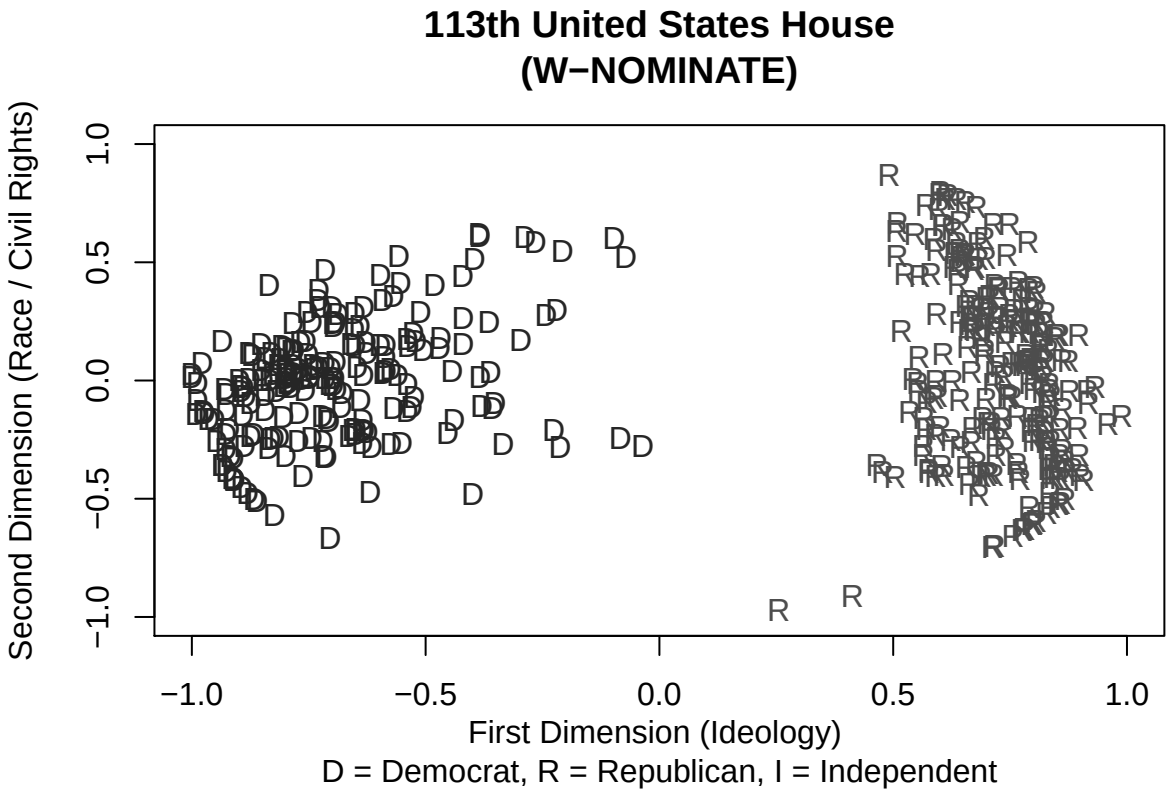
```
## Attempting to read file in Keith Poole/Howard Rosenthal (KH) format.
## Attempting to create roll call object
## 113th_House_Roll_Call_Data
## 445 legislators and 1202 roll calls
## Frequency counts for vote types:
## rollCallMatrix
##      0      1      6      7      9
##  14576 295753 202943    290  21328
```

```
wnom_result <- wnominate(house113,
                         dims = 2,
                         minvotes = 20,
                         lop = 0.025,
                         polarity = c(2,2))
```

```
##
## Preparing to run W-NOMINATE...
##
##  Checking data...
##
##      ... 1 of 445 total members dropped.
##
##      Votes dropped:
##      ... 181 of 1202 total votes dropped.
##
##  Running W-NOMINATE...
##
##      Getting bill parameters...
##      Getting legislator coordinates...
##      Starting estimation of Beta...
##      Getting bill parameters...
##      Getting legislator coordinates...
##      Starting estimation of Beta...
##      Getting bill parameters...
##      Getting legislator coordinates...
##      Getting bill parameters...
##      Getting legislator coordinates...
##      Estimating weights...
##      Getting bill parameters...
##      Getting legislator coordinates...
##      Estimating weights...
```

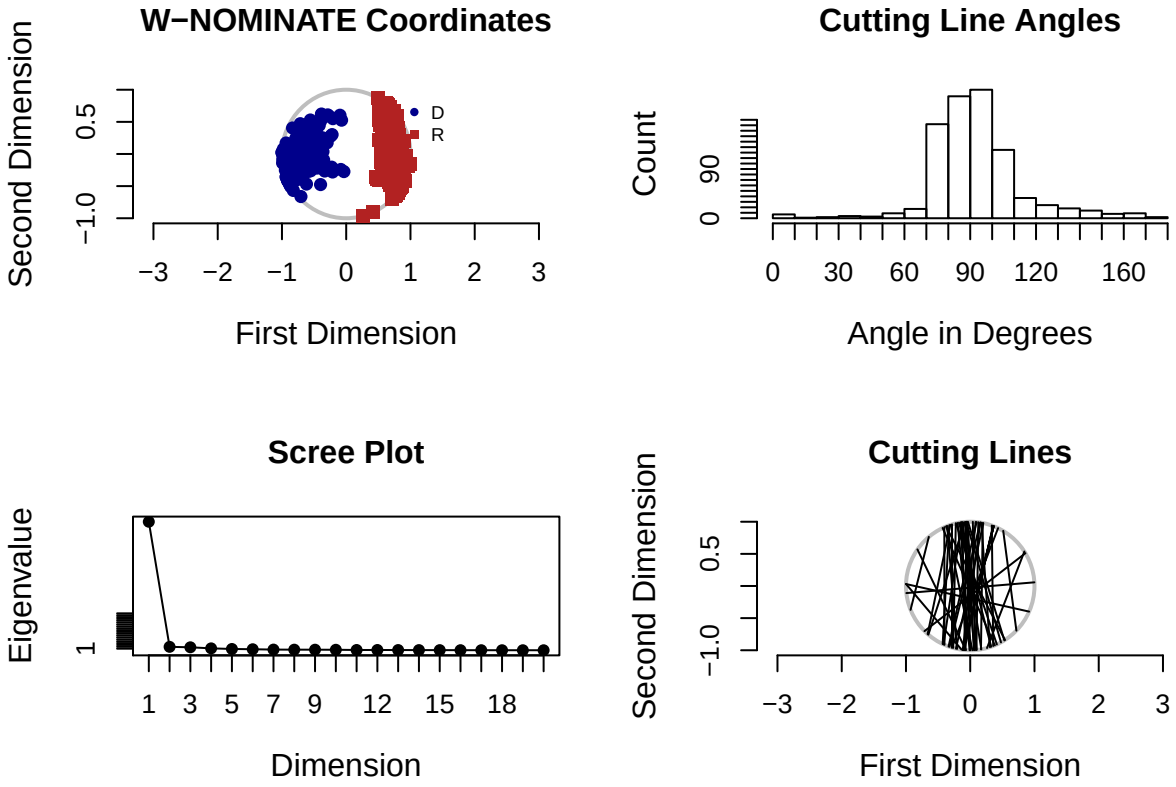
```
##      Getting bill parameters...
##      Getting legislator coordinates...
##
##
## W-NOMINATE estimation completed successfully.
## W-NOMINATE took 222.29 seconds to execute.
```

```r
# store a few things for plotting
wnom1 <- wnom_result$legislators$coord1D
wnom2 <- wnom_result$legislators$coord2D
party <- house113$legis.data$party

# custom plot
plot(wnom1, wnom2,
     main="113th United States House\n(W-NOMINATE)",
     xlab="First Dimension (Ideology) \nD = Democrat, R = Republican, I = Independent",
     ylab="Second Dimension (Race / Civil Rights)",
     xlim=c(-1,1), ylim=c(-1,1), type="n")
points(wnom1[party=="D"], wnom2[party=="D"], pch="D", col="gray15")
points(wnom1[party=="R"], wnom2[party=="R"], pch="R", col="gray30")
points(wnom1[party=="Indep"], wnom2[party=="Indep"], pch="I", col="red")
```



After running the wnom algorithm, we see that the 113th Congress was polarized, with a clear divide in ideology between the Democrats and the Republicans. We also see that Republicans are much more similar in their views regarding ideology, given the dense cluster of Republicans between 0.5-1.0 on ideology (minus the two outliers between 0 and 0.5 on ideology) and less similar in their views on race. Conversely, we see that the Democrats have a larger spread of ideological views, given the wide spread of democrats across ideology from 0.0 to -1.0, but a smaller spread of views on race, given the dense clustering between -0.5 and 0.5. Looking beyond the plot, we see that the results from the wnom algorithm are fairly accurate, with an accuracy rate of 94.3% of yeas predictions correct and 92.8% of the nays predictions correct.

```
#scree plots
# canned plot(s)
plot(wnom_result)
```




```
## NULL
```

```
#apre
wnom_result$fits
```

```
## correctclass1D correctclass2D         apre1D         apre2D         gmp1D
##     92.7916336     93.6033859      0.8167768      0.8374099     0.8398947
##          gmp2D
##      0.8566379
```

Let us now consider the fit of the algorithm. From the scree plot, we see that the first dimension fits the data extremely well, given its high eigenvalue. The second dimension fits slightly better than the rest of the dimensions, but still does not have a particularly good of a fit for our data. Similarly, most of the cutting lines cut at 0 degrees and 90 degrees, indicating the first dimension is a good fit for our data.

The APRE rates show that there is a 77.6% reduction of error for predicting the roll call votes in the first dimension, and a 79.8% reduction of error for predicting the roll call votes in the second dimension. The APRE rates in each dimension are strong, given they are greater than 0.4. The GMP also shows both dimensions are accurate for predicting roll call votes, with high values close to 1.

3. There are three main methods for unfolding binary data: NOMINATE, IRT and Optimal Classification. These methods differ in their assumptions and the way they calculate errors. For example, NOMINATE and IRT use a random utility model, while Optimal Classification uses quadratic utility. Under a random utility model, when the Senator's move away from their ideal point, they lose lots of utility, whereas under the quadratic utility model, when the Senator's move away from their ideal point their loss of utility is less severe. Also, NOMINATE and IRT treat voting as probabilistic, while Optimal Classification does not treat voting probabilistically. In terms of politics, NOMINATE and IRT seem to

be better for more controversial votes, when there are dramatic changes in utility. Conversely, Optimal Classification may better capture less controversial votes, as there is less dramatic change in utility over voting changes.