Gov 50: 10. Election Prediction

Matthew Blackwell

Harvard University

Fall 2018

1. Today's agenda
2. Predicting presidential elections
3. Loops
4. Evaluating the predictions

1/ Today's agenda

Logistics

- Great job on HW2 + midterm!!

Logistics

- Great job on HW2 + midterm!!
- Mid-semseter evaluation:

Logistics

- Great job on HW2 + midterm!!
- Mid-semseter evaluation:
  - Going live today.

Logistics

- Great job on HW2 + midterm!!
- Mid-semseter evaluation:
  - Going live today.
  - Important to get your feedback on how the course is going.

Logistics

- Great job on HW2 + midterm!!
- Mid-semseter evaluation:
  - Going live today.
  - Important to get your feedback on how the course is going.
  - Will discuss results next week.

Logistics

- Great job on HW2 + midterm!!
- Mid-semseter evaluation:
  - Going live today.
  - Important to get your feedback on how the course is going.
  - Will discuss results next week.
- Govt department climate survey

Logistics

- Great job on HW2 + midterm!!
- Mid-semseter evaluation:
  - Going live today.
  - Important to get your feedback on how the course is going.
  - Will discuss results next week.
- Govt department climate survey
  - Gov concentrators will receive email from "Harvard College Institutional Research."

Logistics

- Great job on HW2 + midterm!!
- Mid-semseter evaluation:
  - Going live today.
  - Important to get your feedback on how the course is going.
  - Will discuss results next week.
- Govt department climate survey
  - Gov concentrators will receive email from "Harvard College Institutional Research."
  - 10 minute survey about your experiences with the department.

Logistics

- Great job on HW2 + midterm!!
- Mid-semseter evaluation:
  - Going live today.
  - Important to get your feedback on how the course is going.
  - Will discuss results next week.
- Govt department climate survey
  - Gov concentrators will receive email from "Harvard College Institutional Research."
  - 10 minute survey about your experiences with the department.
  - Please help us lower the non-response bias!

Where are we? Where are going?

- Up to now: two uses for statistics in research

Where are we? Where are going?

- Up to now: two uses for statistics in research
  - Causality: how one thing affects another

Where are we? Where are going?

- Up to now: two uses for statistics in research
  - Causality: how one thing affects another
  - Measurement: amorphous concept $\rightsquigarrow$ data

Where are we? Where are going?

- Up to now: two uses for statistics in research
  - Causality: how one thing affects another
  - Measurement: amorphous concept $\rightsquigarrow$ data
- Now: third use of statistics

Where are we? Where are going?

- Up to now: two uses for statistics in research
  - Causality: how one thing affects another
  - Measurement: amorphous concept $\rightsquigarrow$ data
- Now: third use of statistics
- **Prediction**: making a best guess about unknown quantity using data.

Where are we? Where are going?

- Up to now: two uses for statistics in research
  - Causality: how one thing affects another
  - Measurement: amorphous concept $\rightsquigarrow$ data
- Now: third use of statistics
- **Prediction**: making a best guess about unknown quantity using data.
- Today: how to make and evaluate predictions.

Where are we? Where are going?

- Up to now: two uses for statistics in research
  - Causality: how one thing affects another
  - Measurement: amorphous concept $\rightsquigarrow$ data
- Now: third use of statistics
- **Prediction**: making a best guess about unknown quantity using data.
- Today: how to make and evaluate predictions.
  - prediction error, bias, (mis)classification

Where are we? Where are going?

- Up to now: two uses for statistics in research
  - Causality: how one thing affects another
  - Measurement: amorphous concept $\rightsquigarrow$ data
- Now: third use of statistics
- **Prediction**: making a best guess about unknown quantity using data.
- Today: how to make and evaluate predictions.
  - prediction error, bias, (mis)classification
- Context: predicting US presidential election results.

Where are we? Where are going?

- Up to now: two uses for statistics in research
  - Causality: how one thing affects another
  - Measurement: amorphous concept $\rightsquigarrow$ data
- Now: third use of statistics
- **Prediction**: making a best guess about unknown quantity using data.
- Today: how to make and evaluate predictions.
  - prediction error, bias, (mis)classification
- Context: predicting US presidential election results.
- R tools: loops for repeated tasks

2/ Predicting presidential elections

Electoral College

- 2016 election popular vote:

Electoral College

- 2016 election popular vote:
  - Clinton: 65,853,516 (48.2%)

Electoral College

- 2016 election popular vote:
  - Clinton: 65,853,516 (48.2%)
  - Trump: 62,984,825 (46.1%)

Electoral College

- 2016 election popular vote:
  - Clinton: 65,853,516 (48.2%)
  - Trump: 62,984,825 (46.1%)
- Why did Trump win? **Electoral college**

Electoral College

- 2016 election popular vote:
  - Clinton: 65,853,516 (48.2%)
  - Trump: 62,984,825 (46.1%)
- Why did Trump win? **Electoral college**
  - Trump: 304, Clinton: 227

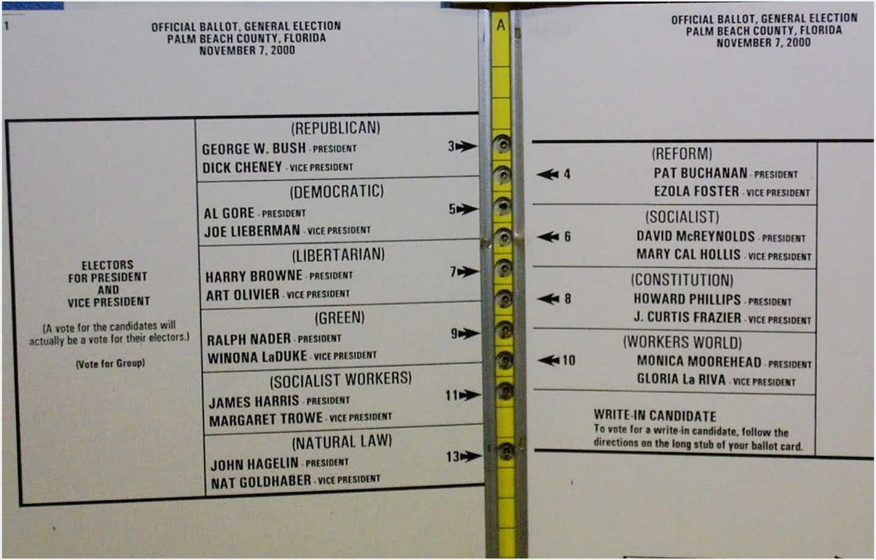
Electoral College

- 2016 election popular vote:
  - Clinton: 65,853,516 (48.2%)
  - Trump: 62,984,825 (46.1%)
- Why did Trump win? **Electoral college**
  - Trump: 304, Clinton: 227
- Election determined by 77,744 votes (margins in WI, MI, and PA)

Electoral College

- 2016 election popular vote:
  - Clinton: 65,853,516 (48.2%)
  - Trump: 62,984,825 (46.1%)
- Why did Trump win? **Electoral college**
  - Trump: 304, Clinton: 227
- Election determined by 77,744 votes (margins in WI, MI, and PA)
  - 0.056% of the electorate (~136 million)

Butterfly ballot



Florida 2000 recount

- National votes: Gore = 50,999,897 vs. Bush = 50,456,002

Florida 2000 recount

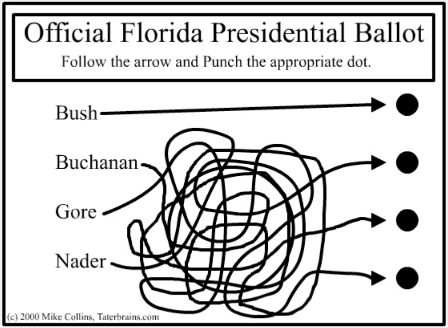
- National votes: Gore = 50,999,897 vs. Bush = 50,456,002
- Margin of victory in Florida: 537 votes (or 0.01% of all FL votes)!

Florida 2000 recount

- National votes: Gore = 50,999,897 vs. Bush = 50,456,002
- Margin of victory in Florida: 537 votes (or 0.01% of all FL votes)!
- Recounts followed by the US Supreme court decision *Bush v. Gore*

Florida 2000 recount

- National votes: Gore = 50,999,897 vs. Bush = 50,456,002
- Margin of victory in Florida: 537 votes (or 0.01% of all FL votes)!
- Recounts followed by the US Supreme court decision *Bush v. Gore*



Predicting US Presidential Elections

- **Electoral college system**

Predicting US Presidential Elections

- **Electoral college system**
  - Must win an absolute majority of 538 electoral votes

Predicting US Presidential Elections

- **Electoral college system**
  - Must win an absolute majority of 538 electoral votes
  - 538 = 435 (House of Representatives) + 100 (Senators) + 3 (DC)

Predicting US Presidential Elections

- **Electoral college system**
  - Must win an absolute majority of 538 electoral votes
  - 538 = 435 (House of Representatives) + 100 (Senators) + 3 (DC)
  - Must win at least 270 votes

Predicting US Presidential Elections

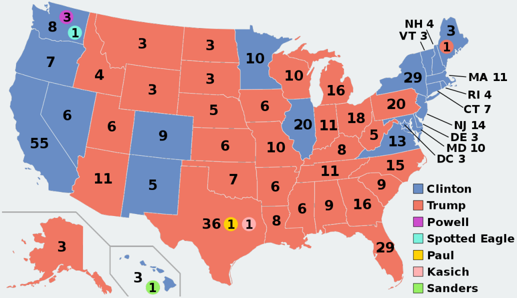
- **Electoral college system**
  - Must win an absolute majority of 538 electoral votes
  - 538 = 435 (House of Representatives) + 100 (Senators) + 3 (DC)
  - Must win at least 270 votes
  - nobody wins an absolute majority $\rightsquigarrow$ House vote

Predicting US Presidential Elections

- **Electoral college system**
  - Must win an absolute majority of 538 electoral votes
  - 538 = 435 (House of Representatives) + 100 (Senators) + 3 (DC)
  - Must win at least 270 votes
  - nobody wins an absolute majority $\rightsquigarrow$ House vote
- Must predict winner of each state

Predicting US Presidential Elections

- **Electoral college system**
  - Must win an absolute majority of 538 electoral votes
  - 538 = 435 (House of Representatives) + 100 (Senators) + 3 (DC)
  - Must win at least 270 votes
  - nobody wins an absolute majority $\rightsquigarrow$ House vote
- Must predict winner of each state



Prediction strategy

- Predict state-level support for each candidate using polls

Prediction strategy

- Predict state-level support for each candidate using polls
- Allocate electoral college votes of that state to its predicted winner

Prediction strategy

- Predict state-level support for each candidate using polls
- Allocate electoral college votes of that state to its predicted winner
- Aggregate EC votes across states to determine the predicted winner

Prediction strategy

- Predict state-level support for each candidate using polls
- Allocate electoral college votes of that state to its predicted winner
- Aggregate EC votes across states to determine the predicted winner
- Coding strategy:

Prediction strategy

- Predict state-level support for each candidate using polls
- Allocate electoral college votes of that state to its predicted winner
- Aggregate EC votes across states to determine the predicted winner
- Coding strategy:
    1. For each state, subset to polls within that state.

Prediction strategy

- Predict state-level support for each candidate using polls
- Allocate electoral college votes of that state to its predicted winner
- Aggregate EC votes across states to determine the predicted winner
- Coding strategy:
    1. For each state, subset to polls within that state.
    2. Further subset the latest polls

Prediction strategy

- Predict state-level support for each candidate using polls
- Allocate electoral college votes of that state to its predicted winner
- Aggregate EC votes across states to determine the predicted winner
- Coding strategy:
    1. For each state, subset to polls within that state.
    2. Further subset the latest polls
    3. Average the latest polls to estimate support for each candidate

Prediction strategy

- Predict state-level support for each candidate using polls
- Allocate electoral college votes of that state to its predicted winner
- Aggregate EC votes across states to determine the predicted winner
- Coding strategy:
    1. For each state, subset to polls within that state.
    2. Further subset the latest polls
    3. Average the latest polls to estimate support for each candidate
    4. Allocate the electoral votes to the candidate who has greatest support

Prediction strategy

- Predict state-level support for each candidate using polls
- Allocate electoral college votes of that state to its predicted winner
- Aggregate EC votes across states to determine the predicted winner
- Coding strategy:
    1. For each state, subset to polls within that state.
    2. Further subset the latest polls
    3. Average the latest polls to estimate support for each candidate
    4. Allocate the electoral votes to the candidate who has greatest support
    5. Repeat this for all states and aggregate the electoral votes

Prediction strategy

- Predict state-level support for each candidate using polls
- Allocate electoral college votes of that state to its predicted winner
- Aggregate EC votes across states to determine the predicted winner
- Coding strategy:
    1. For each state, subset to polls within that state.
    2. Further subset the latest polls
    3. Average the latest polls to estimate support for each candidate
    4. Allocate the electoral votes to the candidate who has greatest support
    5. Repeat this for all states and aggregate the electoral votes
- Sounds like a lot of subsets, ugh...

3/ Loops

Multiplication

```r
values <- c(2, 4, 6)
```

- Let’s say you want to create a new variable that multiplies each value in a vector by 2.

Multiplication

```r
values <- c(2, 4, 6)
```

- Let’s say you want to create a new variable that multiplies each value in a vector by 2.
  - Easy in R: `values * 2`

Multiplication

```r
values <- c(2, 4, 6)
```

- Let's say you want to create a new variable that multiplies each value in a vector by 2.
  - Easy in R: `values * 2`
  - Pretend you didn't know this approach

Manually changing values

```r
values <- c(2, 4, 6)

## number of values
n <- length(values)

## create container to hold results
results <- rep(NA, times = n)

## multiply each value by 2
results[1] <- values[1] * 2
results[2] <- values[2] * 2
results[3] <- values[3] * 2

## print results
results
```

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

Loops in R

- Basic structure:

```
for (i in X) {
  expression1
  expression2
  ...
  expression3
}
```

Loops in R

- Basic structure:

```
for (i in X) {
  expression1
  expression2
  ...
  expression3
}
```

- Elements of a loop:

Loops in R

- Basic structure:

```
for (i in X) {
  expression1
  expression2
  ...
  expression3
}
```

- Elements of a loop:
  1. `i`: counter (can use any name)

Loops in R

- Basic structure:

```
for (i in X) {
  expression1
  expression2
  ...
  expression3
}
```

- Elements of a loop:
    1. `i`: counter (can use any name)
    2. `X`: vector containing a set of ordered values the counter takes.

Loops in R

- Basic structure:

```
for (i in X) {
  expression1
  expression2
  ...
  expression3
}
```

- Elements of a loop:
  1. `i`: counter (can use any name)
  2. `X`: vector containing a set of ordered values the counter takes.
  3. `expression`: a set of expressions that will be repeatedly evaluated.

Loops in R

- Basic structure:

```
for (i in X) {
  expression1
  expression2
  ...
  expression3
}
```

- Elements of a loop:
  1. `i`: counter (can use any name)
  2. `X`: vector containing a set of ordered values the counter takes.
  3. `expression`: a set of expressions that will be repeatedly evaluated.
  4. `{ }`: curly braces to define beginning and end of the loop.

Loops in R

- Basic structure:

```
for (i in X) {
  expression1
  expression2
  ...
  expression3
}
```

- Elements of a loop:
  1. `i`: counter (can use any name)
  2. `X`: vector containing a set of ordered values the counter takes.
  3. `expression`: a set of expressions that will be repeatedly evaluated.
  4. `{ }`: curly braces to define beginning and end of the loop.
- Indentation is important for readability of the code.

Loops in R

- Basic structure:

```
for (i in X) {
  expression1
  expression2
  ...
  expression3
}
```

- Elements of a loop:
  1. `i`: counter (can use any name)
  2. `X`: vector containing a set of ordered values the counter takes.
  3. `expression`: a set of expressions that will be repeatedly evaluated.
  4. `{ }`: curly braces to define beginning and end of the loop.
- Indentation is important for readability of the code.
- Code without loops first by setting counter to specific value.

Loop example

```r
values <- c(2, 4, 6)

## number of values
n <- length(values)

## create container to hold results
results <- rep(NA, n)

## begin loop
for (i in 1:n) {
  results[i] <- values[i] * 2

  ## use cat() to display output
  cat(values[i], "times 2 is equal to ", results[i], "\n")
}
```

```
## 2 times 2 is equal to  4
## 4 times 2 is equal to  8
## 6 times 2 is equal to  12
```

2016 polling prediction

- Election data: `pres16.csv`

2016 polling prediction

- Election data: `pres16.csv`

Name	Description
`state`	abbreviated name of state
`state.name`	unabbreviated name of state
`clinton`	Clinton’s vote share (percentage)
`trump`	Trump’s vote share (percentage)
`ev`	number of electoral college votes for the state

2016 polling prediction

- Election data: `pres16.csv`

Name	Description
`state`	abbreviated name of state
`state.name`	unabbreviated name of state
`clinton`	Clinton's vote share (percentage)
`trump`	Trump's vote share (percentage)
`ev`	number of electoral college votes for the state

- Polling data `polls16.csv`

2016 polling prediction

- Election data: `pres16.csv`

Name	Description
`state`	abbreviated name of state
`state.name`	unabbreviated name of state
`clinton`	Clinton's vote share (percentage)
`trump`	Trump's vote share (percentage)
`ev`	number of electoral college votes for the state

- Polling data `polls16.csv`

Name	Description
`state`	abbreviated name of state in which poll was conducted
`middate`	middate of the period when poll was conducted
`daysleft`	number of days between middate and election day
`pollster`	name of organization conducting poll
`clinton`	predicted support for Obama (percentage)
`trump`	predicted support for McCain (percentage)

Some preprocessing

```r
# election results by state
pres16 <- read.csv("data/pres16.csv")

# polling data
polls16 <- read.csv("data/polls16.csv")

# calculate Trump's margin of victory
polls16$margin <- polls16$trump - polls16$clinton
pres16$margin <- pres16$trump - pres16$clinton
```

What does the data look like?

```r
head(polls16)
```

```
##   state  middate daysleft                 pollster
## 1    AK  8/11/16       89  Lake Research Partners
## 2    AK  8/20/16       80             SurveyMonkey
## 3    AK 10/20/16       19                   YouGov
## 4    AK 10/26/16       13 Google Consumer Surveys
## 5    AK  9/30/16       39 Google Consumer Surveys
## 6    AK 10/12/16       27 Google Consumer Surveys
##   clinton trump margin
## 1    30.0  38.0   8.00
## 2    31.0  38.0   7.00
## 3    37.4  37.7   0.30
## 4    38.0  39.0   1.00
## 5    47.5  36.7 -10.76
## 6    34.6  30.0  -4.62
```

Poll prediction for each state

```r
poll.pred <- rep(NA, 51) # place holder
```

Poll prediction for each state

```
poll.pred <- rep(NA, 51) # place holder

# get list of unique state names to iterate over
st.names <- unique(polls16$state)
```

Poll prediction for each state

```r
poll.pred <- rep(NA, 51) # place holder

# get list of unique state names to iterate over
st.names <- unique(polls16$state)

# add labels to holder
names(poll.pred) <- st.names
```

Poll prediction for each state

```
poll.pred <- rep(NA, 51) # place holder

# get list of unique state names to iterate over
st.names <- unique(polls16$state)

# add labels to holder
names(poll.pred) <- st.names

for (i in 1:51) {
```

Poll prediction for each state

```
poll.pred <- rep(NA, 51) # place holder

# get list of unique state names to iterate over
st.names <- unique(polls16$state)

# add labels to holder
names(poll.pred) <- st.names

for (i in 1:51) {
  state.data <- subset(polls16, subset = (state == st.names[i]))
```

Poll prediction for each state

```
poll.pred <- rep(NA, 51) # place holder

# get list of unique state names to iterate over
st.names <- unique(polls16$state)

# add labels to holder
names(poll.pred) <- st.names

for (i in 1:51) {
  state.data <- subset(polls16, subset = (state == st.names[i]))

  latest <- state.data$daysleft == min(state.data$daysleft)
```

Poll prediction for each state

```r
poll.pred <- rep(NA, 51) # place holder

# get list of unique state names to iterate over
st.names <- unique(polls16$state)

# add labels to holder
names(poll.pred) <- st.names

for (i in 1:51) {
  state.data <- subset(polls16, subset = (state == st.names[i]))

  latest <- state.data$daysleft == min(state.data$daysleft)

  poll.pred[i] <- mean(state.data$margin[latest])
}
```

Poll prediction for each state

```r
poll.pred <- rep(NA, 51) # place holder

# get list of unique state names to iterate over
st.names <- unique(polls16$state)

# add labels to holder
names(poll.pred) <- st.names

for (i in 1:51) {
  state.data <- subset(polls16, subset = (state == st.names[i]))

  latest <- state.data$daysleft == min(state.data$daysleft)

  poll.pred[i] <- mean(state.data$margin[latest])
}

head(poll.pred)
```

```
##     AK     AL     AR     AZ     CA     CO
##  14.73  29.72  20.02   2.50 -23.00  -7.05
```

4/ Evaluating the predictions

Polling errors

- **prediction error** = actual outcome − predicted outcome

Polling errors

- **prediction error** = actual outcome − predicted outcome

```
errors <- pres16$margin - poll.pred
names(errors) <- st.names
```

Polling errors

- **prediction error** = actual outcome − predicted outcome

```r
errors <- pres16$margin - poll.pred
names(errors) <- st.names
```

- **Bias**: average prediction error

Polling errors

- **prediction error** = actual outcome − predicted outcome

```
errors <- pres16$margin - poll.pred
names(errors) <- st.names
```

- **Bias**: average prediction error

```
mean(errors)
```

Polling errors

- **prediction error** = actual outcome − predicted outcome

```
errors <- pres16$margin - poll.pred
names(errors) <- st.names
```

- **Bias**: average prediction error

```
mean(errors)
```

```
## [1] 3.81
```

Polling errors

- **prediction error** $=$ actual outcome $-$ predicted outcome

```
errors <- pres16$margin - poll.pred
names(errors) <- st.names
```

- **Bias**: average prediction error

```
mean(errors)
```

```
## [1] 3.81
```

- **Root mean-square error**: average magnitude of the prediction error

Polling errors

- **prediction error** = actual outcome − predicted outcome

```
errors <- pres16$margin - poll.pred
names(errors) <- st.names
```

- **Bias**: average prediction error

```
mean(errors)
```

```
## [1] 3.81
```

- **Root mean-square error**: average magnitude of the prediction error

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

Polling errors

- **prediction error** = actual outcome − predicted outcome

```
errors <- pres16$margin - poll.pred
names(errors) <- st.names
```

- **Bias**: average prediction error

```
mean(errors)
```

```
## [1] 3.81
```

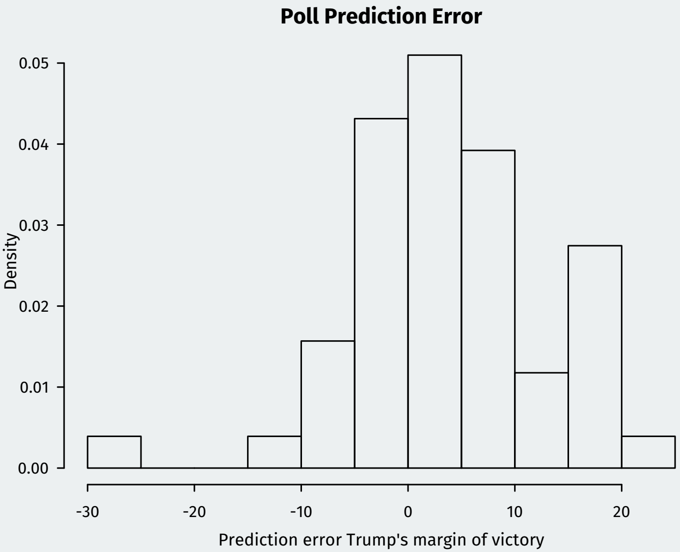
- **Root mean-square error**: average magnitude of the prediction error

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

```
## [1] 9.6
```

Histogram of the errors

```
hist(errors, freq = FALSE, main = ”Poll Prediction Error”,
     xlab = ”Prediction error Trump's margin of victory”)
```

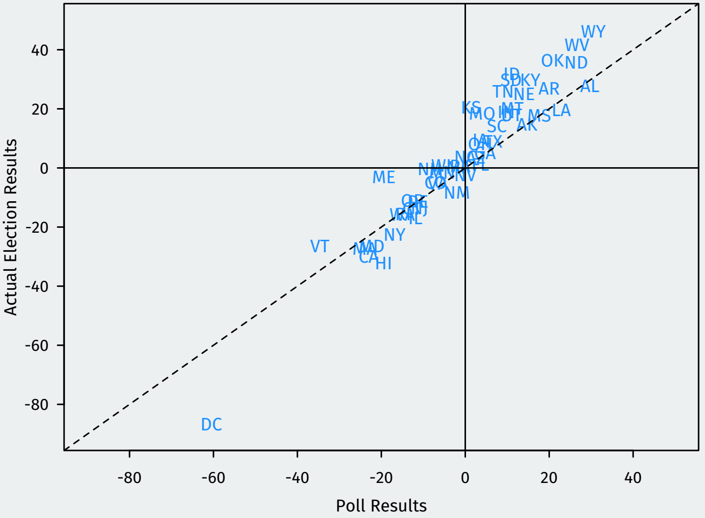

Poll Prediction Error
Density
0.05
0.04
0.03
0.02
0.01
0.00
-30
-20
-10
0
10
20
Prediction error Trump's margin of victory

State-by-state errors

```r
plot(poll.pred, pres16$margin, type = "n", main = "",
     xlim = c(-90, 50), ylim = c(-90, 50),
     xlab = "Poll Results",
     ylab = "Actual Election Results")

text(poll.pred, pres16$margin, pres16$state,
     col = "dodgerblue")

abline(a = 0, b = 1, lty = "dashed") ## 45-degree line
abline(v = 0)
abline(h = 0)
```



Actual Election Results
Poll Results
40
20
0
-20
-40
-60
-80
-80
-60
-40
-20
0
20
40
WY
WV
OK
ND
AR
AL
TN
NE
KS
LA
MS
SC
AK
ME
NM
NY
VT
CA
HI
DC

Classification

- Election prediction: need to predict winner in each state:

Classification

- Election prediction: need to predict winner in each state:

```
sum(pres16$ev[pres16$margin > 0])
```

Classification

- Election prediction: need to predict winner in each state:

```
sum(pres16$ev[pres16$margin > 0])
```

```
## [1] 305
```

Classification

- Election prediction: need to predict winner in each state:

```r
sum(pres16$ev[pres16$margin > 0])
```

```
## [1] 305
```

```r
sum(pres16$ev[poll.pred > 0])
```

Classification

- Election prediction: need to predict winner in each state:

```
sum(pres16$ev[pres16$margin > 0])
```

```
## [1] 305
```

```
sum(pres16$ev[poll.pred > 0])
```

```
## [1] 244
```

Classification

- Election prediction: need to predict winner in each state:

```
sum(pres16$ev[pres16$margin > 0])
```

```
## [1] 305
```

```
sum(pres16$ev[poll.pred > 0])
```

```
## [1] 244
```

- Prediction of binary outcome variable = **classification problem**

Classification

- Election prediction: need to predict winner in each state:

```
sum(pres16$ev[pres16$margin > 0])
```

```
## [1] 305
```

```
sum(pres16$ev[poll.pred > 0])
```

```
## [1] 244
```

- Prediction of binary outcome variable = **classification problem**
- Wrong prediction $\rightsquigarrow$ misclassification

Classification

- Election prediction: need to predict winner in each state:

```
sum(pres16$ev[pres16$margin > 0])
```

```
## [1] 305
```

```
sum(pres16$ev[poll.pred > 0])
```

```
## [1] 244
```

- Prediction of binary outcome variable = **classification problem**
- Wrong prediction $\rightsquigarrow$ misclassification
    1. **true positive**: predict Trump wins when he actually wins.

Classification

- Election prediction: need to predict winner in each state:

```r
sum(pres16$ev[pres16$margin > 0])
```

```
## [1] 305
```

```r
sum(pres16$ev[poll.pred > 0])
```

```
## [1] 244
```

- Prediction of binary outcome variable = **classification problem**
- Wrong prediction $\rightsquigarrow$ misclassification
  1. **true positive**: predict Trump wins when he actually wins.
  2. **false positive**: predict Trump wins when he actually loses.

Classification

- Election prediction: need to predict winner in each state:

```
sum(pres16$ev[pres16$margin > 0])
```

```
## [1] 305
```

```
sum(pres16$ev[poll.pred > 0])
```

```
## [1] 244
```

- Prediction of binary outcome variable = **classification problem**
- Wrong prediction $\rightsquigarrow$ misclassification
  1. **true positive**: predict Trump wins when he actually wins.
  2. **false positive**: predict Trump wins when he actually loses.
  3. **true negative**: predict Trump loses when he actually loses.

Classification

- Election prediction: need to predict winner in each state:

```
sum(pres16$ev[pres16$margin > 0])
```

```
## [1] 305
```

```
sum(pres16$ev[poll.pred > 0])
```

```
## [1] 244
```

- Prediction of binary outcome variable = **classification problem**
- Wrong prediction $\rightsquigarrow$ misclassification
    1. **true positive**: predict Trump wins when he actually wins.
    2. **false positive**: predict Trump wins when he actually loses.
    3. **true negative**: predict Trump loses when he actually loses.
    4. **false negative**: predict Trump loses when he actually wins.

Classification

- Election prediction: need to predict winner in each state:

```
sum(pres16$ev[pres16$margin > 0])
```

```
## [1] 305
```

```
sum(pres16$ev[poll.pred > 0])
```

```
## [1] 244
```

- Prediction of binary outcome variable = **classification problem**
- Wrong prediction $\rightsquigarrow$ misclassification
    1. **true positive**: predict Trump wins when he actually wins.
    2. **false positive**: predict Trump wins when he actually loses.
    3. **true negative**: predict Trump loses when he actually loses.
    4. **false negative**: predict Trump loses when he actually wins.
- Sometimes false negatives are more/less important: e.g., civil war.

Classification based on polls

- Accuracy: `sign()` returns `1` for a positive number, `-1` for a negative number, and `0` for 0.

Classification based on polls

- Accuracy: `sign()` returns `1` for a positive number, `-1` for a negative number, and `0` for 0.

```
mean(sign(poll.pred) == sign(pres16$margin))
```

Classification based on polls

- Accuracy: `sign()` returns `1` for a positive number, `-1` for a negative number, and `0` for 0.

```
mean(sign(poll.pred) == sign(pres16$margin))
```

```
## [1] 0.902
```

Classification based on polls

- Accuracy: `sign()` returns `1` for a positive number, `-1` for a negative number, and `0` for 0.

```
mean(sign(poll.pred) == sign(pres16$margin))
```

```
## [1] 0.902
```

- Which states did polls call wrong?

Classification based on polls

- Accuracy: `sign()` returns `1` for a positive number, `-1` for a negative number, and `0` for 0.

```
mean(sign(poll.pred) == sign(pres16$margin))
```

```
## [1] 0.902
```

- Which states did polls call wrong?

```
pres16$state[sign(poll.pred) != sign(pres16$margin)]
```

Classification based on polls

- Accuracy: `sign()` returns `1` for a positive number, `-1` for a negative number, and `0` for 0.

```
mean(sign(poll.pred) == sign(pres16$margin))
```

```
## [1] 0.902
```

- Which states did polls call wrong?

```
pres16$state[sign(poll.pred) != sign(pres16$margin)]
```

```
## [1] MI NC NV PA WI
## 51 Levels: AK AL AR AZ CA CO CT DC DE FL GA HI ... WY
```

Classification based on polls

- Accuracy: `sign()` returns `1` for a positive number, `-1` for a negative number, and `0` for 0.

```r
mean(sign(poll.pred) == sign(pres16$margin))
```

```
## [1] 0.902
```

- Which states did polls call wrong?

```r
pres16$state[sign(poll.pred) != sign(pres16$margin)]
```

```
## [1] MI NC NV PA WI
## 51 Levels: AK AL AR AZ CA CO CT DC DE FL GA HI ... WY
```

- What were the actual margins?

Classification based on polls

- Accuracy: `sign()` returns `1` for a positive number, `-1` for a negative number, and `0` for 0.

```
mean(sign(poll.pred) == sign(pres16$margin))
```

```
## [1] 0.902
```

- Which states did polls call wrong?

```
pres16$state[sign(poll.pred) != sign(pres16$margin)]
```

```
## [1] MI NC NV PA WI
## 51 Levels: AK AL AR AZ CA CO CT DC DE FL GA HI ... WY
```

- What were the actual margins?

```
pres16$margin[sign(poll.pred) != sign(pres16$margin)]
```

Classification based on polls

- Accuracy: `sign()` returns `1` for a positive number, `-1` for a negative number, and `0` for 0.

```
mean(sign(poll.pred) == sign(pres16$margin))
```

```
## [1] 0.902
```

- Which states did polls call wrong?

```
pres16$state[sign(poll.pred) != sign(pres16$margin)]
```

```
## [1] MI NC NV PA WI
## 51 Levels: AK AL AR AZ CA CO CT DC DE FL GA HI ... WY
```

- What were the actual margins?

```
pres16$margin[sign(poll.pred) != sign(pres16$margin)]
```

```
## [1]  0.22  3.66 -2.42  0.71  0.77
```

How does 538/NYT difer?

- What we did is the core idea behind election forecasters like 538 and the NYT election prediction.

How does 538/NYT difer?

- What we did is the core idea behind election forecasters like 538 and the NYT election prediction.
- What do they do differently?

How does 538/NYT difer?

- What we did is the core idea behind election forecasters like 538 and the NYT election prediction.
- What do they do differently?
  - Use a longer history of polls but down-weight older polls.

How does 538/NYT difer?

- What we did is the core idea behind election forecasters like 538 and the NYT election prediction.
- What do they do differently?
  - Use a longer history of polls but down-weight older polls.
  - Up-weight/down-weight polls from polling firms with low/high past prediction error.

How does 538/NYT difer?

- What we did is the core idea behind election forecasters like 538 and the NYT election prediction.
- What do they do differently?
  - Use a longer history of polls but down-weight older polls.
  - Up-weight/down-weight polls from polling firms with low/high past prediction error.
  - Up-weight polls with better methodologies.

How does 538/NYT difer?

- What we did is the core idea behind election forecasters like 538 and the NYT election prediction.
- What do they do differently?
  - Use a longer history of polls but down-weight older polls.
  - Up-weight/down-weight polls from polling firms with low/high past prediction error.
  - Up-weight polls with better methodologies.
  - Combine poll-based predictions with predictions based on “fundamentals” like economic performance, popularity of the incumbent president.

Next week

- Prediction using linear regression.

Next week

- Prediction using linear regression.
- DataCamp assignment 4 due on Thursday

Next week

- Prediction using linear regression.
- DataCamp assignment 4 due on Thursday
- HW 3 goes out on Tuesday

Next week

- Prediction using linear regression.
- DataCamp assignment 4 due on Thursday
- HW 3 goes out on Tuesday
- Mid-semester evaluation survey online now.