Code for QSS tidyverse Chapter 4: Prediction

Kosuke Imai and Nora Webb Williams

First Printing

Prediction

Predicting Election Outcomes

Loops in R

```
for (i in X) {
    expression1
    expression2
    ...
    expressionN
}
```

```
library(tidyverse)
library(stringr)

values <- c(2, 4, 6)
n <- length(values) # number of elements in 'values'
results <- rep(NA, n) # empty container vector for storing the results

## loop counter 'i' will take values of 1, 2, ..., n in that order,
## up to the length of 'values'
for (i in seq_along(values)) {
    ## store the result of multiplication as the ith element of
    ## 'results' vector
    results[i] <- values[i] * 2
    print(str_c(values[i], " times 2 is equal to ", results[i]))
}
```

```
## [1] "2 times 2 is equal to 4"
## [1] "4 times 2 is equal to 8"
## [1] "6 times 2 is equal to 12"
```

```
results
```

```
## [1]  4  8 12
```

```r
## check if the code runs when i = 1
i <- 1 # set i to a sample value
x <- values[i] * 2 # the first expression in the loop
print(str_c(values[i], " times 2 is equal to ", x)) # the second expression
```

```
## [1] "2 times 2 is equal to 4"
```

```r
## a toy data frame
data <- data.frame("a" = 1:2, "b" = c("hi", "hey"), "c" = 3:4)
## we see an error occurring at iteration 2
results <- rep(NA, 3)
for (i in seq_along(data)) {
    print(str_c("iteration", i))
    results[i] <- median(data[, i]) # for the i-th column
}
```

```
## [1] "iteration1"
## [1] "iteration2"
```

```
## Warning in mean.default(sort(x, partial = half + 0L:1L)[half
## + 0L:1L]): argument is not numeric or logical: returning NA
```

```
## [1] "iteration3"
```

```r
results
```

```
## [1] 1.5  NA 3.5
```

General Conditional Statements in R

```r
if (X) {
    expression1
    expression2
    ...
    expressionN
}
```

```r
## define the operation to be executed
operation <- "add"
if (operation == "add") {
    print("I will perform addition 4 + 4")
    4 + 4
}
```

```
## [1] "I will perform addition 4 + 4"
```

```
## [1] 8
```

```
if (operation == "multiply") {
    print("I will perform multiplication 4 * 4")
    4 * 4
}
```

```
if (X) {
    expression1a
    ...
    expressionNa
} else {
    expression1b
    ...
    expressionNb
}
```

```
operation <- "multiply"
if (operation == "add") {
    print("I will perform addition 4 + 4")
    4 + 4
} else {
    print("I will perform multiplication 4 * 4")
    4 * 4
}
```

```
## [1] "I will perform multiplication 4 * 4"
```

```
## [1] 16
```

```
if (X) {
    expression1a
    ...
    expressionNa
} else if (Y) {
    expression1b
    ...
    expressionNb
} else {
    expression1c
    ...
    expressionNc
}
```

```
## Note that 'operation' is redefined
operation <- "subtract"
if (operation == "add") {
    print("I will perform addition 4 + 4")
    4 + 4
} else if (operation == "multiply") {
    print("I will perform multiplication 4 * 4")
    4 * 4
} else {
    print(str_c("'", operation, "' is invalid. Use either 'add' or 'multiply'."))
}
```

```
## [1] "'subtract' is invalid. Use either 'add' or 'multiply'."
```

```
values <- 1:5
n <-  length(values)
results <- rep(NA, n)
for (i in seq_along(values)) {
    ## x and r get overwritten in each iteration
    x <- values[i]
    r <- x %% 2  # remainder when divided by 2 to check whether even or odd
    if (r == 0) { # remainder is zero
        print(str_c(x, " is even and I will perform addition: " , x, "+", x))
        results[i] <- x + x
    } else { # remainder is not zero
        print(str_c(x, " is odd and I will perform multiplication: ", x, "*", x))
        results[i] <- x * x
    }
}
```

```
## [1] "1 is odd and I will perform multiplication: 1*1"
## [1] "2 is even and I will perform addition: 2+2"
## [1] "3 is odd and I will perform multiplication: 3*3"
## [1] "4 is even and I will perform addition: 4+4"
## [1] "5 is odd and I will perform multiplication: 5*5"
```

```
results
```

```
## [1]  1  4  9  8 25
```

Poll Predictions

```
## Load the data
data("pres08", package = "qss")
data("polls08", package = "qss")
## Add the Obama margin
polls08 <-
  polls08 %>% mutate(margin = Obama - McCain)
pres08 <-
  pres08 %>% mutate(margin = Obama - McCain)
```

```
library(lubridate)
## what class is middate?
class(polls08$middate)
```

```
## [1] "Date"
```

```
## two example dates and subtraction
x <- ymd("2008-11-04")
y <- ymd("2008/9/1")
subtraction <- x - y
subtraction
```

```
## Time difference of 64 days
```

```
class(subtraction)
```

```
## [1] "difftime"
```

```
as.numeric(subtraction)
```

```
## [1] 64
```

```
## the election date
election_date <- ymd("2008-11-04")

## add DaysToElection
polls08 <- polls08 %>%
  mutate(DaysToElection = as.numeric(election_date - middate))

poll.pred <- rep(NA, 51) # initialize a vector place holder
## extract unique state names which the loop will iterate through
st.names <- unique(polls08$state)
## add state names as labels for easy interpretation later on
names(poll.pred) <- as.character(st.names)

## loop across 50 states plus DC
for (i in seq_along(st.names)){
    ## subset the ith state
    state.data <- polls08 %>%
      filter(state == st.names[i])

    ## pull out the closest date (minimum days to election)
    min_days <- min(state.data$DaysToElection)

    ## subset only the latest polls within the state
    state.data <- state.data %>%
      filter(DaysToElection == min_days)

    ## compute the mean of latest polls and store it
    poll.pred[i] <- mean(state.data$margin)
}
```

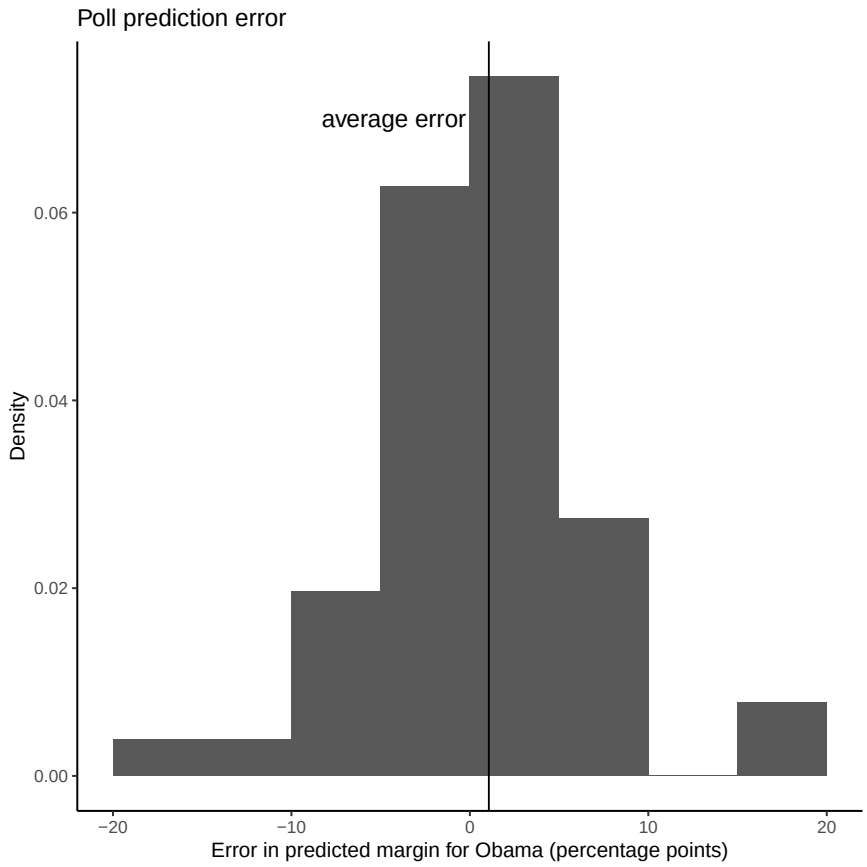
```
# Assuming the states are in the same order in both data:
errors <- pres08$margin - poll.pred
names(errors) <- st.names # add state names
mean(errors) # mean prediction error
```

```
## [1] 1.062092
```

```
sqrt(mean(errors^2))
```

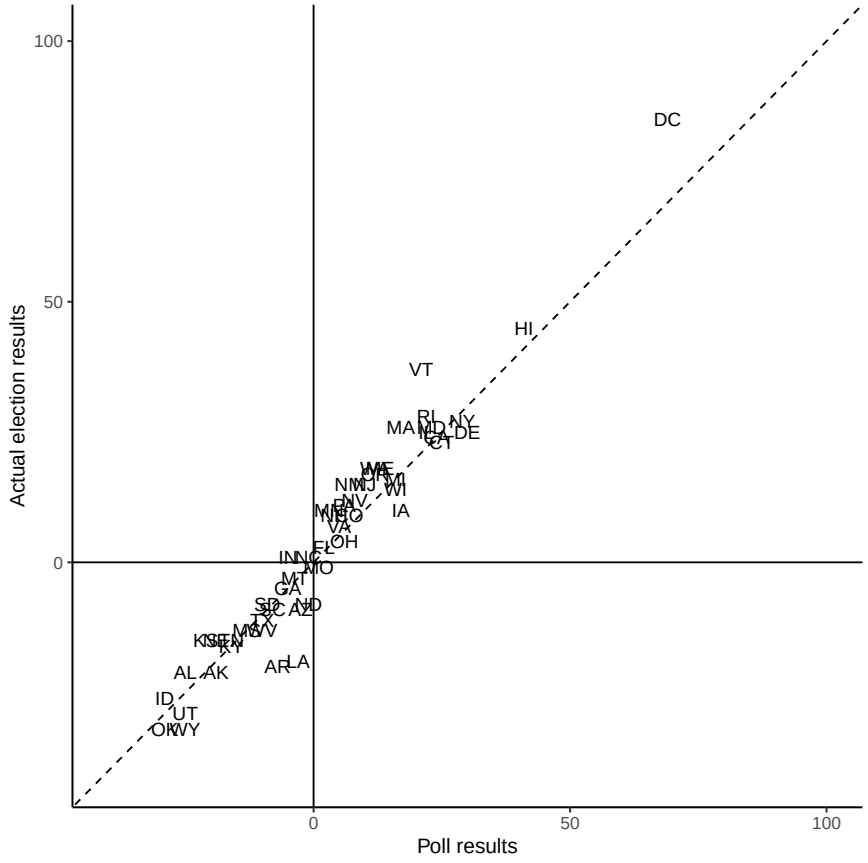
```
## [1] 5.90894
```

```
## Errors as tibble
errors_tib <- as_tibble(errors)

## Plot the histogram
ggplot(errors_tib, aes(x = errors, y = ..density..)) +
  geom_histogram(binwidth = 5, boundary = 0) +
  geom_vline(xintercept = mean(errors_tib$value)) +
  ggplot2::annotate("text", x = -.2,
           y = 0.07, hjust = 1, label = "average error",
           size = 14/.pt) +
  labs(title = "Poll prediction error", y = "Density",
       x="Error in predicted margin for Obama (percentage points)") +
  theme_classic(base_size = 12)
```




```
pres08 <- pres08 %>%
  cbind(poll.pred = poll.pred)

ggplot(data = pres08,
       aes(x = poll.pred, y = margin)) +
    geom_text(aes(label = state)) +
  geom_abline(linetype = "dashed") +
  geom_vline(xintercept = 0) +
  geom_hline(yintercept = 0) +
  ylim(-40, 100) +
  xlim(-40, 100) +
```

```
  labs(x = "Poll results",
       y = "Actual election results")
```




```
pres08 <- pres08 %>%
  mutate(correct = if_else(sign(poll.pred) == sign(margin), 1, 0))

## Which states were miss-called?
filter(pres08, correct == 0) %>%
  select(state.name, Obama, McCain, margin, poll.pred,correct)
```

```
##        state.name Obama McCain margin poll.pred correct
## IN        Indiana    50     49      1        -5       0
## MO       Missouri    48     49     -1         1       0
## NC North Carolina    50     49      1        -1       0
```

```
## actual results: total number of electoral votes won by Obama
pres08 %>%
  filter(margin > 0) %>%
  summarize(total_EV = sum(EV))
```

```
##   total_EV
## 1      364
```

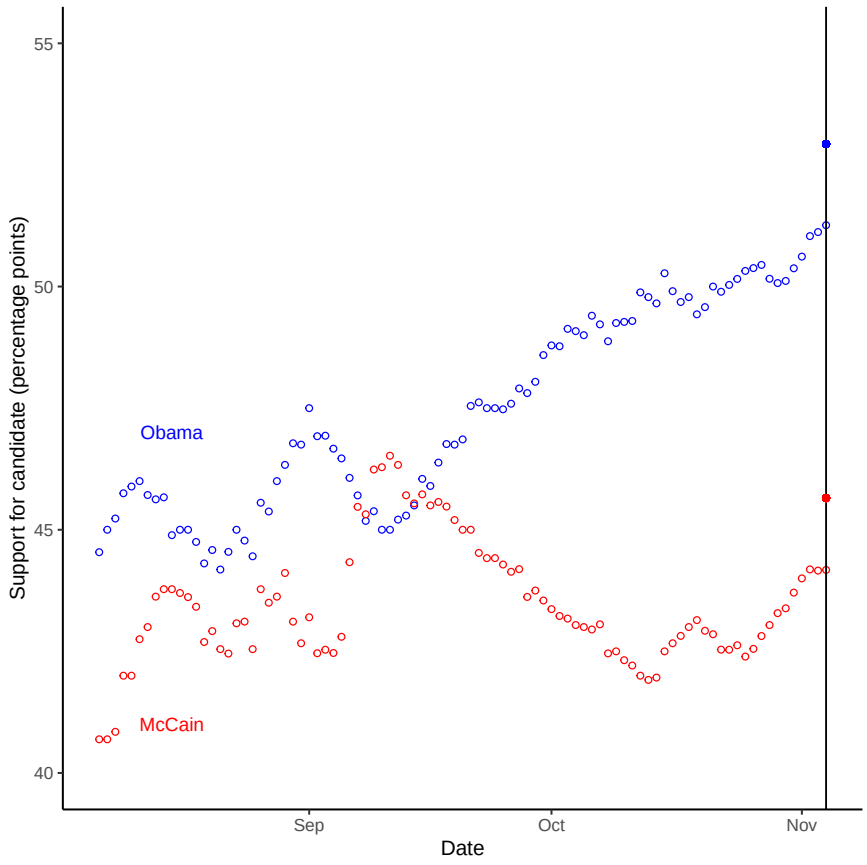
```r
## poll prediction
pres08 %>%
  filter(poll.pred >0) %>%
  summarize(pred_EV = sum(EV))
```

```
##   pred_EV
## 1     349
```

```r
data("pollsUS08", package = "qss")

## What days should we loop over?
## Every day from the earliest poll to the election date
## election_date created earlier
all_dates <- seq(min(pollsUS08$middate), election_date, by = "days")

# How many prior days of polling to use?
prior_days <- 7

## Create an object to hold the loop results
vote_avg <- vector(length(all_dates), mode = "list")

## The loop
for (i in seq_along(all_dates)) {
  date <- all_dates[i]
  # summarize the polls from the prior seven day
  week_data <- filter(pollsUS08,
          # want only days prior to the current loop date
                      as.numeric(date - middate) >= 0,
          # want 7 days prior to the current loop date
                      as.numeric(date - middate) < prior_days) %>%
     summarize(Obama = mean(Obama, na.rm = TRUE),
               McCain = mean(McCain, na.rm = TRUE))
  # add date for the observation
  week_data$date <- date
  # save the data as an item in the results list
  vote_avg[[i]] <- week_data
}

## Convert the list of results to a data frame
vote_avg_df <- bind_rows(vote_avg)
```

```r
## Focus only on last 90 days
vote_avg_df %>%
  filter(election_date - date <= 90) %>%
  ggplot() +
  geom_point(aes(x = date, y = Obama),
             color = "blue", shape = 1) +
  geom_point(aes(x = date, y = McCain),
             color = "red", shape = 1) +
  ylim(40, 55) +
  labs(y = "Support for candidate (percentage points)",
       x = "Date") +
  ggplot2::annotate("text", x = ymd("2008-08-15"),
```

```
          y = 47, label = "Obama", color = "blue") +
  ggplot2::annotate("text", x = ymd("2008-08-15"),
          y = 41, label = "McCain", color = "red") +
  geom_vline(xintercept = election_date) +
  geom_point(aes(x = election_date, y = 52.93), color = "blue") +
  geom_point(aes(x = election_date, y = 45.65), color = "red")
```



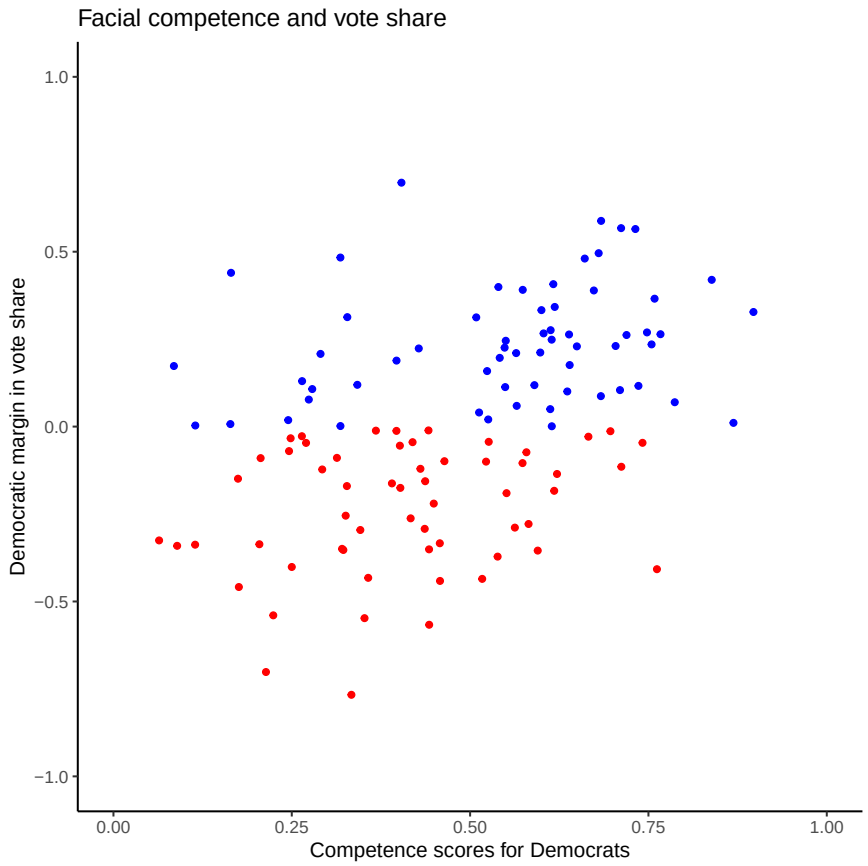
Linear Regression

Facial Appearance and Election Outcomes

```
## load the data
data("face", package = "qss")

## add the shares and differences
face <- mutate(face,
               d.share = d.votes / (d.votes + r.votes),
               r.share = r.votes / (d.votes + r.votes),
               diff.share = d.share - r.share)
```

```
ggplot(face, aes(x = d.comp, y = diff.share, color = w.party)) +
  geom_point() +
```

```
  scale_colour_manual(values = c(D = "blue", R = "red")) +
  labs(x = "Competence scores for Democrats",
       y = "Democratic margin in vote share",
       title = "Facial competence and vote share") +
  ylim(-1, 1) +
  xlim(0, 1) +
  theme(legend.position = "none")
```



Correlation and Scatter Plots

```
cor(face$d.comp, face$diff.share)
```

```
## [1] 0.4327743
```

Least Squares

```
fit <- lm(diff.share ~ d.comp, data = face) # fit the model
fit
```

```
##
```

```
## Call:
## lm(formula = diff.share ~ d.comp, data = face)
##
## Coefficients:
## (Intercept)       d.comp
##     -0.3122       0.6604
```

```
lm(face$diff.share ~ face$d.comp)
```

```
coef(fit) # get estimated coefficients
```

```
## (Intercept)      d.comp
##  -0.3122259   0.6603815
```

```
head(fitted(fit)) # show the first few fitted or predicted values
```

```
##           1           2           3           4           5
##  0.06060411 -0.08643340  0.09217061  0.04539236  0.13698690
##           6
## -0.10057206
```

```
library(tidymodels)
glance(fit)
```

```
## # A tibble: 1 x 12
##   r.squared adj.r.squared sigma statistic     p.value    df
##       <dbl>         <dbl> <dbl>     <dbl>       <dbl> <dbl>
## 1     0.187         0.180 0.266      27.0 0.000000885     1
## # ... with 6 more variables: logLik <dbl>, AIC <dbl>,
## #   BIC <dbl>, deviance <dbl>, df.residual <int>,
## #   nobs <int>
```

```
tidy(fit)
```

```
## # A tibble: 2 x 5
##   term        estimate std.error statistic     p.value
##   <chr>          <dbl>     <dbl>     <dbl>       <dbl>
## 1 (Intercept)   -0.312    0.0660     -4.73 0.00000624
## 2 d.comp         0.660    0.127       5.19 0.000000885
```

```
augment(fit) %>% head()
```

```
## # A tibble: 6 x 8
##   diff.share d.comp .fitted  .resid    .hat .sigma  .cooksd
##        <dbl>  <dbl>   <dbl>   <dbl>   <dbl>  <dbl>    <dbl>
## 1     0.210   0.565  0.0606  0.150  0.00996  0.267 0.00160
## 2     0.119   0.342 -0.0864  0.206  0.0129   0.267 0.00394
## 3     0.0499  0.612  0.0922 -0.0423 0.0123   0.268 0.000158
## 4     0.197   0.542  0.0454  0.151  0.00922  0.267 0.00151
## 5     0.496   0.680  0.137   0.359  0.0174   0.266 0.0163
## 6    -0.350   0.321 -0.101  -0.249  0.0143   0.267 0.00644
## # ... with 1 more variable: .std.resid <dbl>
```

```
ggplot() +
  geom_point(data = face,
             mapping = aes(x = d.comp, y = diff.share), shape = 1) +
  geom_abline(slope = coef(fit)["d.comp"],
              intercept = coef(fit)["(Intercept)"]) +
  scale_y_continuous("Competence scores for Democrats",
                     breaks = seq(-1, 1, by = 0.5), limits = c(-1, 1)) +
  scale_x_continuous("Democratic margin in vote shares",
                     breaks = seq(0, 1, by = 0.2), limits = c(0, 1)) +
  geom_vline(xintercept = mean(face$d.comp),
             linetype = "dashed") +
  geom_hline(yintercept =  mean(face$diff.share),
                linetype = "dashed") +
  ggtitle("Facial competence and vote share")
```

```
ggplot(data = face, mapping = aes(x = d.comp, y = diff.share)) +
  geom_point() +
  geom_smooth(method = "lm", se = FALSE)
```

```
epsilon.hat <- resid(fit)  # residuals
sqrt(mean(epsilon.hat^2))  # RMSE
```

```
## [1] 0.2642361
```

Regression towards the Mean

Merging Data Sets in R

```
data("pres12", package = "qss")
glimpse(pres08)
```

```
## Rows: 51
## Columns: 8
## $ state.name <chr> "Alabama", "Alaska", "Arizona", "Arkans~
## $ state      <chr> "AL", "AK", "AZ", "AR", "CA", "CO", "CT~
## $ Obama      <int> 39, 38, 45, 39, 61, 54, 61, 92, 62, 51,~
## $ McCain     <int> 60, 59, 54, 59, 37, 45, 38, 7, 37, 48, ~
## $ EV         <int> 9, 3, 10, 6, 55, 9, 7, 3, 3, 27, 15, 4,~
## $ margin     <int> -21, -21, -9, -20, 24, 9, 23, 85, 25, 3~
## $ poll.pred  <dbl> -25.0, -19.0, -2.5, -7.0, 24.0, 7.0, 25~
## $ correct    <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ~
```

```
glimpse(pres12)
```

```
## Rows: 51
## Columns: 4
## $ state  <chr> "AL", "AK", "AZ", "AR", "CA", "CO", "CT", "~
## $ Obama  <int> 38, 41, 45, 37, 60, 51, 58, 59, 91, 50, 45,~
## $ Romney <int> 61, 55, 54, 61, 37, 46, 41, 40, 7, 49, 53, ~
## $ EV     <int> 9, 3, 11, 6, 55, 9, 7, 3, 3, 29, 16, 4, 4, ~
```

```r
pres <- full_join(x = pres08, y = pres12, by = "state")
summary(pres)
```

```
##   state.name           state              Obama.x
##  Length:51          Length:51          Min.   :33.00
##  Class :character   Class :character   1st Qu.:43.00
##  Mode  :character   Mode  :character   Median :51.00
##                                        Mean   :51.37
##                                        3rd Qu.:57.50
##                                        Max.   :92.00
##      McCain           EV.x           margin
##  Min.   : 7.00   Min.   : 3.00   Min.   :-32.000
##  1st Qu.:40.00   1st Qu.: 4.50   1st Qu.:-13.000
##  Median :47.00   Median : 8.00   Median :  4.000
##  Mean   :47.06   Mean   :10.55   Mean   :  4.314
##  3rd Qu.:56.00   3rd Qu.:11.50   3rd Qu.: 17.500
##  Max.   :66.00   Max.   :55.00   Max.   : 85.000
##    poll.pred          correct          Obama.y
##  Min.   :-29.000   Min.   :0.0000   Min.   :25.00
##  1st Qu.: -9.500   1st Qu.:1.0000   1st Qu.:40.50
##  Median :  3.000   Median :1.0000   Median :51.00
##  Mean   :  3.252   Mean   :0.9412   Mean   :49.06
##  3rd Qu.: 16.000   3rd Qu.:1.0000   3rd Qu.:56.00
##  Max.   : 69.000   Max.   :1.0000   Max.   :91.00
##      Romney           EV.y
##  Min.   : 7.00   Min.   : 3.00
##  1st Qu.:41.00   1st Qu.: 4.50
##  Median :48.00   Median : 8.00
##  Mean   :49.04   Mean   :10.55
##  3rd Qu.:58.00   3rd Qu.:11.50
##  Max.   :73.00   Max.   :55.00
```

```r
## change the variable name for illustration
pres12 <- rename(pres12, state.abbrev = state)
## merging data sets using the variables of different names
## and specifying the suffix
pres <- full_join(pres08, pres12,
             by = c("state" = "state.abbrev"),
             suffix = c("_08", "_12"))
glimpse(pres)
```

```
## Rows: 51
## Columns: 11
## $ state.name <chr> "Alabama", "Alaska", "Arizona", "Arkans~
## $ state      <chr> "AL", "AK", "AZ", "AR", "CA", "CO", "CT~
## $ Obama_08   <int> 39, 38, 45, 39, 61, 54, 61, 92, 62, 51,~
## $ McCain     <int> 60, 59, 54, 59, 37, 45, 38, 7, 37, 48, ~
## $ EV_08      <int> 9, 3, 10, 6, 55, 9, 7, 3, 3, 27, 15, 4,~
## $ margin     <int> -21, -21, -9, -20, 24, 9, 23, 85, 25, 3~
## $ poll.pred  <dbl> -25.0, -19.0, -2.5, -7.0, 24.0, 7.0, 25~
## $ correct    <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ~
## $ Obama_12   <int> 38, 41, 45, 37, 60, 51, 58, 91, 59, 50,~
```

```
## $ Romney     <int> 61, 55, 54, 61, 37, 46, 41, 7, 40, 49, ~
## $ EV_12      <int> 9, 3, 11, 6, 55, 9, 7, 3, 3, 29, 16, 4,~
```

```
## cbinding two data frames
pres_cbind <- cbind(pres08, pres12)
## DC and DE are flipped
pres_cbind[8:9, ]
```

```
##    state.name state Obama McCain EV margin poll.pred
## DC       D.C.    DC    92      7  3     85        69
## DE   Delaware    DE    62     37  3     25        30
##    correct state.abbrev Obama Romney EV
## DC       1           DE    59     40  3
## DE       1           DC    91      7  3
```

```
## bind_cols two data frames
pres_bind_cols <- bind_cols(pres08, pres12)
## odd variable names
summary(pres_bind_cols)
```

```
##   state.name           state              Obama...3
##  Length:51          Length:51          Min.   :33.00
##  Class :character   Class :character   1st Qu.:43.00
##  Mode  :character   Mode  :character   Median :51.00
##                                        Mean   :51.37
##                                        3rd Qu.:57.50
##                                        Max.   :92.00
##      McCain          EV...5          margin
##  Min.   : 7.00   Min.   : 3.00   Min.   :-32.000
##  1st Qu.:40.00   1st Qu.: 4.50   1st Qu.:-13.000
##  Median :47.00   Median : 8.00   Median :  4.000
##  Mean   :47.06   Mean   :10.55   Mean   :  4.314
##  3rd Qu.:56.00   3rd Qu.:11.50   3rd Qu.: 17.500
##  Max.   :66.00   Max.   :55.00   Max.   : 85.000
##    poll.pred          correct       state.abbrev
##  Min.   :-29.000   Min.   :0.0000   Length:51
##  1st Qu.: -9.500   1st Qu.:1.0000   Class :character
##  Median :  3.000   Median :1.0000   Mode  :character
##  Mean   :  3.252   Mean   :0.9412
##  3rd Qu.: 16.000   3rd Qu.:1.0000
##  Max.   : 69.000   Max.   :1.0000
##    Obama...10        Romney         EV...12
##  Min.   :25.00   Min.   : 7.00   Min.   : 3.00
##  1st Qu.:40.50   1st Qu.:41.00   1st Qu.: 4.50
##  Median :51.00   Median :48.00   Median : 8.00
##  Mean   :49.06   Mean   :49.04   Mean   :10.55
##  3rd Qu.:56.00   3rd Qu.:58.00   3rd Qu.:11.50
##  Max.   :91.00   Max.   :73.00   Max.   :55.00
```

```
## more control with full_join!
```

```r
pres <- pres %>%
  mutate(Obama2008.z = as.numeric(scale(Obama_08)),
         Obama2012.z = as.numeric(scale(Obama_12)))
```

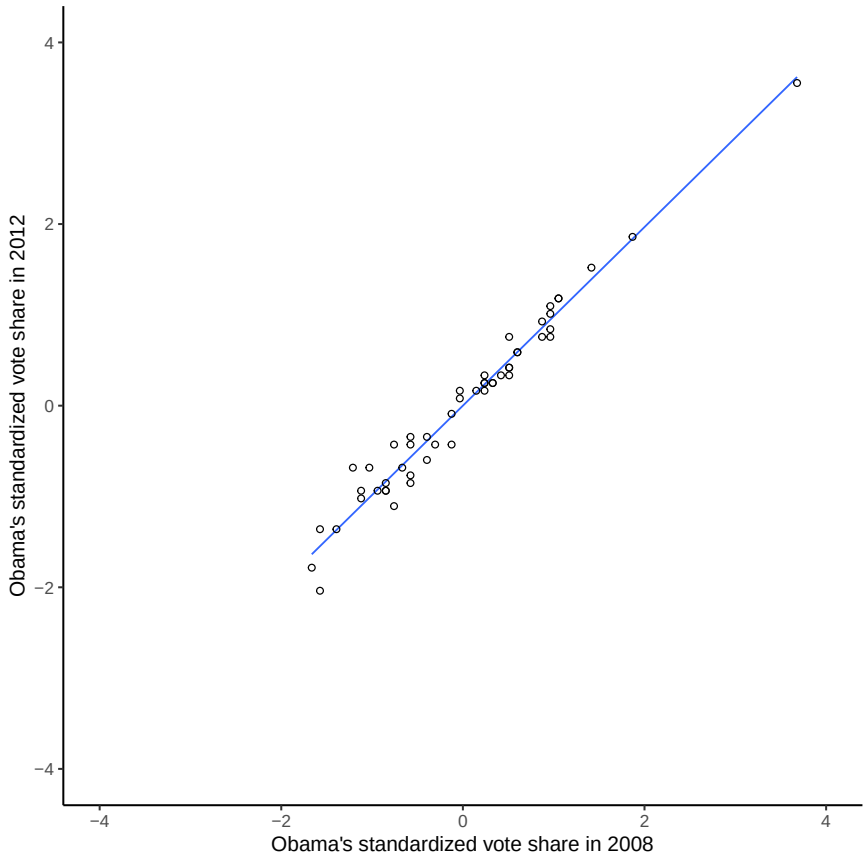
```r
## intercept is estimated essentially zero
fit1 <- lm(Obama2012.z ~ Obama2008.z, data = pres)
fit1
```

```
##
## Call:
## lm(formula = Obama2012.z ~ Obama2008.z, data = pres)
##
## Coefficients:
## (Intercept)  Obama2008.z
##  -5.076e-17    9.834e-01
```

```r
## regression without an intercept; estimated slope is identical
fit1 <- lm(Obama2012.z ~ -1 + Obama2008.z, data = pres)
fit1
```

```
##
## Call:
## lm(formula = Obama2012.z ~ -1 + Obama2008.z, data = pres)
##
## Coefficients:
## Obama2008.z
##      0.9834
```

```r
ggplot(pres, aes(x = Obama2008.z, y = Obama2012.z)) +
  geom_smooth(method = "lm", se=F, size = 0.5) +
  geom_point(shape = 1) +
  coord_fixed() +
  scale_x_continuous("Obama's standardized vote share in 2008",
                     limits = c(-4, 4)) +
  scale_y_continuous("Obama's standardized vote share in 2012",
                     limits = c(-4, 4))
```




```
pres %>%
  filter(Obama2008.z < quantile(Obama2008.z, 0.25)) %>%
  summarize(improve = mean(Obama2012.z > Obama2008.z))
```

```
##     improve
## 1 0.5833333
```

```
pres %>%
  filter(Obama2008.z < quantile(Obama2008.z, 0.75)) %>%
  summarize(improve = mean(Obama2012.z > Obama2008.z))
```

```
##   improve
## 1     0.5
```

Model Fit

```
data("florida", package = "qss")
fit2 <- lm(Buchanan00 ~ Perot96, data = florida)
fit2
```

```
##
## Call:
```

```
## lm(formula = Buchanan00 ~ Perot96, data = florida)
##
## Coefficients:
## (Intercept)      Perot96
##     1.34575      0.03592
```

```
## compute TSS (total sum of squares) and SSR (sum of squared residuals)
TSS2 <- florida %>%
  mutate(diff_sq = (Buchanan00 - mean(florida$Buchanan00))^2) %>%
  summarize(TSS = sum(diff_sq))

SSR2 <- sum(resid(fit2)^2)

## Coefficient of determination
(TSS2 - SSR2) / TSS2
```

```
##         TSS
## 1 0.5130333
```

```
R2 <- function(fit) {
    resid <- resid(fit) # residuals
    y <- fitted(fit) + resid # outcome variable
    TSS <- sum((y - mean(y))^2)
    SSR <- sum(resid^2)
    R2 <- (TSS - SSR) / TSS
    return(R2)
}
R2(fit2)
```

```
## [1] 0.5130333
```

```
## built-in R function
fit2summary <- summary(fit2)
fit2summary$r.squared
```

```
## [1] 0.5130333
```

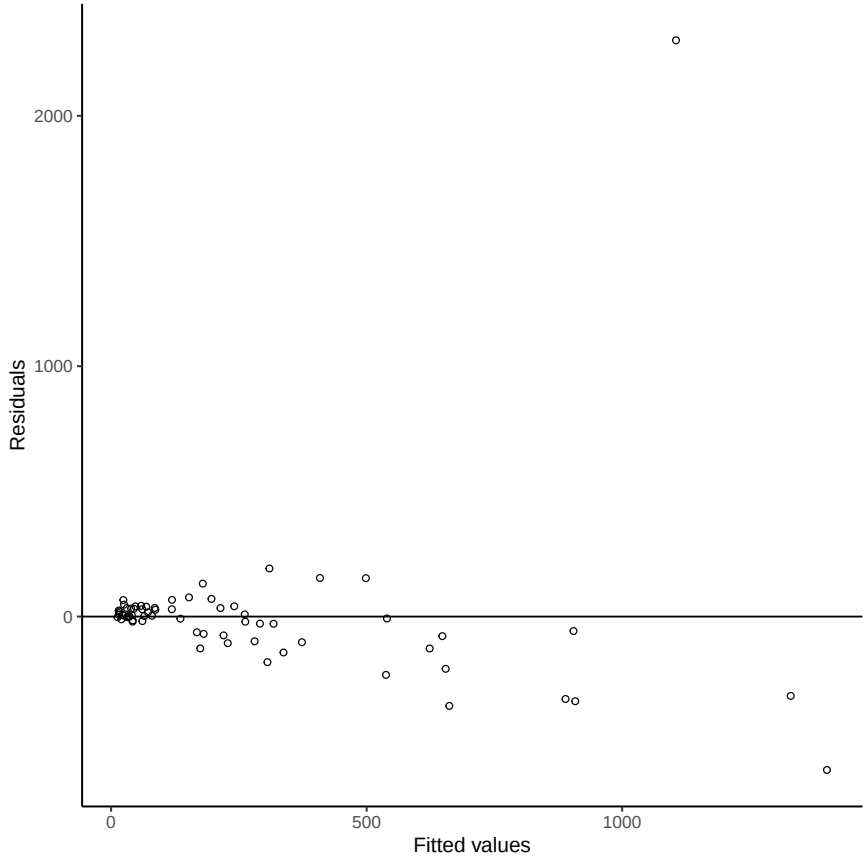
```
## with broom function
glance(fit2)
```

```
## # A tibble: 1 x 12
##   r.squared adj.r.squared sigma statistic  p.value    df
##       <dbl>         <dbl> <dbl>     <dbl>    <dbl> <dbl>
## 1     0.513         0.506  316.      68.5 9.47e-12     1
## # ... with 6 more variables: logLik <dbl>, AIC <dbl>,
## #   BIC <dbl>, deviance <dbl>, df.residual <int>,
## #   nobs <int>
```

```
R2(fit1)
```

```
## [1] 0.9671579
```

```
augment_fit2 <- augment(fit2)

ggplot(augment_fit2, aes(x = .fitted, y = .resid)) +
  geom_point(shape = 1) +
  geom_hline(yintercept = 0) +
  labs(x = "Fitted values", y = "Residuals")
```




```
library(modelr)

florida_fit2 <- florida %>%
  add_predictions(fit2) %>%
  add_residuals(fit2)

filter(florida_fit2, resid == max(resid)) %>%
  select(county) %>%
  pull()
```

```
## [1] "PalmBeach"
```

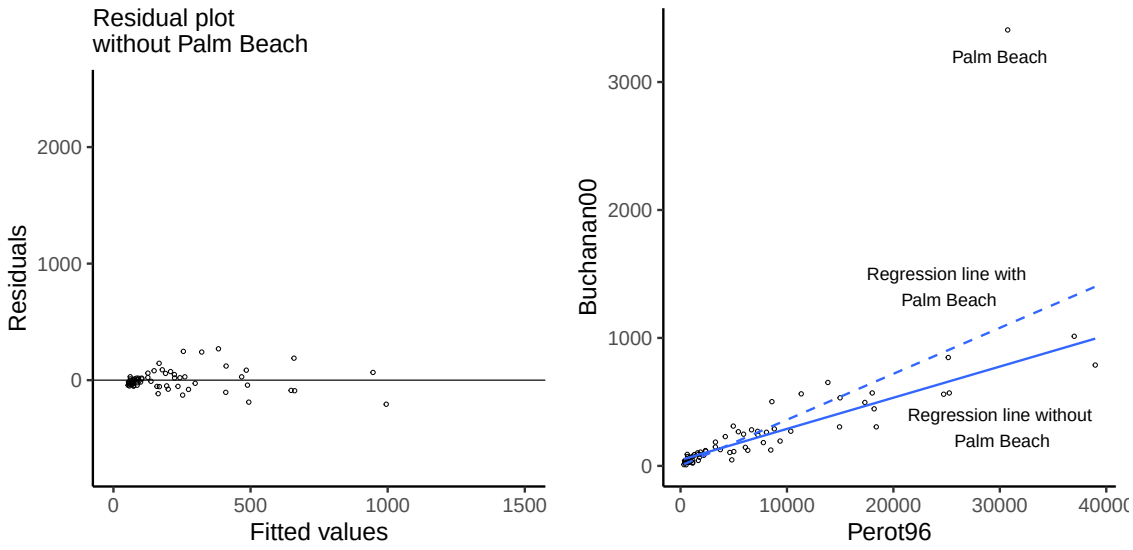
```
## data without Palm Beach
florida.pb <- filter(florida, county != "PalmBeach")
fit3 <- lm(Buchanan00 ~ Perot96, data = florida.pb)
fit3
```

```
##
## Call:
## lm(formula = Buchanan00 ~ Perot96, data = florida.pb)
##
## Coefficients:
## (Intercept)      Perot96
##    45.84193      0.02435
```

```
## R^2 or coefficient of determination
R2(fit3)
```

```
## [1] 0.8511675
```

```
## Residual plot
florida.pb %>%
  add_residuals(fit3) %>%
  add_predictions(fit3) %>%
  ggplot(aes(x = pred, y = resid)) +
  geom_hline(yintercept = 0) +
  geom_point(shape = 1) +
  ylim(-750, 2500) +
  xlim(0, 1500) +
  labs(x = "Fitted values", y = "Residuals",
       title = "Residual plot \nwithout Palm Beach") +
  theme(plot.title = element_text(size = 22))

## Scatter plot with regression lines
ggplot() +
  geom_point(data = florida, aes(x = Perot96, y = Buchanan00),
             shape = 1) +
  geom_smooth(data = florida, aes(x = Perot96, y = Buchanan00),
              method = 'lm', se = FALSE, linetype = "dashed") +
  geom_smooth(data = florida.pb, aes(x = Perot96, y = Buchanan00),
              method = 'lm', se = FALSE) +
  ggplot2::annotate("text", x = 30000, y = 3200, label = "Palm Beach",
           size = 16/.pt)+
  ggplot2::annotate("text", x = 30000, y = 300,
                    label = "Regression line without\n Palm Beach",
           size = 16/.pt)+
  ggplot2::annotate("text", x = 25000, y = 1400,
                    label = "Regression line with\n Palm Beach",
           size = 16/.pt)
```



Regression and Causation

Randomized Experiments

```
data("women", package = "qss")
women %>%
  group_by(reserved) %>%
  summarize(prop_female = mean(female))
```

```
## # A tibble: 2 x 2
##   reserved prop_female
##      <int>       <dbl>
## 1        0      0.0748
## 2        1      1
```

```
women %>%
  group_by(reserved) %>%
  summarize(irrigation = mean(irrigation),
            water = mean(water)) %>%
  pivot_longer(names_to = "variable", - reserved) %>%
  pivot_wider(names_from = reserved) %>%
  rename("not_reserved" = `0`,
         "reserved" = `1` ) %>%
  mutate(diff = reserved - not_reserved)
```

```
## # A tibble: 2 x 4
##   variable   not_reserved reserved   diff
##   <chr>             <dbl>    <dbl>  <dbl>
## 1 irrigation         3.39     3.02 -0.369
## 2 water             14.7     24.0   9.25
```

```
lm(water ~ reserved, data = women)
```

```
##
## Call:
## lm(formula = water ~ reserved, data = women)
##
## Coefficients:
## (Intercept)     reserved
##      14.738        9.252
```

```
lm(irrigation ~ reserved, data = women)
```

```
##
## Call:
## lm(formula = irrigation ~ reserved, data = women)
##
## Coefficients:
## (Intercept)     reserved
##      3.3879      -0.3693
```

Regression with Multiple Predictors

```
data("social", package = "qss")
unique(social$messages)
```

```
## [1] "Civic Duty" "Hawthorne"  "Control"    "Neighbors"
```

```
fit <- lm(primary2006 ~ messages, data = social)
fit # the Civic message is the reference category
```

```
##
## Call:
## lm(formula = primary2006 ~ messages, data = social)
##
## Coefficients:
##       (Intercept)    messagesControl  messagesHawthorne
##          0.314538          -0.017899           0.007837
## messagesNeighbors
##          0.063411
```

```
## create indicator variables
social <- social %>%
  mutate(Control = if_else(messages == "Control", 1, 0),
         Hawthorne = if_else(messages == "Hawthorne", 1, 0),
         Neighbors = if_else(messages == "Neighbors", 1, 0))
## fit the same regression as above by directly using indicator variables
lm(primary2006 ~ Control + Hawthorne + Neighbors, data = social)
```

```
##
## Call:
## lm(formula = primary2006 ~ Control + Hawthorne + Neighbors, data = social)
##
## Coefficients:
## (Intercept)      Control    Hawthorne    Neighbors
##    0.314538    -0.017899     0.007837     0.063411
```

```
unique_messages <-
  data_grid(social, messages) %>% #What does this create?
  add_predictions(fit)
unique_messages
```

```
## # A tibble: 4 x 2
##   messages    pred
##   <chr>      <dbl>
## 1 Civic Duty 0.315
## 2 Control    0.297
## 3 Hawthorne  0.322
## 4 Neighbors  0.378
```

```
social %>%
  group_by(messages) %>%
  summarize(mean(primary2006))
```

```
## # A tibble: 4 x 2
##   messages   `mean(primary2006)`
##   <chr>                    <dbl>
## 1 Civic Duty               0.315
## 2 Control                  0.297
## 3 Hawthorne                0.322
## 4 Neighbors                0.378
```

```
fit.noint <- lm(primary2006 ~ -1 + messages, data = social)
fit.noint
```

```
##
## Call:
## lm(formula = primary2006 ~ -1 + messages, data = social)
##
## Coefficients:
## messagesCivic Duty     messagesControl   messagesHawthorne
##             0.3145              0.2966              0.3224
##  messagesNeighbors
##             0.3779
```

```
## difference in means
social %>%
  group_by(messages) %>%
  summarize(primary2006 = mean(primary2006)) %>%
  mutate(Control = primary2006[messages == "Control"],
         diff = primary2006 - Control)
```

```
## # A tibble: 4 x 4
##   messages   primary2006 Control   diff
##   <chr>            <dbl>   <dbl>  <dbl>
## 1 Civic Duty       0.315   0.297 0.0179
## 2 Control          0.297   0.297 0
## 3 Hawthorne        0.322   0.297 0.0257
## 4 Neighbors        0.378   0.297 0.0813
```

```
## an adjusted Rsquare function
adjR2 <- function(fit) {
    resid <- resid(fit) # residuals
    y <- fitted(fit) + resid # outcome
    n <- length(y)
    TSS.adj <- sum((y - mean(y))^2) / (n - 1)
    SSR.adj <- sum(resid^2) / (n - length(coef(fit)))
    R2.adj <- 1 - SSR.adj / TSS.adj
    return(R2.adj)
}
adjR2(fit)
```

```
## [1] 0.003272788
```

```
R2(fit) # unadjusted Rsquare calculation
```

```
## [1] 0.003282564
```

```
fitsummary <- summary(fit)
fitsummary$adj.r.squared
```

```
## [1] 0.003272788
```

Heterogeneous Treatment Effects

```
## average treatment effect (ate) among those who voted in 2004 primary
ate <- social %>%
  group_by(primary2004, messages) %>%
  summarize(primary2006 = mean(primary2006)) %>%
  pivot_wider(names_from = messages,
              values_from = primary2006) %>%
  mutate(ate_Neighbors = Neighbors - Control) %>%
  select(primary2004, Neighbors, Control, ate_Neighbors)
ate
```

```
## # A tibble: 2 x 4
## # Groups:   primary2004 [2]
##   primary2004 Neighbors Control ate_Neighbors
##         <int>     <dbl>   <dbl>         <dbl>
## 1           0     0.306   0.237        0.0693
## 2           1     0.482   0.386        0.0965
```

```r
## ATE for 2004 voters and nonvoters
ate.voter <- filter(ate, primary2004 == 1) %>%
  select(ate_Neighbors) %>% pull()

ate.nonvoter <- filter(ate, primary2004 == 0) %>%
  select(ate_Neighbors) %>% pull()

## Difference in ate based on 2004 voting
ate.voter - ate.nonvoter
```

```
## [1] 0.02722908
```

```r
## subset neighbors and control groups
social.neighbor <- filter(social,
                          messages == "Control"
                          | messages == "Neighbors")

## standard way to generate main and interaction effects
fit.int <- lm(primary2006 ~ primary2004 + messages + primary2004:messages,
              data = social.neighbor)
fit.int
```

```
##
## Call:
## lm(formula = primary2006 ~ primary2004 + messages + primary2004:messages,
##     data = social.neighbor)
##
## Coefficients:
##                   (Intercept)
##                       0.23711
##                   primary2004
##                       0.14870
##             messagesNeighbors
##                       0.06930
## primary2004:messagesNeighbors
##                       0.02723
```

```r
lm(primary2006 ~ primary2004 * messages, data = social.neighbor)
```

```r
## create an age variable (at time of election)
social.neighbor <- social.neighbor %>%
  mutate(age = 2008 - yearofbirth)

summary(social.neighbor$age)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   22.00   43.00   52.00   51.82   61.00  108.00
```

```r
fit.age <- lm(primary2006 ~ age * messages, data = social.neighbor)
fit.age
```

```
##
## Call:
## lm(formula = primary2006 ~ age * messages, data = social.neighbor)
##
## Coefficients:
##           (Intercept)                    age
##             0.0894768              0.0039982
##     messagesNeighbors  age:messagesNeighbors
##             0.0485728              0.0006283
```

```
ate.age <- tidyr::crossing(age = seq(from = 20, to = 80, by = 20),
         messages = c("Neighbors", "Control")) %>%
  add_predictions(fit.age) %>%
  pivot_wider(names_from = messages,
              values_from = pred) %>%
  mutate(diff = Neighbors - Control)
ate.age
```

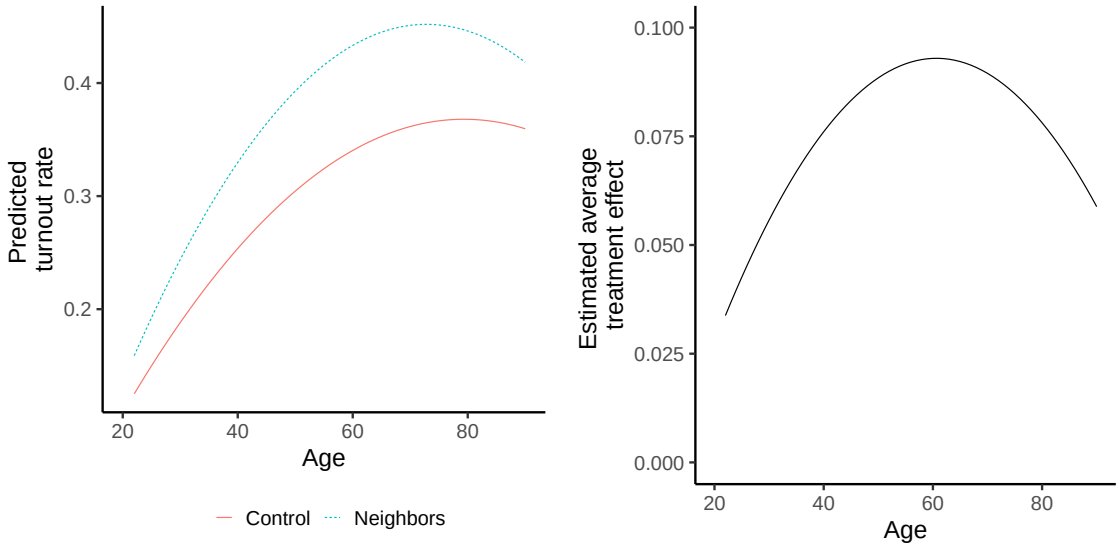
```
## # A tibble: 4 x 4
##     age Control Neighbors   diff
##   <dbl>   <dbl>     <dbl>  <dbl>
## 1    20   0.169     0.231 0.0611
## 2    40   0.249     0.323 0.0737
## 3    60   0.329     0.416 0.0863
## 4    80   0.409     0.508 0.0988
```

```
fit.age2 <- lm(primary2006 ~ age + I(age^2) + messages +
                 age:messages + I(age^2):messages, data = social.neighbor)
fit.age2
```

```
##
## Call:
## lm(formula = primary2006 ~ age + I(age^2) + messages + age:messages +
##     I(age^2):messages, data = social.neighbor)
##
## Coefficients:
##                (Intercept)                         age
##                 -9.700e-02                   1.172e-02
##                   I(age^2)           messagesNeighbors
##                 -7.389e-05                  -5.275e-02
##      age:messagesNeighbors  I(age^2):messagesNeighbors
##                  4.804e-03                  -3.961e-05
```

```
## predicted turnout rate under the two conditions
## and many ages
y.hat <- data_grid(social.neighbor, age, messages) %>%
  add_predictions(fit.age2)

## the ATE
social.neighbor.ate <- y.hat %>%
  pivot_wider(names_from = messages,
              values_from = pred) %>%
  mutate(ate = Neighbors - Control)
```

```
## Plot the predictions
ggplot(y.hat, aes(x = age, y = pred)) +
  geom_line(aes(linetype = messages,
                color = messages)) +
  labs(color = "",
       linetype = "", y = "Predicted \nturnout rate",
       x = "Age") +
  xlim(20, 90) +
  theme(legend.position = "bottom")

# Plot the ATE
ggplot(social.neighbor.ate, aes(x = age, y = ate)) +
  geom_line() +
  labs(y = "Estimated average \ntreatment effect",
       x = "Age") +
  xlim(20, 90) +
  ylim(0, 0.1)
```



Regression Discontinuity Design

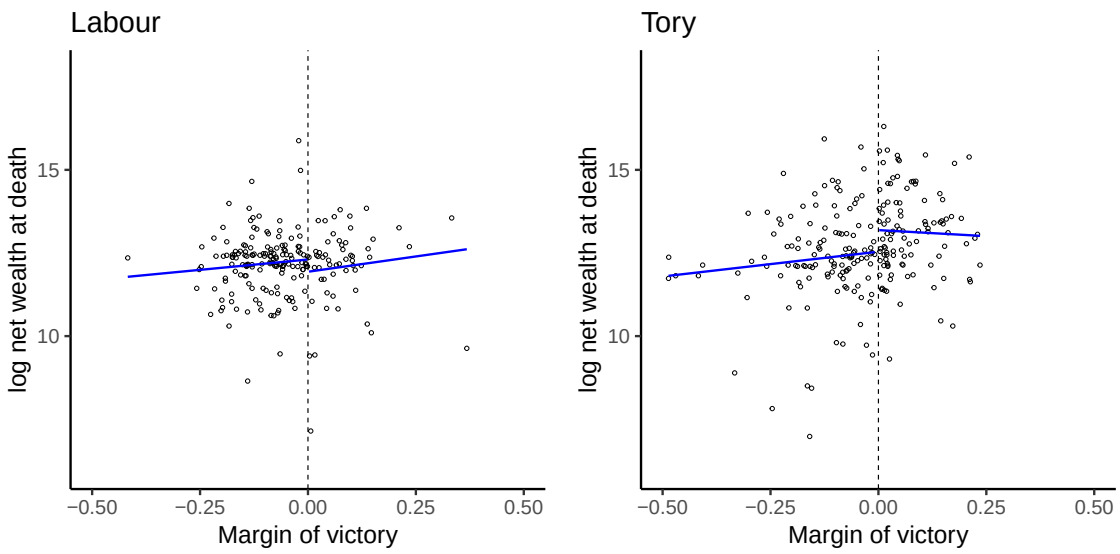
```
## load the data
data("MPs", package = "qss")

## Subset the data
labour_winners <- filter(MPs, party == "labour", margin > 0)
labour_losers <- filter(MPs, party == "labour", margin < 0)
tory_winners <- filter(MPs, party == "tory", margin > 0)
tory_losers <- filter(MPs, party == "tory", margin < 0)

### the the regressions
labour_fit_win <- lm(ln.net ~ margin, data = labour_winners)
labour_fit_lose <- lm(ln.net ~ margin, data = labour_losers)
```

```r
tory_fit_win <- lm(ln.net ~ margin, data = tory_winners)
tory_fit_lose <- lm(ln.net ~ margin, data = tory_losers)
```

```r
y1_labour_win <- labour_winners %>%
  data_grid(margin) %>%
  add_predictions(labour_fit_win)

y2_labour_lose <- labour_losers %>%
  data_grid(margin) %>%
  add_predictions(labour_fit_lose)

y1_tory_win <- tory_winners %>%
  data_grid(margin) %>%
  add_predictions(tory_fit_win)

y2_tory_lose <- tory_losers %>%
  data_grid(margin) %>%
  add_predictions(tory_fit_lose)
```

```r
## Labour
ggplot() +
  geom_point(data = labour_winners,
             mapping = aes(x = margin, y = ln.net), shape = 1) +
  geom_point(data = labour_losers,
             mapping = aes(x = margin, y = ln.net), shape = 1) +
  geom_line(data = y1_labour_win,
            mapping = aes(x = margin, y = pred),
            color = "blue", size = 1) +
  geom_line(data = y2_labour_lose,
            mapping = aes(x = margin, y = pred),
            color = "blue", size = 1) +
  geom_vline(xintercept = 0,
             linetype = "dashed") +
  labs(x = "Margin of victory",
       y = "log net wealth at death",
       title = "Labour") +
  xlim(-0.5, 0.5) +
  ylim(6, 18)

## Tory
ggplot() +
  geom_point(data = tory_winners,
             mapping = aes(x = margin, y = ln.net), shape = 1) +
  geom_point(data = tory_losers,
             mapping = aes(x = margin, y = ln.net), shape = 1) +
  geom_line(data = y1_tory_win,
            mapping = aes(x = margin, y = pred),
            color = "blue", size = 1) +
  geom_line(data = y2_tory_lose,
            mapping = aes(x = margin, y = pred),
            color = "blue", size = 1) +
  geom_vline(xintercept = 0,
             linetype = "dashed") +
```

```
  labs(x = "Margin of victory",
       y = "log net wealth at death",
       title = "Tory") +
  xlim(-0.5, 0.5) +
  ylim(6, 18)
```




```
spread_predictions(tibble(margin = 0),
                   tory_fit_win, tory_fit_lose) %>%
  mutate(rd_est = exp(tory_fit_win) - exp(tory_fit_lose)) %>%
  select(rd_est) %>%
  pull()
```

```
##        1
## 255050.9
```

```
## two regressions for Tory: negative and positive margin
tory_fit_win_placebo <- lm(margin.pre ~ margin, data = tory_winners)
tory_fit_lose_placebo <- lm(margin.pre ~ margin, data = tory_losers)

## the difference between two intercepts is the estimated effect
win_intercept <- tidy(tory_fit_win_placebo) %>%
  filter(term == "(Intercept)") %>%
  select(estimate) %>%
  pull()

lose_intercept <- tidy(tory_fit_lose_placebo) %>%
  filter(term == "(Intercept)") %>%
  select(estimate) %>%
  pull()

win_intercept - lose_intercept
```

```
## [1] -0.01725578
```