MC - Ridge Regression

Andi Mellyn

2025-12-01

Monte Carlo Simulation using a Ridge Regression Model

```
# sourcing the simulated functions

source("simulate_functions.R")
```

```
# ridge logistic regression function

fit_logistic_ridge <- function(dat) {
  # dat: data.frame with first column y and the rest predictors

  X <- as.matrix(dat[ , -1])  # all columns except the first
  y <- dat$y                  # first column is y

  # alpha = 0 -> ridge penalty
  cvfit <- cv.glmnet(
    x = X,
    y = y,
    alpha = 0,
    family = "binomial"
  )

  cvfit
}
```

```
# ridge-specific evaluate model

evaluate_ridge_model <- function(cvfit, dat) {
  X <- as.matrix(dat[ , -1])
  y <- dat$y

  # lambda.min from CV (the value that minimizes the error)
  probs <- predict(cvfit, newx = X, type = "response", s = "lambda.min")

  auc_val <- pROC::auc(y, probs)
  brier <- mean((y - probs)^2)

  tibble::tibble(
    AUC = as.numeric(auc_val), # AUC: area under the ROC curve
    Brier = brier, # brier: measure of the accuracy of probabilistic predictions
```

```
    lambda = cvfit$lambda.min
  )
}
```

```
# testing the ridge model for one run

set.seed(400)

# simulating one dataset
dat_test <- simulate_logit(N = 1000, P = 8, rho = 0.5, beta_pattern = "strong")

ridge_fit <- fit_logistic_ridge(dat_test)
ridge_summary <- evaluate_ridge_model(ridge_fit, dat_test)

ridge_summary
```

```
## # A tibble: 1 x 3
##     AUC Brier lambda
##   <dbl> <dbl>  <dbl>
## 1 0.864 0.150 0.0274
```

```
# Monte Carlo simulation settings

EPV_vals <- c(3, 10, 50)          # subset -> research study uses 7 values
event_frac_vals <- c(1/2, 1/8)    # subset -> research study uses 4 fractions
P_vals <- c(4, 8, 12)             # same p-vals as study
rho_vals <- c(0, 0.3, 0.5)        # same continuous predictors
beta_patterns <- c("equal", "strong", "noise", "halfnoise")    # same beta patterns

B <- 10  # number of Monte Carlo repetitions per condition
```

```
# Monte Carlo loop for ridge regression

set.seed(400)

ridge_results <- list()

iter <- 1

for (EPV in EPV_vals) {
  for (event_frac in event_frac_vals) {
    for (P in P_vals) {
      for (rho in rho_vals) {
        for (bp in beta_patterns) {
          for (b in 1:B) {

            # N <- hypothetical individuals
            N <- ceiling(EPV * P / event_frac)

            # simulate one dataset
            dat <- simulate_logit(N = N, P = P, rho = rho, beta_pattern = bp)

            # fit ridge model
            ridge_fit <- fit_logistic_ridge(dat)
```

```
            # evaluate performance on this dataset
            ridge_summary <- evaluate_ridge_model(ridge_fit, dat)

            # store results
            ridge_results[[iter]] <- tibble(
              method = "ridge",
              sim = b,
              P = P,
              rho = rho,
              beta_pattern = bp,
              AUC = ridge_summary$AUC,
              Brier = ridge_summary$Brier,
              lambda = ridge_summary$lambda
            )
            iter <- iter + 1
          }

        }

      }

    }

  }

}


ridge_results_df <- bind_rows(ridge_results)
head(ridge_results_df)
```

```
## # A tibble: 6 x 8
##   method   sim     P   rho beta_pattern   AUC Brier  lambda
##   <chr>  <int> <dbl> <dbl> <chr>        <dbl> <dbl>   <dbl>
## 1 ridge      1     4     0 equal        0.729 0.243  96.2
## 2 ridge      2     4     0 equal        0.846 0.179   0.226
## 3 ridge      3     4     0 equal        0.736 0.250 103.
## 4 ridge      4     4     0 equal        0.814 0.193   0.353
## 5 ridge      5     4     0 equal        0.821 0.195   0.310
## 6 ridge      6     4     0 equal        0.741 0.248  84.2
```

```
# matching the research paper summary stats

ridge_condition_summary <- ridge_results_df |>
  group_by(P, rho, beta_pattern) |>
  summarise(
    n_sims = n(),
    mean_AUC = mean(AUC),
    sd_AUC = sd(AUC),
    mean_Brier = mean(Brier),
    sd_Brier = sd(Brier),
    mean_lambda = mean(lambda),
    .groups = "drop"
  )
```

```
ridge_condition_summary
```

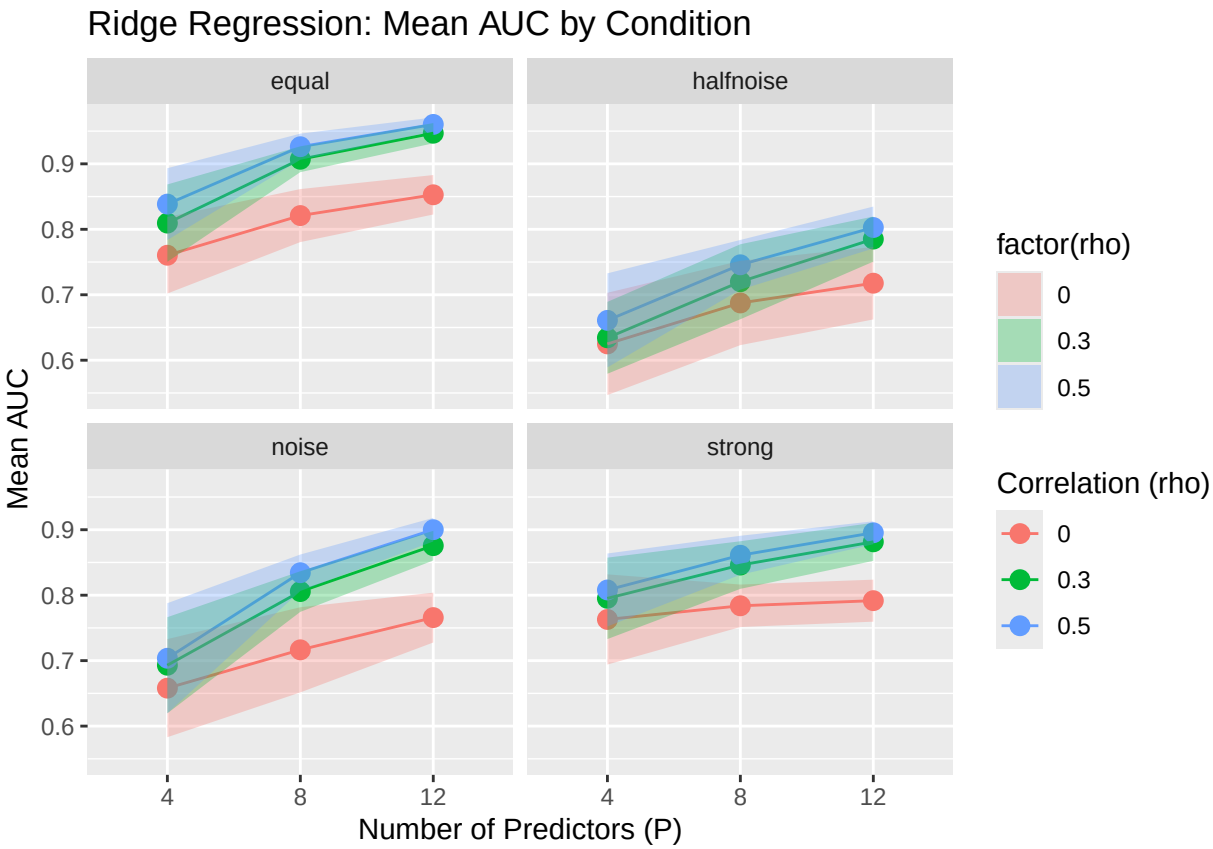
```
## # A tibble: 36 x 9
##        P   rho beta_pattern n_sims mean_AUC sd_AUC mean_Brier sd_Brier
##    <dbl> <dbl> <chr>         <int>    <dbl>  <dbl>      <dbl>    <dbl>
##  1     4   0   equal            60    0.760 0.0585      0.200  0.0219
##  2     4   0   halfnoise        60    0.625 0.0781      0.237  0.0162
##  3     4   0   noise            60    0.658 0.0749      0.228  0.0202
##  4     4   0   strong           60    0.763 0.0689      0.196  0.0235
##  5     4   0.3 equal            60    0.809 0.0591      0.176  0.0253
##  6     4   0.3 halfnoise        60    0.634 0.0549      0.235  0.00893
##  7     4   0.3 noise            60    0.693 0.0734      0.220  0.0265
##  8     4   0.3 strong           60    0.795 0.0622      0.182  0.0265
##  9     4   0.5 equal            60    0.839 0.0548      0.162  0.0299
## 10     4   0.5 halfnoise        60    0.661 0.0718      0.227  0.0208
## # i 26 more rows
## # i 1 more variable: mean_lambda <dbl>
```

```
knitr::kable(ridge_condition_summary, digits = 3)
```

P	rho	beta_pattern	n_sims	mean_AUC	sd_AUC	mean_Brier	sd_Brier	mean_lambda
4	0.0	equal	60	0.760	0.058	0.200	0.022	7.257
4	0.0	halfnoise	60	0.625	0.078	0.237	0.016	20.250
4	0.0	noise	60	0.658	0.075	0.228	0.020	9.198
4	0.0	strong	60	0.763	0.069	0.196	0.023	3.648
4	0.3	equal	60	0.809	0.059	0.176	0.025	1.373
4	0.3	halfnoise	60	0.634	0.055	0.235	0.009	17.057
4	0.3	noise	60	0.693	0.073	0.220	0.027	7.080
4	0.3	strong	60	0.795	0.062	0.182	0.026	1.407
4	0.5	equal	60	0.839	0.055	0.162	0.030	1.228
4	0.5	halfnoise	60	0.661	0.072	0.227	0.021	12.611
4	0.5	noise	60	0.704	0.084	0.216	0.022	6.888
4	0.5	strong	60	0.808	0.056	0.176	0.034	0.081
8	0.0	equal	60	0.821	0.040	0.174	0.019	1.318
8	0.0	halfnoise	60	0.687	0.064	0.223	0.016	4.399
8	0.0	noise	60	0.717	0.065	0.214	0.016	4.610
8	0.0	strong	60	0.784	0.032	0.193	0.016	2.355
8	0.3	equal	60	0.907	0.020	0.124	0.014	0.041
8	0.3	halfnoise	60	0.720	0.057	0.214	0.016	5.175
8	0.3	noise	60	0.806	0.031	0.180	0.015	0.096
8	0.3	strong	60	0.846	0.036	0.160	0.022	0.060
8	0.5	equal	60	0.926	0.020	0.110	0.016	0.039
8	0.5	halfnoise	60	0.746	0.038	0.205	0.015	0.174
8	0.5	noise	60	0.834	0.028	0.166	0.014	0.096
8	0.5	strong	60	0.861	0.030	0.151	0.020	0.069
12	0.0	equal	60	0.853	0.030	0.157	0.016	0.027
12	0.0	halfnoise	60	0.718	0.055	0.215	0.018	1.736
12	0.0	noise	60	0.766	0.038	0.199	0.015	0.089
12	0.0	strong	60	0.792	0.032	0.188	0.014	1.997
12	0.3	equal	60	0.947	0.015	0.093	0.015	0.032
12	0.3	halfnoise	60	0.785	0.034	0.189	0.017	0.135
12	0.3	noise	60	0.876	0.023	0.144	0.014	0.059
12	0.3	strong	60	0.882	0.029	0.140	0.020	0.047

P	rho	beta_pattern	n_sims	mean_AUC	sd_AUC	mean_Brier	sd_Brier	mean_lambda
12	0.5	equal	60	0.960	0.011	0.081	0.012	0.035
12	0.5	halfnoise	60	0.803	0.032	0.181	0.015	0.121
12	0.5	noise	60	0.900	0.018	0.129	0.011	0.063
12	0.5	strong	60	0.895	0.017	0.132	0.012	0.064

```
# AUC plot

ggplot(ridge_condition_summary,
       aes(x = factor(P), y = mean_AUC, group = factor(rho), color = factor(rho))) +
  geom_point(size = 3) +
  geom_line() +
  facet_wrap(~ beta_pattern) +
  labs(
    title = "Ridge Regression: Mean AUC by Condition",
    x = "Number of Predictors (P)",
    y = "Mean AUC",
    color = "Correlation (rho)"
  ) +
  geom_ribbon(aes(ymin = mean_AUC - sd_AUC,
                  ymax = mean_AUC + sd_AUC,
                  fill = factor(rho)),
              alpha = 0.30, color = NA)
```




```
# Brier plot
```

```
ggplot(ridge_condition_summary,
       aes(x = factor(P), y = mean_Brier, group = factor(rho), color = factor(rho))) +
  geom_point(size = 3) +
  geom_line() +
  facet_wrap(~ beta_pattern) +
  labs(
    title = "Ridge Regression: Mean Brier Score by Condition",
    x = "Number of Predictors (P)",
    y = "Mean Brier Score",
    color = "Correlation (rho)"
  ) +
  geom_ribbon(aes(ymin = mean_Brier - sd_Brier,
                  ymax = mean_Brier + sd_Brier,
                  fill = factor(rho)),
              alpha = 0.15, color = NA)
```

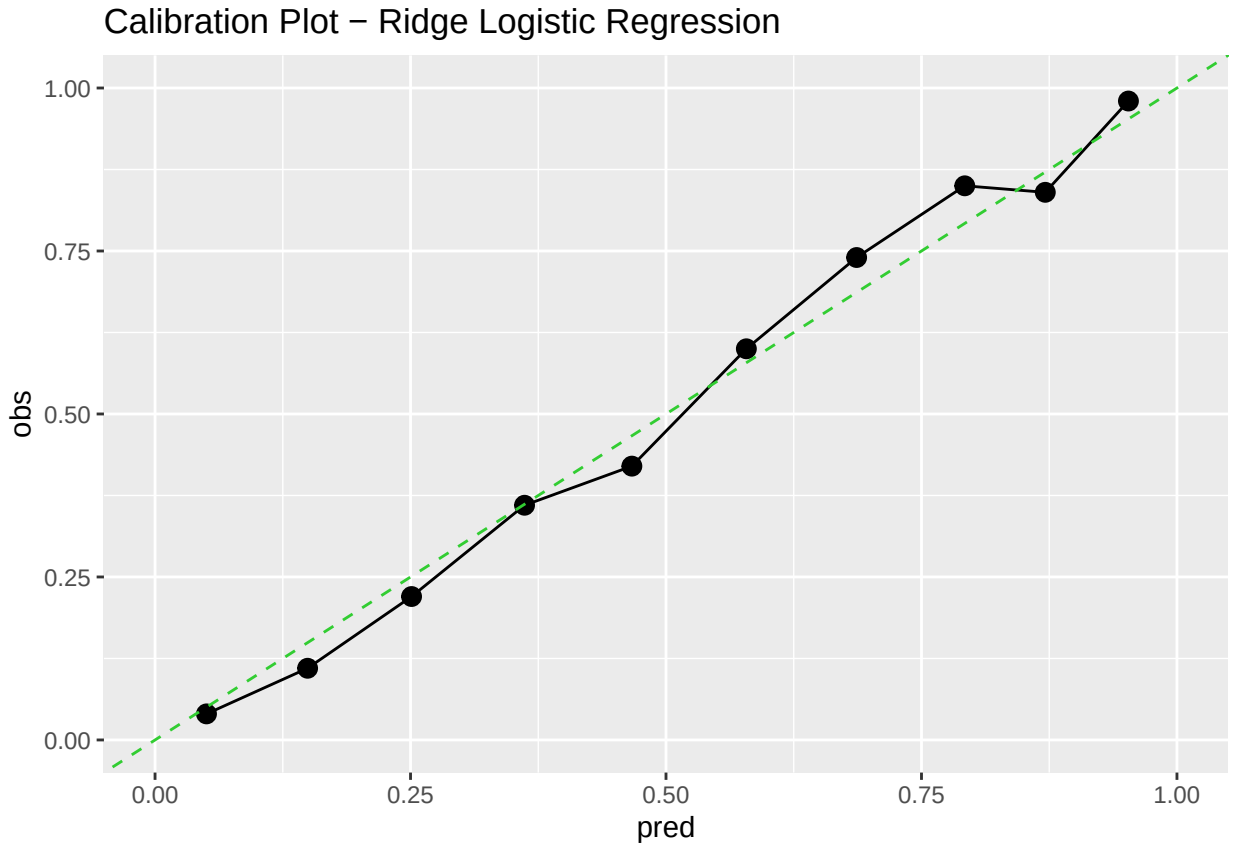

```
# calibration plot - how well do the predicted values line up with the actual observed values

set.seed(400)

# simulating a single dataset for calibration
dat_cal <- simulate_logit(
  N = 1000,
  P = 8,
  rho = 0.5,
  beta_pattern = "strong"
)
```

```r
# fit ridge logistic regression
ridge_fit_cal <- fit_logistic_ridge(dat_cal)
```

```r
# design matrix for glmnet
X_cal <- as.matrix(dat_cal[ , -1])

# predicted probabilities from the ridge model
ridge_probs <- predict(
  ridge_fit_cal,
  newx = X_cal,
  type = "response",
  s = "lambda.min"
)

# build calibration dataframe
ridge_cal_df <- dat_cal |>
  mutate(prob = as.numeric(ridge_probs),
         decile = ntile(prob, 10)) |>
  group_by(decile) |>
  summarise(obs = mean(y), pred = mean(prob), .groups = "drop")

ridge_cal_df
```

```
## # A tibble: 10 x 3
##    decile   obs   pred
##     <int> <dbl>  <dbl>
##  1      1  0.04 0.0503
##  2      2  0.11 0.149
##  3      3  0.22 0.251
##  4      4  0.36 0.362
##  5      5  0.42 0.467
##  6      6  0.6  0.579
##  7      7  0.74 0.687
##  8      8  0.85 0.792
##  9      9  0.84 0.871
## 10     10  0.98 0.953
```

```r
ggplot(ridge_cal_df, aes(x = pred, y = obs)) +
  geom_point(size = 3) +
  geom_line() +
  geom_abline(slope = 1, intercept = 0, color = "limegreen", linetype = 2) +
  xlim(0, 1) + ylim(0, 1) +
  labs(
    title = "Calibration Plot - Ridge Logistic Regression"
  )
```


Calibration Plot − Ridge Logistic Regression
1.00
0.75
0.50
0.25
0.00
obs
0.00
0.25
0.50
0.75
1.00
pred