# Simulation Results

Justin M. Leach

## 1 What is contained herein?

We ran a set of simulations inspired by the original SSL paper. Specifically, we generate a training set with $N = 300$ subjects and a test set with $N = 100$ subjects. The design vector $\boldsymbol{X}$ contains 1,000 predictors, which are generated from the multivariate normal distribution with mean zero and unit variance, but with a block correlation structure consisting of 20 blocks of 50 predictors, where within each block the predictors have a 0.90 correlation with each other. A total of 6 of these predictors are given non-zero values, specifically the predictors with indices, 1, 51, 101, 151, 201, 251. We examine 3 effect size scenarios. Each scenario treats one group as a reference, i.e., all its parameters are 0. The second and third groups are given non-zero parameter values at the appropriate indices. The settings are as follows:

(1) Group 2: $\beta_0 = -0.15$, $\beta = 0.0075$; Group 3: $\beta_0 = -0.5$, $\beta = 0.015$.
(2) Group 2: $\beta_0 = -0.15$, $\beta = 0.0125$; Group 3: $\beta_0 = -0.5$, $\beta = 0.025$.
(3) Group 2: $\beta_0 = -0.15$, $\beta = 0.025$; Group 3: $\beta_0 = -0.5$, $\beta = 0.05$.

These scenarios generate imbalanced data, where group 1 is most common, group 2 secondmost common, and group 3 least common.

Table 1: Sample Balance in Scenario 1

| | Group_1 | Group_2 | Group_3 |
|---|---|---|---|
| Freq. (Mean) | 121.94 | 103.44 | 74.62 |
| Freq. (SD) | 8.33 | 8.19 | 7.29 |
| Percent (Mean) | 40.51 | 34.62 | 24.87 |
| Percent (SD) | 4.80 | 4.69 | 4.39 |

Table 2: Sample Balance in Scenario 2

| | Group_1 | Group_2 | Group_3 |
|---|---|---|---|
| Freq. (Mean) | 122.23 | 101.75 | 76.02 |
| Freq. (SD) | 8.34 | 8.16 | 7.40 |
| Percent (Mean) | 40.62 | 34.04 | 25.34 |
| Percent (SD) | 4.90 | 4.65 | 4.39 |

Table 3: Sample Balance in Scenario 3

| | Group_1 | Group_2 | Group_3 |
|---|---|---|---|
| Freq. (Mean) | 123.67 | 94.82 | 81.51 |

| | Group_1 | Group_2 | Group_3 |
|---|---|---|---|
| Freq. (SD) | 8.47 | 7.99 | 7.65 |
| Percent (Mean) | 41.08 | 31.72 | 27.20 |
| Percent (SD) | 4.86 | 4.67 | 4.52 |

The training set is used to perform 5-fold cross validation to select optimal penalty parameters for the elastic net ($\alpha = 0.5$), lasso ($\alpha = 1$), and spike-and-slab elastic net, and spike-and-slab lasso. That is, $\lambda = s_0 = s_1$ for the traditional models and a spike scale, $s_0$, and slab scale, $s_1$, for the spike-and-slab models. For the traditional models, the range of $\lambda$ is chosen internally by the `R` package `glmnet`. A grid of values is preselected for the spike-and-slab models, $s_0 = \{0.01, 0.02, 0.03, \ldots, 0.1\}$ and $s1 = \{1, 2, 3, 4\}$, which are then fit with the `R` package `ssnet`. We then use the selected penalty parameters to fit the model on the entire training data. This model is then applied to predict the observations in the test data and calculate measures of model fitness.

# 2 Combining files

As usual, simulation analyses take too long, so we break them up into separate (array) jobs on UAB's Cheaha. The following function puts them back together again.

```
combine_files <- function(
  path = "C:/Users/cotto/Documents/Publications/multinomial models in R/simulations/results/",
  files, # names of files to combine
  rename_sims = TRUE, # When combining files simulation indices repeat, so fix it here
  number_outcomes = 3, #
  outfile = NULL # specify file name to write results to path directory
) {

  fit_time <- NULL
  model_fitness <- NULL
  param_est <- NULL
  for (i in 1:length(files)) {

    results_i <- readRDS(
      paste0(path, files[i])
    )

    if (rename_sims == TRUE) {
      if (i == 1) {
        sim_num_i <- 1:nrow(results_i$model_fitness)
      } else {
        sim_num_i <- max(sim_num_i) + 1:nrow(results_i$model_fitness)
      }
      print(c(min(sim_num_i), max(sim_num_i)))
      # print(length(sim_num_i))
    }

    fit_time <- rbind(
      fit_time,
      results_i$fit_time
    )
```

```
    if (rename_sims == TRUE) {
      # print(length(results_i$model_fitness$sim.num))
      results_i$model_fitness$sim.num <- sim_num_i
      cat("Model fitness sim IDs renamed for file ", i, "\n")
    }
    model_fitness <- rbind(
      model_fitness,
      results_i$model_fitness
    )

    if (rename_sims == TRUE) {
      results_i$param_est$sim.num <- rep(
        x = sim_num_i,
        each = number_outcomes
      )
      cat("Parameter sim IDs renamed for file ", i, "\n")
    }

    param_est <- rbind(
      param_est,
      results_i$param_est
    )
  }
  out <- list(
    fit_time = apply(fit_time[, 1:3], 2, sum),
    model_fitness = model_fitness,
    param_est = param_est
  )
  if (is.null(outfile) == FALSE) {
    saveRDS(
      out,
      file = paste0(path, outfile)
    )
  }
  return(out)
}
```

### 2.1 B6_BV0075015

```
results_lasso_M5000_X1k_B6_BV0075015_NTR300_NTE100_NCV5 <- combine_files(
  files = c(
    "results_lasso_M5000_X1k_B6_BV0075015_NTR300_NTE100_NCV5_1.RDS",
    "results_lasso_M5000_X1k_B6_BV0075015_NTR300_NTE100_NCV5_2.RDS",
    "results_lasso_M5000_X1k_B6_BV0075015_NTR300_NTE100_NCV5_3.RDS",
    "results_lasso_M5000_X1k_B6_BV0075015_NTR300_NTE100_NCV5_4.RDS",
    "results_lasso_M5000_X1k_B6_BV0075015_NTR300_NTE100_NCV5_5.RDS"
  ),
  outfile = "results_lasso_M5000_X1k_B6_BV0075015_NTR300_NTE100_NCV5.RDS"
)

results_en_M5000_X1k_B6_BV0075015_NTR300_NTE100_NCV5 <- combine_files(
```

```
  files = c(
    "results_en_M5000_X1k_B6_BV0075015_NTR300_NTE100_NCV5_1.RDS",
    "results_en_M5000_X1k_B6_BV0075015_NTR300_NTE100_NCV5_2.RDS",
    "results_en_M5000_X1k_B6_BV0075015_NTR300_NTE100_NCV5_3.RDS",
    "results_en_M5000_X1k_B6_BV0075015_NTR300_NTE100_NCV5_4.RDS",
    "results_en_M5000_X1k_B6_BV0075015_NTR300_NTE100_NCV5_5.RDS"
  ),
  outfile = "results_en_M5000_X1k_B6_BV0075015_NTR300_NTE100_NCV5.RDS"
)

results_ssl_M5000_X1k_B6_BV0075015_NTR300_NTE100_NCV5 <- combine_files(
  files = c(
    "results_ssl_M5000_X1k_B6_BV0075015_NTR300_NTE100_NCV5_1.RDS",
    "results_ssl_M5000_X1k_B6_BV0075015_NTR300_NTE100_NCV5_2.RDS",
    "results_ssl_M5000_X1k_B6_BV0075015_NTR300_NTE100_NCV5_3.RDS",
    "results_ssl_M5000_X1k_B6_BV0075015_NTR300_NTE100_NCV5_4.RDS",
    "results_ssl_M5000_X1k_B6_BV0075015_NTR300_NTE100_NCV5_5.RDS",
    "results_ssl_M5000_X1k_B6_BV0075015_NTR300_NTE100_NCV5_6.RDS",
    "results_ssl_M5000_X1k_B6_BV0075015_NTR300_NTE100_NCV5_7.RDS",
    "results_ssl_M5000_X1k_B6_BV0075015_NTR300_NTE100_NCV5_8.RDS",
    "results_ssl_M5000_X1k_B6_BV0075015_NTR300_NTE100_NCV5_9.RDS",
    "results_ssl_M5000_X1k_B6_BV0075015_NTR300_NTE100_NCV5_10.RDS",
    "results_ssl_M5000_X1k_B6_BV0075015_NTR300_NTE100_NCV5_11.RDS",
    "results_ssl_M5000_X1k_B6_BV0075015_NTR300_NTE100_NCV5_12.RDS",
    "results_ssl_M5000_X1k_B6_BV0075015_NTR300_NTE100_NCV5_13.RDS",
    "results_ssl_M5000_X1k_B6_BV0075015_NTR300_NTE100_NCV5_14.RDS",
    "results_ssl_M5000_X1k_B6_BV0075015_NTR300_NTE100_NCV5_15.RDS",
    "results_ssl_M5000_X1k_B6_BV0075015_NTR300_NTE100_NCV5_16.RDS",
    "results_ssl_M5000_X1k_B6_BV0075015_NTR300_NTE100_NCV5_17.RDS",
    "results_ssl_M5000_X1k_B6_BV0075015_NTR300_NTE100_NCV5_18.RDS",
    "results_ssl_M5000_X1k_B6_BV0075015_NTR300_NTE100_NCV5_19.RDS",
    "results_ssl_M5000_X1k_B6_BV0075015_NTR300_NTE100_NCV5_20.RDS",
    "results_ssl_M5000_X1k_B6_BV0075015_NTR300_NTE100_NCV5_21.RDS",
    "results_ssl_M5000_X1k_B6_BV0075015_NTR300_NTE100_NCV5_22.RDS",
    "results_ssl_M5000_X1k_B6_BV0075015_NTR300_NTE100_NCV5_23.RDS",
    "results_ssl_M5000_X1k_B6_BV0075015_NTR300_NTE100_NCV5_24.RDS",
    "results_ssl_M5000_X1k_B6_BV0075015_NTR300_NTE100_NCV5_25.RDS"
  ),
  outfile = "results_ssl_M5000_X1k_B6_BV0075015_NTR300_NTE100_NCV5.RDS"
)

results_ssen_M5000_X1k_B6_BV0075015_NTR300_NTE100_NCV5 <- combine_files(
  files = c(
    "results_ssen_M5000_X1k_B6_BV0075015_NTR300_NTE100_NCV5_1.RDS",
    "results_ssen_M5000_X1k_B6_BV0075015_NTR300_NTE100_NCV5_2.RDS",
    "results_ssen_M5000_X1k_B6_BV0075015_NTR300_NTE100_NCV5_3.RDS",
    "results_ssen_M5000_X1k_B6_BV0075015_NTR300_NTE100_NCV5_4.RDS",
    "results_ssen_M5000_X1k_B6_BV0075015_NTR300_NTE100_NCV5_5.RDS",
    "results_ssen_M5000_X1k_B6_BV0075015_NTR300_NTE100_NCV5_6.RDS",
    "results_ssen_M5000_X1k_B6_BV0075015_NTR300_NTE100_NCV5_7.RDS",
    "results_ssen_M5000_X1k_B6_BV0075015_NTR300_NTE100_NCV5_8.RDS",
    "results_ssen_M5000_X1k_B6_BV0075015_NTR300_NTE100_NCV5_9.RDS",
    "results_ssen_M5000_X1k_B6_BV0075015_NTR300_NTE100_NCV5_10.RDS",
```

```
    "results_ssen_M5000_X1k_B6_BV0075015_NTR300_NTE100_NCV5_11.RDS",
    "results_ssen_M5000_X1k_B6_BV0075015_NTR300_NTE100_NCV5_12.RDS",
    "results_ssen_M5000_X1k_B6_BV0075015_NTR300_NTE100_NCV5_13.RDS",
    "results_ssen_M5000_X1k_B6_BV0075015_NTR300_NTE100_NCV5_14.RDS",
    "results_ssen_M5000_X1k_B6_BV0075015_NTR300_NTE100_NCV5_15.RDS",
    "results_ssen_M5000_X1k_B6_BV0075015_NTR300_NTE100_NCV5_16.RDS",
    "results_ssen_M5000_X1k_B6_BV0075015_NTR300_NTE100_NCV5_17.RDS",
    "results_ssen_M5000_X1k_B6_BV0075015_NTR300_NTE100_NCV5_18.RDS",
    "results_ssen_M5000_X1k_B6_BV0075015_NTR300_NTE100_NCV5_19.RDS",
    "results_ssen_M5000_X1k_B6_BV0075015_NTR300_NTE100_NCV5_20.RDS",
    "results_ssen_M5000_X1k_B6_BV0075015_NTR300_NTE100_NCV5_21.RDS",
    "results_ssen_M5000_X1k_B6_BV0075015_NTR300_NTE100_NCV5_22.RDS",
    "results_ssen_M5000_X1k_B6_BV0075015_NTR300_NTE100_NCV5_23.RDS",
    "results_ssen_M5000_X1k_B6_BV0075015_NTR300_NTE100_NCV5_24.RDS",
    "results_ssen_M5000_X1k_B6_BV0075015_NTR300_NTE100_NCV5_25.RDS"
  ),
  outfile = "results_ssen_M5000_X1k_B6_BV0075015_NTR300_NTE100_NCV5.RDS"
)
```

## 2.2 B6_BV012025

```
results_lasso_M5000_X1k_B6_BV012025_NTR300_NTE100_NCV5 <- combine_files(
  files = c(
    "results_lasso_M5000_X1k_B6_BV012025_NTR300_NTE100_NCV5_1.RDS",
    "results_lasso_M5000_X1k_B6_BV012025_NTR300_NTE100_NCV5_2.RDS",
    "results_lasso_M5000_X1k_B6_BV012025_NTR300_NTE100_NCV5_3.RDS",
    "results_lasso_M5000_X1k_B6_BV012025_NTR300_NTE100_NCV5_4.RDS",
    "results_lasso_M5000_X1k_B6_BV012025_NTR300_NTE100_NCV5_5.RDS"
  ),
  outfile = "results_lasso_M5000_X1k_B6_BV012025_NTR300_NTE100_NCV5.RDS"
)

results_en_M5000_X1k_B6_BV012025_NTR300_NTE100_NCV5 <- combine_files(
  files = c(
    "results_en_M5000_X1k_B6_BV012025_NTR300_NTE100_NCV5_1.RDS",
    "results_en_M5000_X1k_B6_BV012025_NTR300_NTE100_NCV5_2.RDS",
    "results_en_M5000_X1k_B6_BV012025_NTR300_NTE100_NCV5_3.RDS",
    "results_en_M5000_X1k_B6_BV012025_NTR300_NTE100_NCV5_4.RDS",
    "results_en_M5000_X1k_B6_BV012025_NTR300_NTE100_NCV5_5.RDS"
  ),
  outfile = "results_en_M5000_X1k_B6_BV012025_NTR300_NTE100_NCV5.RDS"
)

results_ssl_M5000_X1k_B6_BV012025_NTR300_NTE100_NCV5 <- combine_files(
  files = c(
    "results_ssl_M5000_X1k_B6_BV012025_NTR300_NTE100_NCV5_1.RDS",
    "results_ssl_M5000_X1k_B6_BV012025_NTR300_NTE100_NCV5_2.RDS",
    "results_ssl_M5000_X1k_B6_BV012025_NTR300_NTE100_NCV5_3.RDS",
    "results_ssl_M5000_X1k_B6_BV012025_NTR300_NTE100_NCV5_4.RDS",
    "results_ssl_M5000_X1k_B6_BV012025_NTR300_NTE100_NCV5_5.RDS",
    "results_ssl_M5000_X1k_B6_BV012025_NTR300_NTE100_NCV5_6.RDS",
    "results_ssl_M5000_X1k_B6_BV012025_NTR300_NTE100_NCV5_7.RDS",
```

```r
    "results_ssl_M5000_X1k_B6_BV012025_NTR300_NTE100_NCV5_8.RDS",
    "results_ssl_M5000_X1k_B6_BV012025_NTR300_NTE100_NCV5_9.RDS",
    "results_ssl_M5000_X1k_B6_BV012025_NTR300_NTE100_NCV5_10.RDS",
    "results_ssl_M5000_X1k_B6_BV012025_NTR300_NTE100_NCV5_11.RDS",
    "results_ssl_M5000_X1k_B6_BV012025_NTR300_NTE100_NCV5_12.RDS",
    "results_ssl_M5000_X1k_B6_BV012025_NTR300_NTE100_NCV5_13.RDS",
    "results_ssl_M5000_X1k_B6_BV012025_NTR300_NTE100_NCV5_14.RDS",
    "results_ssl_M5000_X1k_B6_BV012025_NTR300_NTE100_NCV5_15.RDS",
    "results_ssl_M5000_X1k_B6_BV012025_NTR300_NTE100_NCV5_16.RDS",
    "results_ssl_M5000_X1k_B6_BV012025_NTR300_NTE100_NCV5_17.RDS",
    "results_ssl_M5000_X1k_B6_BV012025_NTR300_NTE100_NCV5_18.RDS",
    "results_ssl_M5000_X1k_B6_BV012025_NTR300_NTE100_NCV5_19.RDS",
    "results_ssl_M5000_X1k_B6_BV012025_NTR300_NTE100_NCV5_20.RDS",
    "results_ssl_M5000_X1k_B6_BV012025_NTR300_NTE100_NCV5_21.RDS",
    "results_ssl_M5000_X1k_B6_BV012025_NTR300_NTE100_NCV5_22.RDS",
    "results_ssl_M5000_X1k_B6_BV012025_NTR300_NTE100_NCV5_23.RDS",
    "results_ssl_M5000_X1k_B6_BV012025_NTR300_NTE100_NCV5_24.RDS",
    "results_ssl_M5000_X1k_B6_BV012025_NTR300_NTE100_NCV5_25.RDS"
  ),
  outfile = "results_ssl_M5000_X1k_B6_BV012025_NTR300_NTE100_NCV5.RDS"
)

results_ssen_M5000_X1k_B6_BV012025_NTR300_NTE100_NCV5 <- combine_files(
  files = c(
    "results_ssen_M5000_X1k_B6_BV012025_NTR300_NTE100_NCV5_1.RDS",
    "results_ssen_M5000_X1k_B6_BV012025_NTR300_NTE100_NCV5_2.RDS",
    "results_ssen_M5000_X1k_B6_BV012025_NTR300_NTE100_NCV5_3.RDS",
    "results_ssen_M5000_X1k_B6_BV012025_NTR300_NTE100_NCV5_4.RDS",
    "results_ssen_M5000_X1k_B6_BV012025_NTR300_NTE100_NCV5_5.RDS",
    "results_ssen_M5000_X1k_B6_BV012025_NTR300_NTE100_NCV5_6.RDS",
    "results_ssen_M5000_X1k_B6_BV012025_NTR300_NTE100_NCV5_7.RDS",
    "results_ssen_M5000_X1k_B6_BV012025_NTR300_NTE100_NCV5_8.RDS",
    "results_ssen_M5000_X1k_B6_BV012025_NTR300_NTE100_NCV5_9.RDS",
    "results_ssen_M5000_X1k_B6_BV012025_NTR300_NTE100_NCV5_10.RDS",
    "results_ssen_M5000_X1k_B6_BV012025_NTR300_NTE100_NCV5_11.RDS",
    "results_ssen_M5000_X1k_B6_BV012025_NTR300_NTE100_NCV5_12.RDS",
    "results_ssen_M5000_X1k_B6_BV012025_NTR300_NTE100_NCV5_13.RDS",
    "results_ssen_M5000_X1k_B6_BV012025_NTR300_NTE100_NCV5_14.RDS",
    "results_ssen_M5000_X1k_B6_BV012025_NTR300_NTE100_NCV5_15.RDS",
    "results_ssen_M5000_X1k_B6_BV012025_NTR300_NTE100_NCV5_16.RDS",
    "results_ssen_M5000_X1k_B6_BV012025_NTR300_NTE100_NCV5_17.RDS",
    "results_ssen_M5000_X1k_B6_BV012025_NTR300_NTE100_NCV5_18.RDS",
    "results_ssen_M5000_X1k_B6_BV012025_NTR300_NTE100_NCV5_19.RDS",
    "results_ssen_M5000_X1k_B6_BV012025_NTR300_NTE100_NCV5_20.RDS",
    "results_ssen_M5000_X1k_B6_BV012025_NTR300_NTE100_NCV5_21.RDS",
    "results_ssen_M5000_X1k_B6_BV012025_NTR300_NTE100_NCV5_22.RDS",
    "results_ssen_M5000_X1k_B6_BV012025_NTR300_NTE100_NCV5_23.RDS",
    "results_ssen_M5000_X1k_B6_BV012025_NTR300_NTE100_NCV5_24.RDS",
    "results_ssen_M5000_X1k_B6_BV012025_NTR300_NTE100_NCV5_25.RDS"
  ),
  outfile = "results_ssen_M5000_X1k_B6_BV012025_NTR300_NTE100_NCV5.RDS"
)
```

## 2.3 B6_BV025050

```
results_lasso_M5000_X1k_B6_BV025050_NTR300_NTE100_NCV5 <- combine_files(
  files = c(
    "results_lasso_M5000_X1k_B6_BV025050_NTR300_NTE100_NCV5_1.RDS",
    "results_lasso_M5000_X1k_B6_BV025050_NTR300_NTE100_NCV5_2.RDS",
    "results_lasso_M5000_X1k_B6_BV025050_NTR300_NTE100_NCV5_3.RDS",
    "results_lasso_M5000_X1k_B6_BV025050_NTR300_NTE100_NCV5_4.RDS",
    "results_lasso_M5000_X1k_B6_BV025050_NTR300_NTE100_NCV5_5.RDS"
  ),
  outfile = "results_lasso_M5000_X1k_B6_BV025050_NTR300_NTE100_NCV5.RDS"
)

results_en_M5000_X1k_B6_BV025050_NTR300_NTE100_NCV5 <- combine_files(
  files = c(
    "results_en_M5000_X1k_B6_BV025050_NTR300_NTE100_NCV5_1.RDS",
    "results_en_M5000_X1k_B6_BV025050_NTR300_NTE100_NCV5_2.RDS",
    "results_en_M5000_X1k_B6_BV025050_NTR300_NTE100_NCV5_3.RDS",
    "results_en_M5000_X1k_B6_BV025050_NTR300_NTE100_NCV5_4.RDS",
    "results_en_M5000_X1k_B6_BV025050_NTR300_NTE100_NCV5_5.RDS"
  ),
  outfile = "results_en_M5000_X1k_B6_BV025050_NTR300_NTE100_NCV5.RDS"
)

results_ssl_M5000_X1k_B6_BV025050_NTR300_NTE100_NCV5 <- combine_files(
  files = c(
    "results_ssl_M5000_X1k_B6_BV025050_NTR300_NTE100_NCV5_1.RDS",
    "results_ssl_M5000_X1k_B6_BV025050_NTR300_NTE100_NCV5_2.RDS",
    "results_ssl_M5000_X1k_B6_BV025050_NTR300_NTE100_NCV5_3.RDS",
    "results_ssl_M5000_X1k_B6_BV025050_NTR300_NTE100_NCV5_4.RDS",
    "results_ssl_M5000_X1k_B6_BV025050_NTR300_NTE100_NCV5_5.RDS",
    "results_ssl_M5000_X1k_B6_BV025050_NTR300_NTE100_NCV5_6.RDS",
    "results_ssl_M5000_X1k_B6_BV025050_NTR300_NTE100_NCV5_7.RDS",
    "results_ssl_M5000_X1k_B6_BV025050_NTR300_NTE100_NCV5_8.RDS",
    "results_ssl_M5000_X1k_B6_BV025050_NTR300_NTE100_NCV5_9.RDS",
    "results_ssl_M5000_X1k_B6_BV025050_NTR300_NTE100_NCV5_10.RDS",
    "results_ssl_M5000_X1k_B6_BV025050_NTR300_NTE100_NCV5_11.RDS",
    "results_ssl_M5000_X1k_B6_BV025050_NTR300_NTE100_NCV5_12.RDS",
    "results_ssl_M5000_X1k_B6_BV025050_NTR300_NTE100_NCV5_13.RDS",
    "results_ssl_M5000_X1k_B6_BV025050_NTR300_NTE100_NCV5_14.RDS",
    "results_ssl_M5000_X1k_B6_BV025050_NTR300_NTE100_NCV5_15.RDS",
    "results_ssl_M5000_X1k_B6_BV025050_NTR300_NTE100_NCV5_16.RDS",
    "results_ssl_M5000_X1k_B6_BV025050_NTR300_NTE100_NCV5_17.RDS",
    "results_ssl_M5000_X1k_B6_BV025050_NTR300_NTE100_NCV5_18.RDS",
    "results_ssl_M5000_X1k_B6_BV025050_NTR300_NTE100_NCV5_19.RDS",
    "results_ssl_M5000_X1k_B6_BV025050_NTR300_NTE100_NCV5_20.RDS",
    "results_ssl_M5000_X1k_B6_BV025050_NTR300_NTE100_NCV5_21.RDS",
    "results_ssl_M5000_X1k_B6_BV025050_NTR300_NTE100_NCV5_22.RDS",
    "results_ssl_M5000_X1k_B6_BV025050_NTR300_NTE100_NCV5_23.RDS",
    "results_ssl_M5000_X1k_B6_BV025050_NTR300_NTE100_NCV5_24.RDS",
    "results_ssl_M5000_X1k_B6_BV025050_NTR300_NTE100_NCV5_25.RDS"
  ),
  outfile = "results_ssl_M5000_X1k_B6_BV025050_NTR300_NTE100_NCV5.RDS"
```

```
)

results_ssen_M5000_X1k_B6_BV025050_NTR300_NTE100_NCV5 <- combine_files(
  files = c(
    "results_ssen_M5000_X1k_B6_BV025050_NTR300_NTE100_NCV5_1.RDS",
    "results_ssen_M5000_X1k_B6_BV025050_NTR300_NTE100_NCV5_2.RDS",
    "results_ssen_M5000_X1k_B6_BV025050_NTR300_NTE100_NCV5_3.RDS",
    "results_ssen_M5000_X1k_B6_BV025050_NTR300_NTE100_NCV5_4.RDS",
    "results_ssen_M5000_X1k_B6_BV025050_NTR300_NTE100_NCV5_5.RDS",
    "results_ssen_M5000_X1k_B6_BV025050_NTR300_NTE100_NCV5_6.RDS",
    "results_ssen_M5000_X1k_B6_BV025050_NTR300_NTE100_NCV5_7.RDS",
    "results_ssen_M5000_X1k_B6_BV025050_NTR300_NTE100_NCV5_8.RDS",
    "results_ssen_M5000_X1k_B6_BV025050_NTR300_NTE100_NCV5_9.RDS",
    "results_ssen_M5000_X1k_B6_BV025050_NTR300_NTE100_NCV5_10.RDS",
    "results_ssen_M5000_X1k_B6_BV025050_NTR300_NTE100_NCV5_11.RDS",
    "results_ssen_M5000_X1k_B6_BV025050_NTR300_NTE100_NCV5_12.RDS",
    "results_ssen_M5000_X1k_B6_BV025050_NTR300_NTE100_NCV5_13.RDS",
    "results_ssen_M5000_X1k_B6_BV025050_NTR300_NTE100_NCV5_14.RDS",
    "results_ssen_M5000_X1k_B6_BV025050_NTR300_NTE100_NCV5_15.RDS",
    "results_ssen_M5000_X1k_B6_BV025050_NTR300_NTE100_NCV5_16.RDS",
    "results_ssen_M5000_X1k_B6_BV025050_NTR300_NTE100_NCV5_17.RDS",
    "results_ssen_M5000_X1k_B6_BV025050_NTR300_NTE100_NCV5_18.RDS",
    "results_ssen_M5000_X1k_B6_BV025050_NTR300_NTE100_NCV5_19.RDS",
    "results_ssen_M5000_X1k_B6_BV025050_NTR300_NTE100_NCV5_20.RDS",
    "results_ssen_M5000_X1k_B6_BV025050_NTR300_NTE100_NCV5_21.RDS",
    "results_ssen_M5000_X1k_B6_BV025050_NTR300_NTE100_NCV5_22.RDS",
    "results_ssen_M5000_X1k_B6_BV025050_NTR300_NTE100_NCV5_23.RDS",
    "results_ssen_M5000_X1k_B6_BV025050_NTR300_NTE100_NCV5_24.RDS",
    "results_ssen_M5000_X1k_B6_BV025050_NTR300_NTE100_NCV5_25.RDS"
  ),
  outfile = "results_ssen_M5000_X1k_B6_BV025050_NTR300_NTE100_NCV5.RDS"
)
```

# 3 Summary function

The following function automates useful summaries.

```
summary_function_A <- function(
  path = "C:/Users/cotto/Documents/Publications/multinomial models in R/simulations/results/",
  simulations_desired = 5000, # used when calculating total times required for all simulations
  files,
  model_names = c("Lasso", "EN", "SSL"),
  spatial = TRUE, # if true use spatial arguments; otherwise use non-spatial arguments
  # the following are for spatial settings
  im.wh = 32, # image (X) width/height
  nzb.wh = 6, # non-zero parameter area width/height
  row.index = 5, col.index = 5, B0 = 0, B.values = 0.01, # as in sim2Dpredictr::beta_builder
  index.type = "ellipse",  # as in sim2Dpredictr::beta_builder
  # the following are for non-spatial settings
  p = 1000, # how many predictors?
  b.indices, # indices for non-zero predictors
  print.progress = FALSE #
```

```r
) {

  if (length(files) != length(model_names)) {
    stop("Number of files and number of model names must be equal.")
  }

  # hold fit times
  fit_time_avg <- NULL

  # hold model fitness summaries
  model_fitness_avg <- NULL
  model_fitness_med <- NULL
  model_fitness_sd <- NULL
  model_fitness_iqr <- NULL

  # hold raw values for FP & Power
  sfp_out <- NULL

  # hold FP & Power summaries
  sfp_avg <- NULL
  sfp_med <- NULL
  sfp_sd <- NULL
  sfp_iqr <- NULL

  # hold inclusion probabilities
  prob_rej <- NULL

  if (spatial == TRUE) {
    # should match the simulation scenario
  B <- sim2Dpredictr::beta_builder(
    im.res = c(im.wh, im.wh),
    row.index = row.index, col.index = col.index,
    B0 = B0, B.values = B.values,
    index.type = index.type,
    h = nzb.wh, w = nzb.wh
  )

  # convert to 1 = Non-zero-valued and 0 = zero-valued parameter
  b.binary <- ifelse(B$B[-1] != 0, 1, 0)
  true.pred <- paste0("x", B$Truth.Indices)
  } else {
    b.binary = rep(0, p)
    b.binary[b.indices] <- 1
    true.pred <- paste0("x", b.indices)
  }

  # rej <- list()
  # sfp <- list()
  for (i in 1:length(files)) {

    # load results
    results_i <- readRDS(
```

```
    paste0(path, files[[i]])
  )

  # how many simulations?
  M <- length(unique(results_i$param_est$sim.num))

  # How long to fit models
  fit_time_avg_i <- dplyr::bind_cols(
    model = model_names[i],
    sec_per_sim = results_i$fit_time[3] / nrow(results_i$model_fitness),
    hours_required = simulations_desired * results_i$fit_time[3] / nrow(results_i$model_fitness) / 60
    )
  fit_time_avg <- dplyr::bind_rows(
    fit_time_avg,
    fit_time_avg_i
  )

  if (print.progress == TRUE) {
    cat("Model fit times for ", model_names[i], " complete \n")
  }

  # model fitness
  model_fitness_avg_i <- dplyr::bind_cols(
    model = model_names[i],
    purrr::map_df(results_i$model_fitness |> dplyr::select(-sim.num), mean, na.rm = TRUE)
    )
  model_fitness_avg <- dplyr::bind_rows(
    model_fitness_avg,
    model_fitness_avg_i
  )
  model_fitness_med_i <- dplyr::bind_cols(
    model = model_names[i],
    purrr::map_df(results_i$model_fitness |> dplyr::select(-sim.num), median, na.rm = TRUE)
  )
  model_fitness_med <- dplyr::bind_rows(
    model_fitness_med,
    model_fitness_med_i
  )
  model_fitness_sd_i <- dplyr::bind_cols(
    model = model_names[i],
    purrr::map_df(results_i$model_fitness |> dplyr::select(-sim.num), sd, na.rm = TRUE)
  )
  model_fitness_sd <- dplyr::bind_rows(
    model_fitness_sd,
    model_fitness_sd_i
  )
  model_fitness_iqr_i <- dplyr::bind_cols(
    model = model_names[i],
    purrr::map_df(results_i$model_fitness |> dplyr::select(-sim.num), IQR, na.rm = TRUE)
  )
  model_fitness_iqr <- dplyr::bind_rows(
    model_fitness_iqr,
    model_fitness_iqr_i
```

```r
  )

  if (print.progress == TRUE) {
    cat("Model fitness for ", model_names[i], " complete. \n")
  }

  # obtain estimates from each simulation
  # obtain indices for predictors with non-zero estimates
  rej_i <- NULL
  # Obtain FDP & Power
  sfp_i <- NULL
  for (m in 1:M) {

    if (!(m %in% results_i$param_est$sim.num)) {
      cat("Simulation ", m, " for model ", model_names[i], "is missing. \n")
    } else {
      # obtain indices for non-zero estimates in results i, simulation m
      # can use outcome == 1 since grouped parameters by predictor
      rej_im <- data.frame(
        ifelse(
          results_i$param_est |>
            dplyr::filter(outcome == 1, sim.num == m) |>
            dplyr::select(-sim.num, -s0, -s1, -outcome, -x0) != 0,
          1, 0
        )
      )
      if (print.progress == TRUE) {
        cat("Non-zero parameter indices for ", model_names[i], " for sim ", m, " complete. \n")
      }

      # false positive rates & power for results i, simulation m
      sfp_im <- sim2Dpredictr::sample_FP_Power(
        rejections = as.numeric(rej_im),
        B = b.binary,
        B.incl.B0 = FALSE
      )

      sfp_i <- dplyr::bind_rows(
        sfp_i,
        sfp_im
      )

      rej_im <- dplyr::tibble(rej_im)

      rej_i <- dplyr::bind_rows(
        rej_i,
        rej_im
      )
    }

    if (print.progress == TRUE) {
      cat("FP and Power for ", model_names[i], " for sim ", m, " complete. \n")
    }
```

```r
  }
  # rej_i <- data.frame(rej_i)
  # print(rej_i[, 1:10])
  prob_rej_i <- dplyr::bind_cols(
    model = model_names[i],
    purrr::map_df(.x = rej_i, .f = mean)
  )
  prob_rej <- dplyr::bind_rows(
    prob_rej,
    prob_rej_i
  )
  sfp_avg_i <- dplyr::bind_cols(
    model = model_names[i],
    purrr::map_df(.x = sfp_i, .f = mean, na.rm = TRUE)
  )
  sfp_avg <- dplyr::bind_rows(
    sfp_avg,
    sfp_avg_i
  )
  sfp_med_i <- dplyr::bind_cols(
    model = model_names[i],
    purrr::map_df(.x = sfp_i, .f = median, na.rm = TRUE)
  )
  sfp_med <- dplyr::bind_rows(
    sfp_med,
    sfp_med_i
  )
  sfp_sd_i <- dplyr::bind_cols(
    model = model_names[i],
    purrr::map_df(.x = sfp_i, .f = sd, na.rm = TRUE)
  )
  sfp_sd <- dplyr::bind_rows(
    sfp_sd,
    sfp_sd_i
  )
  sfp_iqr_i <- dplyr::bind_cols(
    model = model_names[i],
    purrr::map_df(.x = sfp_i, .f = IQR, na.rm = TRUE)
  )
  sfp_iqr <- dplyr::bind_rows(
    sfp_iqr,
    sfp_iqr_i
  )

  sfp_out_i <- cbind(model = model_names[i], sfp_i)
  sfp_out <- dplyr::bind_rows(
    sfp_out,
    sfp_out_i
  )

  # rej[[i]] <- data.frame(dplyr::bind_cols(model = model_names[i], rej_i))
  # sfp[[i]] <- dplyr::bind_cols(model = model_names[i], sfp_i)
}
```

```
  rownames(fit_time_avg) <- NULL
  # names(rej) <- model_names
  # rej_final <- dplyr::bind_rows(rej)
  # names(sfp) <- model_names
  # sfp_final <- dplyr::bind_rows(sfp)

  return(
    list(
      fit_time_avg = data.frame(fit_time_avg),
      model_fitness_avg = data.frame(model_fitness_avg),
      model_fitness_sd = data.frame(model_fitness_sd),
      model_fitness_med = data.frame(model_fitness_med),
      model_fitness_iqr = data.frame(model_fitness_iqr),
      FP_Power_avg = data.frame(sfp_avg),
      FP_Power_sd = data.frame(sfp_sd),
      FP_Power_med = data.frame(sfp_med),
      FP_Power_iqr = data.frame(sfp_iqr),
      Prob_Included = data.frame(prob_rej),
      Raw_out = data.frame(sfp_out)
      # fpr_power = sfp_final,
      # nonzero_est = rej_final
      )
  )
}
```

For false positives and power we need locations of non-zero parameters. While the parameter values change in each of the 3 scenarios examined here, the number of non-zero parameters and the number of total predictors is the same, so `b.binary` can be reused.

```
# how many parameters?
p <- 1000

# non-zero locations
nzl <- c(1, 51, 101, 151, 201, 251)

bb.ns <- rep(0, p)
bb.ns[nzl] <- 1
tp.ns <- paste0("x", nzl)
```

## 3.1 B6_BV0075015 Summary (ALL)

Table 4: Mean Model Fitness: X1k_B6_BV0075015

| model | s0 | s1 | deviance | avg_acc | pce | ppv_macro | sn_macro | f1_macro | ppv_micro | sn_micro | f1_micro |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Lasso | 0.1031 | 0.1031 | 216.8022 | 0.5976 | 0.4024 | 0.3451 | 0.3340 | 0.3353 | 0.3965 | 0.3965 | 0.3965 |
| EN | 0.2020 | 0.2020 | 216.7978 | 0.5977 | 0.4023 | 0.3521 | 0.3341 | 0.3389 | 0.3965 | 0.3965 | 0.3965 |
| SSL | 0.0243 | 1.7258 | 217.2746 | 0.5965 | 0.4035 | 0.3414 | 0.3346 | 0.3358 | 0.3948 | 0.3948 | 0.3948 |
| SSEN | 0.0152 | 1.6154 | 217.2341 | 0.5967 | 0.4033 | 0.3412 | 0.3344 | 0.3362 | 0.3950 | 0.3950 | 0.3950 |

Table 5: SD Model Fitness: X1k_B6_BV0075015

| model | s0 | s1 | deviance | avg_acc | pce | ppv_macro | sn_macro | f1_macro | ppv_micro | sn_micro | f1_micro |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Lasso | 0.0144 | 0.0144 | 4.5535 | 0.0342 | 0.0342 | 0.1136 | 0.0184 | 0.0660 | 0.0513 | 0.0513 | 0.0513 |
| EN | 0.0311 | 0.0311 | 4.5431 | 0.0342 | 0.0342 | 0.1150 | 0.0182 | 0.0665 | 0.0513 | 0.0513 | 0.0513 |
| SSL | 0.0181 | 1.2832 | 5.0504 | 0.0345 | 0.0345 | 0.1004 | 0.0217 | 0.0625 | 0.0518 | 0.0518 | 0.0518 |
| SSEN | 0.0074 | 1.2107 | 5.0495 | 0.0344 | 0.0344 | 0.0928 | 0.0208 | 0.0598 | 0.0516 | 0.0516 | 0.0516 |

Table 6: Median Model Fitness: X1k_B6_BV0075015

| model | s0 | s1 | deviance | avg_acc | pce | ppv_macro | sn_macro | f1_macro | ppv_micro | sn_micro | f1_micro |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Lasso | 0.1052 | 0.1052 | 216.7385 | 0.6 | 0.4 | 0.3232 | 0.3333 | 0.3301 | 0.4 | 0.4 | 0.4 |
| EN | 0.2077 | 0.2077 | 216.7407 | 0.6 | 0.4 | 0.3334 | 0.3333 | 0.3388 | 0.4 | 0.4 | 0.4 |
| SSL | 0.0100 | 1.0000 | 217.0349 | 0.6 | 0.4 | 0.3286 | 0.3333 | 0.3348 | 0.4 | 0.4 | 0.4 |
| SSEN | 0.0100 | 1.0000 | 216.9702 | 0.6 | 0.4 | 0.3311 | 0.3333 | 0.3333 | 0.4 | 0.4 | 0.4 |

Table 7: IQR Model Fitness: X1k_B6_BV0075015

| model | s0 | s1 | deviance | avg_acc | pce | ppv_macro | sn_macro | f1_macro | ppv_micro | sn_micro | f1_micro |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Lasso | 0.0180 | 0.0180 | 5.5983 | 0.0467 | 0.0467 | 0.1541 | 0 | 0.0957 | 0.07 | 0.07 | 0.07 |
| EN | 0.0397 | 0.0397 | 5.5560 | 0.0467 | 0.0467 | 0.1609 | 0 | 0.1013 | 0.07 | 0.07 | 0.07 |
| SSL | 0.0300 | 0.0000 | 5.9760 | 0.0467 | 0.0467 | 0.1217 | 0 | 0.0823 | 0.07 | 0.07 | 0.07 |
| SSEN | 0.0100 | 0.0000 | 5.9911 | 0.0467 | 0.0467 | 0.1217 | 0 | 0.0864 | 0.07 | 0.07 | 0.07 |

Table 8: Mean False Positive Rates and Power: X1k_B6_BV0075015

| model | FDP | FWE | Power |
|---|---|---|---|
| Lasso | 0.6437 | 0.6500 | 0.0090 |
| EN | 0.5637 | 0.5706 | 0.0143 |
| SSL | 0.4052 | 0.4116 | 0.0150 |
| SSEN | 0.3661 | 0.3718 | 0.0211 |

Table 9: SD False Positive Rates and Power: X1k_B6_BV0075015

| model | FDP | FWE | Power |
|---|---|---|---|
| Lasso | 0.4739 | 0.4770 | 0.0406 |
| EN | 0.4902 | 0.4950 | 0.0518 |
| SSL | 0.4854 | 0.4922 | 0.0534 |
| SSEN | 0.4764 | 0.4833 | 0.0676 |

Table 10: Median False Positive Rates and Power: X1k_B6_BV0075015

| model | FDP | FWE | Power |
|---|---|---|---|
| Lasso | 1.0000 | 1 | 0 |
| EN | 0.9798 | 1 | 0 |
| SSL | 0.0000 | 0 | 0 |
| SSEN | 0.0000 | 0 | 0 |

Table 11: IQR False Positive Rates and Power: X1k_B6_BV0075015

| model | FDP | FWE | Power |
|---|---|---|---|
| Lasso | 1 | 1 | 0 |
| EN | 1 | 1 | 0 |
| SSL | 1 | 1 | 0 |
| SSEN | 1 | 1 | 0 |

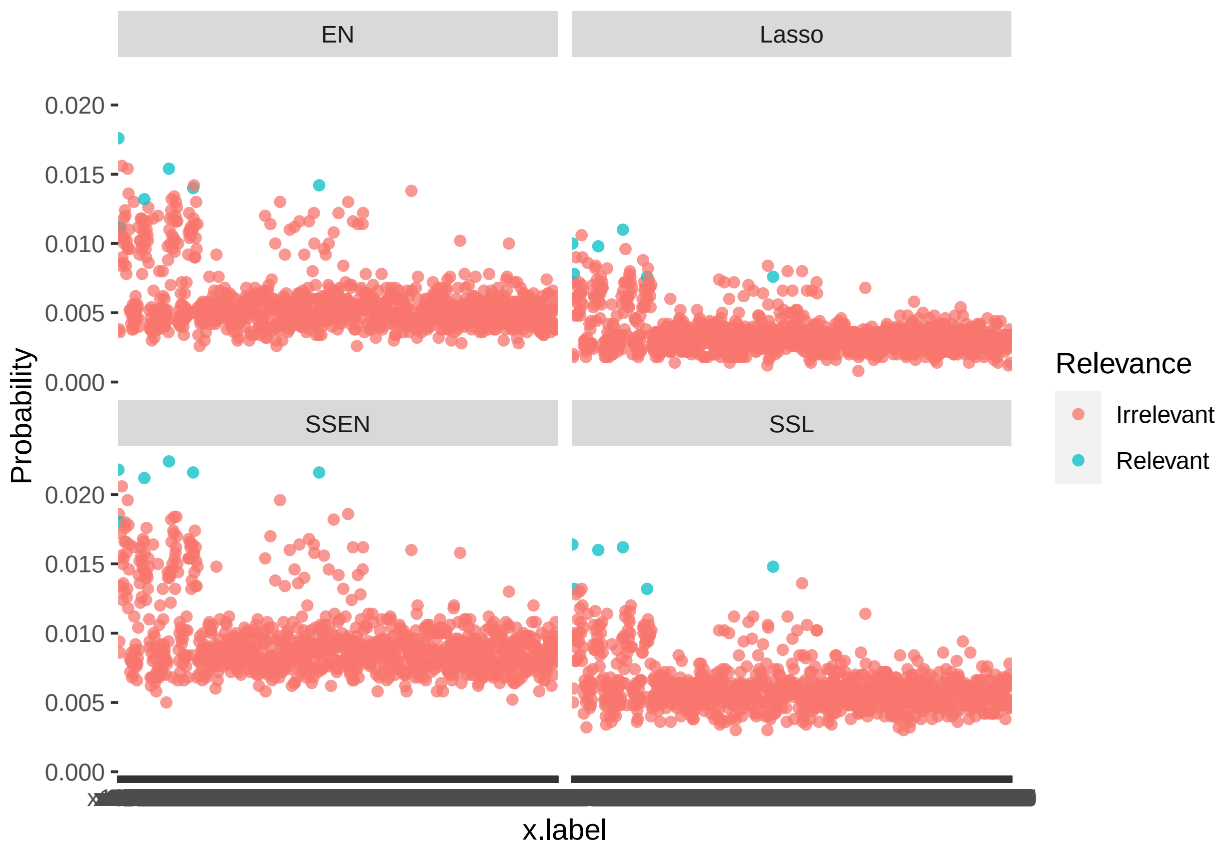


Table 12: Mean Probability of Inclusion for True Non-zero Paramters

| Model | X1 | X51 | X101 | X151 | X201 | X251 |
|---|---|---|---|---|---|---|
| Lasso | 0.0100 | 0.0076 | 0.0078 | 0.0098 | 0.0110 | 0.0076 |
| EN | 0.0176 | 0.0142 | 0.0112 | 0.0132 | 0.0154 | 0.0140 |
| SSL | 0.0164 | 0.0148 | 0.0132 | 0.0160 | 0.0162 | 0.0132 |
| SSEN | 0.0218 | 0.0216 | 0.0180 | 0.0212 | 0.0224 | 0.0216 |

Table 13: Probability of Inclusion for Relevant/Irrelevant Parameters: X1k_B6_BV0075015

| Model | Relevance | Mean | SD |
|---|---|---|---|
| Lasso | Irrelevant | 0.0034 | 0.0014 |
| EN | Irrelevant | 0.0057 | 0.0021 |
| SSL | Irrelevant | 0.0061 | 0.0017 |
| SSEN | Irrelevant | 0.0094 | 0.0025 |
| Lasso | Relevant | 0.0090 | 0.0015 |
| EN | Relevant | 0.0143 | 0.0021 |
| SSL | Relevant | 0.0150 | 0.0015 |
| SSEN | Relevant | 0.0211 | 0.0016 |

### 3.2 B6_BV0075015 Summary (Reduced for Manuscript)

Table 14: Mean Model Fitness: X1k_B6_BV0075015

| model | s0 | s1 | deviance | FDP | Power |
|---|---|---|---|---|---|
| Lasso | 0.1031 | 0.1031 | 216.8022 | 0.6437 | 0.0090 |
| EN | 0.2020 | 0.2020 | 216.7978 | 0.5637 | 0.0143 |
| SSL | 0.0243 | 1.7258 | 217.2746 | 0.4052 | 0.0150 |
| SSEN | 0.0152 | 1.6154 | 217.2341 | 0.3661 | 0.0211 |

Table 15: Mean Probability of Inclusion for True Non-zero Paramters

| Model | X1 | X51 | X101 | X151 | X201 | X251 |
|---|---|---|---|---|---|---|
| Lasso | 0.0100 | 0.0076 | 0.0078 | 0.0098 | 0.0110 | 0.0076 |
| EN | 0.0176 | 0.0142 | 0.0112 | 0.0132 | 0.0154 | 0.0140 |
| SSL | 0.0164 | 0.0148 | 0.0132 | 0.0160 | 0.0162 | 0.0132 |
| SSEN | 0.0218 | 0.0216 | 0.0180 | 0.0212 | 0.0224 | 0.0216 |

### 3.3 B6_B012025 Summary

Table 16: Mean Model Fitness: X1k_B6_BV012025

| model | s0 | s1 | deviance | avg_acc | pce | ppv_macro | sn_macro | f1_macro | ppv_micro | sn_micro | f1_micro |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Lasso | 0.1001 | 0.1001 | 216.7096 | 0.5998 | 0.4002 | 0.3635 | 0.3381 | 0.3500 | 0.3996 | 0.3996 | 0.3996 |
| EN | 0.1937 | 0.1937 | 216.6557 | 0.6000 | 0.4000 | 0.3621 | 0.3386 | 0.3501 | 0.4000 | 0.4000 | 0.4000 |
| SSL | 0.0308 | 2.0642 | 217.2401 | 0.5987 | 0.4013 | 0.3636 | 0.3406 | 0.3533 | 0.3981 | 0.3981 | 0.3981 |
| SSEN | 0.0180 | 1.9542 | 217.1768 | 0.5992 | 0.4008 | 0.3673 | 0.3406 | 0.3563 | 0.3987 | 0.3987 | 0.3987 |

Table 17: SD Model Fitness: X1k_B6_BV012025

| model | s0 | s1 | deviance | avg_acc | pce | ppv_macro | sn_macro | f1_macro | ppv_micro | sn_micro | f1_micro |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Lasso | 0.0172 | 0.0172 | 4.6489 | 0.0341 | 0.0341 | 0.1135 | 0.0227 | 0.0665 | 0.0511 | 0.0511 | 0.0511 |
| EN | 0.0368 | 0.0368 | 4.6383 | 0.0341 | 0.0341 | 0.1122 | 0.0236 | 0.0674 | 0.0512 | 0.0512 | 0.0512 |
| SSL | 0.0198 | 1.4328 | 5.3637 | 0.0341 | 0.0341 | 0.0989 | 0.0278 | 0.0620 | 0.0512 | 0.0512 | 0.0512 |
| SSEN | 0.0086 | 1.3963 | 5.2928 | 0.0341 | 0.0341 | 0.0958 | 0.0271 | 0.0606 | 0.0511 | 0.0511 | 0.0511 |

Table 18: Median Model Fitness: X1k_B6_BV012025

| model | s0 | s1 | deviance | avg_acc | pce | ppv_macro | sn_macro | f1_macro | ppv_micro | sn_micro | f1_micro |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Lasso | 0.1029 | 0.1029 | 216.6698 | 0.6 | 0.4 | 0.3510 | 0.3333 | 0.3471 | 0.4 | 0.4 | 0.4 |
| EN | 0.1994 | 0.1994 | 216.6339 | 0.6 | 0.4 | 0.3487 | 0.3333 | 0.3483 | 0.4 | 0.4 | 0.4 |
| SSL | 0.0400 | 1.0000 | 216.9986 | 0.6 | 0.4 | 0.3527 | 0.3333 | 0.3500 | 0.4 | 0.4 | 0.4 |
| SSEN | 0.0200 | 1.0000 | 216.9323 | 0.6 | 0.4 | 0.3593 | 0.3333 | 0.3552 | 0.4 | 0.4 | 0.4 |

Table 19: IQR Model Fitness: X1k_B6_BV012025

| model | s0 | s1 | deviance | avg_acc | pce | ppv_macro | sn_macro | f1_macro | ppv_micro | sn_micro | f1_micro |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Lasso | 0.0254 | 0.0254 | 5.5223 | 0.0417 | 0.0417 | 0.1449 | 0.0050 | 0.0932 | 0.0625 | 0.0625 | 0.0625 |
| EN | 0.0579 | 0.0579 | 5.5248 | 0.0400 | 0.0400 | 0.1497 | 0.0074 | 0.0992 | 0.0600 | 0.0600 | 0.0600 |
| SSL | 0.0400 | 3.0000 | 6.2586 | 0.0467 | 0.0467 | 0.1229 | 0.0126 | 0.0865 | 0.0700 | 0.0700 | 0.0700 |
| SSEN | 0.0200 | 3.0000 | 6.1332 | 0.0467 | 0.0467 | 0.1168 | 0.0120 | 0.0858 | 0.0700 | 0.0700 | 0.0700 |

Table 20: Mean False Positive Rates and Power: X1k_B6_BV012025

| model | FDP | FWE | Power |
|---|---|---|---|
| Lasso | 0.7166 | 0.7396 | 0.0338 |
| EN | 0.6555 | 0.6802 | 0.0557 |
| SSL | 0.5435 | 0.5608 | 0.0479 |
| SSEN | 0.5100 | 0.5262 | 0.0697 |

Table 21: SD False Positive Rates and Power: X1k_B6_BV012025

| model | FDP | FWE | Power |
|---|---|---|---|
| Lasso | 0.4308 | 0.4389 | 0.0789 |
| EN | 0.4536 | 0.4664 | 0.1038 |
| SSL | 0.4828 | 0.4963 | 0.0984 |
| SSEN | 0.4849 | 0.4994 | 0.1235 |

Table 22: Median False Positive Rates and Power: X1k_B6_BV012025

| model | FDP | FWE | Power |
|---|---|---|---|
| Lasso | 1.0000 | 1 | 0 |
| EN | 0.9565 | 1 | 0 |
| SSL | 0.9000 | 1 | 0 |
| SSEN | 0.8889 | 1 | 0 |

Table 23: IQR False Positive Rates and Power: X1k_B6_BV012025

| model | FDP | FWE | Power |
|---|---|---|---|
| Lasso | 1.0000 | 1 | 0.0000 |
| EN | 1.0000 | 1 | 0.1667 |
| SSL | 1.0000 | 1 | 0.0000 |
| SSEN | 0.9828 | 1 | 0.1667 |

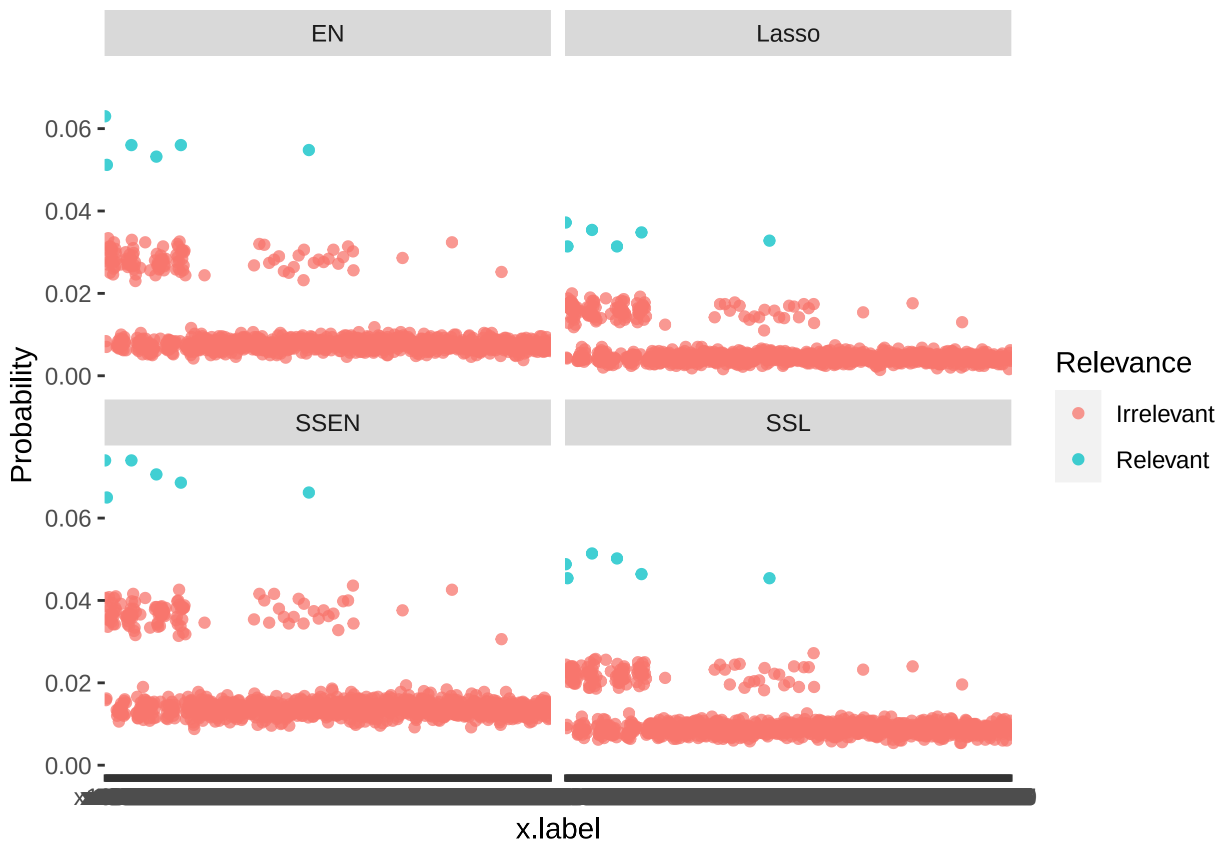



Table 24: Mean Probability of Inclusion for True Non-zero Paramters

| Model | X1 | X51 | X101 | X151 | X201 | X251 |
|---|---|---|---|---|---|---|
| Lasso | 0.0372 | 0.0328 | 0.0314 | 0.0354 | 0.0314 | 0.0348 |
| EN | 0.0630 | 0.0548 | 0.0512 | 0.0560 | 0.0532 | 0.0560 |
| SSL | 0.0488 | 0.0454 | 0.0454 | 0.0514 | 0.0502 | 0.0464 |
| SSEN | 0.0740 | 0.0662 | 0.0650 | 0.0740 | 0.0706 | 0.0686 |

Table 25: Probability of Inclusion for Relevant/Irrelevant Parameters: X1k_B6_BV012025

| Model | Relevance | Mean | SD |
|---|---|---|---|
| Lasso | Irrelevant | 0.0056 | 0.0038 |
| EN | Irrelevant | 0.0099 | 0.0067 |
| SSL | Irrelevant | 0.0103 | 0.0044 |
| SSEN | Irrelevant | 0.0163 | 0.0076 |
| Lasso | Relevant | 0.0338 | 0.0024 |
| EN | Relevant | 0.0557 | 0.0040 |
| SSL | Relevant | 0.0479 | 0.0026 |
| SSEN | Relevant | 0.0697 | 0.0038 |

## 3.4 B6_BV012025 Summary (Reduced for Manuscript)

Table 26: Mean Model Fitness: X1k_B6_BV012025

| model | s0 | s1 | deviance | FDP | Power |
|---|---|---|---|---|---|
| Lasso | 0.1001 | 0.1001 | 216.7096 | 0.7166 | 0.0338 |
| EN | 0.1937 | 0.1937 | 216.6557 | 0.6555 | 0.0557 |
| SSL | 0.0308 | 2.0642 | 217.2401 | 0.5435 | 0.0479 |
| SSEN | 0.0180 | 1.9542 | 217.1768 | 0.5100 | 0.0697 |

Table 27: Mean Probability of Inclusion for True Non-zero Paramters

| Model | X1 | X51 | X101 | X151 | X201 | X251 |
|---|---|---|---|---|---|---|
| Lasso | 0.0372 | 0.0328 | 0.0314 | 0.0354 | 0.0314 | 0.0348 |
| EN | 0.0630 | 0.0548 | 0.0512 | 0.0560 | 0.0532 | 0.0560 |
| SSL | 0.0488 | 0.0454 | 0.0454 | 0.0514 | 0.0502 | 0.0464 |
| SSEN | 0.0740 | 0.0662 | 0.0650 | 0.0740 | 0.0706 | 0.0686 |

## 3.5 B6_B025050 Summary

Table 28: Mean Model Fitness: X1k_B6_BV025050

| model | s0 | s1 | deviance | avg_acc | pce | ppv_macro | sn_macro | f1_macro | ppv_micro | sn_micro | f1_micro |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Lasso | 0.0770 | 0.0770 | 210.9040 | 0.6259 | 0.3741 | 0.4224 | 0.3923 | 0.4081 | 0.4389 | 0.4389 | 0.4389 |
| EN | 0.1445 | 0.1445 | 210.6891 | 0.6268 | 0.3732 | 0.4243 | 0.3935 | 0.4094 | 0.4402 | 0.4402 | 0.4402 |
| SSL | 0.0553 | 3.0590 | 210.9524 | 0.6275 | 0.3725 | 0.4210 | 0.4030 | 0.4129 | 0.4412 | 0.4412 | 0.4412 |
| SSEN | 0.0301 | 2.9224 | 210.8284 | 0.6278 | 0.3722 | 0.4231 | 0.4036 | 0.4145 | 0.4417 | 0.4417 | 0.4417 |

Table 29: SD Model Fitness: X1k_B6_BV025050

| model | s0 | s1 | deviance | avg_acc | pce | ppv_macro | sn_macro | f1_macro | ppv_micro | sn_micro | f1_micro |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Lasso | 0.0149 | 0.0149 | 6.5346 | 0.0351 | 0.0351 | 0.0876 | 0.0470 | 0.0591 | 0.0526 | 0.0526 | 0.0526 |
| EN | 0.0290 | 0.0290 | 6.4617 | 0.0348 | 0.0348 | 0.0882 | 0.0469 | 0.0593 | 0.0521 | 0.0521 | 0.0521 |
| SSL | 0.0132 | 1.3769 | 7.7893 | 0.0348 | 0.0348 | 0.0716 | 0.0489 | 0.0550 | 0.0522 | 0.0522 | 0.0522 |
| SSEN | 0.0067 | 1.4355 | 7.7913 | 0.0352 | 0.0352 | 0.0720 | 0.0490 | 0.0552 | 0.0527 | 0.0527 | 0.0527 |

Table 30: Median Model Fitness: X1k_B6_BV025050

| model | s0 | s1 | deviance | avg_acc | pce | ppv_macro | sn_macro | f1_macro | ppv_micro | sn_micro | f1_micro |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Lasso | 0.0746 | 0.0746 | 211.1127 | 0.6267 | 0.3733 | 0.4212 | 0.3905 | 0.4091 | 0.44 | 0.44 | 0.44 |
| EN | 0.1397 | 0.1397 | 210.8755 | 0.6267 | 0.3733 | 0.4220 | 0.3914 | 0.4109 | 0.44 | 0.44 | 0.44 |
| SSL | 0.0600 | 4.0000 | 211.0422 | 0.6267 | 0.3733 | 0.4201 | 0.4019 | 0.4134 | 0.44 | 0.44 | 0.44 |
| SSEN | 0.0300 | 4.0000 | 210.9342 | 0.6267 | 0.3733 | 0.4223 | 0.4034 | 0.4155 | 0.44 | 0.44 | 0.44 |

Table 31: IQR Model Fitness: X1k_B6_BV025050

| model | s0 | s1 | deviance | avg_acc | pce | ppv_macro | sn_macro | f1_macro | ppv_micro | sn_micro | f1_micro |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Lasso | 0.0167 | 0.0167 | 8.4828 | 0.0467 | 0.0467 | 0.1064 | 0.0690 | 0.0775 | 0.07 | 0.07 | 0.07 |
| EN | 0.0318 | 0.0318 | 8.4177 | 0.0467 | 0.0467 | 0.1057 | 0.0686 | 0.0787 | 0.07 | 0.07 | 0.07 |
| SSL | 0.0100 | 3.0000 | 9.9019 | 0.0467 | 0.0467 | 0.0899 | 0.0673 | 0.0726 | 0.07 | 0.07 | 0.07 |
| SSEN | 0.0000 | 3.0000 | 10.0360 | 0.0467 | 0.0467 | 0.0886 | 0.0690 | 0.0723 | 0.07 | 0.07 | 0.07 |

Table 32: Mean False Positive Rates and Power: X1k_B6_BV025050

| model | FDP | FWE | Power |
|---|---|---|---|
| Lasso | 0.8917 | 0.9802 | 0.2519 |
| EN | 0.9067 | 0.9808 | 0.3799 |
| SSL | 0.9016 | 0.9616 | 0.2955 |
| SSEN | 0.9139 | 0.9656 | 0.4264 |

Table 33: SD False Positive Rates and Power: X1k_B6_BV025050

| model | FDP | FWE | Power |
|---|---|---|---|
| Lasso | 0.1519 | 0.1393 | 0.1894 |
| EN | 0.1384 | 0.1372 | 0.2158 |
| SSL | 0.1867 | 0.1922 | 0.1977 |
| SSEN | 0.1758 | 0.1823 | 0.2211 |

Table 34: Median False Positive Rates and Power: X1k_B6_BV025050

| model | FDP | FWE | Power |
|---|---|---|---|
| Lasso | 0.9231 | 1 | 0.1667 |
| EN | 0.9322 | 1 | 0.3333 |
| SSL | 0.9444 | 1 | 0.3333 |
| SSEN | 0.9500 | 1 | 0.5000 |

Table 35: IQR False Positive Rates and Power: X1k_B6_BV025050

| model | FDP | FWE | Power |
|---|---|---|---|
| Lasso | 0.0934 | 0 | 0.1667 |
| EN | 0.0583 | 0 | 0.3333 |
| SSL | 0.0570 | 0 | 0.3333 |
| SSEN | 0.0357 | 0 | 0.1667 |

Table 36: Mean Probability of Inclusion for True Non-zero Paramters

| Model | X1 | X51 | X101 | X151 | X201 | X251 |
|---|---|---|---|---|---|---|
| Lasso | 0.2514 | 0.2522 | 0.2492 | 0.2492 | 0.2504 | 0.2588 |
| EN | 0.3772 | 0.3792 | 0.3800 | 0.3770 | 0.3778 | 0.3880 |
| SSL | 0.2922 | 0.2900 | 0.2916 | 0.2964 | 0.2982 | 0.3046 |
| SSEN | 0.4196 | 0.4238 | 0.4292 | 0.4270 | 0.4286 | 0.4302 |

Table 37: Probability of Inclusion for Relevant/Irrelevant Parameters: X1k_B6_BV025050

| Model | Relevance | Mean | SD |
|---|---|---|---|
| Lasso | Irrelevant | 0.0181 | 0.0174 |
| EN | Irrelevant | 0.0327 | 0.0347 |
| SSL | Irrelevant | 0.0302 | 0.0184 |
| SSEN | Irrelevant | 0.0513 | 0.0356 |
| Lasso | Relevant | 0.2519 | 0.0036 |
| EN | Relevant | 0.3799 | 0.0042 |
| SSL | Relevant | 0.2955 | 0.0054 |
| SSEN | Relevant | 0.4264 | 0.0040 |

### 3.6 B6_BV025050 Summary (Reduced for Manuscript)

Table 38: Mean Model Fitness: X1k_B6_BV025050

| model | s0 | s1 | deviance | FDP | Power |
|---|---|---|---|---|---|
| Lasso | 0.0770 | 0.0770 | 210.9040 | 0.8917 | 0.2519 |
| EN | 0.1445 | 0.1445 | 210.6891 | 0.9067 | 0.3799 |
| SSL | 0.0553 | 3.0590 | 210.9524 | 0.9016 | 0.2955 |
| SSEN | 0.0301 | 2.9224 | 210.8284 | 0.9139 | 0.4264 |

Table 39: Mean Probability of Inclusion for True Non-zero Paramters

| Model | X1 | X51 | X101 | X151 | X201 | X251 |
|---|---|---|---|---|---|---|
| Lasso | 0.2514 | 0.2522 | 0.2492 | 0.2492 | 0.2504 | 0.2588 |
| EN | 0.3772 | 0.3792 | 0.3800 | 0.3770 | 0.3778 | 0.3880 |
| SSL | 0.2922 | 0.2900 | 0.2916 | 0.2964 | 0.2982 | 0.3046 |
| SSEN | 0.4196 | 0.4238 | 0.4292 | 0.4270 | 0.4286 | 0.4302 |

### 3.7 Summary (Reduced for Manuscript)

| scenario | model | s0 | s1 | deviance | FDP | Power |
|---|---|---|---|---|---|---|
| Scenario 1 | Lasso | 0.1031141 | 0.1031141 | 216.8022 | 0.6437452 | 0.0089667 |

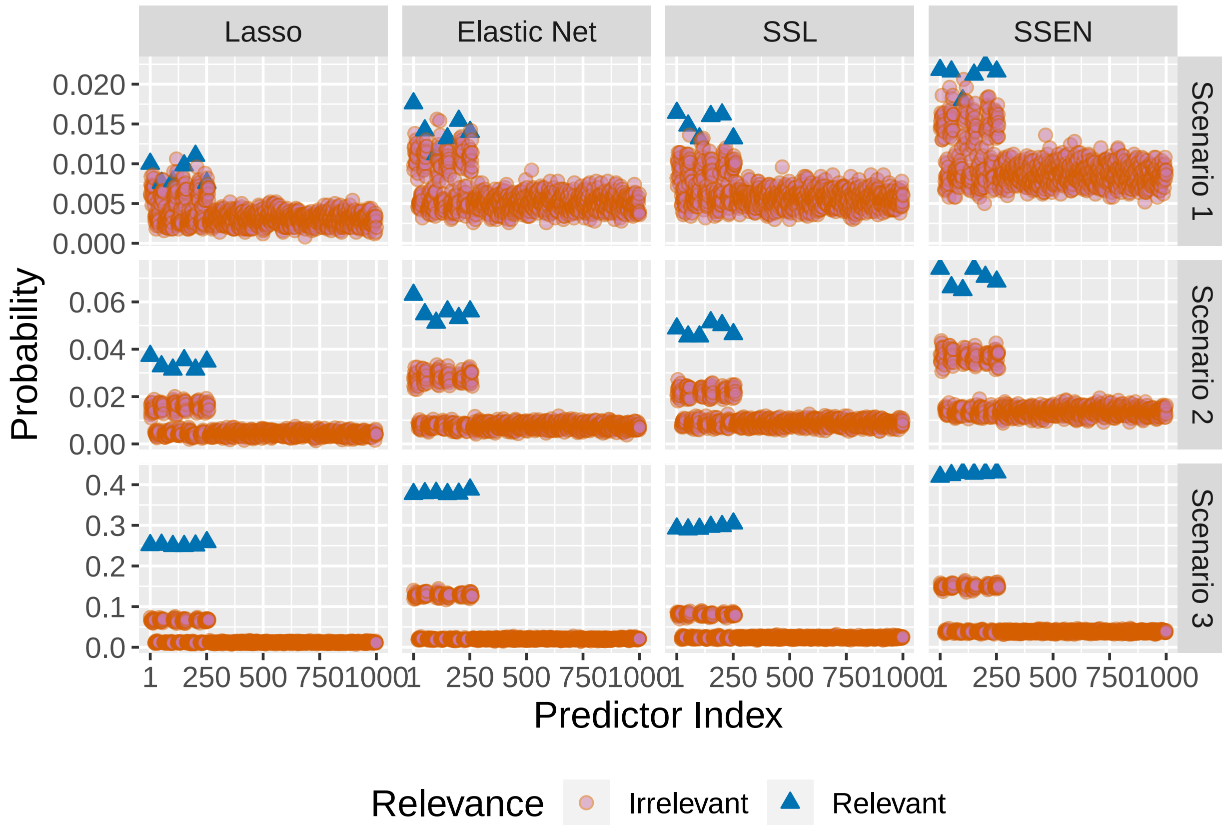
| scenario | model | s0 | s1 | deviance | FDP | Power |
|---|---|---|---|---|---|---|
| Scenario 1 | EN | 0.2020326 | 0.2020326 | 216.7978 | 0.5637058 | 0.0142667 |
| Scenario 1 | SSL | 0.0242820 | 1.7258000 | 217.2746 | 0.4052290 | 0.0149667 |
| Scenario 1 | SSEN | 0.0151680 | 1.6154000 | 217.2341 | 0.3661322 | 0.0211000 |
| Scenario 2 | Lasso | 0.1000553 | 0.1000553 | 216.7096 | 0.7166263 | 0.0338333 |
| Scenario 2 | EN | 0.1936949 | 0.1936949 | 216.6557 | 0.6555374 | 0.0557000 |
| Scenario 2 | SSL | 0.0307700 | 2.0642000 | 217.2401 | 0.5434850 | 0.0479333 |
| Scenario 2 | SSEN | 0.0179940 | 1.9542000 | 217.1768 | 0.5099825 | 0.0697333 |
| Scenario 3 | Lasso | 0.0770487 | 0.0770487 | 210.9040 | 0.8917342 | 0.2518667 |
| Scenario 3 | EN | 0.1445050 | 0.1445050 | 210.6891 | 0.9066802 | 0.3798667 |
| Scenario 3 | SSL | 0.0553260 | 3.0590000 | 210.9524 | 0.9015955 | 0.2955000 |
| Scenario 3 | SSEN | 0.0301220 | 2.9224000 | 210.8284 | 0.9139263 | 0.4264000 |

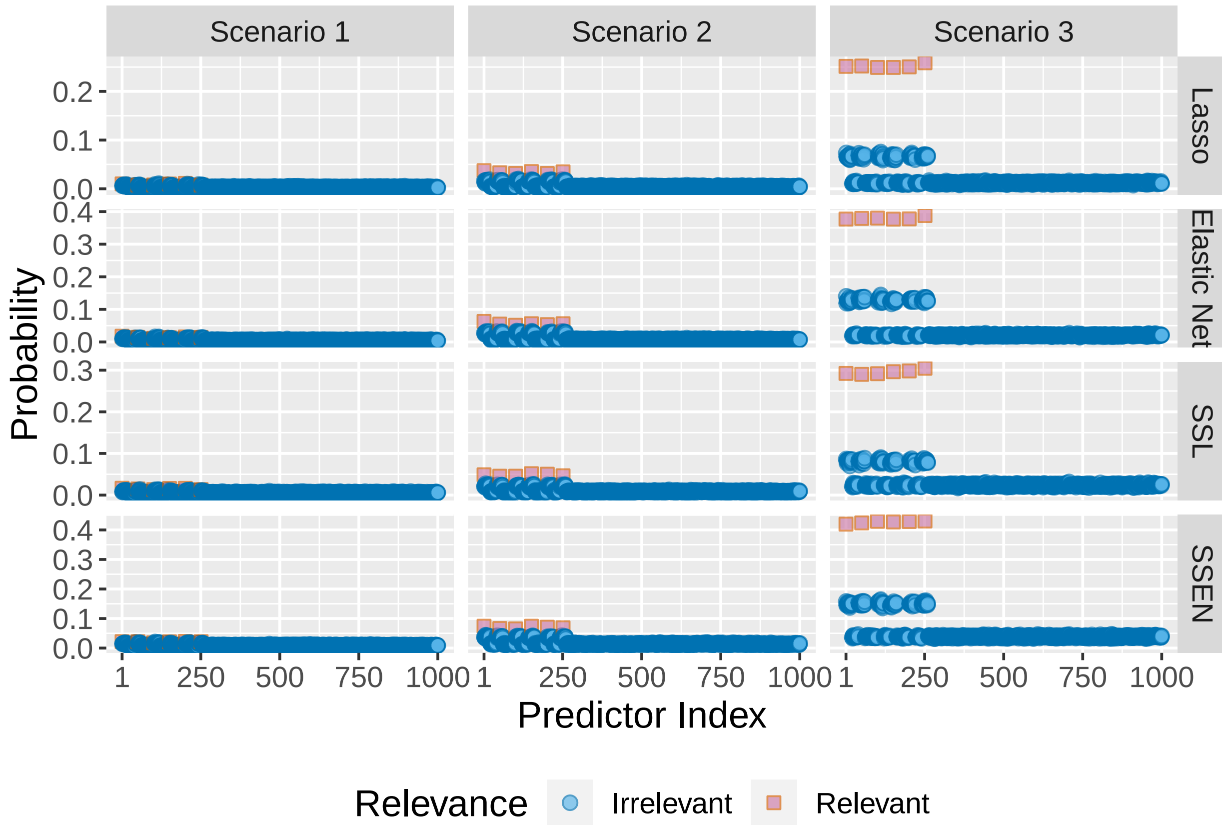
| Scenario | Model | X1 | X51 | X101 | X151 | X201 | X251 |
|---|---|---|---|---|---|---|---|
| Scenario 1 | Lasso | 0.0100 | 0.0076 | 0.0078 | 0.0098 | 0.0110 | 0.0076 |
| Scenario 1 | EN | 0.0176 | 0.0142 | 0.0112 | 0.0132 | 0.0154 | 0.0140 |
| Scenario 1 | SSL | 0.0164 | 0.0148 | 0.0132 | 0.0160 | 0.0162 | 0.0132 |
| Scenario 1 | SSEN | 0.0218 | 0.0216 | 0.0180 | 0.0212 | 0.0224 | 0.0216 |
| Scenario 2 | Lasso | 0.0372 | 0.0328 | 0.0314 | 0.0354 | 0.0314 | 0.0348 |
| Scenario 2 | EN | 0.0630 | 0.0548 | 0.0512 | 0.0560 | 0.0532 | 0.0560 |
| Scenario 2 | SSL | 0.0488 | 0.0454 | 0.0454 | 0.0514 | 0.0502 | 0.0464 |
| Scenario 2 | SSEN | 0.0740 | 0.0662 | 0.0650 | 0.0740 | 0.0706 | 0.0686 |
| Scenario 3 | Lasso | 0.2514 | 0.2522 | 0.2492 | 0.2492 | 0.2504 | 0.2588 |
| Scenario 3 | EN | 0.3772 | 0.3792 | 0.3800 | 0.3770 | 0.3778 | 0.3880 |
| Scenario 3 | SSL | 0.2922 | 0.2900 | 0.2916 | 0.2964 | 0.2982 | 0.3046 |
| Scenario 3 | SSEN | 0.4196 | 0.4238 | 0.4292 | 0.4270 | 0.4286 | 0.4302 |

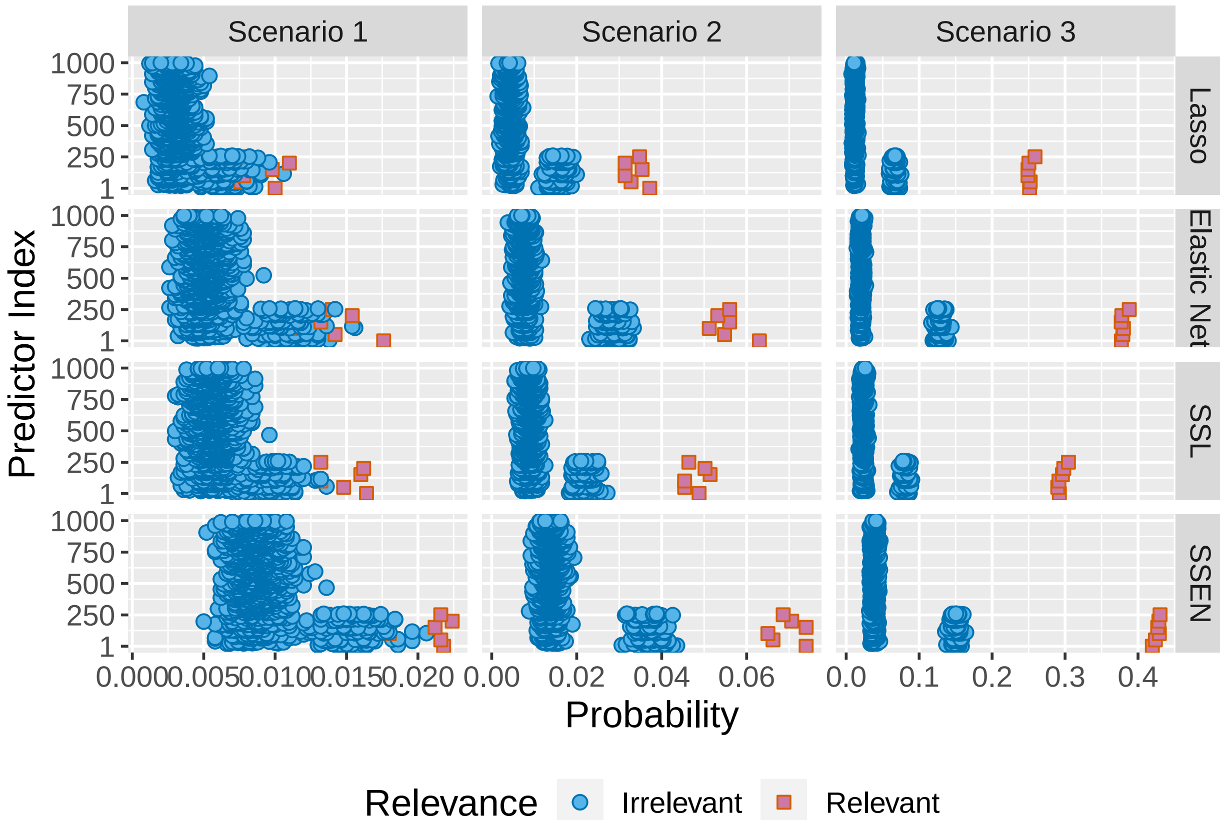
# 4 Plots

## 4.1 Inclusion Probabilities

```
## Warning: Using alpha for a discrete variable is not advised.
```

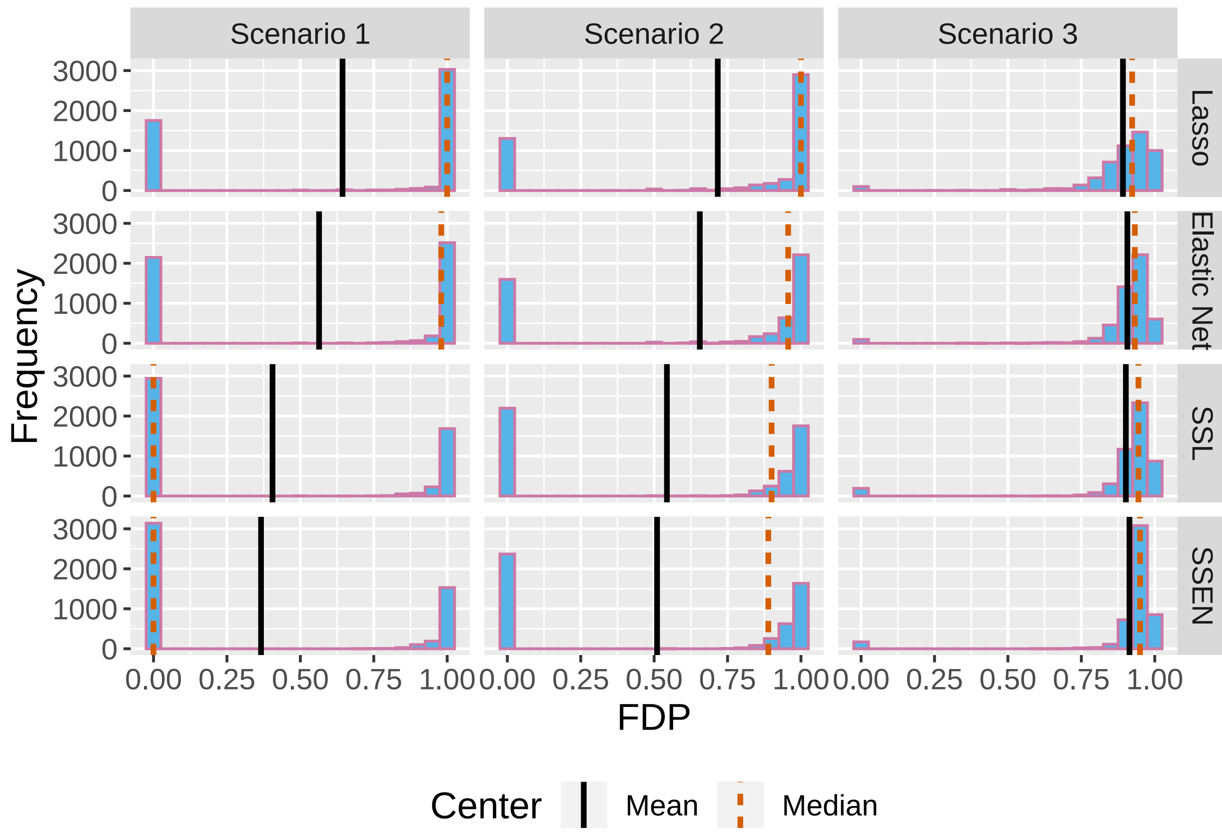

Lasso
Elastic Net
SSL
SSEN
Scenario 1
Scenario 2
Scenario 3
Probability
0.020
0.015
0.010
0.005
0.000
0.06
0.04
0.02
0.00
0.4
0.3
0.2
0.1
0.0
1
250
500
750
1000
Predictor Index
Relevance
Irrelevant
Relevant

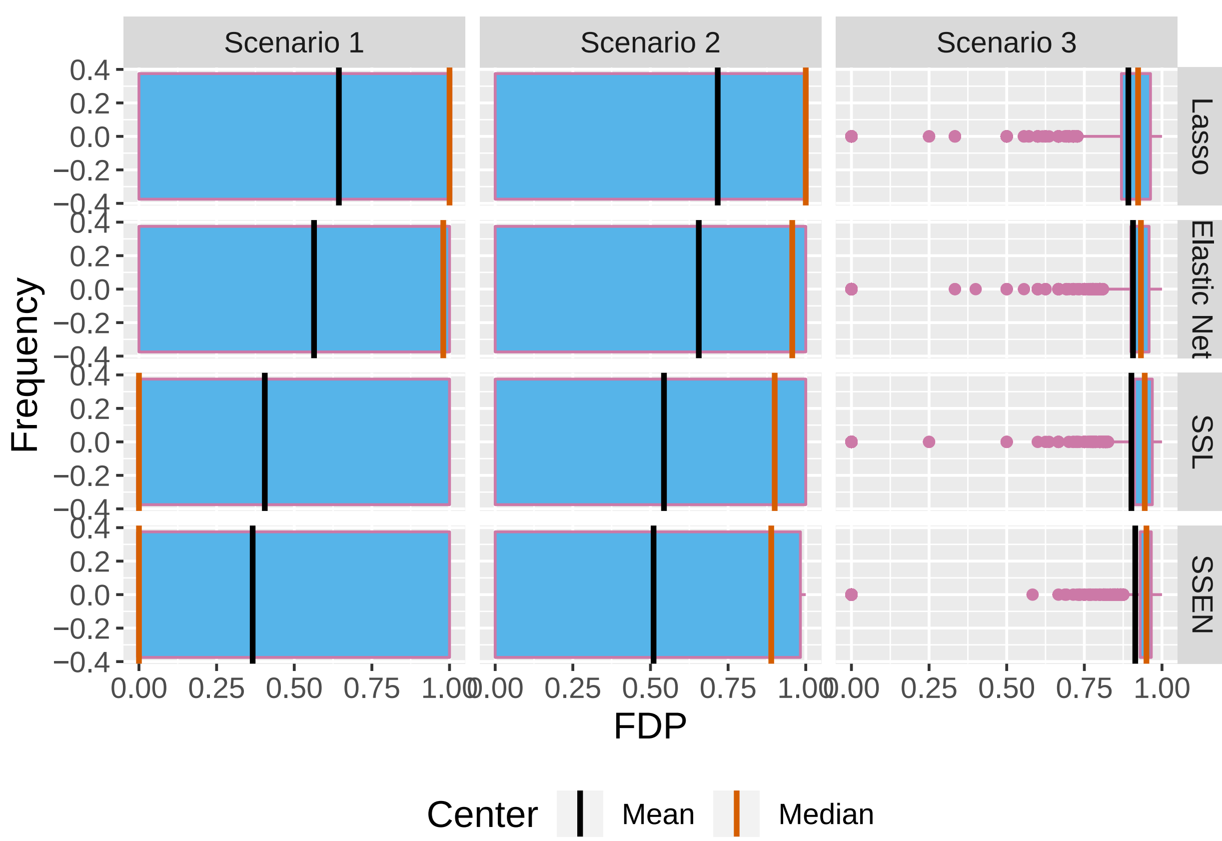

Scenario 1
Scenario 2
Scenario 3
Lasso
Elastic Net
SSL
SSEN
Probability
Predictor Index
Relevance
Irrelevant
Relevant



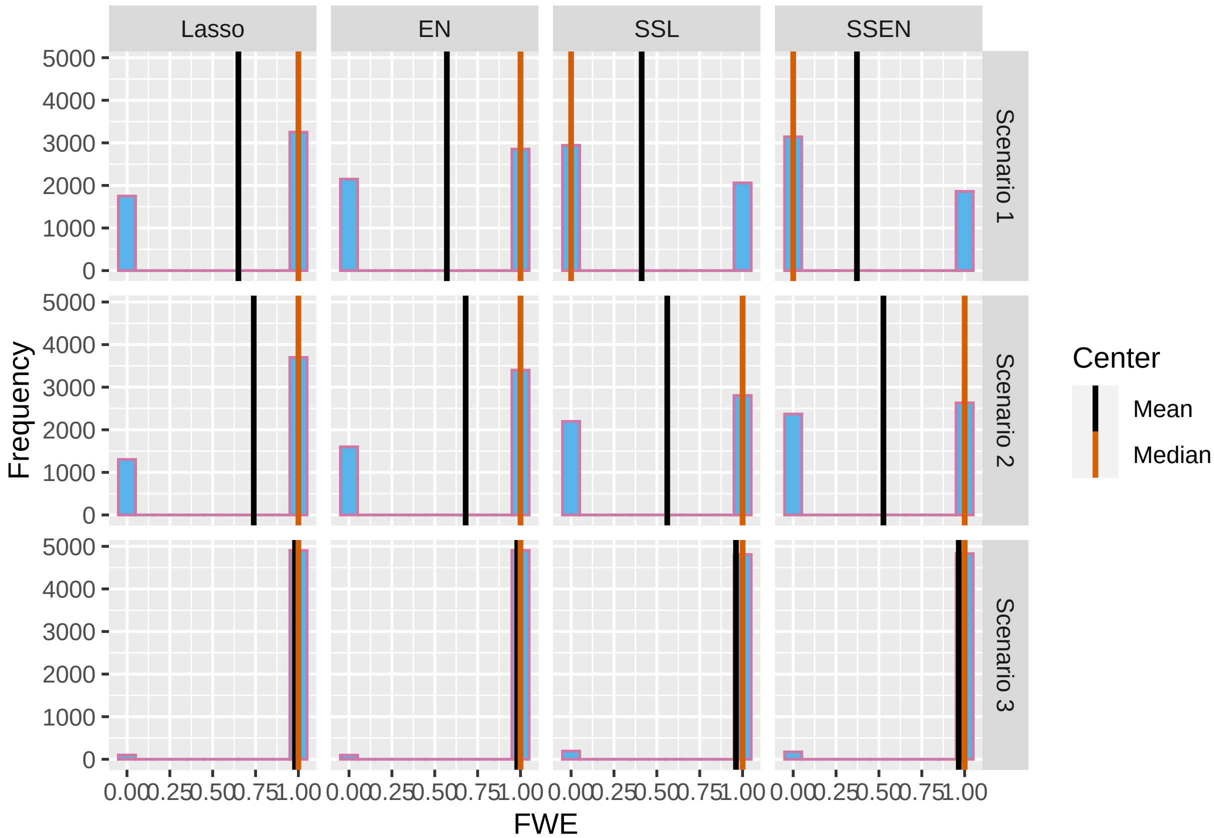
Scenario 1
Scenario 2
Scenario 3
Lasso
Elastic Net
SSL
SSEN
Predictor Index
1000
750
500
250
1
0.0000.0050.0100.0150.020
0.00 0.02 0.04 0.06
0.0 0.1 0.2 0.3 0.4
Probability
Relevance
Irrelevant
Relevant

### 4.2 FDP


Scenario 1
Scenario 2
Scenario 3
Lasso
Elastic Net
SSL
SSEN
Frequency
3000
2000
1000
0
0.00 0.25 0.50 0.75 1.00
FDP
Center
Mean
Median

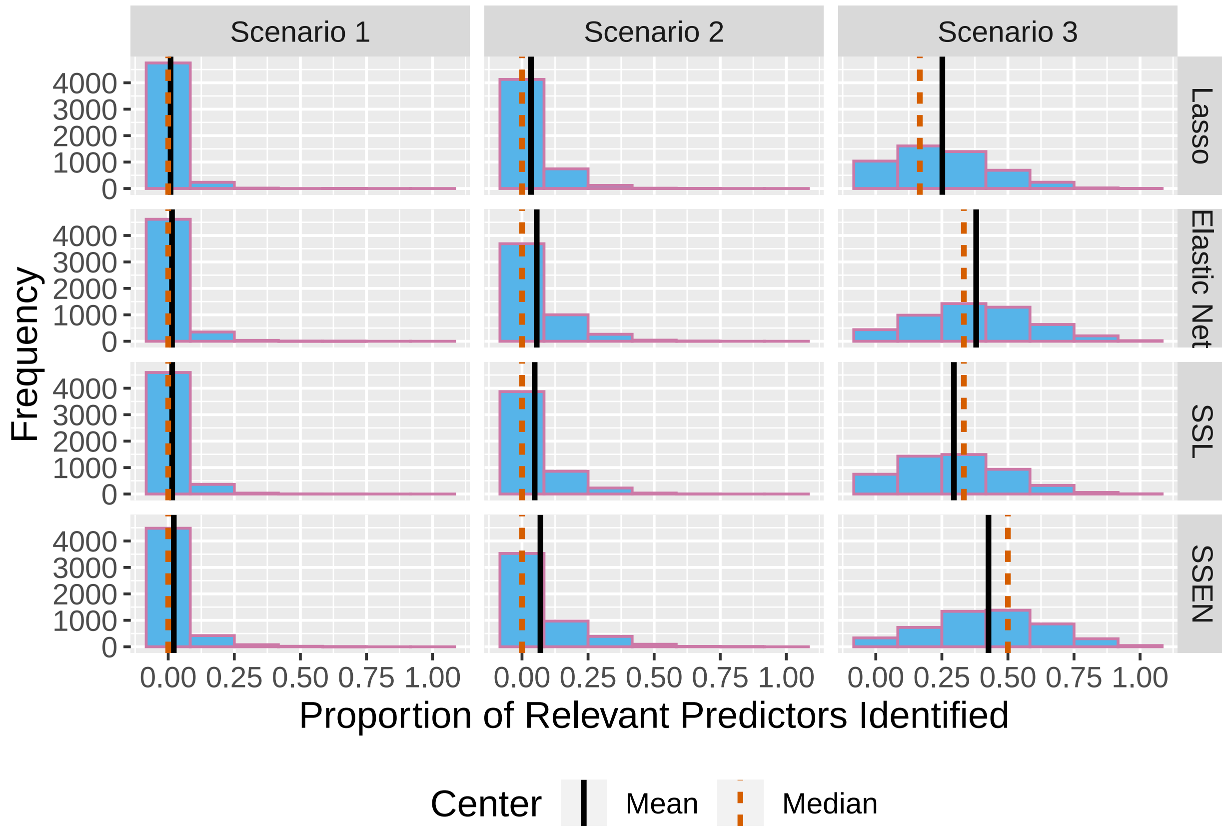

Scenario 1
Scenario 2
Scenario 3
Lasso
Elastic Net
SSL
SSEN
Frequency
0.4
0.2
0.0
−0.2
−0.4
0.00 0.25 0.50 0.75 1.00
FDP
Center
Mean
Median

### 4.3 FWE

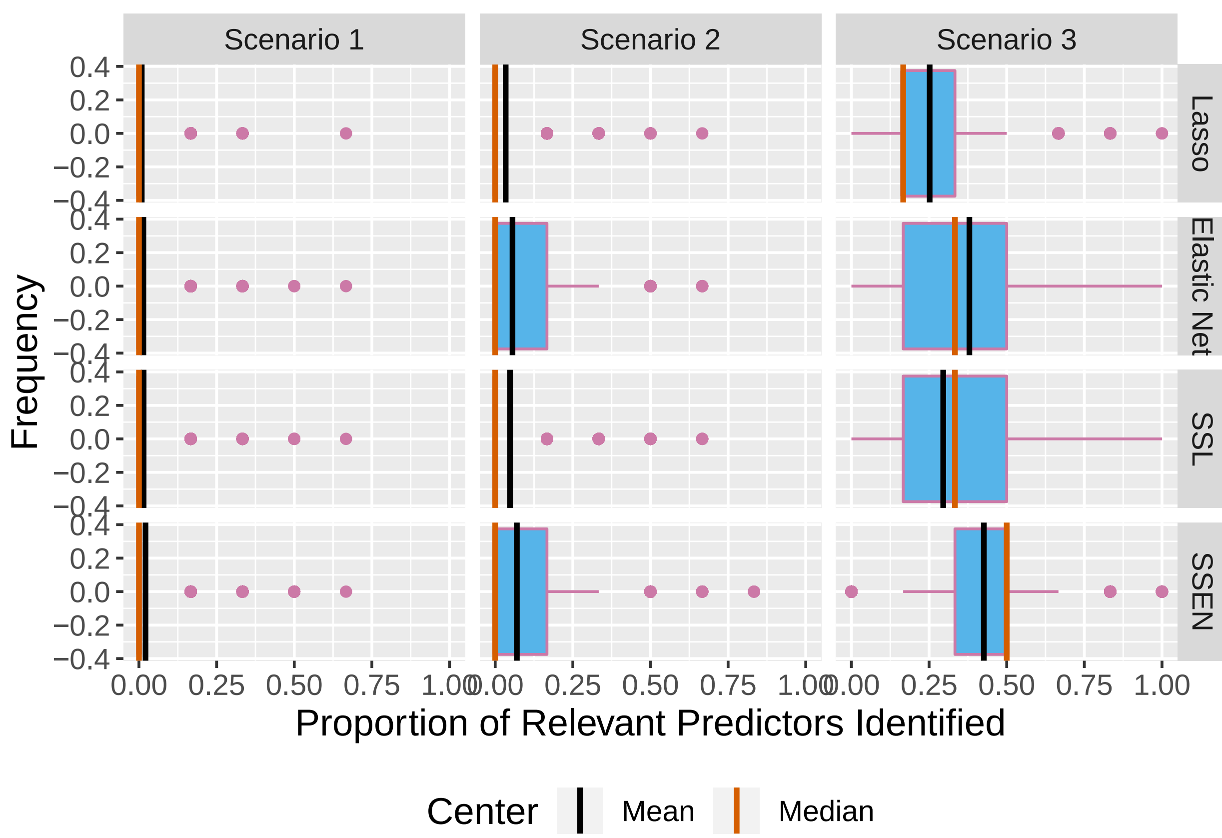


Lasso
EN
SSL
SSEN
Scenario 1
Scenario 2
Scenario 3
Frequency
5000
4000
3000
2000
1000
0
0.00
0.25
0.50
0.75
1.00
FWE
Center
Mean
Median

### 4.4 Power



Scenario 1
Scenario 2
Scenario 3
Lasso
Elastic Net
SSL
SSEN
Frequency
4000
3000
2000
1000
0
0.00 0.25 0.50 0.75 1.00
Proportion of Relevant Predictors Identified
Center
Mean
Median

Scenario 1
Scenario 2
Scenario 3
Lasso
Elastic Net
SSL
SSEN
Frequency
0.4
0.2
0.0
−0.2
−0.4
0.00 0.25 0.50 0.75 1.00
Proportion of Relevant Predictors Identified
Center
Mean
Median