Penalized regression, logistic regression, and classification

Tom Ben-Shahar

February 22, 2025

```r
# Load Packages
pacman::p_load(tidyverse, tidymodels, magrittr, skimr, glmnet)

# Load Data
election <- read.csv("data/election.csv")

# Clean Data
election %<>% mutate(
  i_republican_2016_f = factor(i_republican_2016),
  i_republican_2012 = factor(i_republican_2012)
)

# Set Constants
set.seed(93284298)
nfold = 5
lambdas = 10^seq(from = 5, to = -2, length = 1e3)
alphas = seq(from = 0, to = 1, by = 0.05)
```

Data Overview

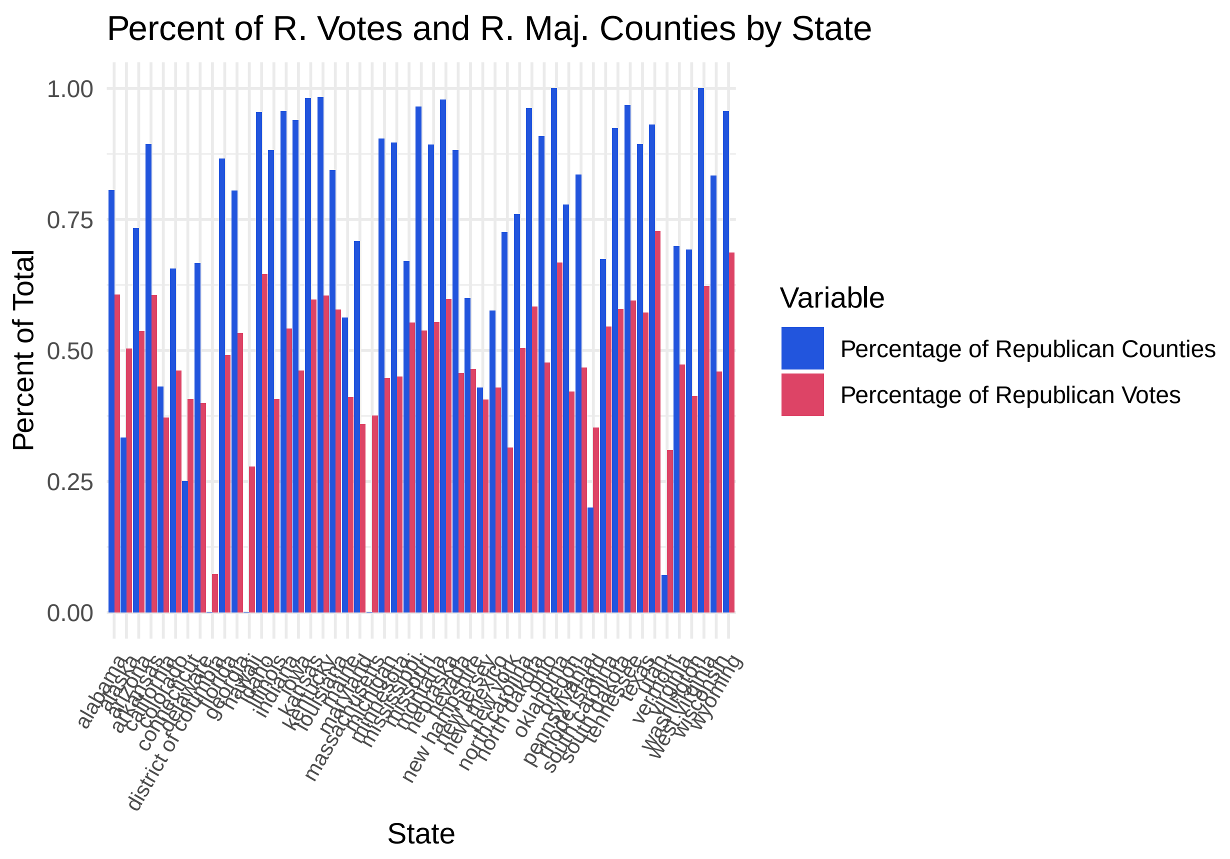
Create some nice plots of the raw data and brief discussion

```r
# Percent of Republican Votes by State
fig_1_df <- election %>%
  group_by(state) %>%
  summarise(
    pct_r_counties = mean(i_republican_2016),
    tot_votes_r = sum(n_votes_republican_2012),
    tot_votes = sum(n_votes_total_2012)
  ) %>%
  mutate(
    pct_r_votes = tot_votes_r / tot_votes
  )

long_df <- fig_1_df %>%
  pivot_longer(cols = c(pct_r_votes, pct_r_counties),
               names_to = "Variable",
               values_to = "Value")

# Create the bar chart
```

```r
ggplot(long_df, aes(x = state, y = Value, fill = Variable)) +
  geom_bar(stat = "identity", position = "dodge") +
  labs(title = "Percent of R. Votes and R. Maj. Counties by State",
       x = "State",
       y = "Percent of Total",
       fill = "Variable") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 60, hjust = 1, size = 8)) +
  scale_fill_manual(values = c("pct_r_counties" = "#25d", "pct_r_votes" = "#d46"),
                    labels = c("Percentage of Republican Counties", "Percentage of Republican Votes"))
```

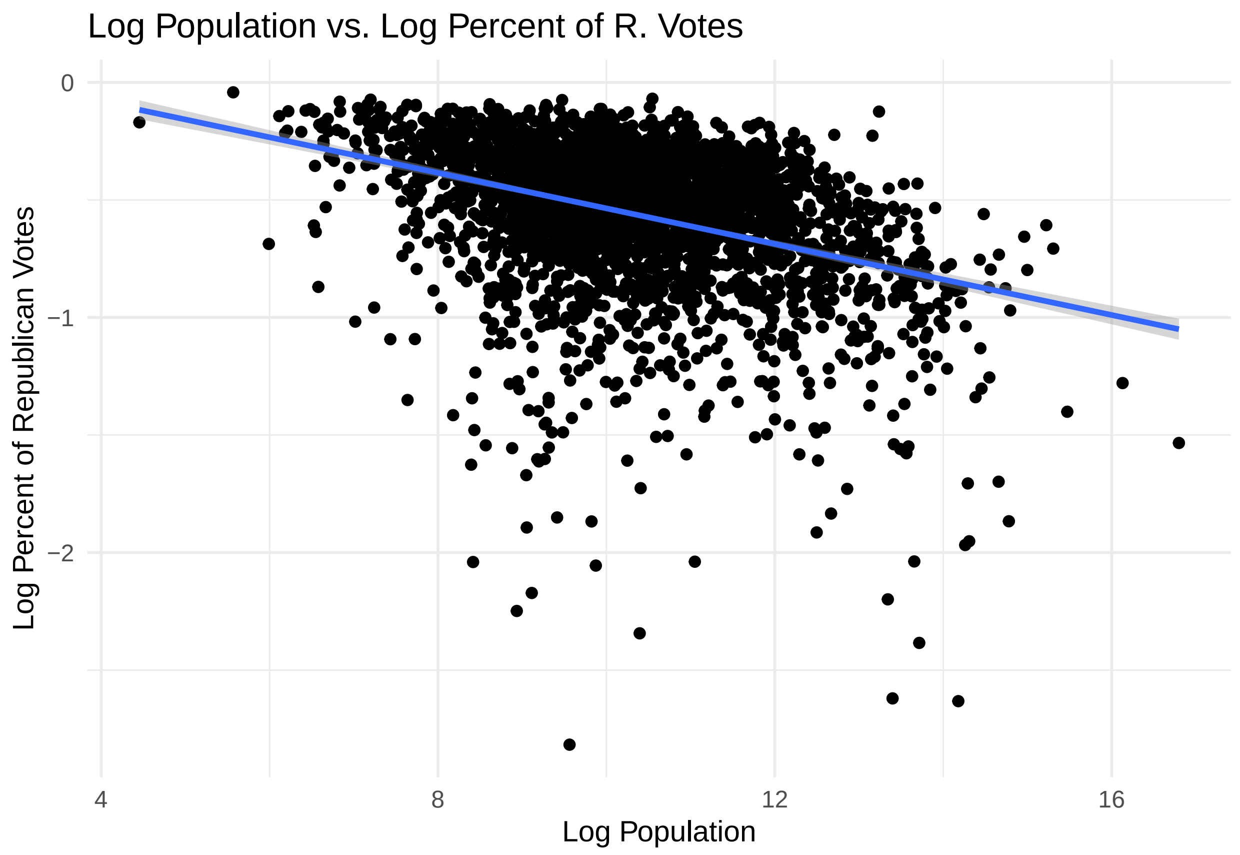


What I find interesting about this visualization is that the proportion of republican-majority counties is consistently larger than the proportion of republican votes, indicating that republican votes may be disproportionately represented. In other words, republicans seem to be winning more counties than they "should" given their voting population. Given the American democratic system, this is expected.

```r
# Plot population vs. pct of r votes
election_plot_df <- election %>%
  mutate(
  pct_r_votes = n_votes_republican_2012 / n_votes_total_2012
  )

ggplot(data = election_plot_df, aes(x = log(pop), y = log(pct_r_votes))) +
  geom_point()+
  geom_smooth(method='lm') +
  labs(
    title = "Log Population vs. Log Percent of R. Votes",
    x = "Log Population",
    y = "Log Percent of Republican Votes"
  ) +
  theme_minimal()
```

```
## `geom_smooth()` using formula = 'y ~ x'
```



This shows a weakly negative relationship between population and republican votes, suggesting more populous areas are more likely democratic-aligned. This is expected.

Penalized Regression

Using 5-fold cross validation, tune a Lasso model

```
# Recipe
rec_1 <- recipe(data = election, i_republican_2016 ~ .) %>%
  update_role(fips, new_role = "id") %>%
  step_novel(county, state) %>%
  step_dummy(i_republican_2016_f, i_republican_2012, county, state) %>%
  step_normalize(all_numeric_predictors())
```

```r
# Model : Lasso
mod_lasso <- linear_reg(penalty = tune(),mixture = 1) %>%
  set_engine("glmnet")

# Workflow
wkfl_lasso <- workflow() %>%
  add_model(mod_lasso) %>%
  add_recipe(rec_1)

# Cross Validate
elec_cv <- election %>% vfold_cv(v = nfold)

cv_lasso <- wkfl_lasso %>%
  tune_grid(
    elec_cv,
    grid = data.frame(penalty = lambdas),
    metrics = metric_set(rmse)
  )
```

```
## > A | warning: !  The following columns have zero variance so scaling cannot be used:
##                    county_new and state_new.
##                  i Consider using ?step_zv (‘?recipes::step_zv()‘) to remove those columns
##                    before normalizing.

## There were issues with some computations   A: x1There were issues with some computations   A: x2There
```

```r
# Find optimal penalty

cv_lasso %>% show_best()
```

```
## # A tibble: 5 x 7
##   penalty .metric .estimator   mean     n  std_err .config
##     <dbl> <chr>   <chr>       <dbl> <int>    <dbl> <chr>
## 1  0.01   rmse    standard   0.0106     5 0.000131 Preprocessor1_Model0001
## 2  0.0102 rmse    standard   0.0106     5 0.000131 Preprocessor1_Model0002
## 3  0.0103 rmse    standard   0.0106     5 0.000131 Preprocessor1_Model0003
## 4  0.0105 rmse    standard   0.0106     5 0.000131 Preprocessor1_Model0004
## 5  0.0107 rmse    standard   0.0107     5 0.000154 Preprocessor1_Model0005
```

```r
# Tune elasticnet prediction model
```

Highest Performing Lambda:

$$\lambda = 0.01$$

Which metric was used? RMSE was the metric used. This is not a very fitting choice, as this is a classification regression. Instead, minimizing Gini or Entropy would be preferable.

Tune an Elasticnet model

```r
# Elasticnet Model
mod_ent <- linear_reg(
  penalty = tune(), mixture = tune()
) %>%
  set_engine(
    "glmnet"
  )

# Workflow
wkfl_ent <- workflow() %>%
  add_model(mod_ent) %>%
  add_recipe(rec_1)

# Cross validating alphas and lambdas
cv_ent <- wkfl_ent %>%
  tune_grid(
    elec_cv,
    grid = expand_grid(mixture = alphas, penalty = lambdas),
    metrics = metric_set(rmse)
  )
```

```
## > A | warning: !  The following columns have zero variance so scaling cannot be used:
##                    county_new and state_new.
##                  i Consider using ?step_zv (`?recipes::step_zv()`) to remove those columns
##                    before normalizing.
```

```
## There were issues with some computations   A: x1There were issues with some computations   A: x2There
```

```r
cv_ent %>% show_best()
```

```
## # A tibble: 5 x 8
##   penalty mixture .metric .estimator   mean     n  std_err .config
##     <dbl>   <dbl> <chr>   <chr>       <dbl> <int>    <dbl> <chr>
## 1  0.01      1    rmse    standard   0.0106     5 0.000131 Preprocessor1_Model2~
## 2  0.0102    1    rmse    standard   0.0106     5 0.000131 Preprocessor1_Model2~
## 3  0.0103    1    rmse    standard   0.0106     5 0.000131 Preprocessor1_Model2~
## 4  0.0105    1    rmse    standard   0.0106     5 0.000131 Preprocessor1_Model2~
## 5  0.01      0.75 rmse    standard   0.0107     5 0.000129 Preprocessor1_Model1~
```

Highest Performing Lambda and Alpha:

$$\lambda = 0.01,\ \ \alpha = 1.00$$

The algorithm tuned α to 1, indicating that a Lasso regression dominates Ridge in this setting.

Logistic Regression

Use 5-fold cross validation to tune a logistic regression model

```r
mod_log <- logistic_reg(
  penalty = tune(),
  mixture = tune()
) %>%
  set_engine("glmnet")

# Recipe
rec_2 <- recipe(data = election, i_republican_2016_f ~ .) %>%
  update_role(fips, new_role = "id") %>%
  step_novel(county, state) %>%
  step_dummy(i_republican_2012, county, state) %>%
  step_normalize(all_numeric_predictors())

wkfl_log <- workflow() %>%
  add_model(mod_log) %>%
  add_recipe(rec_2)

# Cross validating alphas and lambdas
cv_log <- wkfl_log %>%
  tune_grid(
    elec_cv,
    grid = expand_grid(mixture = alphas, penalty = lambdas),
    metrics = metric_set(accuracy)
  )
```

```
## > A | warning: !  The following columns have zero variance so scaling cannot be used:
##                    county_new and state_new.
##                  i Consider using ?step_zv (‘?recipes::step_zv()‘) to remove those columns
##                    before normalizing.

## There were issues with some computations   A: x1There were issues with some computations   A: x2There
```

```r
cv_log %>% show_best()
```

```
## Warning in show_best(.): No value of ‘metric‘ was given; "accuracy" will be
## used.

## # A tibble: 5 x 8
##   penalty mixture .metric  .estimator  mean     n std_err .config
##     <dbl>   <dbl> <chr>    <chr>      <dbl> <int>   <dbl> <chr>
## 1  0.01       0.1 accuracy binary         1     5       0 Preprocessor1_Model02~
## 2  0.0102     0.1 accuracy binary         1     5       0 Preprocessor1_Model02~
## 3  0.0103     0.1 accuracy binary         1     5       0 Preprocessor1_Model02~
## 4  0.0105     0.1 accuracy binary         1     5       0 Preprocessor1_Model02~
## 5  0.0107     0.1 accuracy binary         1     5       0 Preprocessor1_Model02~
```

```r
# logistic_pred = predict(wkfl_log %>% fit(new_data = election), type = "response")



# Find accuracy, precision, specificity, sensitivity, ROC AUC
```

Highest Performing Lambda and Alpha:

$$\lambda = 0.01, \ \alpha = 0.1$$

Logistic Lasso Regression

```r
# Use 5-fold cross validation to tune a logistic lasso regression model

# Find accuracy, precision, specificity, sensitivity, ROC AUC
```

Reflection