Charles Coffey 17.835 PSET3

Charles Coffey

10/4/2020

PROBLEM 1

1.1

```
rm(list = ls(all.names = TRUE))
library(rddtools)

## Loading required package: AER

## Loading required package: car

## Loading required package: carData

## Loading required package: lmtest

## Loading required package: zoo

##
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric

## Loading required package: sandwich

## Loading required package: survival

## Loading required package: np

## Nonparametric Kernel Methods for Mixed Datatypes (version 0.60-10)
## [vignette("np_faq",package="np") provides answers to frequently asked ques
tions]
## [vignette("np",package="np") an overview]
## [vignette("entropy_np",package="np") an overview of entropy-based methods]

#KEPT
#TRUE = 1; FALSE = 0;

#ALLOC_TYPE
#Flat = 1;
#Late_High = 0;

#MESSAGE
#YES = 1;
```

```r
#NO = 0;

Retro_Vote = read.csv("retro_vote.csv")

Retro_Vote[Retro_Vote$kept,"kept"] = 1
Retro_Vote[Retro_Vote$alloc_type == "Flat", "alloc_type"] = 0 #control
Retro_Vote[Retro_Vote$alloc_type == "Late_High", "alloc_type"] = 1 #treated
Retro_Vote[Retro_Vote$message == "NO", "message"] = 0 #control
Retro_Vote[Retro_Vote$message == "YES", "message"] = 1 #treated
```

1.2

```r
retention_rates = tapply(Retro_Vote$kept, list(Retro_Vote$alloc_type, Retro_V
ote$message), mean)

#alloc_type is the rows & message is the columns
print(retention_rates)
```

```
##           0         1
## 0 0.5666667 0.5625000
## 1 0.7412698 0.8669725
```

1.3

```r
#standard error function defined below

std_error = function(X){
  #function that computes standard error
  #INPUT: vector object
  #OUTPUT: standard error scalar value for vector object

  sqrt(var(X)/length(X))
}

standard_errors = tapply(Retro_Vote$kept, list(Retro_Vote$alloc_type, Retro_V
ote$message), std_error)

flat_nomes_interval = c(retention_rates["0", "0"] - 1.96*standard_errors["0",
"0"], retention_rates["0", "0"] + 1.96*standard_errors["0", "0"])
flat_yesmes_interval = c(retention_rates["0", "1"] - 1.96*standard_errors["0"
, "1"], retention_rates["0", "1"] + 1.96*standard_errors["0", "1"])
lahi_nomes_interval = c(retention_rates["1", "0"] - 1.96*standard_errors["1",
"0"], retention_rates["1", "0"] + 1.96*standard_errors["1", "0"])
lahi_yesmes_interval = c(retention_rates["1", "1"] - 1.96*standard_errors["1"
, "1"], retention_rates["1", "1"] + 1.96*standard_errors["1", "1"])

print(paste("The 95% confidence interval for Flat & No Message is", toString(
flat_nomes_interval[1]), "< x <",
            toString(flat_nomes_interval[2]), sep = " "))
```

```
## [1] "The 95% confidence interval for Flat & No Message is 0.386310101991421 < x < 0.747023231341912"
```

```
print(paste("The 95% confidence interval for Flat & Yes Message is", toString(flat_yesmes_interval[1]), "< x <",
            toString(flat_yesmes_interval[2]), sep = " "))
```

```
## [1] "The 95% confidence interval for Flat & Yes Message is 0.470212076571142 < x < 0.654787923428858"
```

```
print(paste("The 95% confidence interval for Late High & No Message is", toString(lahi_nomes_interval[1]), "< x <",
            toString(lahi_nomes_interval[2]), sep = " "))
```

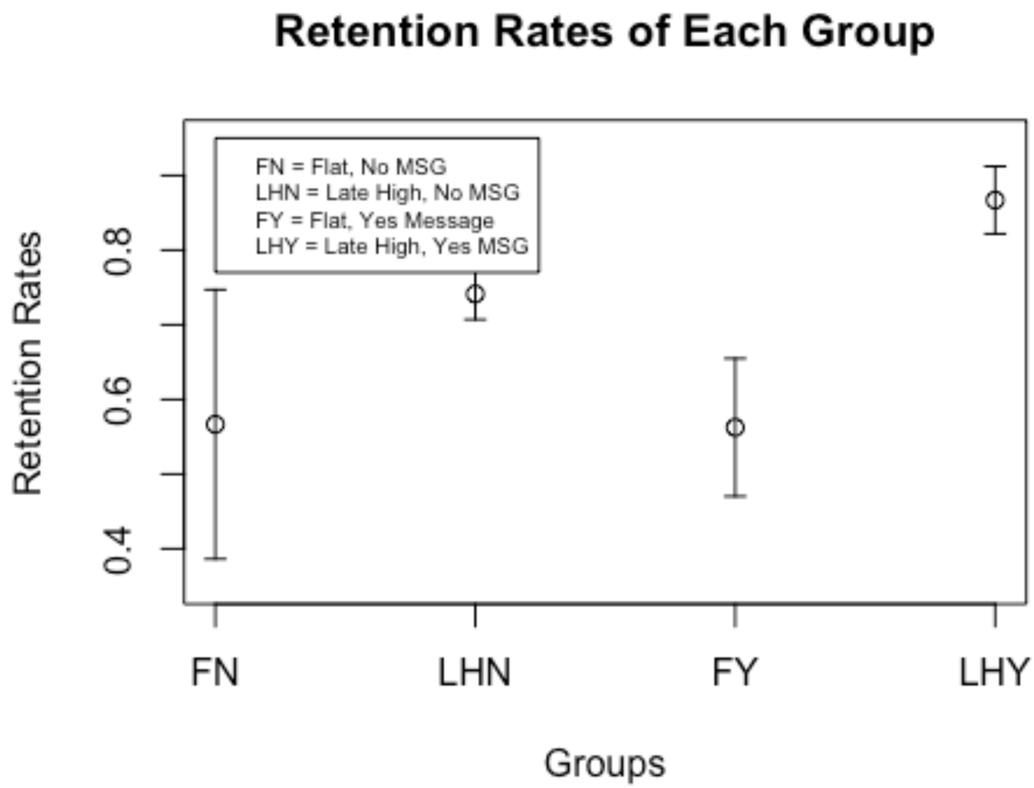
```
## [1] "The 95% confidence interval for Late High & No Message is 0.707044921268903 < x < 0.77549476127078"
```

```
print(paste("The 95% confidence interval for Late High & Yes Message is", toString(lahi_yesmes_interval[1]), "< x <", toString(lahi_yesmes_interval[2]), sep = " "))
```

```
## [1] "The 95% confidence interval for Late High & Yes Message is 0.821786923009837 < x < 0.912158031118604"
```

1.4

```
plot(c(1:4), retention_rates,
     xaxt='n',
     ylim = c(0.35, 0.95),
     ylab = "Retention Rates",
     xlab = "Groups",
     main = "Retention Rates of Each Group")
arrows(c(1:4), retention_rates-1.96*standard_errors, c(1:4), retention_rates+1.96*standard_errors, length=0.05, angle=90, code=3)
axis(1, at=c(1:4), labels=c("FN", "LHN", "FY", "LHY"))
legend(1, 0.95, c("FN = Flat, No MSG", "LHN = Late High, No MSG", "FY = Flat, Yes Message", "LHY = Late High, Yes MSG"), cex = 0.6)
```



The large confidence intervals indicate the small sample sizes for the Flat allocator types. If we had more flat samples, the confidence intervals would be smaller.

1.5

```
#calculating ATE of receiving high payment in the second 30 days (effect on k
ept among all respondents)

alloc_treated = Retro_Vote[Retro_Vote$alloc_type == 1, ]
alloc_control = Retro_Vote[Retro_Vote$alloc_type == 0, ]

#ATE calculation
dif_mean_alloc = mean(alloc_treated$kept) - mean(alloc_control$kept)

ols_alloc = lm(Retro_Vote$kept ~ Retro_Vote$alloc_type)

#ATE from regression
Beta_alloc = summary(ols_alloc)$coefficients[2,1]

print(paste("Difference-in-means method: ", toString(dif_mean_alloc)))

## [1] "Difference-in-means method:  0.210204623970237"
```

```
print(paste("OLS regression: ", toString(Beta_alloc)))

## [1] "OLS regression:  0.210204623970237"
```

I notice that the average treatment affect on the treated is a positive 0.21, showing us that a person is 21% more likely to vote for a candidate if they follow the Late High policy. This may point to a finding that voters may be more inclined to vote for a president who had the economy running very well in the year leading up to his re-election. This year, it could mean that Trump has this factor working against him as the economy crashed with introduction of this global pandemic. Also, his administration's handling of the pandemic in the U.S. has not be received well so I suspect that this will hurt his vote share as well. It it interesting to see how easily elections can be manipulated in the U.S. We can see that elections are not always, even many times, not decided by the candidate that is best fit for the position.

1.6

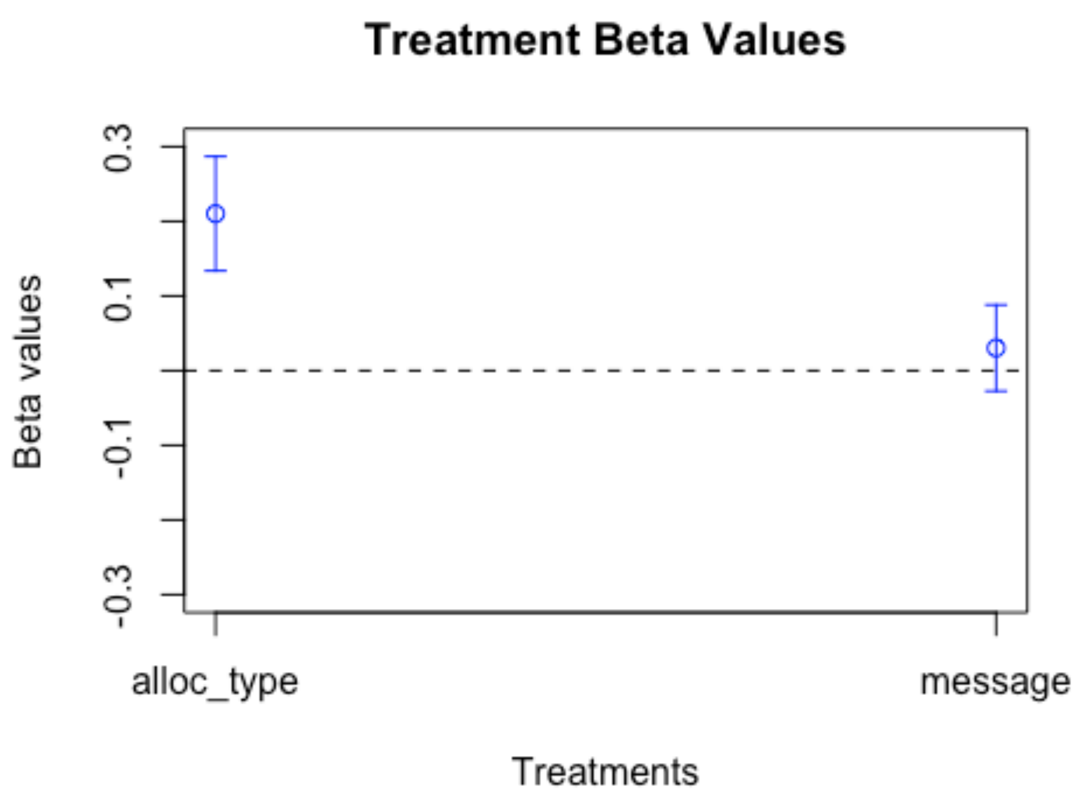
```
#calculating ATE of receiving campaign primer (effect on kept among all respo
ndents)
mes_treated = Retro_Vote[Retro_Vote$message == 1, ]
mes_control = Retro_Vote[Retro_Vote$message == 0, ]

#ATE calculation
dif_mean_mes = mean(mes_treated$kept) - mean(mes_control$kept)

ols_mes = lm(Retro_Vote$kept ~ Retro_Vote$message)

#ATE from regression
Beta_mes = summary(ols_mes)$coefficients[2,1]

print(paste("Difference-in-means method: ", toString(dif_mean_mes)))

## [1] "Difference-in-means method:  0.0303030303030304"

print(paste("OLS regression: ", toString(Beta_mes)))

## [1] "OLS regression:  0.030303030303031"
```

This data shows us that campaign primer messages on average have much lower treatment affect than the allocator types. It could be argued that the messages have nearly no effect.

1.7

```
ols_alloc_se = summary(ols_alloc)$coefficients[2,2]
ols_mes_se = summary(ols_mes)$coefficients[2,2]

#confidence intervals
alloc_Beta_interval = c(Beta_alloc - 1.96*ols_alloc_se, Beta_alloc + 1.96*ols
_alloc_se)
mes_Beta_interval = c(Beta_mes - 1.96*ols_mes_se, Beta_mes + 1.96*ols_mes_se)
```

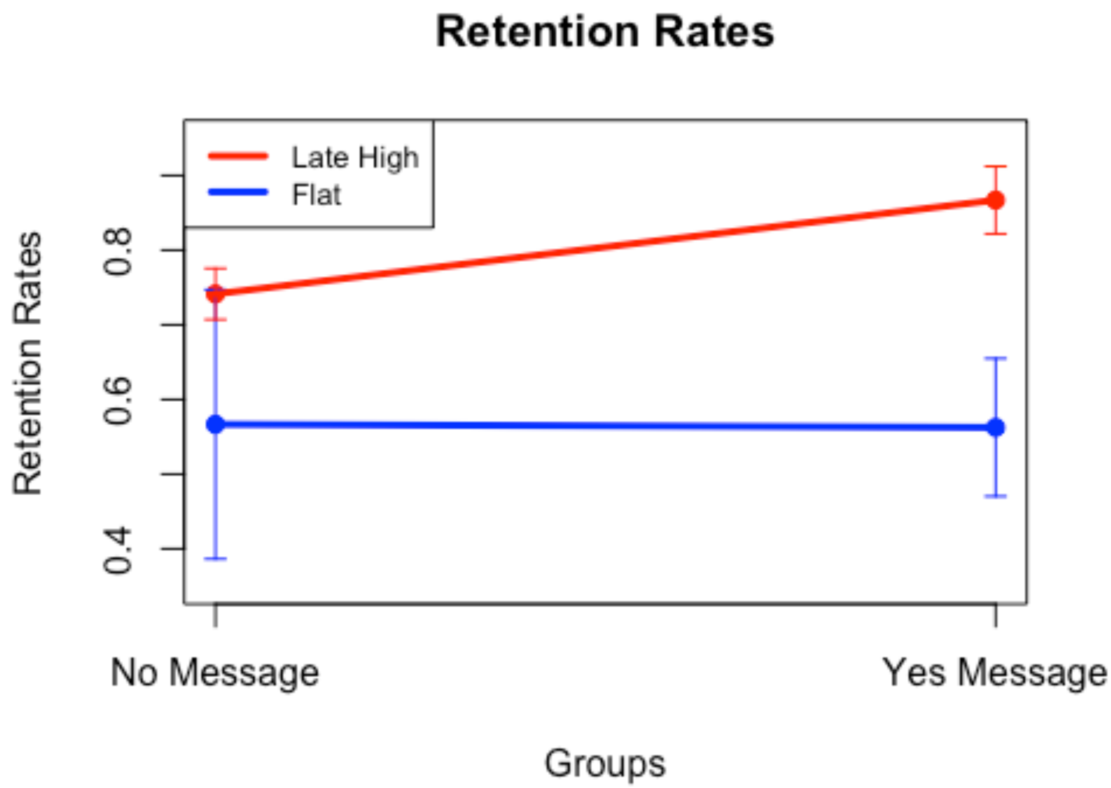
```
print(paste("The 95% confidence interval for the alloc_type Beta value is", t
oString(alloc_Beta_interval[1]), "< x <", toString(alloc_Beta_interval[2]), s
ep = " "))

## [1] "The 95% confidence interval for the alloc_type Beta value is 0.133623
579979903 < x < 0.28678566796057"

print(paste("The 95% confidence interval for the message Beta value is", toSt
ring(mes_Beta_interval[1]),
            "< x <", toString(mes_Beta_interval[2]), sep = " "))

## [1] "The 95% confidence interval for the message Beta value is -0.02743633
29066788 < x < 0.0880423935127407"

plot(c(1,2), c(Beta_alloc, Beta_mes), col = "blue", xaxt = "n",
     xlab = "Treatments", ylab = "Beta values", ylim = c(-0.3, 0.3), main = "
Treatment Beta Values")
arrows(c(1:2), c(alloc_Beta_interval[1], mes_Beta_interval[1]), c(1:2), c(all
oc_Beta_interval[2], mes_Beta_interval[2]),
       col = "blue", length=0.05, angle=90, code=3)
axis(1, at=c(1:2), labels=c("alloc_type", "message"))
abline(h=0, lty = 2)
```



The error bar touching the Beta = 0 line implies that the campaign primer message may have no real effect on the retention rates of a candidate.

1.8

```
#same code as 1.5

print(paste("The conditional treatment affect of the Late_High allocator: ",
toString(dif_mean_alloc)))

## [1] "The conditional treatment affect of the Late_High allocator:  0.21020
4623970237"

plot(c(1,1,2,2), retention_rates,
     xaxt='n',
     ylim = c(0.35, 0.95),
     ylab = "Retention Rates",
     xlab = "Groups",
     pch = 19,
     col = c("blue", "red", "blue", "red"),
     main = "Retention Rates")
arrows(c(1,1,2,2), retention_rates-1.96*standard_errors , c(1,1,2,2), retenti
on_rates+1.96*standard_errors,
       length=0.05,
       angle=90,
       code=3,
       col = c("blue", "red", "blue", "red"),)
axis(1, at=c(1,2), labels=c("No Message", "Yes Message"))

tr_points = lm(c(retention_rates[2,1], retention_rates[2,2]) ~ c(1,2))
cl_points = lm(c(retention_rates[1,1], retention_rates[1,2]) ~ c(1,2))

curve(tr_points$coefficients[1] + tr_points$coefficients[2]*x, 1, 2, add = T,
lwd = 3, lty = 1, col = "red")
curve(cl_points$coefficients[1] + cl_points$coefficients[2]*x, 1, 2, add = T,
lwd = 3, lty = 1, col = "blue")

legend("topleft", c("Late High", "Flat"), col = c("red", "blue"), lty = 1, lw
d = 3, cex = 0.8)
```



The slopes of these lines indicate the whether having a campaign primer message had positive or negative effect on each of the allocator types. It seems that the primer message had a positive effect on retention rates for candidates who had the Late High policy and a negative effect for candidates who had the flat policy.

PROBLEM 2

2.1

This is not good evidence for an incumbency advantage because we may not be comparing like candidates to like candidates. There are many other factors that distinguish candidates besides incumbency and these factors could very easily affect the vote share of the candidate. For example, if a mayor of a town is much more active and engaging on social media than a non-incumbent candidate, this could greatly increase their vote share relative to the other candidate. This disparity would not be due strictly due to the mayor's incumbency.

2.2

```
brazil_mayor = read.csv("brazil_mayor.csv")

#only looking at margin for 1996 vote because if we looked at both it would l
iterally be only 30 observations??
brazil_mayor = brazil_mayor[abs(brazil_mayor$vote_margin_pct) < 2,]

#subsetting treated and control groups
brazil_mayor_treated = brazil_mayor[brazil_mayor$incumbent == 1,]
brazil_mayor_control = brazil_mayor[brazil_mayor$incumbent == 0,]

reg_treated = lm(brazil_mayor_treated$margin_2000 ~ brazil_mayor_treated$vote
_margin_pct)
reg_control = lm(brazil_mayor_control$margin_2000 ~ brazil_mayor_control$vote
_margin_pct)

#intercept, slope
mod.tr = c(summary(reg_treated)$coefficients[1,1], summary(reg_treated)$coeff
icients[2,1])
mod.cl = c(summary(reg_control)$coefficients[1,1], summary(reg_control)$coeff
icients[2,1])

summary(reg_treated)

##
## Call:
## lm(formula = brazil_mayor_treated$margin_2000 ~ brazil_mayor_treated$vote_
margin_pct)
##
## Residuals:
##    Min      1Q Median      3Q     Max
## -50.40 -12.94   0.17   11.58 100.97
##
## Coefficients:
##                                        Estimate Std. Error t value Pr(>|t|)
## (Intercept)                              -7.368      2.443  -3.016  0.00283
**
```

```
## brazil_mayor_treated$vote_margin_pct    5.686      2.128   2.672  0.00804 
**
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 19.92 on 251 degrees of freedom
##   (110 observations deleted due to missingness)
## Multiple R-squared:  0.02766,    Adjusted R-squared:  0.02378
## F-statistic: 7.139 on 1 and 251 DF,  p-value: 0.008037
```

```
summary(reg_control)
```

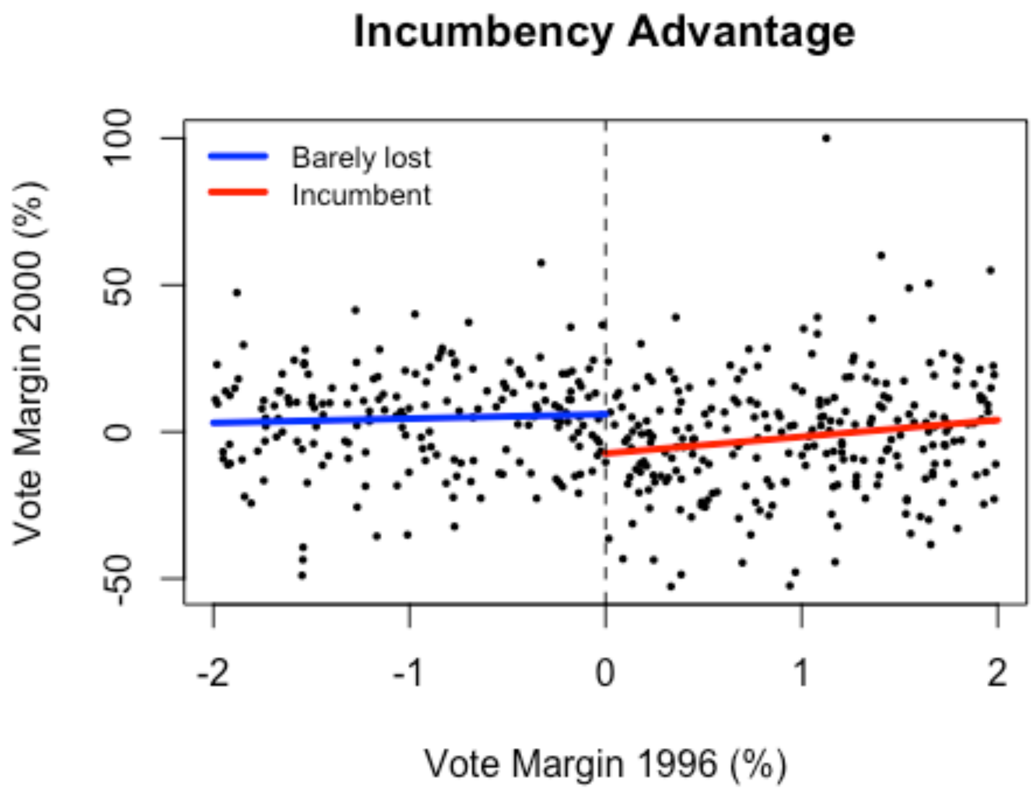
```
##
## Call:
## lm(formula = brazil_mayor_control$margin_2000 ~ brazil_mayor_control$vote_
margin_pct)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -52.634 -11.041   2.158  10.704  52.039
##
## Coefficients:
##                                        Estimate Std. Error t value Pr(>|t|)
## (Intercept)                               5.947      2.325   2.558   0.0113 
*
## brazil_mayor_control$vote_margin_pct    1.451      2.079   0.698   0.4862
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 17.01 on 184 degrees of freedom
##   (97 observations deleted due to missingness)
## Multiple R-squared:  0.002639,   Adjusted R-squared:  -0.002781
## F-statistic: 0.4869 on 1 and 184 DF,  p-value: 0.4862
```

Trimming the data to only candidates who won or lost by 2% seems to be a reasonable enough margin to cut out noisy data. This helps us to make sure that we are comparing candidates that had an actual chance at having the opposite outcome in the election. These candidates are more likely to be affected by incumbency advantages/disadvantages.

2.3

```
plot(brazil_mayor$vote_margin_pct, brazil_mayor$margin_2000,
     xlab = "Vote Margin 1996 (%)",
     ylab = "Vote Margin 2000 (%)",
     main = "Incumbency Advantage",
     cex = 0.5,
     pch = 16)

curve(reg_treated$coefficient[1] + reg_treated$coefficient[2] * x, 0, 2, add
= T, lwd = 3, lty = 1, col = "red")
curve(reg_control$coefficient[1] + reg_control$coefficient[2] * x, -2, 0, add
```

```
= T, lwd = 3, lty = 1, col = "blue")
abline(v = 0, lty = 2)

legend("topleft", legend = c("Barely lost", "Incumbent"), col = c("blue", "re
d"), bty = "n", lty = 1, lwd = 3, cex = 0.8)
```



2.4

```
tau = as.numeric(reg_control$coefficient[1] - reg_treated$coefficient[1])

print(paste("The difference of the fitted values of the two lines at x=0 is:
", toString(tau), "%", sep=""))

## [1] "The difference of the fitted values of the two lines at x=0 is: 13.31
54313857919%"
```

The two lines do not meet at the center. Once we hit the cutoff point, the treatment group jumps down. Based on this visualization, incumbency provides a disadvantage for a mayoral candidate in Brazil.