MC Model Comparisons

Group 3

2025-12-01

Minimum Liklihood, Ridge, and Random Forest: A Monte Carlo Model Comparison

0. Source functions and simulated data for model comparisons

```r
# sourcing method functions

source("simulate_functions.R")  # for
```

```r
# sourcing simulated data

simulated_data <- readRDS("data/simulated_data.rds")
```

1. Background

replicating "Sample size for binary logistic prediction models: Beyond events per variable criteria" by Smeden et al.

out-of-sample performance of developed prediction models studied before and after regression shrinkage and variable selection

concluded EPV does not have a strong relation with metrics of predictive performance and is not an appropriate cirterian for binary prediction model development studies

better approximated by considering number of predictors, total sample size, and events fraction

key contributing factor to predictive performance is size of data set relative to number of predictors

events per variable: ratio of number of events, the number of observations in smaller of two outcome groups, relative to the number of degrees of freedom/parameters required to represent the predictors considered in the model

medical literature uses EPV of 10 as minimum but there is skepticism as to the validity of the scientific backing of this benchmark. in step-wise predictor selection, EPV greater than 50 may be needed. Good out-of-sample predictive performance demonstrated with regression shrinkage techniques with EPV less than 10.

modeling: predictive performance evaluated on large sample validation data sets maximum likelihood (ML), ridge regression, LASSO, Firth's correction, heuristic shrinkage after ML estimation, backwards elimination predictor selection

EPV, the events fraction, number of candidate predictors, area under the ROC curve, distribution of predictor variables, type of predictor variable effects

factorial simulation study with 4,032 models 5,000 simulation runs

1. development data set generated
    a. for each of N = EPV * P/Pr(Y = 1) individuals, draw predictor variable vector $\mathbf{x}_i$
    b. for each individual, generate y_i = Bernoulli(π_i)
2. fit binary logistic prediction models with each of the considered regression shrinkage and predictor selection strategies
3. large validation data set generated, N* = 5000/Pr(Y = 1), using sampling approach from (1)
4. evaluate performance of prediction models (2) on validation data (3)

issues with maximum likelihood:

1. not optimal for making model predictions of the expected risk in new individuals – in most circumstances shrinkage estimators can be defined that have lower expected error for estimating probabilities in new individuals than ML estimators; benefit of shrinkage decreases with increasing EPV
2. ML coefficients are biased toward more extreme effects; bias reduces with increasing EPV but may not completely disappear
3. model estimation becomes unstable when predictor effects are large or sparse – results in extreme probability estimates close to boundaries of 0 or 1; less likely with increasing EPV
4. estimation becomes unstable when predictors are strongly correlated, inflates standard errors for predictor effects; spurious collinearity less likely with increasing EPV

Ridge regression: penalizes the likelihood proportionally to the sum of squared predictor effects

predictors standardized to have mean zero and unit variance

10-fold cross validation used to determine the optimal value for the tuning parameter

used to deal with collinearity (4), can also be used to deal with separation (3)

2. Maximum Liklihood

```
# Maximum Likelihood
```

```
simulate_logit <- function(N, P, rho, beta_pattern) {

    # Predictor distribution Correlation = rho (0 or 0.5)
    Sigma <- matrix(rho, P, P)
    diag(Sigma) <- 1

    X <- mvrnorm(n = N, mu = rep(0, P), Sigma = Sigma)

    # Predictor effects
    if (beta_pattern == "equal") {
        beta <- rep(0.5, P)

    } else if (beta_pattern == "strong") {
        beta <- c(1, rep(0.2, P - 1))  # 1 strong effect

    } else if (beta_pattern == "noise") {
        beta <- c(0, rep(0.3, P - 1))  # first predictor is noise
```

```r
    } else if (beta_pattern == "halfnoise") {
        beta <- c(rep(0, P/2), rep(0.3, P/2))  # first half noise

    } else {
        stop("Unknown beta pattern")
    }

    # Linear predictor + logistic link
    eta <- X %*% beta
    pi <- 1/(1 + exp(-eta))

    # Binary outcome from Bernoulli
    y <- rbinom(N, size = 1, prob = pi)

    data.frame(y = y, X)
}
```

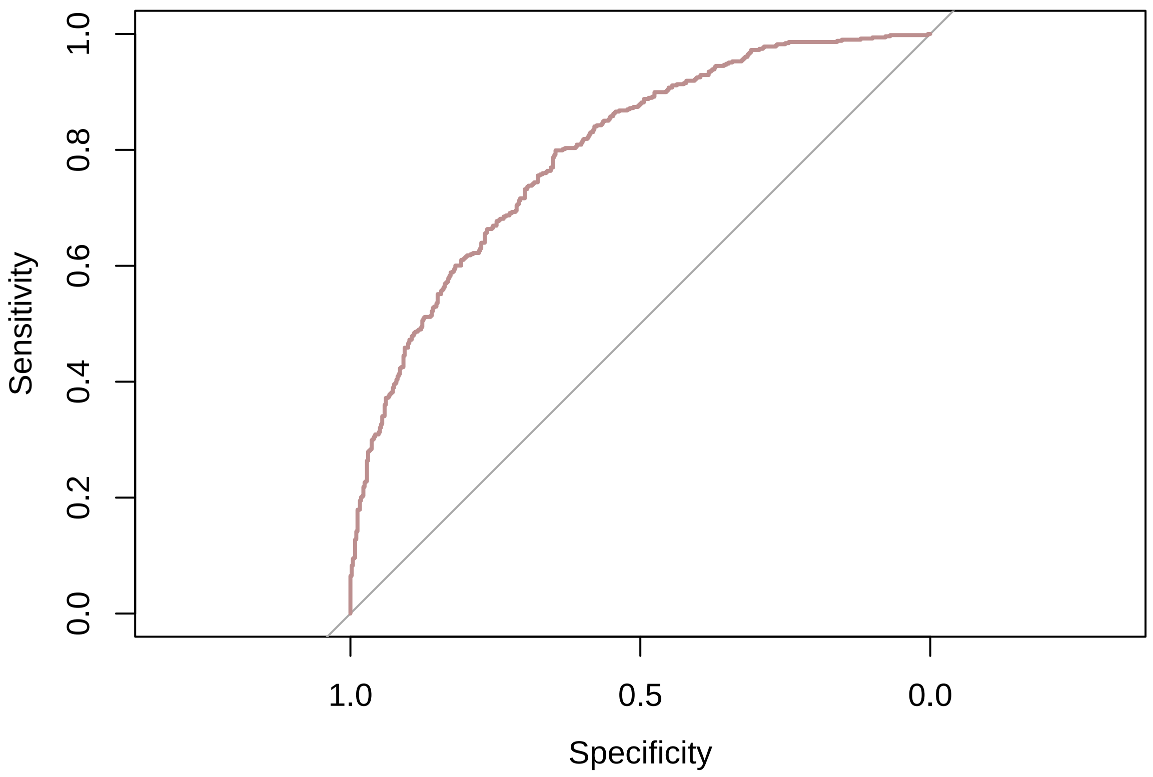
```r
fit_logistic_mle <- function(dat) {
    glm(y ~ ., data = dat, family = binomial)
}
```

```r
evaluate_model <- function(fit, dat) {

  probs <- predict(fit, type = "response")

  auc_val <- auc(dat$y, probs)
  brier <- mean((dat$y - probs)^2)

  tibble::tibble(
    # AUC: Area Under the ROC Curve
    AUC = as.numeric(auc_val),
    # brier: measure of the accuracy of probabilistic predictions
    Brier = brier
  )
}
```

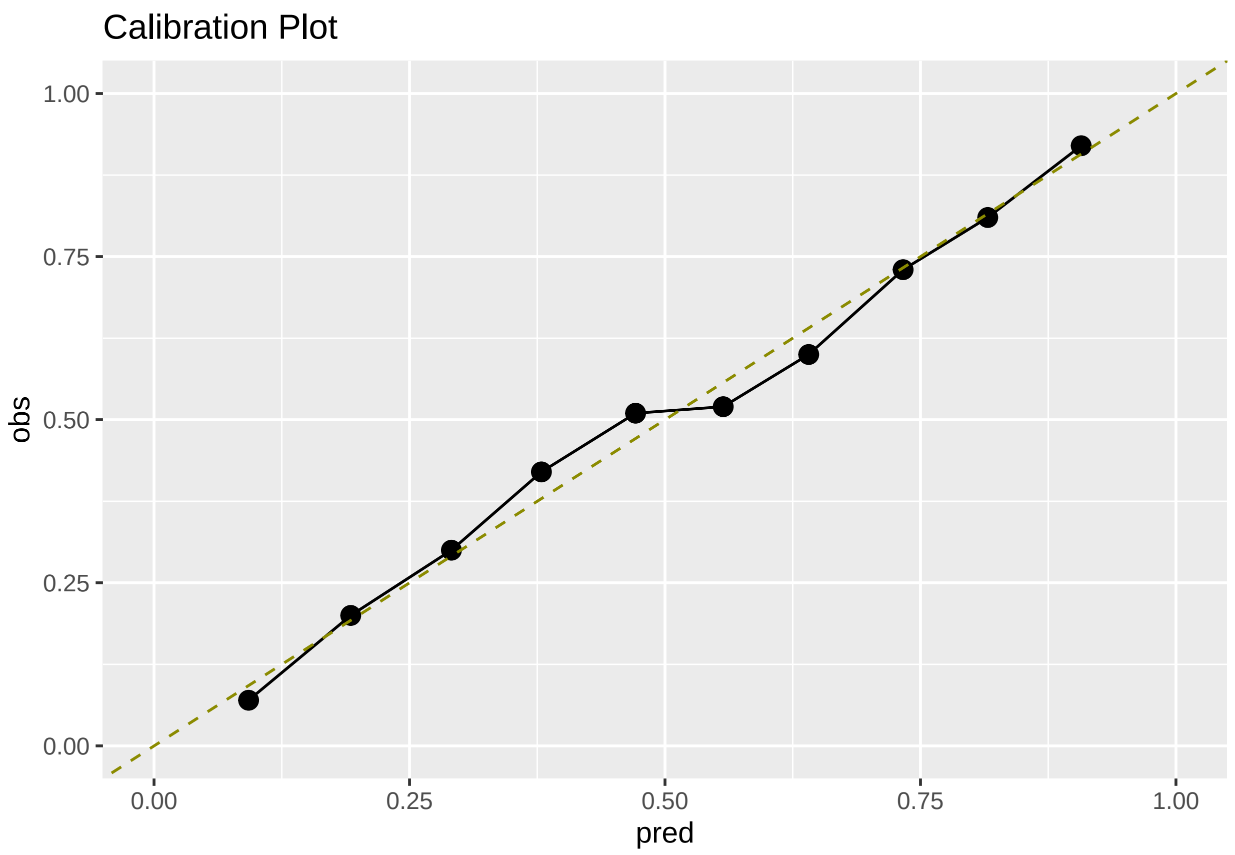
```r
P_vals <- c(4, 8, 12)
rho_vals <- c(0, 0.5)
beta_patterns <- c("equal", "strong", "noise", "halfnoise")

results <- list()
iter <- 1

for (P in P_vals) {
    for (rho in rho_vals) {
        for (bp in beta_patterns) {

            # One simulation run
            dat <- simulate_logit(N = 1000, P = P, rho = rho, beta_pattern = bp)

            # Maximum likelihood fit
            fit <- fit_logistic_mle(dat)

            # Evaluate predictive performance
            perf <- evaluate_model(fit, dat)

            # Save results
            results[[iter]] <- cbind(P = P, rho = rho, beta_pattern = bp, perf)

            iter <- iter + 1
        }
    }
}

final_results <- bind_rows(results)
knitr::kable(final_results)
```

P	rho	beta_pattern	AUC	Brier
4	0.0	equal	0.7305510	0.2100762
4	0.0	strong	0.7490588	0.2031413
4	0.0	noise	0.6436932	0.2349203
4	0.0	halfnoise	0.6123144	0.2401857
4	0.5	equal	0.8147801	0.1757036
4	0.5	strong	0.8035815	0.1805105
4	0.5	noise	0.6808568	0.2237958
4	0.5	halfnoise	0.6703052	0.2278287
8	0.0	equal	0.8001808	0.1822946
8	0.0	strong	0.7732583	0.1943832
8	0.0	noise	0.7272596	0.2113413
8	0.0	halfnoise	0.6778889	0.2257083
8	0.5	equal	0.9285414	0.1083215
8	0.5	strong	0.8652514	0.1484563
8	0.5	noise	0.8259852	0.1704028
8	0.5	halfnoise	0.7164916	0.2143960
12	0.0	equal	0.8318367	0.1673038
12	0.0	strong	0.7872535	0.1875502
12	0.0	noise	0.7517311	0.2021622
12	0.0	halfnoise	0.6745791	0.2256199
12	0.5	equal	0.9558096	0.0832955
12	0.5	strong	0.8755092	0.1431618
12	0.5	noise	0.8982046	0.1297300
12	0.5	halfnoise	0.7964959	0.1838043

```
roc_obj <- roc(dat$y, predict(fit, type = "response"))
plot(roc_obj, col = "rosybrown", lwd = 2)
```



```r
library(ggplot2)

cal_df <- dat %>%
    mutate(prob = predict(fit, type = "response"), decile = ntile(prob, 10)) %>%
    group_by(decile) %>%
    summarize(obs = mean(y), pred = mean(prob))

ggplot(cal_df, aes(pred, obs)) + geom_point(size = 3) + geom_line() + geom_abline(slope = 1,
    intercept = 0, color = "yellow4", lty = 2) + ylim(0, 1) + xlim(0, 1) + ggtitle("Calibration Plot")
```



3. Ridge Regression

```
# ridge regression
```

4. Random Forest

```
# random forest
```

5. Comparisons

```
# comparisons
```