Polling and ML Concepts

GOV 1347 Lab: Week III

Matthew E. Dardet

Harvard University

September 18, 2024

Check-In

- Any questions or feedback?

Review I: Multivariate (Multiple Regression)

Univariate regression: $\underbrace{Y}_{PV} = \beta_0 + \beta_1 \underbrace{X_1}_{GDP}$

Review I: Multivariate (Multiple Regression)

Univariate regression: $\underbrace{Y}_{PV} = \beta_0 + \beta_1 \underbrace{X_1}_{GDP}$

Multivariate regression: $\underbrace{Y}_{PV} = \beta_0 + \beta_1 \underbrace{X_1}_{GDP} + \beta_2 \underbrace{X_2}_{Approval}$

- Assumption: Linear Additivity between features $(\beta_1 X_1 + \beta_2 X_2)$

Review I: Multivariate (Multiple Regression)

Univariate regression: $\underbrace{Y}_{PV} = \beta_0 + \beta_1 \underbrace{X_1}_{GDP}$

Multivariate regression: $\underbrace{Y}_{PV} = \beta_0 + \beta_1 \underbrace{X_1}_{GDP} + \beta_2 \underbrace{X_2}_{Approval}$

- Assumption: <u>Linear Additivity</u> between features ($\beta_1 X_1 + \beta_2 X_2$)
- Estimation: same <u>OLS</u> procedure (minimizing sum of squared residuals).

Review I: Multivariate (Multiple Regression)

Univariate regression: $\underbrace{Y}_{PV} = \beta_0 + \beta_1 \underbrace{X_1}_{GDP}$

Multivariate regression: $\underbrace{Y}_{PV} = \beta_0 + \beta_1 \underbrace{X_1}_{GDP} + \beta_2 \underbrace{X_2}_{Approval}$

- Assumption: Linear Additivity between features ($\beta_1 X_1 + \beta_2 X_2$)
- Estimation: same OLS procedure (minimizing sum of squared residuals).
  - Solution now a **hyperplane** rather than a **line** (i.e., just a line in multiple dimensions).

Review I: Multivariate (Multiple Regression)

Univariate regression: $\underbrace{Y}_{PV} = \beta_0 + \beta_1 \underbrace{X_1}_{GDP}$

Multivariate regression: $\underbrace{Y}_{PV} = \beta_0 + \beta_1 \underbrace{X_1}_{GDP} + \beta_2 \underbrace{X_2}_{Approval}$

- Assumption: <u>Linear Additivity</u> between features ($\beta_1 X_1 + \beta_2 X_2$)
- Estimation: same <u>OLS</u> procedure (minimizing sum of squared residuals).
  - Solution now a **hyperplane** rather than a **line** (i.e., just a line in multiple dimensions).
- Better in-sample fit metric: $\underline{R^2_{adjusted}} = 1 - \frac{(1-R^2)(n-1)}{n-p-1}$ for p features

Review I: Multivariate (Multiple Regression)

Univariate regression: $\underbrace{Y}_{PV} = \beta_0 + \beta_1 \underbrace{X_1}_{GDP}$

Multivariate regression: $\underbrace{Y}_{PV} = \beta_0 + \beta_1 \underbrace{X_1}_{GDP} + \beta_2 \underbrace{X_2}_{Approval}$

- Assumption: Linear Additivity between features $(\beta_1 X_1 + \beta_2 X_2)$
- Estimation: same OLS procedure (minimizing sum of squared residuals).
  - Solution now a **hyperplane** rather than a **line** (i.e., just a line in multiple dimensions).
- Better in-sample fit metric: $R^2_{adjusted} = 1 - \frac{(1-R^2)(n-1)}{n-p-1}$ for p features
- Adding new features will change parameter estimates
  - Intuitively, β_1 is the unique effect of X_1

Review I: Multivariate (Multiple Regression)

Univariate regression: $\underbrace{Y}_{PV} = \beta_0 + \beta_1 \underbrace{X_1}_{GDP}$

Multivariate regression: $\underbrace{Y}_{PV} = \beta_0 + \beta_1 \underbrace{X_1}_{GDP} + \beta_2 \underbrace{X_2}_{Approval}$

- Assumption: Linear Additivity between features ($\beta_1 X_1 + \beta_2 X_2$)
- Estimation: same OLS procedure (minimizing sum of squared residuals).
  - Solution now a **hyperplane** rather than a **line** (i.e., just a line in multiple dimensions).
- Better in-sample fit metric: $R^2_{adjusted} = 1 - \frac{(1-R^2)(n-1)}{n-p-1}$ for p features
- Adding new features will change parameter estimates
  - Intuitively, β_1 is the unique effect of X_1
  - Only two rare cases where $\hat{\beta}_{1\,univariate} = \hat{\beta}_{1\,multivariate}$:

Review I: Multivariate (Multiple Regression)

Univariate regression: $\underbrace{Y}_{PV} = \beta_0 + \beta_1 \underbrace{X_1}_{GDP}$

Multivariate regression: $\underbrace{Y}_{PV} = \beta_0 + \beta_1 \underbrace{X_1}_{GDP} + \beta_2 \underbrace{X_2}_{Approval}$

- Assumption: <u>Linear Additivity</u> between features ($\beta_1 X_1 + \beta_2 X_2$)
- Estimation: same <u>OLS</u> procedure (minimizing sum of squared residuals).
  - Solution now a **hyperplane** rather than a **line** (i.e., just a line in multiple dimensions).
- Better in-sample fit metric: $\underline{R^2_{adjusted}} = 1 - \frac{(1-R^2)(n-1)}{n-p-1}$ for p features
- Adding new features will <u>change parameter estimates</u>
  - Intuitively, β_1 is the unique effect of X_1
  - Only two rare cases where $\hat{\beta}_{1\,univariate} = \hat{\beta}_{1\,multivariate}$:
    1. $\beta_2 = 0$

Review I: Multivariate (Multiple Regression)

Univariate regression: $\underbrace{Y}_{PV} = \beta_0 + \beta_1 \underbrace{X_1}_{GDP}$

Multivariate regression: $\underbrace{Y}_{PV} = \beta_0 + \beta_1 \underbrace{X_1}_{GDP} + \beta_2 \underbrace{X_2}_{Approval}$

- Assumption: <u>Linear Additivity</u> between features ($\beta_1 X_1 + \beta_2 X_2$)
- Estimation: same <u>OLS</u> procedure (minimizing sum of squared residuals).
  - Solution now a **hyperplane** rather than a **line** (i.e., just a line in multiple dimensions).
- Better in-sample fit metric: $\underline{R^2_{adjusted}} = 1 - \frac{(1-R^2)(n-1)}{n-p-1}$ for p features
- Adding new features will <u>change parameter estimates</u>
  - Intuitively, β_1 is the unique effect of X_1
  - Only two rare cases where $\hat{\beta}_{1\,univariate} = \hat{\beta}_{1\,multivariate}$:
    1. $\beta_2 = 0$
    2. $\rho_{X_1, X_2} = 0$

Review I: Multivariate (Multiple Regression)

Univariate regression: $\underbrace{Y}_{PV} = \beta_0 + \beta_1 \underbrace{X_1}_{GDP}$

Multivariate regression: $\underbrace{Y}_{PV} = \beta_0 + \beta_1 \underbrace{X_1}_{GDP} + \beta_2 \underbrace{X_2}_{Approval}$

- Assumption: <u>Linear Additivity</u> between features ($\beta_1 X_1 + \beta_2 X_2$)
- Estimation: same <u>OLS</u> procedure (minimizing sum of squared residuals).
  - Solution now a **hyperplane** rather than a **line** (i.e., just a line in multiple dimensions).
- Better in-sample fit metric: $\underline{R^2_{adjusted}} = 1 - \frac{(1-R^2)(n-1)}{n-p-1}$ for p features
- Adding new features will <u>change parameter estimates</u>
  - Intuitively, β_1 is the unique effect of X_1
  - Only two rare cases where $\hat{\beta}_{1\,univariate} = \hat{\beta}_{1\,multivariate}$:
    1. $\beta_2 = 0$
    2. $\rho_{X_1, X_2} = 0$

Review II: Interaction Effects

Sociopolitical outcomes are often predicted by **interactions** of features:

Review II: Interaction Effects

Sociopolitical outcomes are often predicted by **interactions** of features:

Examples:

- `race x gender`:

Review II: Interaction Effects

Sociopolitical outcomes are often predicted by **interactions** of features:

Examples:

- `race x gender`:
  - The effect of X_1 (`race`) can vary depending on the value of X_2 (`gender`)

Review II: Interaction Effects

Sociopolitical outcomes are often predicted by **interactions** of features:

Examples:

- `race x gender`:
  - The effect of X_1 (`race`) can vary depending on the value of X_2 (`gender`)
- From last week's blog and extensions:

Review II: Interaction Effects

Sociopolitical outcomes are often predicted by **interactions** of features:

Examples:

- `race x gender`:
  - The effect of X_1 (`race`) can vary depending on the value of X_2 (`gender`)
- From last week's blog and extensions:
  - `sitting president x economic performance`
  - `year x economic performance`

Review II: Interaction Effects

Sociopolitical outcomes are often predicted by **interactions** of features:

Examples:

- `race x gender`:
  - The effect of X_1 (`race`) can vary depending on the value of X_2 (`gender`)
- From last week's blog and extensions:
  - `sitting president x economic performance`
  - `year x economic performance`
  - `party x unemployment`

Review II: Interaction Effects

Sociopolitical outcomes are often predicted by **interactions** of features:

Examples:

- `race x gender`:
    - The effect of X_1 (`race`) can vary depending on the value of X_2 (`gender`)
- From last week's blog and extensions:
    - `sitting president x economic performance`
    - `year x economic performance`
    - `party x unemployment`
        - In state s, incumbent benefits if Democratic, hurt if Republican

Review II: Interaction Effects

Sociopolitical outcomes are often predicted by **interactions** of features:

Examples:

- `race x gender`:
  - The effect of X_1 (`race`) can vary depending on the value of X_2 (`gender`)
- From last week's blog and extensions:
  - `sitting president x economic performance`
  - `year x economic performance`
  - `party x unemployment`
    - In state s, incumbent benefits if Democratic, hurt if Republican Why?

Review III: Overfitting

> *A common criticism of fundamentals models is that they are extremely easy to* ***over-fit****—the statistical term for deriving equations that provide a close match to historical data, but break down when used to predict the future. To avoid this risk, we borrow two techniques from the world of machine learning, with appropriately inscrutable names: elastic-net regularisation and* ***leave-one-out cross-validation****.*

(Morris 2020a)

Review III: Overfitting

> *A common criticism of fundamentals models is that they are extremely easy to **over-fit**—the statistical term for deriving equations that provide a close match to historical data, but break down when used to predict the future. To avoid this risk, we borrow two techniques from the world of machine learning, with appropriately inscrutable names: elastic-net regularisation and **leave-one-out cross-validation**.*

(Morris 2020a)

Explicit tension between **in-sample fit** ("close match to historical data") and **out-of-sample performance**.

Review III: Overfitting

> *A common criticism of fundamentals models is that they are extremely easy to* ***over-fit****—the statistical term for deriving equations that provide a close match to historical data, but break down when used to predict the future. To avoid this risk, we borrow two techniques from the world of machine learning, with appropriately inscrutable names: elastic-net regularisation and* ***leave-one-out cross-validation****.*

(Morris 2020a)

Explicit tension between **in-sample fit** ("close match to historical data") and **out-of-sample performance**. What can we do to ameliorate this?

- Cross-validation

Review III: Overfitting

> *A common criticism of fundamentals models is that they are extremely easy to* ***over-fit****—the statistical term for deriving equations that provide a close match to historical data, but break down when used to predict the future. To avoid this risk, we borrow two techniques from the world of machine learning, with appropriately inscrutable names: elastic-net regularisation and* ***leave-one-out cross-validation****.*

(Morris 2020a)

Explicit tension between **in-sample fit** ("close match to historical data") and **out-of-sample performance**. What can we do to ameliorate this?

- Cross-validation
- Elastic-net regularization: Parsimony of a model reduces out-of-sample error

Review III: Overfitting

> *A common criticism of fundamentals models is that they are extremely easy to* ***over-fit****—the statistical term for deriving equations that provide a close match to historical data, but break down when used to predict the future. To avoid this risk, we borrow two techniques from the world of machine learning, with appropriately inscrutable names: elastic-net regularisation and* ***leave-one-out cross-validation****.*

(Morris 2020a)

Explicit tension between **in-sample fit** ("close match to historical data") and **out-of-sample performance**. What can we do to ameliorate this?

- Cross-validation
- Elastic-net regularization: Parsimony of a model reduces out-of-sample error
- Bottom-line: An $R^2 > 0.9$ might actually be *bad* for prediction

This Week's Goal

Main Question: How can we best use polls to predict election outcomes?

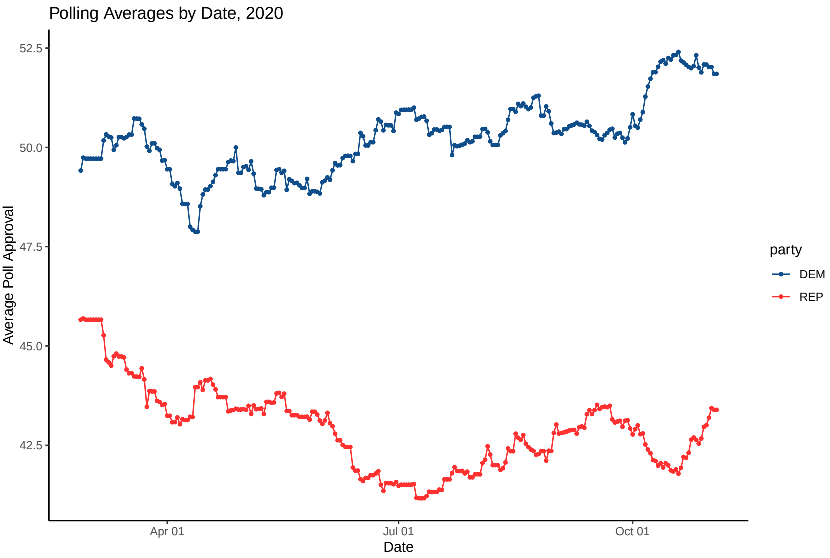
1. **Visualizing Poll Variation Over Time.**
2. **Regularization.**
3. **Regularized Regression with Polling Data.**
4. **(Time Allowing. . .) Ensemble Learning, Model Ensembling, and Combining Predictions.**

FiveThirtyEight Polling Data

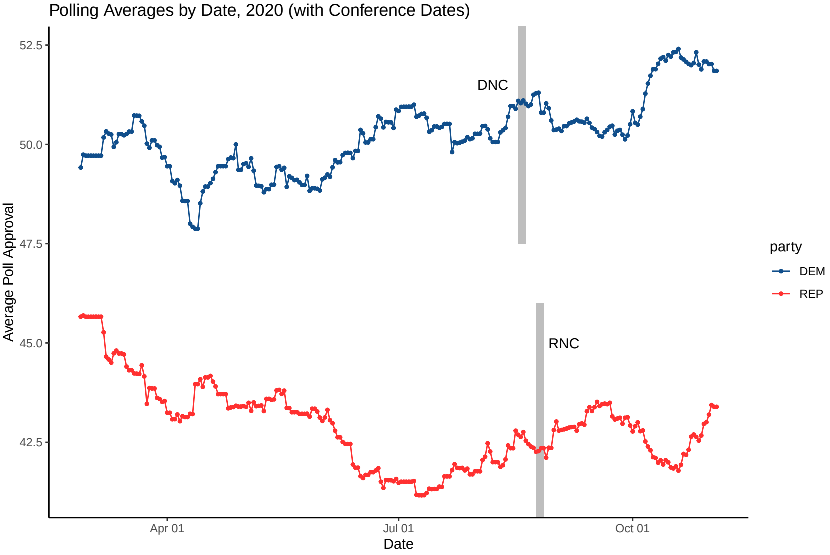
Section 1

Visualizing Poll Variation Over Time

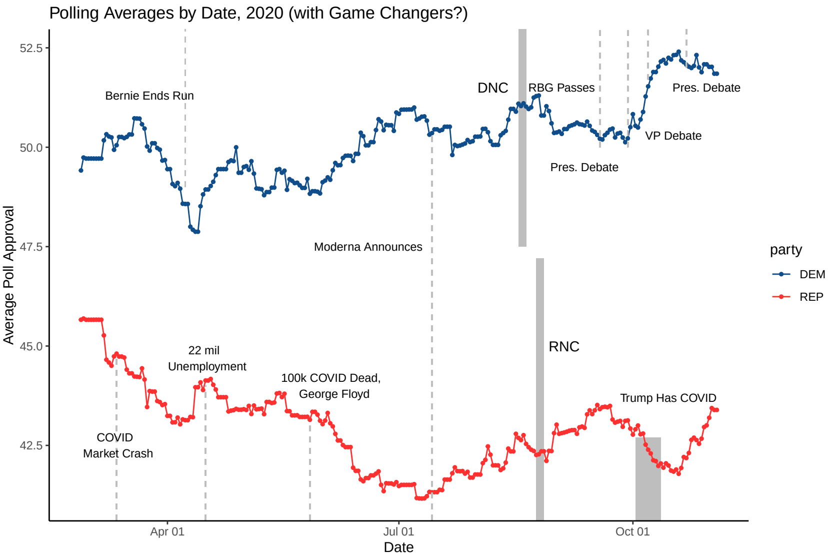
How Do Polls Fluctuate Over Time? 2020



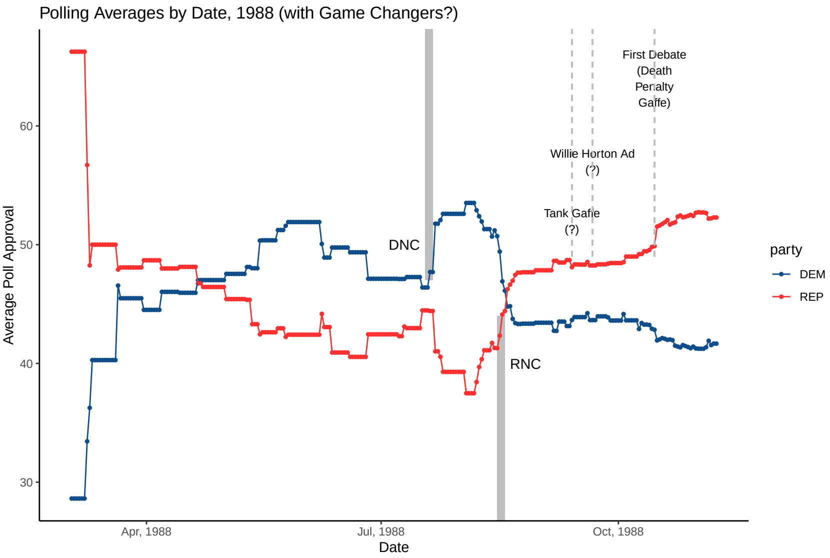
How Do Polls Fluctuate Over Time? 2020



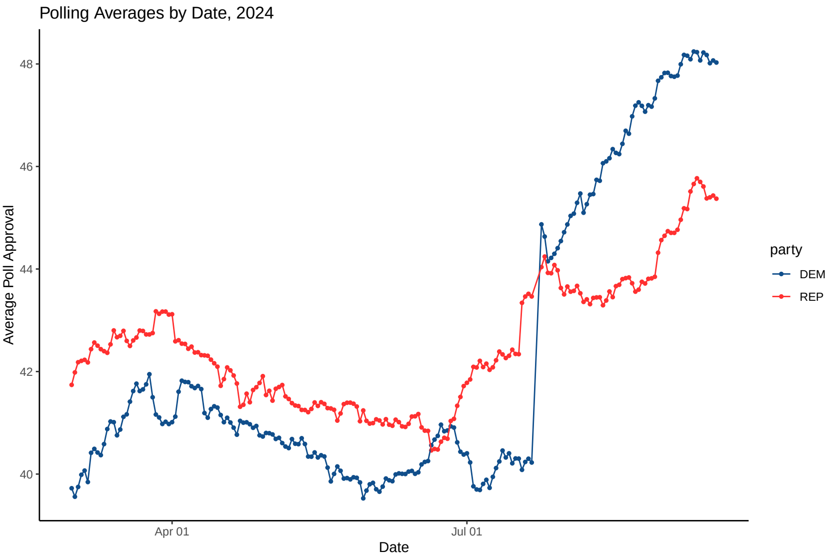
How Do Polls Fluctuate Over Time? 2020



How Do Polls Fluctuate Over Time? 1988



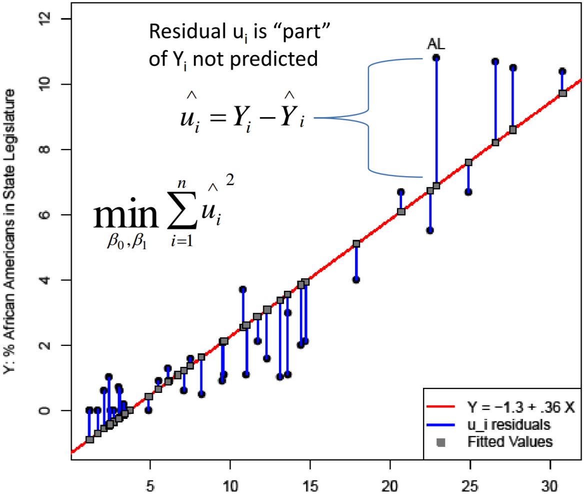
How Do Polls Fluctuate Over Time? 2024



Section 2

Regularization 101

Review of OLS Objective



OLS Works Well When...

- Linearity assumption is satisfied.

OLS Works Well When...

- Linearity assumption is satisfied.
- Features are not highly correlated (no or minimal multicollinearity).

OLS Works Well When...

- Linearity assumption is satisfied.
- Features are not highly correlated (no or minimal multicollinearity).
- Number of features is less than number of observations ($p < n$).

But. . .

- Particularly for predicting national popular vote, we have soon run into the curse of dimensionality where $p > n$.

But. . .

- Particularly for predicting national popular vote, we have soon run into the curse of dimensionality where $p > n$.
  - Possibility of overfitting increases $\rightsquigarrow$ increase out-of-sample prediction error.

But. . .

- Particularly for predicting national popular vote, we have soon run into the curse of dimensionality where $p > n$.
  - Possibility of overfitting increases $\rightsquigarrow$ increase out-of-sample prediction error.
  - Model becomes uninterpretable.

But...

- Particularly for predicting national popular vote, we have soon run into the curse of dimensionality where $p > n$.
  - Possibility of overfitting increases $\rightsquigarrow$ increase out-of-sample prediction error.
  - Model becomes uninterpretable.
  - If we wanted to include all (or, at least $n + 1$) features in our model, OLS will have infinite solutions.

But. . .

- Particularly for predicting national popular vote, we have soon run into the curse of dimensionality where $p > n$.
  - Possibility of overfitting increases $\rightsquigarrow$ increase out-of-sample prediction error.
  - Model becomes uninterpretable.
  - If we wanted to include all (or, at least $n + 1$) features in our model, OLS will have infinite solutions.
- Bet on sparsity principle: smaller subset of features in model exhibit the strongest effects.

Solution: Regularization & Feature Selection

- **Regularization:** Constrain or *regularize* coefficients

Solution: Regularization & Feature Selection

- **Regularization:** Constrain or *regularize* coefficients $\rightsquigarrow$ reduce variance and decrease out-of-sample prediction error.

Solution: Regularization & Feature Selection

- **Regularization:** Constrain or *regularize* coefficients $\rightsquigarrow$ reduce variance and decrease out-of-sample prediction error.
- **Regularized regression:** apply a penalty term to OLS objective function in order to shrink coefficients.

Solution: Regularization & Feature Selection

- **Regularization:** Constrain or *regularize* coefficients $\rightsquigarrow$ reduce variance and decrease out-of-sample prediction error.
- **Regularized regression:** apply a penalty term to OLS objective function in order to shrink coefficients. Why?
- Constraint makes it such that the only way coefficients can increase is if they decrease the sum of squared residuals.

Solution: Regularization & Feature Selection

- **Regularization:** Constrain or *regularize* coefficients $\rightsquigarrow$ reduce variance and decrease out-of-sample prediction error.
- **Regularized regression:** apply a penalty term to OLS objective function in order to shrink coefficients. Why?
- Constraint makes it such that the only way coefficients can increase is if they decrease the sum of squared residuals.
- Selects for features that are particularly important.

Ridge Regression

- Goal: Constrain size of OLS parameters.

Ridge Regression

- Goal: Constrain size of OLS parameters.
- New objective function:

Ridge Regression

- Goal: Constrain size of OLS parameters.
- New objective function:

$$\min_{\beta} \; SSR + P$$

Ridge Regression

- Goal: Constrain size of OLS parameters.
- New objective function:

$$\min_{\beta} \; SSR + P$$

With complexity (a.k.a. tuning) parameter $\lambda \geq 0$ times the L^2 (Euclidean) norm:

$$= \min_{\beta} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 + \lambda \sum_{j=1}^{p} \beta_j^2$$

Ridge Regression

- Goal: Constrain size of OLS parameters.
- New objective function:

$$\min_{\beta} \; SSR + P$$

With complexity (a.k.a. tuning) parameter $\lambda \geq 0$ times the L^2 (Euclidean) norm:

$$= \min_{\beta} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 + \lambda \sum_{j=1}^{p} \beta_j^2$$

$$= \min_{\beta} \sum_{i=1}^{n} (y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij})^2 + \lambda \sum_{j=1}^{p} \beta_j^2$$

Ridge Regression

- Goal: Constrain size of OLS parameters.
- New objective function:

$$\min_{\beta} \; SSR + P$$

With complexity (a.k.a. tuning) parameter $\lambda \geq 0$ times the L^2 (Euclidean) norm:

$$= \min_{\beta} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 + \lambda \sum_{j=1}^{p} \beta_j^2$$

$$= \min_{\beta} \sum_{i=1}^{n} (y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij})^2 + \lambda \sum_{j=1}^{p} \beta_j^2$$

As $\lambda \to \infty, \beta_j \to 0$. As $\lambda \to 0, \beta_j \to \hat{\beta}_j^{OLS}$.

Ridge Regression

- Goal: Constrain size of OLS parameters.
- New objective function:

$$\min_{\beta} \ SSR + P$$

With complexity (a.k.a. tuning) parameter $\lambda \geq 0$ times the L^2 (Euclidean) norm:

$$= \min_{\beta} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 + \lambda \sum_{j=1}^{p} \beta_j^2$$

$$= \min_{\beta} \sum_{i=1}^{n} (y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij})^2 + \lambda \sum_{j=1}^{p} \beta_j^2$$

As $\lambda \to \infty, \beta_j \to 0$. As $\lambda \to 0, \beta_j \to \hat{\beta}_j^{OLS}$. And there are other methods like . . .

The LASSO

- Like Ridge, but with the L_1 norm instead of L_2:

The LASSO

- Like Ridge, but with the L_1 norm instead of L_2:

The LASSO

- New objective function:

$$\min_{\beta} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 + \lambda \sum_{j=1}^{p} |\beta_j|$$

The LASSO

- New objective function:

$$\min_{\beta} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 + \lambda \sum_{j=1}^{p} |\beta_j|$$

- *Key difference* from Ridge: LASSO coefficients can be exactly zero.

Elastic-Net

- You may be asking:

Elastic-Net

- You may be asking:

Elastic-Net

- Elastic-Net is linear combination of Ridge (L_2) and LASSO (L_1) penalties.

Elastic-Net

- Elastic-Net is linear combination of Ridge (L_2) and LASSO (L_1) penalties.
- New objective function with mixing parameter $\alpha \in [0, 1]$:

$$\min_{\beta} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 + \lambda \sum_{j=1}^{p} \alpha \beta_j^2 + (1 - \alpha)|\beta_j|$$

Elastic-Net

- Elastic-Net is linear combination of Ridge (L_2) and LASSO (L_1) penalties.
- New objective function with mixing parameter $\alpha \in [0, 1]$:

$$\min_{\beta} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 + \lambda \sum_{j=1}^{p} \alpha \beta_j^2 + (1 - \alpha)|\beta_j|$$

- Typically, $\alpha = 0.5$ is a good starting point.

Elastic-Net

- Elastic-Net is linear combination of Ridge (L_2) and LASSO (L_1) penalties.
- New objective function with mixing parameter $\alpha \in [0, 1]$:

$$\min_{\beta} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 + \lambda \sum_{j=1}^{p} \alpha \beta_j^2 + (1 - \alpha)|\beta_j|$$

- Typically, $\alpha = 0.5$ is a good starting point.
- If $\alpha = 0$, Elastic-Net is Ridge. If $\alpha = 1$, Elastic-Net is LASSO.

Section 3

Regularized Regression with Polling Data

Using November Polls Average to Predict National Popular Vote

```r
# OLS: Party-stacked pv2p on November polling average.
ols.nov.2 <- lm(pv2p ~ nov_poll,
                data = d_poll_nov)
```

```
##
## Call:
## lm(formula = pv2p ~ nov_poll, data = d_poll_nov)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.6190 -1.6523 -0.5808  1.3629  6.0220
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 17.92577    4.15543   4.314 0.000205 ***
## nov_poll     0.70787    0.09099   7.780 2.97e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.75 on 26 degrees of freedom
```

What If We Want to Use Weekly Poll Averages Throughout the Election Year?

- Soon, may run into curse of dimensionality given that we only have 14 election years.

What If We Want to Use Weekly Poll Averages Throughout the Election Year?

- Soon, may run into curse of dimensionality given that we only have 14 election years.
- In fact, OLS with 31 weekly polls is inestimable.

What If We Want to Use Weekly Poll Averages Throughout the Election Year?

```
##
## Call:
## lm(formula = paste0("pv2p ~ ", paste0("poll_weeks_left_", 0:30,
##     collapse = " + ")), data = d_poll_weeks_train)
##
## Residuals:
## ALL 28 residuals are 0: no residual degrees of freedom!
##
## Coefficients: (4 not defined because of singularities)
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)         28.25534        NaN     NaN      NaN
## poll_weeks_left_0    3.24113        NaN     NaN      NaN
## poll_weeks_left_1    0.02516        NaN     NaN      NaN
## poll_weeks_left_2   -8.87360        NaN     NaN      NaN
## poll_weeks_left_3    7.91455        NaN     NaN      NaN
## poll_weeks_left_4    0.74573        NaN     NaN      NaN
## poll_weeks_left_5    1.41567        NaN     NaN      NaN
## poll_weeks_left_6   -4.58444        NaN     NaN      NaN
## poll_weeks_left_7    4.63361        NaN     NaN      NaN
## poll_weeks_left_8   -0.95121        NaN     NaN      NaN
## poll_weeks_left_9   -1.55307        NaN     NaN      NaN
## poll_weeks_left_10 -1.38062        NaN     NaN      NaN
## poll_weeks_left_11  1.74881        NaN     NaN      NaN
## poll_weeks_left_12 -1.28871        NaN     NaN      NaN
## poll_weeks_left_13 -0.08482        NaN     NaN      NaN
## poll_weeks_left_14  0.87498        NaN     NaN      NaN
## poll_weeks_left_15 -0.16310        NaN     NaN      NaN
## poll_weeks_left_16 -0.34501        NaN     NaN      NaN
## poll_weeks_left_17 -0.38689        NaN     NaN      NaN
## poll_weeks_left_18 -0.06281        NaN     NaN      NaN
## poll_weeks_left_19 -0.17204        NaN     NaN      NaN
## poll_weeks_left_20  1.52230        NaN     NaN      NaN
## poll_weeks_left_21 -0.72487        NaN     NaN      NaN
## poll_weeks_left_22 -2.76531        NaN     NaN      NaN
```

Ridge Regression Using Weekly Poll Averages

```
# Separate data into X and Y for training.
x.train <- d_poll_weeks_train |>
  ungroup() |>
  select(all_of(starts_with("poll_weeks_left_"))) |>
  as.matrix()
y.train <- d_poll_weeks_train$pv2p

# Ridge.
ridge.pollsweeks <- glmnet(x = x.train, y = y.train, alpha = 0)
  # Set ridge using alpha = 0.
```

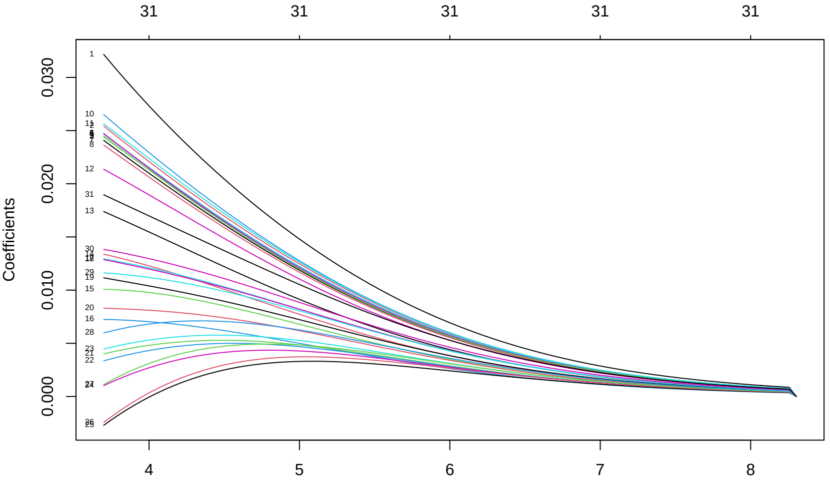
Ridge Regression Using Weekly Poll Averages

```r
# Get particular coefficients.
coef(ridge.pollsweeks, s = 0.1)
```

```
## 32 x 1 sparse Matrix of class "dgCMatrix"
##                              s1
## (Intercept)         29.951147799
## poll_weeks_left_0    0.032163983
## poll_weeks_left_1    0.025440084
## poll_weeks_left_2    0.024404320
## poll_weeks_left_3    0.024688870
## poll_weeks_left_4    0.024695646
## poll_weeks_left_5    0.024725772
## poll_weeks_left_6    0.024080438
## poll_weeks_left_7    0.023636908
## poll_weeks_left_8    0.024487501
## poll_weeks_left_9    0.026498950
## poll_weeks_left_10   0.025642838
## poll_weeks_left_11   0.021361476
## poll_weeks_left_12   0.017386999
## poll_weeks_left_13   0.013378030
## poll_weeks_left_14   0.010078675
## poll_weeks_left_15   0.007248494
## poll_weeks_left_16   0.012943440
## poll_weeks_left_17   0.012879654
## poll_weeks_left_18   0.011157452
## poll_weeks_left_19   0.008302783
## poll_weeks_left_20   0.004012987
## poll_weeks_left_21   0.003350434
## poll_weeks_left_22   0.004458406
## poll_weeks_left_23   0.001019583
## poll_weeks_left_24  -0.002711193
## poll_weeks_left_25  -0.002447895
## poll_weeks_left_26   0.001121142
## poll_weeks_left_27   0.005975853
## poll_weeks_left_28   0.011623984
## poll_weeks_left_29   0.013833925
## poll_weeks_left_30   0.018964139
```

Ridge Shrinkage Visualization

```r
# Visualize shrinkage.
plot(ridge.pollsweeks, xvar = "lambda", label = TRUE)
```



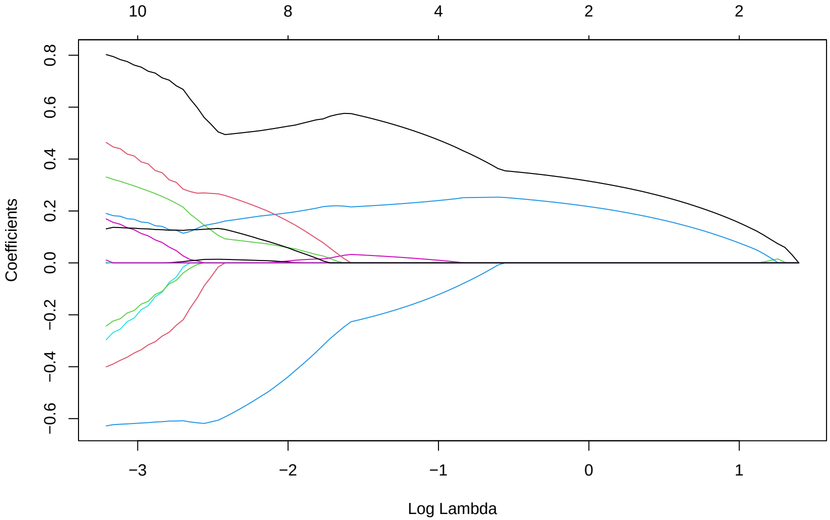
LASSO Using Weekly Poll Averages

```
# Lasso.
lasso.pollsweeks <- glmnet(x = x.train, y = y.train, alpha = 1) # Se
```

```
## 32 x 1 sparse Matrix of class "dgCMatrix"
##                             s1
## (Intercept)        24.57897724
## poll_weeks_left_0   0.50149421
## poll_weeks_left_1   .
## poll_weeks_left_2   .
## poll_weeks_left_3   .
## poll_weeks_left_4   .
## poll_weeks_left_5   0.08461518
## poll_weeks_left_6   .
## poll_weeks_left_7   .
## poll_weeks_left_8   .
## poll_weeks_left_9   0.17064525
## poll_weeks_left_10  .
## poll_weeks_left_11  .
## poll_weeks_left_12  .
## poll_weeks_left_13  .
## poll_weeks_left_14  .
## poll_weeks_left_15  0.01147512
## poll_weeks_left_16  .
## poll_weeks_left_17  .
## poll_weeks_left_18  0.23694416
## poll_weeks_left_19  .
## poll_weeks_left_20  .
## poll_weeks_left_21  .
## poll_weeks_left_22  .
## poll_weeks_left_23  .
## poll_weeks_left_24  .
## poll_weeks_left_25 -0.55693209
## poll_weeks_left_26  .
## poll_weeks_left_27  .
```

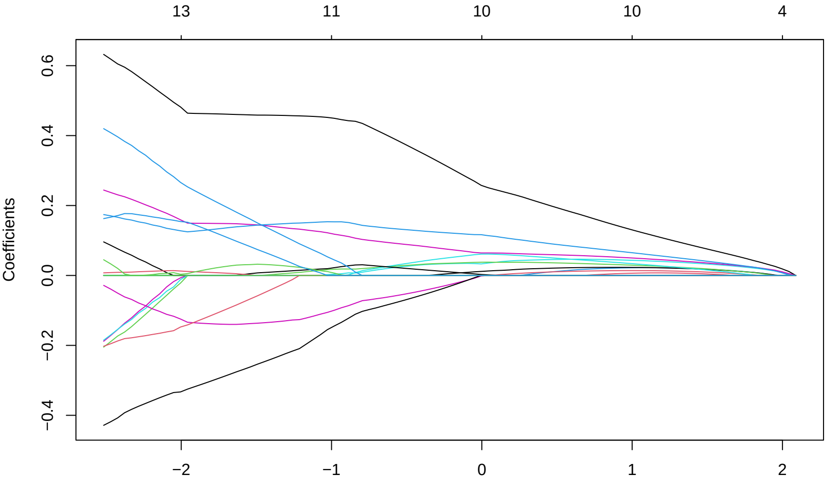
LASSO Shrinkage Visualization

```
plot(lasso.pollsweeks, xvar = "lambda")
```



Elastic-Net Using Weekly Poll Averages

```
# Elastic net.
enet.pollsweeks <- glmnet(x = x.train, y = y.train, alpha = 0.5) # S
```



Cross-Validated Regularized Regression

```
# Can use cross-validated versions to find the optimal values of lambda that minimiz
cv.ridge.pollweeks <- cv.glmnet(x = x.train, y = y.train, alpha = 0)
cv.lasso.pollweeks <- cv.glmnet(x = x.train, y = y.train, alpha = 1)
cv.enet.pollweeks <- cv.glmnet(x = x.train, y = y.train, alpha = 0.5)

# Get minimum lambda values
lambda.min.ridge <- cv.ridge.pollweeks$lambda.min
lambda.min.lasso <- cv.lasso.pollweeks$lambda.min
lambda.min.enet <- cv.enet.pollweeks$lambda.min

# Predict on training data using lambda values that minimize MSE.
(mse.ridge <- mean((predict(ridge.pollsweeks, s = lambda.min.ridge, newx = x.train)
```

```
## [1] 9.575001
```

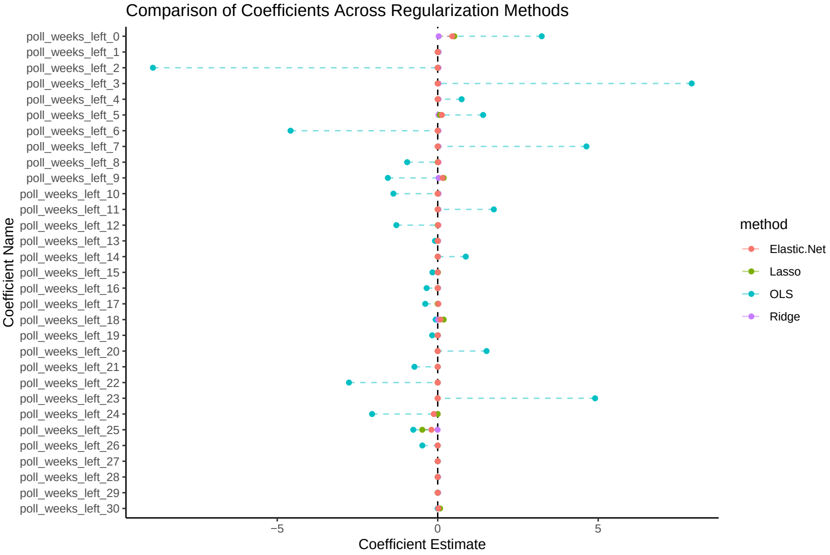
```
(mse.lasso <- mean((predict(lasso.pollsweeks, s = lambda.min.lasso, newx = x.train)
```

```
## [1] 3.663094
```

```
(mse.enet <- mean((predict(enet.pollsweeks, s = lambda.min.enet, newx = x.train) - y
```

```
## [1] 4.228987
```

Comparing Coefficients from OLS, Ridge, LASSO, E-net



Example: 2024 Predictions Using E-Net and Weekly Poll Averages

```r
# First check how many weeks of polling we have for 2024.
d_pollav_natl |>
  filter(year == 2024) |>
  select(weeks_left) |>
  distinct() |>
  range() # Let's take week 30 - 7 as predictors since those are the weeks we have
```

```
## [1]  7 36
```

```r
x.train <- d_poll_weeks_train |>
  ungroup() |>
  select(all_of(paste0("poll_weeks_left_", 7:30))) |>
  as.matrix()
y.train <- d_poll_weeks_train$pv2p
x.test <- d_poll_weeks_test |>
  ungroup() |>
  select(all_of(paste0("poll_weeks_left_", 7:30))) |>
  as.matrix()
```

Example: 2024 Predictions Using E-Net and Weekly Poll Averages

```
# Using elastic-net for simplicity.
set.seed(02138)
enet.poll <- cv.glmnet(x = x.train, y = y.train, alpha = 0.5)
lambda.min.enet.poll <- enet.poll$lambda.min

# Predict 2024 national pv2p share using elastic-net.
(polls.pred <- predict(enet.poll, s = lambda.min.enet.poll, newx =
```

```
##              s1
## [1,] 51.79268
## [2,] 50.65879
```

Section 4

Ensemble Learning, Model Ensembling, and Combining Predictions

We Have a Bunch of Predictions, Now What?

- We can combine different predictions to create a **meta-model** or **ensemble** that is more accurate than any individual model.

We Have a Bunch of Predictions, Now What?

- We can combine different predictions to create a **meta-model** or **ensemble** that is more accurate than any individual model.
- *Many* different ways to create an ensemble:

We Have a Bunch of Predictions, Now What?

- We can combine different predictions to create a **meta-model** or **ensemble** that is more accurate than any individual model.
- *Many* different ways to create an ensemble:
  - Unweighted (or equally weighted) ensemble: average predictions from different models.

We Have a Bunch of Predictions, Now What?

- We can combine different predictions to create a **meta-model** or **ensemble** that is more accurate than any individual model.
- *Many* different ways to create an ensemble:
  - Unweighted (or equally weighted) ensemble: average predictions from different models.
    - PollyVote 2020 model: equally weighted vote intention, expectation, models, and naive forecast.

We Have a Bunch of Predictions, Now What?

- We can combine different predictions to create a **meta-model** or **ensemble** that is more accurate than any individual model.
- *Many* different ways to create an ensemble:
  - Unweighted (or equally weighted) ensemble: average predictions from different models.
    - PollyVote 2020 model: equally weighted vote intention, expectation, models, and naive forecast.
  - Weighted ensemble: assign weights to different models based on performance.

We Have a Bunch of Predictions, Now What?

- We can combine different predictions to create a **meta-model** or **ensemble** that is more accurate than any individual model.
- *Many* different ways to create an ensemble:
  - Unweighted (or equally weighted) ensemble: average predictions from different models.
    - PollyVote 2020 model: equally weighted vote intention, expectation, models, and naive forecast.
  - Weighted ensemble: assign weights to different models based on performance.
    - Weight on days until the election (pollsters vs. Gelman & King, 1993)

We Have a Bunch of Predictions, Now What?

- We can combine different predictions to create a **meta-model** or **ensemble** that is more accurate than any individual model.
- *Many* different ways to create an ensemble:
  - Unweighted (or equally weighted) ensemble: average predictions from different models.
    - PollyVote 2020 model: equally weighted vote intention, expectation, models, and naive forecast.
  - Weighted ensemble: assign weights to different models based on performance.
    - Weight on days until the election (pollsters vs. Gelman & King, 1993)
    - Weight on R^2 (Silver 2020)
    - Weight on cross-validation error (this is called "Super Learning")
    - Weight on human priors.

We Have a Bunch of Predictions, Now What?

- We can combine different predictions to create a **meta-model** or **ensemble** that is more accurate than any individual model.
- *Many* different ways to create an ensemble:
  - Unweighted (or equally weighted) ensemble: average predictions from different models.
    - PollyVote 2020 model: equally weighted vote intention, expectation, models, and naive forecast.
  - Weighted ensemble: assign weights to different models based on performance.
    - Weight on days until the election (pollsters vs. Gelman & King, 1993)
    - Weight on R^2 (Silver 2020)
    - Weight on cross-validation error (this is called "Super Learning")
    - Weight on human priors.
    - The most "cutting edge" methods combine fundamentals and polls using Bayesian methods (e.g., Lauderdale, Linzer (2015))

Ensembling

- Experiment with combining your predictions using vote share, economic fundamentals, and polls, either using combined models with feature selection or ensembles.
- I have a few examples at the bottom of the code and will go into greater detail next week.

Blog Extensions

1. **Incorporating Pollster Quality.** Consider 2016-2024 pollster ratings from FiveThirtyEight. (1.) How much variation is there in pollster quality? (2.) Using tools and knowledge you have gained so far, build a model (or ensemble model) that uses individual polls from 2016 (`president_polls_2016.csv`), 2020 (`...2020.csv`), and 2024 (`...2024.csv`). How does your model compare to the models this week in lab?
2. **State-Level Polls.** How do state-level polls differ from national level polls? Using careful model evaluation techniques (possibly with ensembling), build a predictive model for 2024 using state-level polls.
3. **What Do Forecasters Do?** Based on what you've learned about fundamentals and poll-based forecasts, (1.) briefly summarize Silver (2024) and Morris (2024) and (2.) compare and contrast their approaches. In your opinion, which of the two is the better approach?