SADM Simulations

Justin M. Leach

1 Introduction

This file contains `R` code details regarding simulations in the paper "The Spike-and-Slab Elastic Net as a Classification Tool in Alzheimer's Disease." Simulations and analyses were run on UAB's cluseter. `R` code and scripts for running code on the cluster are in folders "Rcode" and "scripts," respectively. In order to replicate simulations and results, you will need to edit path names and possibly re-write scripts from scratch if your institution does not use slurm workload manager (https://slurm.schedmd.com/overview.html).

2 What simulation?

Note that we use the UAB cluster to run simulations. We have included the template for reproducing the simulations here, noting what changes are necessary to reproduce the simulations used in the paper. The exact `R` code for each scenario is found in the folder "Rcode." We ultimately include analyses on 2,500 data sets of sample size $N = 250$ using 40×40 images as predictors to generate binary outcomes as described below. Note that in order to properly reproduce the results, it will be necessary to edit the path names to match your machine.

2.1 Code for simulating data

We will generate the data this using `sim2Dpredictr`. Below is the general code structure used to generate the data. When necessary, we can change the `B.values` argument in `beta_builder()` to assign different parameter sizes (e.g., 0.05). Similarly, unbalanced data was generated by changing the `mu` argument in `sim_Y_MVN_X()`; `mu = 0` results in approximately balanced data sets, `mu = -1` and `mu = -1.25` were used to obtain unbalanced data for $\beta_j = 0.10$ and $\beta_j = 0.05$, respectively. Differing values of `mu` were used to obtain a balance in outcomes similar (but of course not exact) to that in ADNI data.

```
library(tidyverse)
library(sim2Dpredictr)
library(reshape2)

# colorblind palattes
#palette using grey
cbpg <- c("#999999", "#E69F00", "#56B4E9", "#009E73",
          "#F0E442", "#0072B2", "#D55E00", "#CC79A7")

#palette using black
cbpb <- c("#000000", "#E69F00", "#56B4E9", "#009E73",
          "#F0E442", "#0072B2", "#D55E00", "#CC79A7")
```

```r
# simulate data for official analysis

library(tidyverse)
library(sim2Dpredictr)

# load data -> use array jobs
runID <- as.numeric(Sys.getenv("SLURM_ARRAY_TASK_ID"))

# different seed for each job
set.seed(31323 + runID)

# How many simulations?
M <- 500

# How many subjects per dataset?
N <- 250

# Continuous or binary predictors?
x.ms <- "continuous"
# x.ms <- "binary"

# Continuous or binary outcomes?
y.ms <- "binomial"
# y.ms <- "gaussian"

# image resolution; i.e., number of predictors
im.res <- c(40, 40)

if (x.ms == "continuous") {
  # can modify these arguments if needed.
  L = chol_s2Dp(im.res = im.res, rho = 0.90,
                corr.structure = "ar1",
                triangle = "lower")
}

# generate parameter vector
betas <- beta_builder(index.type = "ellipse",
                      w = 8, h = 8,
                      row.index = 15, col.index = 24,
                      B.values = 0.1, im.res = im.res)

# to store data
data.list <- list()

# run simulations
set.seed(23432)
t1 <- proc.time()
for (m in 1:M) {

  # generate data
  if (x.ms == "continuous") {
    datm <- sim_Y_MVN_X(N = N, dist = y.ms,
                        mu = -1,
```

```
                        L = L$L, S = L$S,
                        B = betas$B)
  } else {
    # from second line on will likely need adjustments
    datm <- sim_Y_Binary_X(N = N, dist = "binomial", B = betas$B, im.res = im.res,
                           lambda = 50, sub.area = TRUE,
                           min.sa = c(0.15, 0.2), max.sa = c(0.25, 0.5),
                           radius.bounds.min.sa = c(0.02, 0.04),
                           radius.bounds.max.sa = c(0.045, 0.06))
  }

  data.list[[m]] <- datm
  cat("Simulation ", m, " has completed. \n")
}
proc.time() - t1

# # break up into more manageable pieces; here 10 independent datasets of 500 each.
# D <- 10
# # size of each data set; i.e., 1k in this case.
# S <- M/D
#
# for (d in 1:D) {
#   min <- 1 + (S * (d - 1))
#   max <- S * d
#   saveRDS(data.list[min:max],
#           file = paste0("/data/user/jleach/sim_06_2021/simdata/simdata_N500_50x50_", d, ".RDS"))
# }

saveRDS(data.list,
        file = paste0("/data/user/jleach/sim_06_2021/simdata/simdata_N250_40x40_", runID, ".RDS"))
```

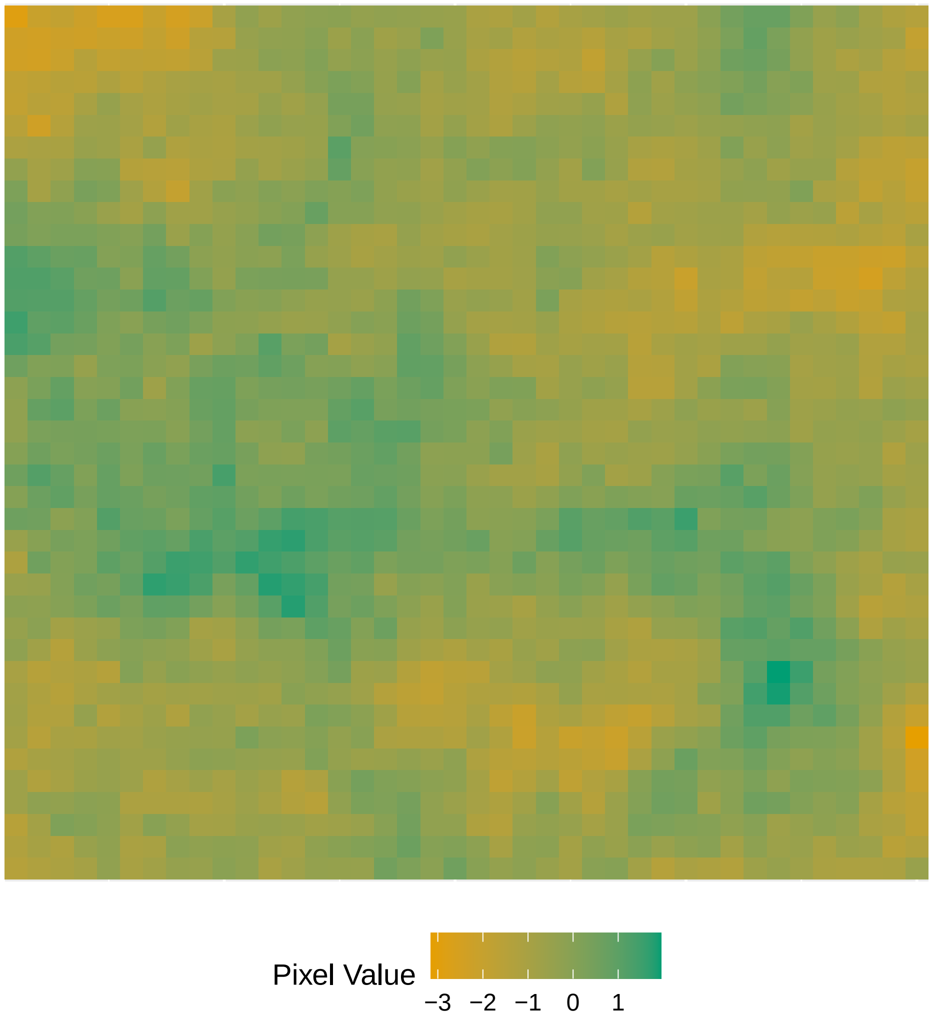
2.2 Example subject image

The outcomes are binary, and below we show an example of predictor images. You can simply change `filter(subjectID == 1)` to any number between 1 and 50 to see other example images.

```
# helps properly display matrices as images
rotate = function(x){
    t(apply(x, 2, rev))
}

# width and height of images is 50 (i.e, 50x50 resolution)
im.wh <- 40

ex.dat <- readRDS(file = "C:/Users/cotto/Documents/Publications/paper3/Simulation R Code/simdata/simdata
ex.dat.smallB <- readRDS(file = "C:/Users/cotto/Documents/Publications/paper3/Simulation R Code/simdata/

ex.dat.1 <- ex.dat[[1]]
ex.x.1 <- matrix(as.numeric(ex.dat.1 %>%
                              filter(subjectID == 1) %>%
                              select(-Y, -subjectID)),
                 nrow = im.wh, ncol = im.wh, byrow = TRUE)
```

```r
# subjectID = 1
X.1.2D <- rotate(
  matrix(
    ex.x.1,
    nrow = im.wh, ncol = im.wh,
    byrow = TRUE
    )
  )


X.1.2D.melt <- melt(
  data.frame(x = 1:im.wh, X.1.2D),
  id = "x")
```

```r
ggplot(X.1.2D.melt) +
  geom_raster(aes(x = x, y = variable, fill = value)) +
  labs(fill = "Pixel Value") +
  scale_fill_gradient(low = cbpg[2], high = cbpg[4]) +
  scale_x_continuous(expand = c(0, 0))+   # get rid of extra space on x-axis
  # guides(fill = FALSE)+                  # turn off color legend
  theme(axis.text = element_blank(),     # turn off the axis annotations
        axis.ticks = element_blank(),
        axis.title = element_blank(),
        legend.position = "bottom")
```



2.3 Balance in outcome

We can check that the balance in the outcomes is what we expect. The cortical thickness data set had 14.29% with dementia and the tau PET data set had 13.53% with dementia. On average, we are close to these percentages in both data sets.

```
y.means.ex <- c()

for (i in 1:length(ex.dat)) {
  y.means.ex[i] <- mean(ex.dat[[i]]$Y)
}

# mean(y.means.ex)
# sd(y.means.ex)

y.means.ex.smallB <- c()

for (i in 1:length(ex.dat.smallB)) {
  y.means.ex.smallB[i] <- mean(ex.dat.smallB[[i]]$Y)
```

```
}

# mean(y.means.ex.smallB)
# sd(y.means.ex.smallB)

balance.smry <- cbind(
  data.frame(
    Beta = c(0.10, 0.05),
    Mean = c(mean(y.means.ex), mean(y.means.ex.smallB)),
    SD = c(sd(y.means.ex), sd(y.means.ex.smallB))
  ),
  rbind(quantile(y.means.ex),
        quantile(y.means.ex.smallB))
)
knitr::kable(balance.smry,
             caption = "Summaries for Percentage of Events in each Simulation",
             digits = 4)
```

Table 1: Summaries for Percentage of Events in each Simulation

Beta	Mean	SD	0%	25%	50%	75%	100%
0.10	0.1384	0.0223	0.072	0.124	0.136	0.152	0.216
0.05	0.1304	0.0219	0.068	0.116	0.128	0.144	0.184

3 Analysis details

Of course the difficult part is analyzing the simulated data. We will analyze using traditional models, spike-and-slab models without spatial structure, and spike-and-slab models with spatial structure for the lasso and the elastic net with compromise parameter 0.5, i.e., halfway compromise between ridge and lasso.

We do not reproduce the files used to run the analyses, but they are found in the folder "Rcode" and easily identified: `glmnet` models run with "rcode_glmnet_N250_40x40.R", `ssnet` models without IAR priors with "rcode_ssnet_N250_40x40.R", and `ssnet` models with IAR priors with "rcode_ssnet_iar_N250_40x40.R".

4 Results ($\beta = 0.1$)

4.1 Loading results

Note that while `glmnet` has only one output file, we have to extract and combine results from the `ssnet` models. For now, we extract only inference, not parameter estimates. Remember to change the path names!

```
# load data

# only take 1st 2500 analyses from glmnet models
glmnet.results <- readRDS(file = "C:/Users/cotto/Documents/Publications/paper3/Simulation R Code/results

ssnet.results <- dplyr::bind_rows(
  readRDS(file = "C:/Users/cotto/Documents/Publications/paper3/Simulation R Code/results/results_ssnet_
  readRDS(file = "C:/Users/cotto/Documents/Publications/paper3/Simulation R Code/results/results_ssnet_
```

```
  readRDS(file = "C:/Users/cotto/Documents/Publications/paper3/Simulation R Code/results/results_ssnet_
  readRDS(file = "C:/Users/cotto/Documents/Publications/paper3/Simulation R Code/results/results_ssnet_
  readRDS(file = "C:/Users/cotto/Documents/Publications/paper3/Simulation R Code/results/results_ssnet_
)

ssnet.iar.results <- dplyr::bind_rows(
  readRDS(file = "C:/Users/cotto/Documents/Publications/paper3/Simulation R Code/results/results_ssnet_
  readRDS(file = "C:/Users/cotto/Documents/Publications/paper3/Simulation R Code/results/results_ssnet_
  readRDS(file = "C:/Users/cotto/Documents/Publications/paper3/Simulation R Code/results/results_ssnet_
  readRDS(file = "C:/Users/cotto/Documents/Publications/paper3/Simulation R Code/results/results_ssnet_
  readRDS(file = "C:/Users/cotto/Documents/Publications/paper3/Simulation R Code/results/results_ssnet_
  readRDS(file = "C:/Users/cotto/Documents/Publications/paper3/Simulation R Code/results/results_ssnet_
  readRDS(file = "C:/Users/cotto/Documents/Publications/paper3/Simulation R Code/results/results_ssnet_
  readRDS(file = "C:/Users/cotto/Documents/Publications/paper3/Simulation R Code/results/results_ssnet_
  readRDS(file = "C:/Users/cotto/Documents/Publications/paper3/Simulation R Code/results/results_ssnet_
  readRDS(file = "C:/Users/cotto/Documents/Publications/paper3/Simulation R Code/results/results_ssnet_
)
```

4.2 Obtaining optimal models

We need to extract means by elastic net penalty (0.5 or 1), s_0, and s_1.

```
glmnet.smry.0 <- glmnet.results %>%
  select(-model) %>%
  group_by(alpha) %>%
  nest() %>%
  mutate(
    means = map(.x = data, .f = function(x) map_df(x, mean, na.rm = TRUE)),
    sds = map(.x = data, .f = function(x) map_df(x, sd, na.rm = TRUE))
    )

#glmnet.smry.0

glmnet.smry <- glmnet.smry.0 %>%
  select(-data, -sds) %>%
  unnest(means)

#glmnet.smry

ssnet.smry.0 <- ssnet.results %>%
  select(-model) %>%
  group_by(alpha, s0, s1) %>%
  nest() %>%
  mutate(
    means = map(.x = data, .f = function(x) map_df(x, mean, na.rm = TRUE)),
    sds = map(.x = data, .f = function(x) map_df(x, sd, na.rm = TRUE))
    )

#ssnet.smry.0

ssnet.smry <- ssnet.smry.0 %>%
  select(-data, -sds) %>%
```

```
  unnest(means)

#ssnet.smry

ssnet.iar.smry.0 <- ssnet.iar.results %>%
  select(-model) %>%
  group_by(alpha, s0, s1) %>%
  nest() %>%
  mutate(
    means = map(.x = data, .f = function(x) map_df(x, mean, na.rm = TRUE)),
    sds = map(.x = data, .f = function(x) map_df(x, sd, na.rm = TRUE))
    )

#ssnet.iar.smry.0

ssnet.iar.smry <- ssnet.iar.smry.0 %>%
  select(-data, -sds) %>%
  unnest(means)

#ssnet.iar.smry
```

Then we need to select the optimal model parameters for each of the 6 modeling approaches:

```
optimal.ssnet.a05 <- ssnet.smry %>%
  ungroup() %>%
  filter(alpha == 0.5) %>%
  filter(deviance == min(deviance))

optimal.ssnet.iar.a05 <- ssnet.iar.smry %>%
  ungroup() %>%
  filter(alpha == 0.5) %>%
  filter(deviance == min(deviance))

optimal.ssnet.a1 <- ssnet.smry %>%
  ungroup() %>%
  filter(alpha == 1) %>%
  filter(deviance == min(deviance))

optimal.ssnet.iar.a1 <- ssnet.iar.smry %>%
  ungroup() %>%
  filter(alpha == 1) %>%
  filter(deviance == min(deviance))

optimal.glmnet <- glmnet.smry %>%
  ungroup()

results.df <- rbind(optimal.glmnet %>% filter(alpha == 1),
                    optimal.ssnet.a1,
                    optimal.ssnet.iar.a1,
                    optimal.glmnet %>% filter(alpha == 0.5),
                    optimal.ssnet.a05,
                    optimal.ssnet.iar.a05)
```

4.3 Tables

```r
knitr::kable(cbind(model = c("Lasso", "SSL", "SSL-IAR",
                             "EN", "SSEN", "SSEN-IAR"),
                   results.df %>%
                     select(s0, s1, deviance, auc, mse, mae, misclassification)),
             caption = "Prediction Error over 2,500 Simulated Data Sets",
             digits = 4)
```

Table 2: Prediction Error over 2,500 Simulated Data Sets

model	s0	s1	deviance	auc	mse	mae	misclassification
Lasso	0.0222	0.0222	99.9108	0.9398	0.0610	0.1257	0.0858
SSL	0.0800	1.0000	84.1390	0.9568	0.0508	0.1027	0.0702
SSL-IAR	0.1000	1.0000	74.3328	0.9667	0.0445	0.0893	0.0606
EN	0.0370	0.0370	97.7755	0.9432	0.0596	0.1240	0.0827
SSEN	0.0700	1.0000	87.1561	0.9541	0.0527	0.1099	0.0726
SSEN-IAR	0.1000	2.0000	63.4708	0.9762	0.0376	0.0716	0.0511

```r
knitr::kable(cbind(model = c("Lasso", "SSL", "SSL-IAR",
                             "EN", "SSEN", "SSEN-IAR"),
                   results.df %>%
                     select(s0, s1, accuracy, sensitivity, specificity, ppv, npv, mcc, f1)),
             caption = "Results over 2,500 Simulated Data Sets",
             digits = 4)
```

Table 3: Results over 2,500 Simulated Data Sets

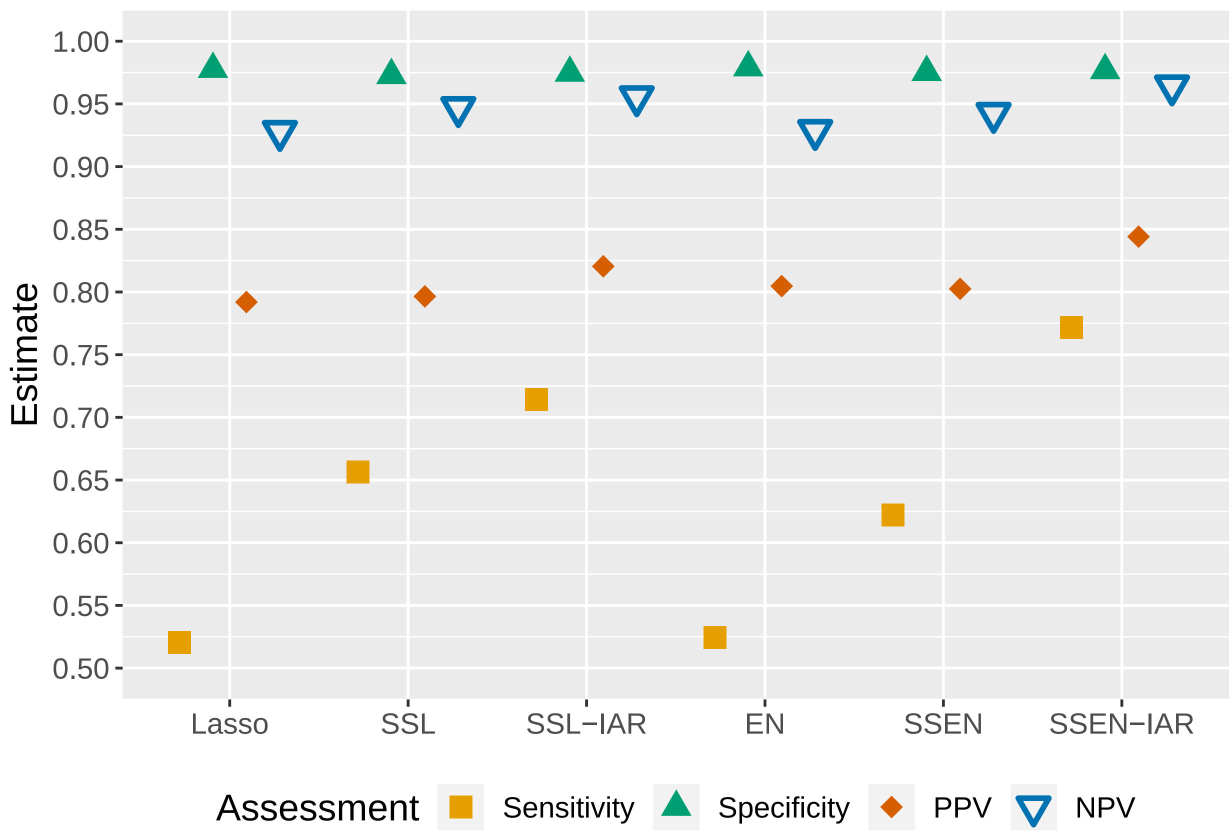
model	s0	s1	accuracy	sensitivity	specificity	ppv	npv	mcc	f1
Lasso	0.0222	0.0222	0.9156	0.5203	0.9778	0.7920	0.9280	0.5963	0.6224
SSL	0.0800	1.0000	0.9298	0.6564	0.9729	0.7965	0.9470	0.6831	0.7173
SSL-IAR	0.1000	1.0000	0.9394	0.7141	0.9747	0.8205	0.9556	0.7305	0.7620
EN	0.0370	0.0370	0.9173	0.5246	0.9790	0.8047	0.9287	0.6049	0.6289
SSEN	0.0700	1.0000	0.9274	0.6222	0.9753	0.8025	0.9424	0.6656	0.6972
SSEN-IAR	0.1000	2.0000	0.9489	0.7717	0.9768	0.8441	0.9643	0.7774	0.8050

```r
#for LaTeX
# xtable::xtable(cbind(model = c("Lasso", "SSL", "SSL-IAR",
#                              "EN", "SSEN", "SSEN-IAR"),
#                    results.df %>%
#                      select(s0, s1, deviance, auc, mse, mae, misclassification)),
#                digits = 4)
# xtable::xtable(cbind(model = c("Lasso", "SSL", "SSL-IAR",
#                              "EN", "SSEN", "SSEN-IAR"),
#                    results.df %>%
#                      select(s0, s1, accuracy, sensitivity, specificity, ppv, npv, mcc, f1)),
#                digits = 4)
```

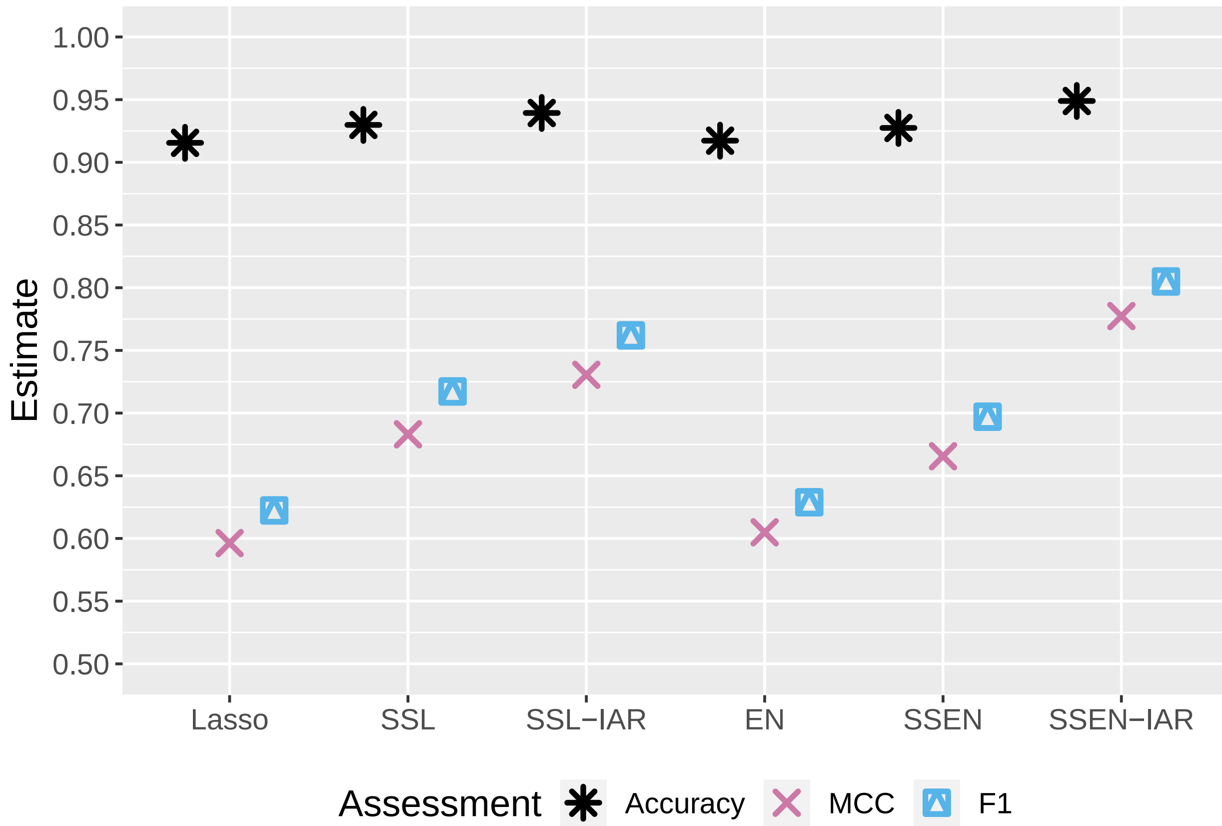
4.4 Figure

```
# wrangle for figures
accuracy.wide <- cbind(Model = c("Lasso", "SSL", "SSL-IAR",
                              "EN", "SSEN", "SSEN-IAR"),
                       results.df %>% select(accuracy, sensitivity, specificity,
                                             ppv, npv, mcc, f1)
)

# number of models in total
nm <- 6
all.long <- cbind(accuracy.wide %>% select(Model),
                  Estimate = c(accuracy.wide$accuracy,
                               accuracy.wide$sensitivity,
                               accuracy.wide$specificity,
                               accuracy.wide$ppv,
                               accuracy.wide$npv,
                               accuracy.wide$mcc,
                               accuracy.wide$f1),
                  Assessment = c(rep("Accuracy", nm),
                                 rep("Sensitivity", nm),
                                 rep("Specificity", nm),
                                 rep("PPV", nm),
                                 rep("NPV", nm),
                                 rep("MCC", nm),
                                 rep("F1", nm)))
names(all.long) <- c("Model", "Estimate", "Assessment")
all.long$Model <- factor(all.long$Model,
                         levels = c("Lasso", "SSL", "SSL-IAR",
                                    "EN", "SSEN", "SSEN-IAR"))
all.long$Assessment <- factor(all.long$Assessment,
                              levels = c("Accuracy", "Sensitivity", "Specificity",
                                         "PPV", "NPV", "MCC", "F1"))

ggplot(data = all.long %>%
         filter(Assessment %in% c("Sensitivity", "Specificity", "PPV", "NPV")),
       mapping = aes(y = Estimate,
                     x = Model,
                     colour = Assessment,
                     shape = Assessment
                     )
       ) +
  geom_point(size = 3, stroke = 1.5,
             position = position_dodge(width = 0.75)) +
  scale_shape_manual(values = c(15, 17, 18, 25)) +
  scale_color_manual(values = cbpb[c(2, 4, 7, 6)]) +
  # scale_fill_manual(values = cbp.b[c(1, 2, 4, 7, 3 )]) +
  # facet_wrap(~Assessment) +
  scale_x_discrete(name = NULL) +
  scale_y_continuous(limits = c(0.5, 1),
                     breaks = seq(0.5, 1, 0.05)) +
  theme(plot.title = element_text(hjust = 0.5),
        text = element_text(size = 14),
```

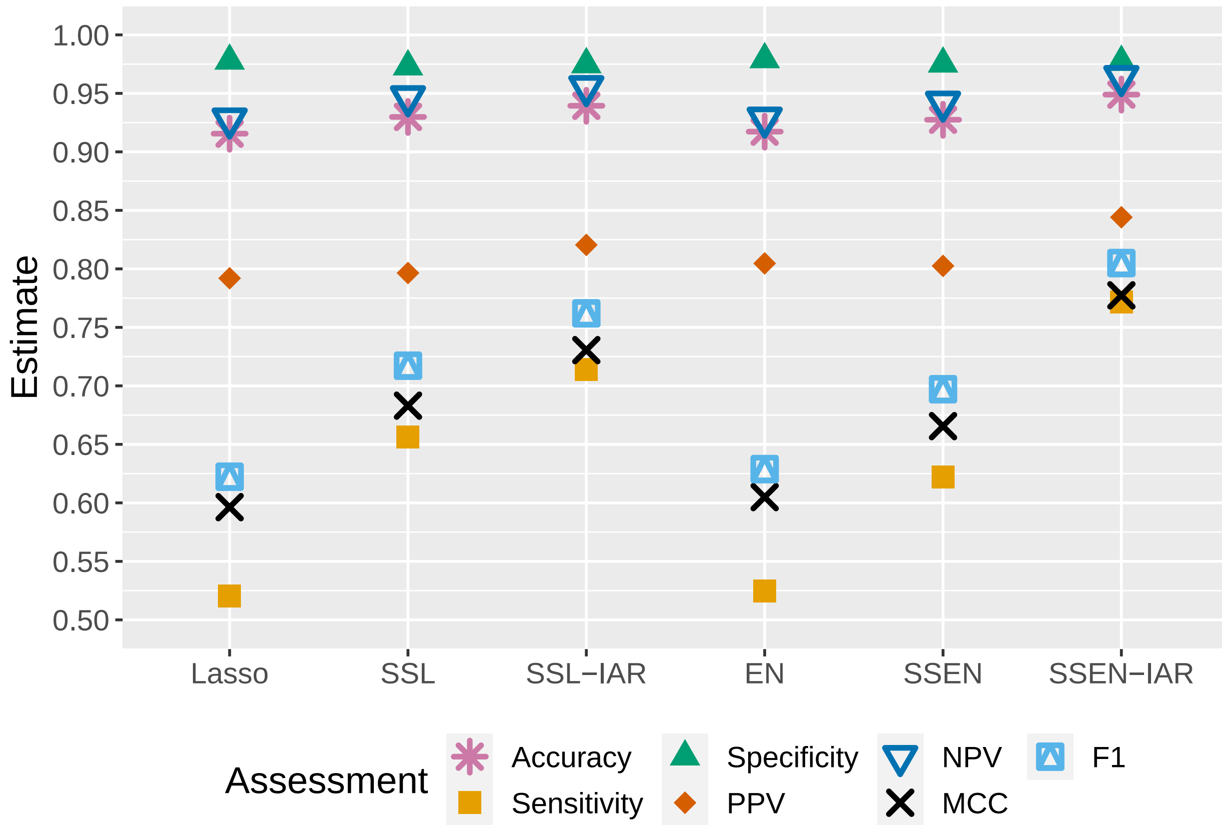
```
       legend.position = "bottom")
```




```
ggplot(data = all.long %>%
         filter(Assessment %in% c("Accuracy", "MCC", "F1")),
       mapping = aes(y = Estimate,
                     x = Model,
                     colour = Assessment,
                     shape = Assessment
                     )
       ) +
  geom_point(size = 3, stroke = 1.5,
             position = position_dodge(width = 0.75)) +
  scale_shape_manual(values = c(8, 4, 14)) +
  scale_color_manual(values = cbpb[c(1, 8, 3)]) +
  # scale_fill_manual(values = cbp.b[c(1, 2, 4, 7, 3 )]) +
  # facet_wrap(~Assessment) +
  scale_x_discrete(name = NULL) +
  scale_y_continuous(limits = c(0.5, 1),
                     breaks = seq(0.5, 1, 0.05)) +
  theme(plot.title = element_text(hjust = 0.5),
        text = element_text(size = 14),
        legend.position = "bottom")
```




```
ggplot(data = all.long,
       mapping = aes(y = Estimate,
                     x = Model,
                     colour = Assessment,
                     shape = Assessment
                     )
       ) +
  geom_point(size = 3, stroke = 1.5
             #,
             #position = position_dodge(width = 0.75)
             ) +
  scale_shape_manual(values = c(8, 15, 17, 18, 25, 4, 14)) +
  scale_color_manual(values = cbpb[c(8, 2, 4, 7, 6, 1, 3)]) +
  # scale_fill_manual(values = cbp.b[c(1, 2, 4, 7, 3 )]) +
  # facet_wrap(~Assessment) +
  scale_x_discrete(name = NULL) +
  scale_y_continuous(limits = c(0.5, 1),
                     breaks = seq(0.5, 1, 0.05)) +
  theme(plot.title = element_text(hjust = 0.5),
        text = element_text(size = 14),
        legend.position = "bottom")
```



5 Results ($\beta_j = 0.05$)

5.1 Loading results

Note that while `glmnet` has only one output file, we have to extract and combine results from the `ssnet` models. For now, we extract only inference, not parameter estimates.

```
# load data

# only take 1st 2500 analyses from glmnet models
glmnet.results.smallB <- readRDS(file = "C:/Users/cotto/Documents/Publications/paper3/Simulation R Code/

ssnet.results.smallB <- dplyr::bind_rows(
  readRDS(file = "C:/Users/cotto/Documents/Publications/paper3/Simulation R Code/results/results_ssnet_
  readRDS(file = "C:/Users/cotto/Documents/Publications/paper3/Simulation R Code/results/results_ssnet_
  readRDS(file = "C:/Users/cotto/Documents/Publications/paper3/Simulation R Code/results/results_ssnet_
  readRDS(file = "C:/Users/cotto/Documents/Publications/paper3/Simulation R Code/results/results_ssnet_
  readRDS(file = "C:/Users/cotto/Documents/Publications/paper3/Simulation R Code/results/results_ssnet_
)

ssnet.iar.results.smallB <- dplyr::bind_rows(
  readRDS(file = "C:/Users/cotto/Documents/Publications/paper3/Simulation R Code/results/results_ssnet_
  readRDS(file = "C:/Users/cotto/Documents/Publications/paper3/Simulation R Code/results/results_ssnet_
  readRDS(file = "C:/Users/cotto/Documents/Publications/paper3/Simulation R Code/results/results_ssnet_
```

```
  readRDS(file = "C:/Users/cotto/Documents/Publications/paper3/Simulation R Code/results/results_ssnet_
  readRDS(file = "C:/Users/cotto/Documents/Publications/paper3/Simulation R Code/results/results_ssnet_
  readRDS(file = "C:/Users/cotto/Documents/Publications/paper3/Simulation R Code/results/results_ssnet_
  readRDS(file = "C:/Users/cotto/Documents/Publications/paper3/Simulation R Code/results/results_ssnet_
  readRDS(file = "C:/Users/cotto/Documents/Publications/paper3/Simulation R Code/results/results_ssnet_
  readRDS(file = "C:/Users/cotto/Documents/Publications/paper3/Simulation R Code/results/results_ssnet_
  readRDS(file = "C:/Users/cotto/Documents/Publications/paper3/Simulation R Code/results/results_ssnet_
)
```

5.2 Obtaining optimal models

We need to extract means by elastic net penalty (0.5 or 1), s_0, and s_1.

```
glmnet.smry.smallB.0 <- glmnet.results.smallB %>%
  select(-model) %>%
  group_by(alpha) %>%
  nest() %>%
  mutate(
    means = map(.x = data, .f = function(x) map_df(x, mean, na.rm = TRUE)),
    sds = map(.x = data, .f = function(x) map_df(x, sd, na.rm = TRUE))
    )

#glmnet.smry.smallB.0

glmnet.smry.smallB <- glmnet.smry.smallB.0 %>%
  select(-data, -sds) %>%
  unnest(means)

#glmnet.smry.smallB

ssnet.smry.smallB.0 <- ssnet.results.smallB %>%
  select(-model) %>%
  group_by(alpha, s0, s1) %>%
  nest() %>%
  mutate(
    means = map(.x = data, .f = function(x) map_df(x, mean, na.rm = TRUE)),
    sds = map(.x = data, .f = function(x) map_df(x, sd, na.rm = TRUE))
    )

#ssnet.smry.smallB.0

ssnet.smry.smallB <- ssnet.smry.smallB.0 %>%
  select(-data, -sds) %>%
  unnest(means)

#ssnet.smry.smallB

ssnet.iar.smry.smallB.0 <- ssnet.iar.results.smallB %>%
  select(-model) %>%
  group_by(alpha, s0, s1) %>%
  nest() %>%
  mutate(
```

```
    means = map(.x = data, .f = function(x) map_df(x, mean, na.rm = TRUE)),
    sds = map(.x = data, .f = function(x) map_df(x, sd, na.rm = TRUE))
    )

#ssnet.iar.smry.smallB.0

ssnet.iar.smry.smallB <- ssnet.iar.smry.smallB.0 %>%
  select(-data, -sds) %>%
  unnest(means)

#ssnet.iar.smry.smallB
```

Then we need to select the optimal model parameters for each of the 6 modeling approaches:

```
optimal.ssnet.a05.smallB <- ssnet.smry.smallB %>%
  ungroup() %>%
  filter(alpha == 0.5) %>%
  filter(deviance == min(deviance))

optimal.ssnet.iar.a05.smallB <- ssnet.iar.smry.smallB %>%
  ungroup() %>%
  filter(alpha == 0.5) %>%
  filter(deviance == min(deviance))

optimal.ssnet.a1.smallB <- ssnet.smry.smallB %>%
  ungroup() %>%
  filter(alpha == 1) %>%
  filter(deviance == min(deviance))

optimal.ssnet.iar.a1.smallB <- ssnet.iar.smry.smallB %>%
  ungroup() %>%
  filter(alpha == 1) %>%
  filter(deviance == min(deviance))

optimal.glmnet.smallB <- glmnet.smry.smallB %>%
  ungroup()

results.df.smallB <- rbind(optimal.glmnet.smallB %>% filter(alpha == 1),
                           optimal.ssnet.a1.smallB,
                           optimal.ssnet.iar.a1.smallB,
                           optimal.glmnet.smallB %>% filter(alpha == 0.5),
                           optimal.ssnet.a05.smallB,
                           optimal.ssnet.iar.a05.smallB)
```

5.3 Tables

```
knitr::kable(cbind(model = c("Lasso", "SSL", "SSL-IAR",
                             "EN", "SSEN", "SSEN-IAR"),
                   results.df.smallB %>%
                     select(s0, s1, deviance, auc, mse, mae, misclassification)),
```

```r
             caption = "Prediction Error over 2,500 Simulated Data Sets",
             digits = 4)
```

Table 4: Prediction Error over 2,500 Simulated Data Sets

model	s0	s1	deviance	auc	mse	mae	misclassification
Lasso	0.0347	0.0347	142.1107	0.8466	0.0858	0.1720	0.1156
SSL	0.0600	1.0000	129.3288	0.8756	0.0776	0.1540	0.1049
SSL-IAR	0.0900	1.0000	119.9155	0.8976	0.0716	0.1378	0.0973
EN	0.0613	0.0613	140.9080	0.8508	0.0851	0.1718	0.1152
SSEN	0.0500	1.0000	132.6004	0.8678	0.0796	0.1597	0.1073
SSEN-IAR	0.1000	2.0000	102.2387	0.9367	0.0595	0.1037	0.0810

```r
knitr::kable(cbind(model = c("Lasso", "SSL", "SSL-IAR",
                              "EN", "SSEN", "SSEN-IAR"),
                   results.df.smallB %>%
                     select(s0, s1, accuracy, sensitivity, specificity, ppv, npv, mcc, f1)),
             caption = "Results over 2,500 Simulated Data Sets",
             digits = 4)
```

Table 5: Results over 2,500 Simulated Data Sets

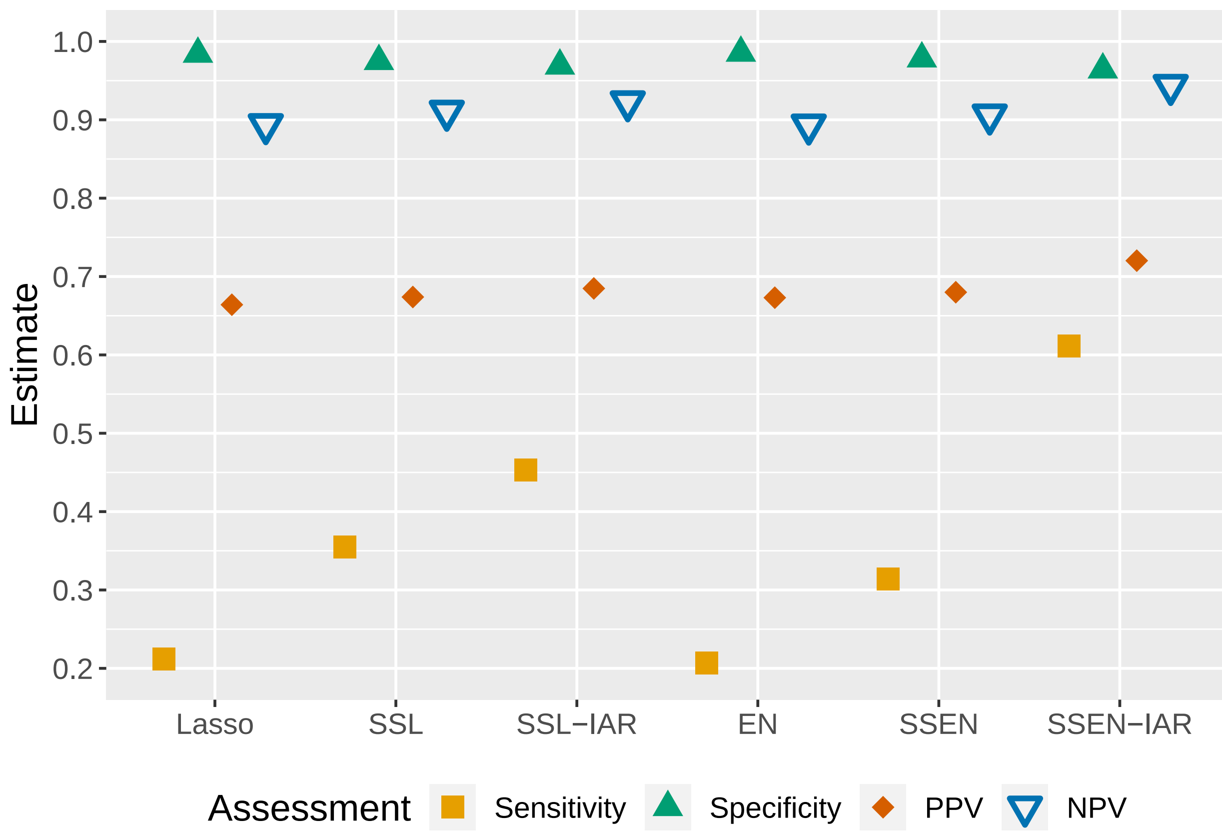
model	s0	s1	accuracy	sensitivity	specificity	ppv	npv	mcc	f1
Lasso	0.0347	0.0347	0.8844	0.2119	0.9841	0.6640	0.8937	0.3242	0.3072
SSL	0.0600	1.0000	0.8951	0.3550	0.9748	0.6739	0.9108	0.4341	0.4555
SSL-IAR	0.0900	1.0000	0.9027	0.4532	0.9690	0.6848	0.9228	0.5045	0.5409
EN	0.0613	0.0613	0.8848	0.2066	0.9854	0.6730	0.8932	0.3254	0.3022
SSEN	0.0500	1.0000	0.8927	0.3140	0.9781	0.6800	0.9060	0.4084	0.4153
SSEN-IAR	0.1000	2.0000	0.9190	0.6114	0.9640	0.7203	0.9437	0.6174	0.6594

```r
#for LaTeX
# xtable::xtable(cbind(model = c("Lasso", "SSL", "SSL-IAR",
#                               "EN", "SSEN", "SSEN-IAR"),
#                    results.df.smallB %>%
#                      select(s0, s1, deviance, auc, mse, mae, misclassification)),
#                digits = 4)
# xtable::xtable(cbind(model = c("Lasso", "SSL", "SSL-IAR",
#                               "EN", "SSEN", "SSEN-IAR"),
#                    results.df.smallB %>%
#                      select(s0, s1, accuracy, sensitivity, specificity, ppv, npv, mcc, f1)),
#                digits = 4)
```

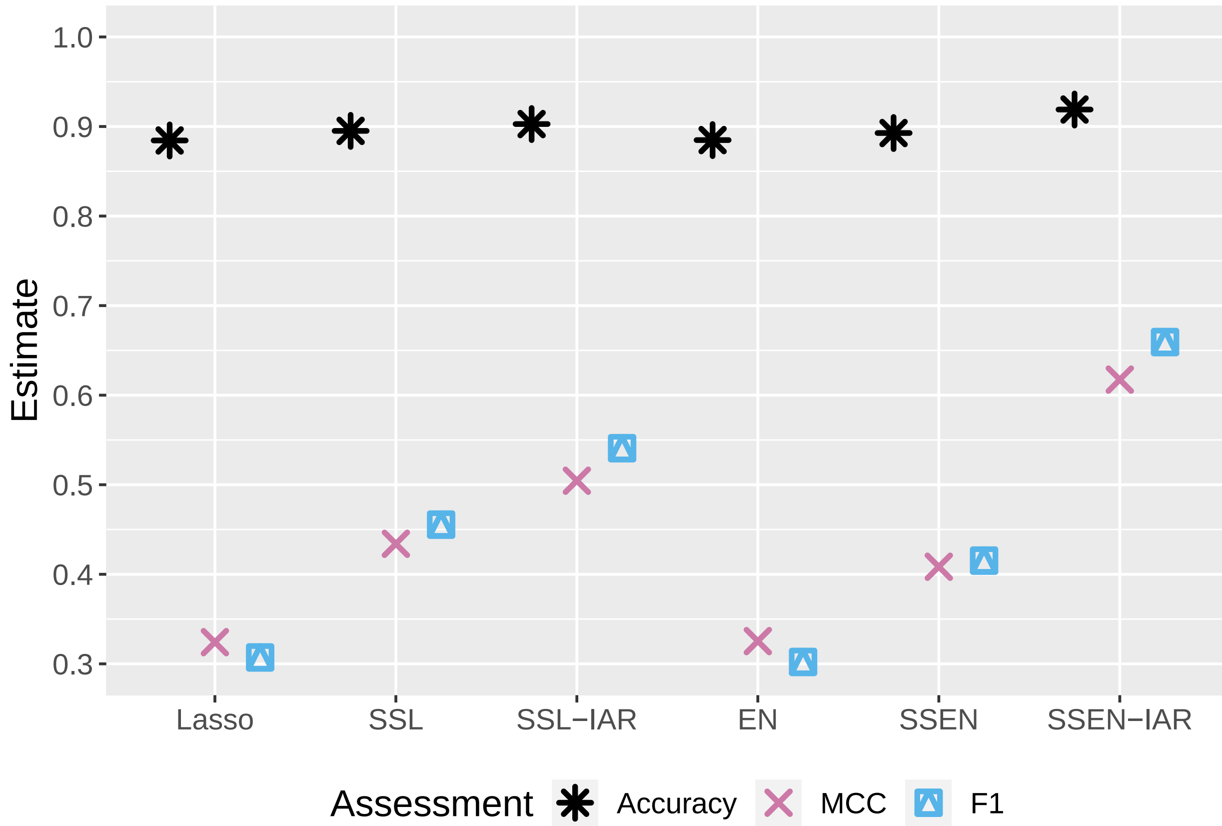
5.4 Figure

```r
# wrangle for figures
accuracy.wide.smallB <- cbind(Model = c("Lasso", "SSL", "SSL-IAR",
                              "EN", "SSEN", "SSEN-IAR"),
```

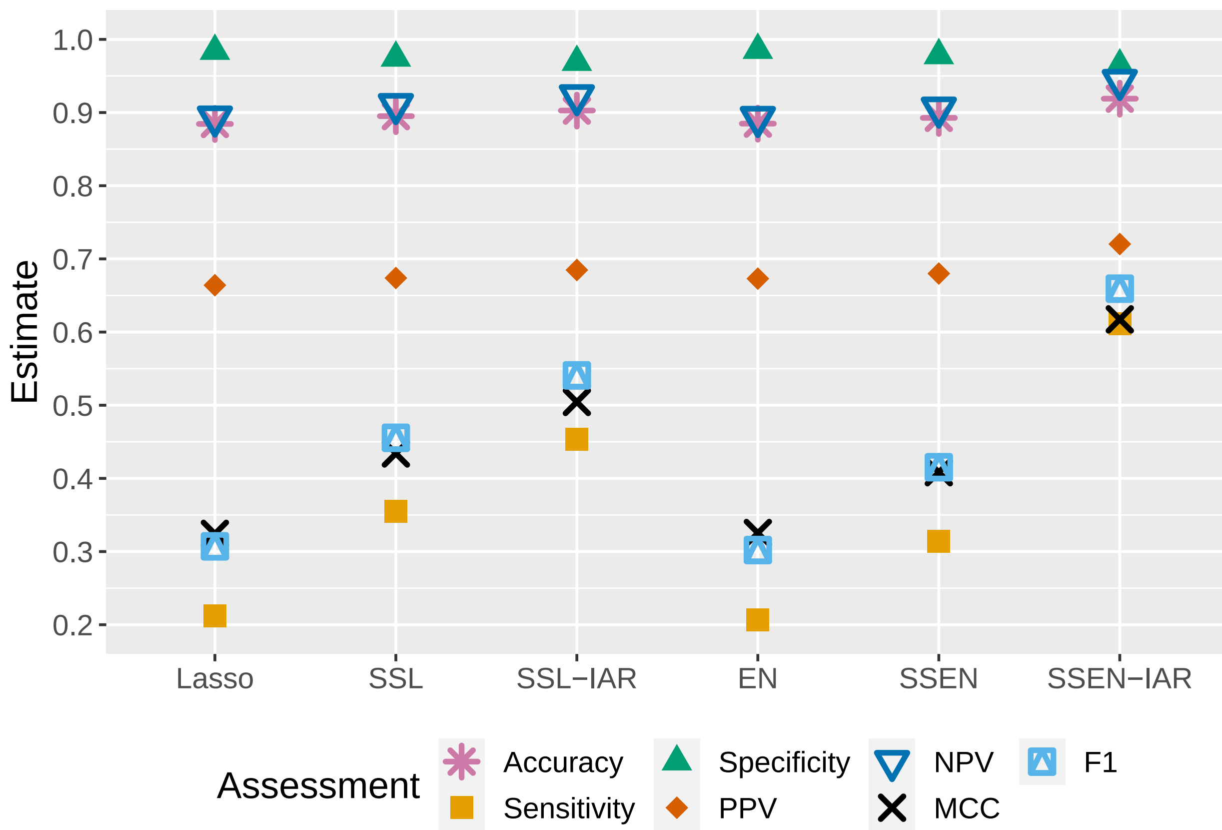
```
                         results.df.smallB %>% select(accuracy, sensitivity, specificity,
                                                      ppv, npv, mcc, f1)
)

# number of models in total
nm <- 6
all.long.smallB <- cbind(accuracy.wide.smallB %>% select(Model),
                  Estimate = c(accuracy.wide.smallB$accuracy,
                               accuracy.wide.smallB$sensitivity,
                               accuracy.wide.smallB$specificity,
                               accuracy.wide.smallB$ppv,
                               accuracy.wide.smallB$npv,
                               accuracy.wide.smallB$mcc,
                               accuracy.wide.smallB$f1),
                  Assessment = c(rep("Accuracy", nm),
                                 rep("Sensitivity", nm),
                                 rep("Specificity", nm),
                                 rep("PPV", nm),
                                 rep("NPV", nm),
                                 rep("MCC", nm),
                                 rep("F1", nm)))
names(all.long.smallB) <- c("Model", "Estimate", "Assessment")
all.long.smallB$Model <- factor(all.long.smallB$Model,
                         levels = c("Lasso", "SSL", "SSL-IAR",
                                    "EN", "SSEN", "SSEN-IAR"))
all.long.smallB$Assessment <- factor(all.long.smallB$Assessment,
                              levels = c("Accuracy", "Sensitivity", "Specificity",
                                         "PPV", "NPV", "MCC", "F1"))

ggplot(data = all.long.smallB %>%
         filter(Assessment %in% c("Sensitivity", "Specificity", "PPV", "NPV")),
       mapping = aes(y = Estimate,
                     x = Model,
                     colour = Assessment,
                     shape = Assessment
                     )
       ) +
  geom_point(size = 3, stroke = 1.5,
             position = position_dodge(width = 0.75)) +
  scale_shape_manual(values = c(15, 17, 18, 25)) +
  scale_color_manual(values = cbpb[c(2, 4, 7, 6)]) +
  # scale_fill_manual(values = cbp.b[c(1, 2, 4, 7, 3 )]) +
  # facet_wrap(~Assessment) +
  scale_x_discrete(name = NULL) +
  scale_y_continuous(limits = c(0.2, 1),
                     breaks = seq(0.2, 1, 0.1)) +
  theme(plot.title = element_text(hjust = 0.5),
        text = element_text(size = 14),
        legend.position = "bottom")
```




```
ggplot(data = all.long.smallB %>%
         filter(Assessment %in% c("Accuracy", "MCC", "F1")),
       mapping = aes(y = Estimate,
                     x = Model,
                     colour = Assessment,
                     shape = Assessment
                     )
       ) +
  geom_point(size = 3, stroke = 1.5,
             position = position_dodge(width = 0.75)) +
  scale_shape_manual(values = c(8, 4, 14)) +
  scale_color_manual(values = cbpb[c(1, 8, 3)]) +
  # scale_fill_manual(values = cbp.b[c(1, 2, 4, 7, 3 )]) +
  # facet_wrap(~Assessment) +
  scale_x_discrete(name = NULL) +
  scale_y_continuous(limits = c(0.3, 1),
                     breaks = seq(0.3, 1, 0.1)) +
  theme(plot.title = element_text(hjust = 0.5),
        text = element_text(size = 14),
        legend.position = "bottom")
```




```
ggplot(data = all.long.smallB,
       mapping = aes(y = Estimate,
                     x = Model,
                     colour = Assessment,
                     shape = Assessment
                     )
       ) +
  geom_point(size = 3, stroke = 1.5
             #,
             #position = position_dodge(width = 0.75)
             ) +
  scale_shape_manual(values = c(8, 15, 17, 18, 25, 4, 14)) +
  scale_color_manual(values = cbpb[c(8, 2, 4, 7, 6, 1, 3)]) +
  # scale_fill_manual(values = cbp.b[c(1, 2, 4, 7, 3 )]) +
  # facet_wrap(~Assessment) +
  scale_x_discrete(name = NULL) +
  scale_y_continuous(limits = c(0.2, 1),
                     breaks = seq(0.2, 1, 0.1)) +
  theme(plot.title = element_text(hjust = 0.5),
        text = element_text(size = 14),
        legend.position = "bottom")
```



6 Variation of metrics

It is good practice to look at the variability in reported metrics over the 2,500 simulations. We produce tables of SD's histograms for all metrics reported in the paper.

6.1 Table of SD for $\beta_j = 0.1$

```
glmnet.sds <- glmnet.smry.0 %>%
  select(-data, -means) %>%
  unnest(sds) %>%
  mutate(model = "glmnet")

ssnet.sds <- ssnet.smry.0 %>%
  select(-data, -means) %>%
  unnest(sds) %>%
  mutate(model = "ssnet")

ssnet.iar.sds <- ssnet.iar.smry.0 %>%
  select(-data, -means) %>%
  unnest(sds) %>%
  mutate(model = "ssnet-iar")
```

```r
sds.01 <- rbind(glmnet.sds,
                ssnet.sds,
                ssnet.iar.sds) %>%
  select(model, alpha, s0, s1, auc, mse, misclassification, accuracy, sensitivity, specificity, mcc, f1)

knitr::kable(sds.01,
             col.names = c("Model", "alpha", "s0", "s1",
                           "AUC", "MSE", "MC", "AC", "SN", "SP", "MCC", "F1"),
             caption = "SD for model fits over 2,500 simulations (beta = 0.10)",
             digits = 4)
```

Table 6: SD for model fits over 2,500 simulations (beta = 0.10)

Model	alpha	s0	s1	AUC	MSE	MC	AC	SN	SP	MCC	F1
glmnet	0.5	0.0131	0.0131	0.0211	0.0112	0.0181	0.0181	0.1129	0.0102	0.0929	0.0983
glmnet	1.0	0.0071	0.0071	0.0230	0.0117	0.0242	0.0189	0.1134	0.0104	0.0972	0.1007
ssnet	0.5	0.0100	1.0000	0.1723	0.0279	0.0298	0.0298	0.2423	0.0143	0.0831	0.2765
ssnet	1.0	0.0100	1.0000	0.1394	0.0244	0.0280	0.0280	0.2177	0.0138	0.0853	0.2371
ssnet	0.5	0.0100	2.0000	0.1945	0.0318	0.0332	0.0332	0.2751	0.0158	0.0839	0.3116
ssnet	1.0	0.0100	2.0000	0.1489	0.0256	0.0288	0.0288	0.2302	0.0144	0.0815	0.2500
ssnet	0.5	0.0200	1.0000	0.0254	0.0152	0.0246	0.0246	0.1627	0.0114	0.1295	0.1682
ssnet	1.0	0.0200	1.0000	0.1202	0.0205	0.0248	0.0248	0.1849	0.0127	0.0838	0.1988
ssnet	0.5	0.0200	2.0000	0.0357	0.0185	0.0296	0.0296	0.2360	0.0144	0.1509	0.2597
ssnet	1.0	0.0200	2.0000	0.1322	0.0222	0.0260	0.0260	0.2028	0.0136	0.0905	0.2198
ssnet	0.5	0.0300	1.0000	0.0218	0.0118	0.0191	0.0191	0.1194	0.0104	0.0936	0.1084
ssnet	1.0	0.0300	1.0000	0.0213	0.0111	0.0183	0.0183	0.1100	0.0099	0.0929	0.0925
ssnet	0.5	0.0300	2.0000	0.0215	0.0121	0.0203	0.0203	0.1432	0.0114	0.1028	0.1405
ssnet	1.0	0.0300	2.0000	0.0216	0.0113	0.0185	0.0185	0.1068	0.0100	0.0905	0.0891
ssnet	0.5	0.0400	1.0000	0.0211	0.0115	0.0185	0.0185	0.1109	0.0097	0.0921	0.0933
ssnet	1.0	0.0400	1.0000	0.0196	0.0110	0.0183	0.0183	0.1058	0.0101	0.0897	0.0877
ssnet	0.5	0.0400	2.0000	0.0210	0.0120	0.0200	0.0200	0.1293	0.0110	0.0978	0.1162
ssnet	1.0	0.0400	2.0000	0.0206	0.0112	0.0185	0.0185	0.1071	0.0098	0.0901	0.0880
ssnet	0.5	0.0500	1.0000	0.0201	0.0114	0.0184	0.0184	0.1083	0.0096	0.0903	0.0908
ssnet	1.0	0.0500	1.0000	0.0179	0.0105	0.0174	0.0174	0.0966	0.0099	0.0816	0.0792
ssnet	0.5	0.0500	2.0000	0.0201	0.0117	0.0192	0.0192	0.1203	0.0105	0.0955	0.1006
ssnet	1.0	0.0500	2.0000	0.0195	0.0110	0.0181	0.0181	0.1005	0.0100	0.0856	0.0827
ssnet	0.5	0.0600	1.0000	0.0190	0.0113	0.0182	0.0182	0.1075	0.0095	0.0899	0.0892
ssnet	1.0	0.0600	1.0000	0.0176	0.0109	0.0178	0.0178	0.0957	0.0101	0.0818	0.0785
ssnet	0.5	0.0600	2.0000	0.0197	0.0115	0.0184	0.0184	0.1157	0.0103	0.0917	0.0944
ssnet	1.0	0.0600	2.0000	0.0191	0.0110	0.0179	0.0179	0.0988	0.0100	0.0843	0.0809
ssnet	0.5	0.0700	1.0000	0.0191	0.0116	0.0185	0.0185	0.1094	0.0099	0.0907	0.0902
ssnet	1.0	0.0700	1.0000	0.0175	0.0111	0.0183	0.0183	0.0972	0.0103	0.0834	0.0794
ssnet	0.5	0.0700	2.0000	0.0200	0.0119	0.0189	0.0189	0.1149	0.0104	0.0923	0.0936
ssnet	1.0	0.0700	2.0000	0.0184	0.0113	0.0183	0.0183	0.1006	0.0102	0.0859	0.0820
ssnet	0.5	0.0800	1.0000	0.0197	0.0118	0.0188	0.0188	0.1092	0.0099	0.0908	0.0902
ssnet	1.0	0.0800	1.0000	0.0180	0.0114	0.0183	0.0183	0.0955	0.0104	0.0822	0.0778
ssnet	0.5	0.0800	2.0000	0.0199	0.0120	0.0192	0.0192	0.1143	0.0106	0.0927	0.0933
ssnet	1.0	0.0800	2.0000	0.0183	0.0116	0.0184	0.0184	0.0971	0.0102	0.0833	0.0787
ssnet	0.5	0.0900	1.0000	0.0206	0.0122	0.0197	0.0197	0.1098	0.0106	0.0924	0.0908
ssnet	1.0	0.0900	1.0000	0.0191	0.0118	0.0188	0.0188	0.0999	0.0103	0.0858	0.0815
ssnet	0.5	0.0900	2.0000	0.0203	0.0122	0.0195	0.0195	0.1177	0.0107	0.0953	0.0962
ssnet	1.0	0.0900	2.0000	0.0195	0.0121	0.0192	0.0192	0.0978	0.0104	0.0847	0.0795

Model	alpha	s0	s1	AUC	MSE	MC	AC	SN	SP	MCC	F1
ssnet	0.5	0.1000	1.0000	0.0210	0.0125	0.0199	0.0199	0.1092	0.0106	0.0925	0.0905
ssnet	1.0	0.1000	1.0000	0.0202	0.0124	0.0196	0.0196	0.1037	0.0106	0.0899	0.0855
ssnet	0.5	0.1000	2.0000	0.0208	0.0124	0.0196	0.0196	0.1165	0.0109	0.0935	0.0936
ssnet	1.0	0.1000	2.0000	0.0204	0.0125	0.0195	0.0195	0.1000	0.0104	0.0866	0.0814
ssnet-iar	0.5	0.0100	1.0000	0.0900	0.0158	0.0212	0.0212	0.1452	0.0115	0.0824	0.1543
ssnet-iar	1.0	0.0100	1.0000	0.1382	0.0243	0.0279	0.0279	0.2166	0.0137	0.0854	0.2358
ssnet-iar	0.5	0.0100	2.0000	0.1083	0.0175	0.0223	0.0223	0.1646	0.0125	0.0814	0.1773
ssnet-iar	1.0	0.0100	2.0000	0.1484	0.0255	0.0287	0.0287	0.2292	0.0144	0.0814	0.2488
ssnet-iar	0.5	0.0200	1.0000	0.0201	0.0113	0.0189	0.0189	0.1016	0.0101	0.0867	0.0862
ssnet-iar	1.0	0.0200	1.0000	0.0909	0.0170	0.0225	0.0225	0.1499	0.0116	0.0838	0.1556
ssnet-iar	0.5	0.0200	2.0000	0.0200	0.0113	0.0189	0.0189	0.0977	0.0101	0.0838	0.0823
ssnet-iar	1.0	0.0200	2.0000	0.1004	0.0177	0.0230	0.0230	0.1600	0.0123	0.0823	0.1683
ssnet-iar	0.5	0.0300	1.0000	0.0185	0.0104	0.0174	0.0174	0.1040	0.0102	0.0870	0.0864
ssnet-iar	1.0	0.0300	1.0000	0.0210	0.0111	0.0183	0.0183	0.1084	0.0100	0.0913	0.0906
ssnet-iar	0.5	0.0300	2.0000	0.0195	0.0107	0.0178	0.0178	0.1023	0.0100	0.0875	0.0856
ssnet-iar	1.0	0.0300	2.0000	0.0214	0.0112	0.0185	0.0185	0.1079	0.0102	0.0910	0.0897
ssnet-iar	0.5	0.0400	1.0000	0.0154	0.0097	0.0163	0.0163	0.0917	0.0097	0.0758	0.0730
ssnet-iar	1.0	0.0400	1.0000	0.0188	0.0105	0.0176	0.0176	0.1016	0.0100	0.0854	0.0835
ssnet-iar	0.5	0.0400	2.0000	0.0170	0.0100	0.0160	0.0160	0.0916	0.0096	0.0766	0.0737
ssnet-iar	1.0	0.0400	2.0000	0.0199	0.0110	0.0183	0.0183	0.1048	0.0101	0.0883	0.0859
ssnet-iar	0.5	0.0500	1.0000	0.0158	0.0105	0.0169	0.0169	0.0853	0.0102	0.0760	0.0708
ssnet-iar	1.0	0.0500	1.0000	0.0170	0.0101	0.0168	0.0168	0.0933	0.0099	0.0793	0.0764
ssnet-iar	0.5	0.0500	2.0000	0.0166	0.0107	0.0170	0.0170	0.0864	0.0102	0.0757	0.0705
ssnet-iar	1.0	0.0500	2.0000	0.0176	0.0105	0.0173	0.0173	0.0941	0.0100	0.0803	0.0769
ssnet-iar	0.5	0.0600	1.0000	0.0161	0.0113	0.0172	0.0172	0.0828	0.0103	0.0760	0.0697
ssnet-iar	1.0	0.0600	1.0000	0.0163	0.0104	0.0170	0.0170	0.0919	0.0099	0.0787	0.0750
ssnet-iar	0.5	0.0600	2.0000	0.0174	0.0114	0.0175	0.0175	0.0828	0.0106	0.0764	0.0697
ssnet-iar	1.0	0.0600	2.0000	0.0168	0.0105	0.0171	0.0171	0.0921	0.0101	0.0792	0.0751
ssnet-iar	0.5	0.0700	1.0000	0.0162	0.0118	0.0176	0.0176	0.0810	0.0106	0.0753	0.0682
ssnet-iar	1.0	0.0700	1.0000	0.0157	0.0103	0.0163	0.0163	0.0875	0.0094	0.0740	0.0699
ssnet-iar	0.5	0.0700	2.0000	0.0178	0.0123	0.0178	0.0178	0.0817	0.0109	0.0777	0.0703
ssnet-iar	1.0	0.0700	2.0000	0.0161	0.0102	0.0166	0.0166	0.0893	0.0096	0.0758	0.0716
ssnet-iar	0.5	0.0800	1.0000	0.0161	0.0118	0.0182	0.0182	0.0824	0.0110	0.0783	0.0705
ssnet-iar	1.0	0.0800	1.0000	0.0155	0.0103	0.0164	0.0164	0.0852	0.0099	0.0724	0.0678
ssnet-iar	0.5	0.0800	2.0000	0.0171	0.0126	0.0181	0.0181	0.0786	0.0113	0.0773	0.0691
ssnet-iar	1.0	0.0800	2.0000	0.0161	0.0106	0.0168	0.0168	0.0870	0.0099	0.0751	0.0701
ssnet-iar	0.5	0.0900	1.0000	0.0160	0.0120	0.0184	0.0184	0.0833	0.0115	0.0787	0.0710
ssnet-iar	1.0	0.0900	1.0000	0.0157	0.0109	0.0172	0.0172	0.0836	0.0103	0.0743	0.0685
ssnet-iar	0.5	0.0900	2.0000	0.0166	0.0128	0.0185	0.0185	0.0786	0.0117	0.0786	0.0700
ssnet-iar	1.0	0.0900	2.0000	0.0161	0.0107	0.0166	0.0166	0.0836	0.0101	0.0733	0.0679
ssnet-iar	0.5	0.1000	1.0000	0.0176	0.0130	0.0197	0.0197	0.0861	0.0125	0.0832	0.0746
ssnet-iar	1.0	0.1000	1.0000	0.0162	0.0112	0.0175	0.0175	0.0853	0.0104	0.0774	0.0711
ssnet-iar	0.5	0.1000	2.0000	0.0165	0.0130	0.0188	0.0188	0.0803	0.0121	0.0792	0.0703
ssnet-iar	1.0	0.1000	2.0000	0.0166	0.0112	0.0172	0.0172	0.0853	0.0101	0.0766	0.0706

6.2 Table of SD for $\beta_j = 0.05$

```
glmnet.sds.smallB <- glmnet.smry.smallB.0 %>%
  select(-data, -means) %>%
  unnest(sds) %>%
```

```
  mutate(model = "glmnet")

ssnet.sds.smallB <- ssnet.smry.smallB.0 %>%
  select(-data, -means) %>%
  unnest(sds) %>%
  mutate(model = "ssnet")

ssnet.iar.sds.smallB <- ssnet.iar.smry.smallB.0 %>%
  select(-data, -means) %>%
  unnest(sds) %>%
  mutate(model = "ssnet-iar")

sds.01.smallB <- rbind(
  glmnet.sds.smallB,
  ssnet.sds.smallB,
  ssnet.iar.sds.smallB) %>%
  select(
    model, alpha, s0, s1, auc, mse, misclassification,
    accuracy, sensitivity, specificity, mcc, f1)

knitr::kable(sds.01.smallB,
             col.names = c("Model", "alpha", "s0", "s1",
                           "AUC", "MSE", "MC", "AC", "SN", "SP", "MCC", "F1"),
             caption = "SD for model fits over 2,500 simulations (0.05)",
             digits = 4)
```

Table 7: SD for model fits over 2,500 simulations (0.05)

Model	alpha	s0	s1	AUC	MSE	MC	AC	SN	SP	MCC	F1
glmnet	0.5	0.0161	0.0161	0.0406	0.0131	0.0212	0.0212	0.1182	0.0098	0.1328	0.1487
glmnet	1.0	0.0087	0.0087	0.0418	0.0132	0.0216	0.0216	0.1189	0.0107	0.1327	0.1465
ssnet	0.5	0.0100	1.0000	0.1717	0.0193	0.0219	0.0219	0.1808	0.0143	0.1055	0.2319
ssnet	1.0	0.0100	1.0000	0.1121	0.0168	0.0220	0.0220	0.1542	0.0129	0.1119	0.1824
ssnet	0.5	0.0100	2.0000	0.1782	0.0206	0.0226	0.0226	0.1909	0.0152	0.1001	0.2413
ssnet	1.0	0.0100	2.0000	0.1171	0.0170	0.0221	0.0221	0.1603	0.0135	0.1105	0.1892
ssnet	0.5	0.0200	1.0000	0.1161	0.0150	0.0206	0.0206	0.1537	0.0127	0.1147	0.1915
ssnet	1.0	0.0200	1.0000	0.1262	0.0161	0.0212	0.0212	0.1591	0.0130	0.1131	0.1933
ssnet	0.5	0.0200	2.0000	0.1303	0.0166	0.0220	0.0220	0.1822	0.0149	0.1094	0.2295
ssnet	1.0	0.0200	2.0000	0.1373	0.0170	0.0214	0.0214	0.1679	0.0140	0.1084	0.2043
ssnet	0.5	0.0300	1.0000	0.0500	0.0138	0.0207	0.0207	0.1508	0.0120	0.1409	0.1818
ssnet	1.0	0.0300	1.0000	0.0922	0.0138	0.0203	0.0203	0.1416	0.0119	0.1163	0.1652
ssnet	0.5	0.0300	2.0000	0.0572	0.0141	0.0213	0.0213	0.1859	0.0149	0.1471	0.2301
ssnet	1.0	0.0300	2.0000	0.1145	0.0142	0.0202	0.0202	0.1534	0.0130	0.1137	0.1823
ssnet	0.5	0.0400	1.0000	0.0423	0.0132	0.0208	0.0208	0.1366	0.0111	0.1282	0.1540
ssnet	1.0	0.0400	1.0000	0.0434	0.0129	0.0208	0.0208	0.1271	0.0108	0.1186	0.1340
ssnet	0.5	0.0400	2.0000	0.0440	0.0133	0.0211	0.0211	0.1621	0.0134	0.1351	0.1933
ssnet	1.0	0.0400	2.0000	0.0477	0.0130	0.0210	0.0210	0.1274	0.0108	0.1171	0.1330
ssnet	0.5	0.0500	1.0000	0.0394	0.0135	0.0217	0.0217	0.1361	0.0115	0.1284	0.1496
ssnet	1.0	0.0500	1.0000	0.0361	0.0130	0.0209	0.0209	0.1269	0.0108	0.1187	0.1312
ssnet	0.5	0.0500	2.0000	0.0390	0.0134	0.0216	0.0216	0.1503	0.0128	0.1395	0.1706
ssnet	1.0	0.0500	2.0000	0.0367	0.0130	0.0212	0.0212	0.1275	0.0111	0.1184	0.1305
ssnet	0.5	0.0600	1.0000	0.0388	0.0136	0.0224	0.0224	0.1380	0.0119	0.1333	0.1519

Model	alpha	s0	s1	AUC	MSE	MC	AC	SN	SP	MCC	F1
ssnet	1.0	0.0600	1.0000	0.0357	0.0134	0.0214	0.0214	0.1287	0.0107	0.1213	0.1317
ssnet	0.5	0.0600	2.0000	0.0393	0.0137	0.0221	0.0221	0.1497	0.0129	0.1390	0.1667
ssnet	1.0	0.0600	2.0000	0.0354	0.0133	0.0217	0.0217	0.1266	0.0112	0.1188	0.1285
ssnet	0.5	0.0700	1.0000	0.0394	0.0138	0.0224	0.0224	0.1365	0.0122	0.1319	0.1476
ssnet	1.0	0.0700	1.0000	0.0370	0.0139	0.0224	0.0224	0.1296	0.0111	0.1247	0.1333
ssnet	0.5	0.0700	2.0000	0.0401	0.0137	0.0222	0.0222	0.1426	0.0132	0.1329	0.1555
ssnet	1.0	0.0700	2.0000	0.0368	0.0139	0.0227	0.0227	0.1290	0.0115	0.1235	0.1320
ssnet	0.5	0.0800	1.0000	0.0404	0.0141	0.0229	0.0229	0.1343	0.0127	0.1342	0.1454
ssnet	1.0	0.0800	1.0000	0.0384	0.0143	0.0231	0.0231	0.1304	0.0113	0.1255	0.1341
ssnet	0.5	0.0800	2.0000	0.0413	0.0139	0.0229	0.0229	0.1400	0.0135	0.1354	0.1513
ssnet	1.0	0.0800	2.0000	0.0378	0.0141	0.0230	0.0230	0.1310	0.0114	0.1257	0.1339
ssnet	0.5	0.0900	1.0000	0.0426	0.0145	0.0233	0.0233	0.1313	0.0129	0.1342	0.1423
ssnet	1.0	0.0900	1.0000	0.0380	0.0141	0.0229	0.0229	0.1308	0.0115	0.1269	0.1357
ssnet	0.5	0.0900	2.0000	0.0428	0.0143	0.0229	0.0229	0.1370	0.0133	0.1364	0.1478
ssnet	1.0	0.0900	2.0000	0.0391	0.0140	0.0227	0.0227	0.1296	0.0112	0.1258	0.1340
ssnet	0.5	0.1000	1.0000	0.0445	0.0148	0.0237	0.0237	0.1294	0.0132	0.1340	0.1407
ssnet	1.0	0.1000	1.0000	0.0399	0.0140	0.0225	0.0225	0.1280	0.0113	0.1273	0.1363
ssnet	0.5	0.1000	2.0000	0.0446	0.0147	0.0237	0.0237	0.1355	0.0139	0.1369	0.1467
ssnet	1.0	0.1000	2.0000	0.0406	0.0141	0.0226	0.0226	0.1308	0.0116	0.1255	0.1375
ssnet-iar	0.5	0.0100	1.0000	0.1505	0.0161	0.0199	0.0199	0.1691	0.0136	0.1063	0.2136
ssnet-iar	1.0	0.0100	1.0000	0.1122	0.0168	0.0220	0.0220	0.1542	0.0129	0.1119	0.1824
ssnet-iar	0.5	0.0100	2.0000	0.1666	0.0175	0.0204	0.0204	0.1891	0.0156	0.1024	0.2364
ssnet-iar	1.0	0.0100	2.0000	0.1153	0.0169	0.0220	0.0220	0.1592	0.0134	0.1100	0.1877
ssnet-iar	0.5	0.0200	1.0000	0.0401	0.0127	0.0204	0.0204	0.1183	0.0105	0.1156	0.1285
ssnet-iar	1.0	0.0200	1.0000	0.1131	0.0151	0.0206	0.0206	0.1538	0.0127	0.1130	0.1842
ssnet-iar	0.5	0.0200	2.0000	0.0397	0.0127	0.0208	0.0208	0.1169	0.0107	0.1136	0.1254
ssnet-iar	1.0	0.0200	2.0000	0.1277	0.0156	0.0206	0.0206	0.1632	0.0137	0.1095	0.1961
ssnet-iar	0.5	0.0300	1.0000	0.0339	0.0124	0.0200	0.0200	0.1213	0.0106	0.1149	0.1250
ssnet-iar	1.0	0.0300	1.0000	0.0627	0.0130	0.0204	0.0204	0.1269	0.0110	0.1156	0.1408
ssnet-iar	0.5	0.0300	2.0000	0.0338	0.0124	0.0200	0.0200	0.1170	0.0108	0.1119	0.1207
ssnet-iar	1.0	0.0300	2.0000	0.0854	0.0133	0.0205	0.0205	0.1358	0.0117	0.1124	0.1543
ssnet-iar	0.5	0.0400	1.0000	0.0320	0.0128	0.0207	0.0207	0.1181	0.0114	0.1117	0.1166
ssnet-iar	1.0	0.0400	1.0000	0.0408	0.0127	0.0206	0.0206	0.1254	0.0108	0.1176	0.1310
ssnet-iar	0.5	0.0400	2.0000	0.0348	0.0131	0.0212	0.0212	0.1235	0.0118	0.1171	0.1227
ssnet-iar	1.0	0.0400	2.0000	0.0414	0.0128	0.0206	0.0206	0.1259	0.0109	0.1166	0.1301
ssnet-iar	0.5	0.0500	1.0000	0.0369	0.0152	0.0231	0.0231	0.1167	0.0129	0.1157	0.1139
ssnet-iar	1.0	0.0500	1.0000	0.0328	0.0124	0.0206	0.0206	0.1237	0.0107	0.1165	0.1263
ssnet-iar	0.5	0.0500	2.0000	0.0393	0.0150	0.0233	0.0233	0.1172	0.0133	0.1133	0.1124
ssnet-iar	1.0	0.0500	2.0000	0.0342	0.0127	0.0205	0.0205	0.1245	0.0108	0.1154	0.1250
ssnet-iar	0.5	0.0600	1.0000	0.0369	0.0157	0.0236	0.0236	0.1139	0.0135	0.1129	0.1086
ssnet-iar	1.0	0.0600	1.0000	0.0328	0.0124	0.0201	0.0201	0.1248	0.0103	0.1148	0.1241
ssnet-iar	0.5	0.0600	2.0000	0.0396	0.0165	0.0241	0.0241	0.1198	0.0142	0.1197	0.1152
ssnet-iar	1.0	0.0600	2.0000	0.0332	0.0126	0.0210	0.0210	0.1264	0.0109	0.1177	0.1258
ssnet-iar	0.5	0.0700	1.0000	0.0384	0.0167	0.0247	0.0247	0.1114	0.0154	0.1140	0.1063
ssnet-iar	1.0	0.0700	1.0000	0.0322	0.0124	0.0199	0.0199	0.1222	0.0106	0.1116	0.1196
ssnet-iar	0.5	0.0700	2.0000	0.0402	0.0172	0.0248	0.0248	0.1155	0.0150	0.1168	0.1093
ssnet-iar	1.0	0.0700	2.0000	0.0327	0.0124	0.0202	0.0202	0.1225	0.0109	0.1117	0.1189
ssnet-iar	0.5	0.0800	1.0000	0.0383	0.0170	0.0247	0.0247	0.1073	0.0155	0.1120	0.1030
ssnet-iar	1.0	0.0800	1.0000	0.0344	0.0132	0.0211	0.0211	0.1174	0.0113	0.1093	0.1129
ssnet-iar	0.5	0.0800	2.0000	0.0396	0.0182	0.0254	0.0254	0.1102	0.0159	0.1134	0.1042
ssnet-iar	1.0	0.0800	2.0000	0.0335	0.0131	0.0210	0.0210	0.1194	0.0112	0.1112	0.1154
ssnet-iar	0.5	0.0900	1.0000	0.0363	0.0170	0.0241	0.0241	0.1024	0.0157	0.1060	0.0977

Model	alpha	s0	s1	AUC	MSE	MC	AC	SN	SP	MCC	F1
ssnet-iar	1.0	0.0900	1.0000	0.0368	0.0146	0.0230	0.0230	0.1160	0.0128	0.1135	0.1125
ssnet-iar	0.5	0.0900	2.0000	0.0411	0.0184	0.0252	0.0252	0.1101	0.0156	0.1158	0.1055
ssnet-iar	1.0	0.0900	2.0000	0.0376	0.0142	0.0220	0.0220	0.1207	0.0119	0.1132	0.1149
ssnet-iar	0.5	0.1000	1.0000	0.0379	0.0174	0.0251	0.0251	0.1104	0.0160	0.1149	0.1061
ssnet-iar	1.0	0.1000	1.0000	0.0393	0.0161	0.0239	0.0239	0.1165	0.0134	0.1167	0.1133
ssnet-iar	0.5	0.1000	2.0000	0.0360	0.0183	0.0250	0.0250	0.1068	0.0162	0.1127	0.1023
ssnet-iar	1.0	0.1000	2.0000	0.0397	0.0157	0.0236	0.0236	0.1191	0.0132	0.1154	0.1133

6.3 Figures

There are really too many combinations of parameters to display all plots. We present plots of the distributions of reported model fitness statistics based on the final models chosen for each case.

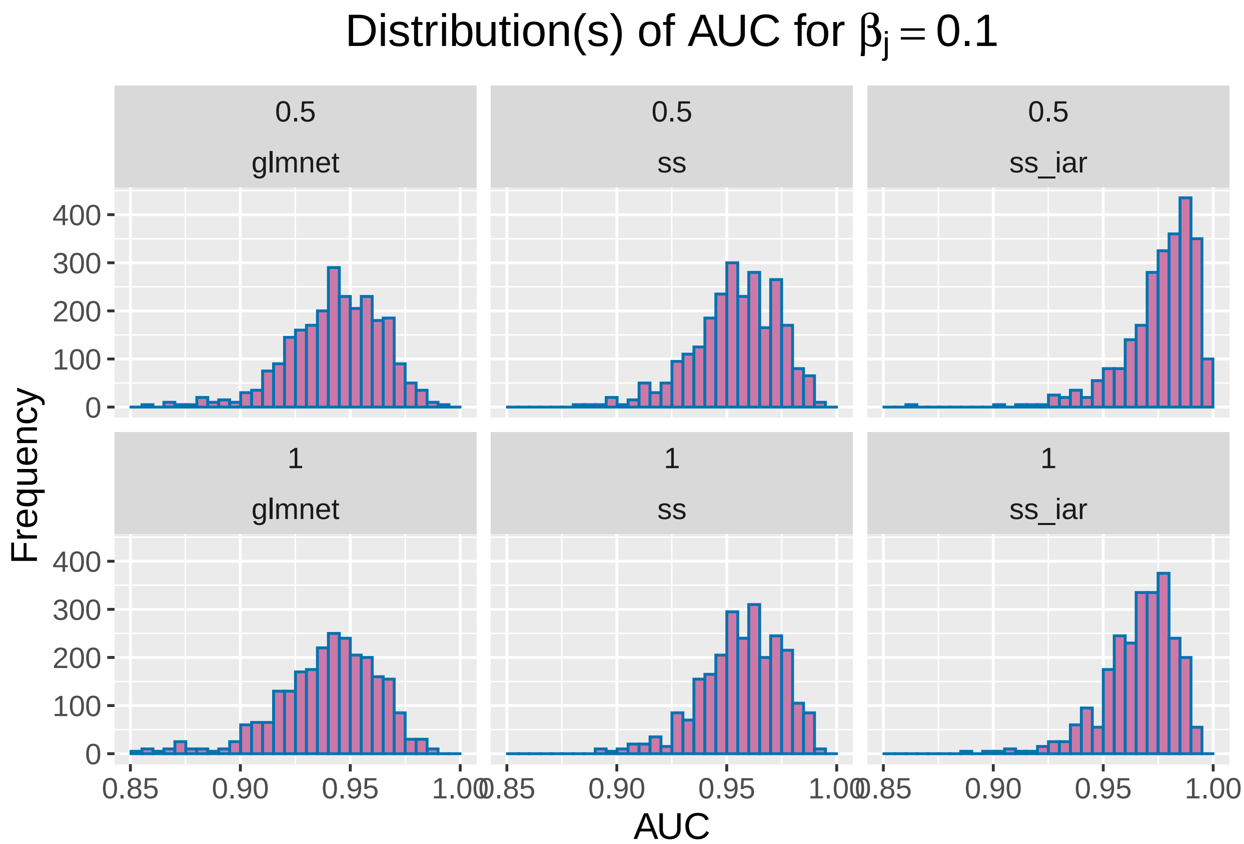
6.3.1 Parameter size 0.1

```
glmnet.results.a1.opt <- glmnet.results %>%
  filter(alpha == 1)
glmnet.results.a05.opt <- glmnet.results %>%
  filter(alpha == 0.5)

ssnet.results$s0 <- round(ssnet.results$s0, digits = 2)
ssnet.results.a1.opt <- ssnet.results %>%
  filter(alpha == 1, s0 == 0.08, s1 == 1)
ssnet.results.a05.opt <- ssnet.results %>%
  filter(alpha == 0.5, s0 == 0.07, s1 == 1)

ssnet.iar.results$s0 <- round(ssnet.iar.results$s0, digits = 2)
ssnet.iar.results.a1.opt <- ssnet.iar.results %>%
  filter(alpha == 1, s0 == 0.10, s1 == 1)
ssnet.iar.results.a05.opt <- ssnet.iar.results %>%
  filter(alpha == 0.5, s0 == 0.10, s1 == 2)

results.opt <- bind_rows(
  glmnet.results.a1.opt,
  ssnet.results.a1.opt,
  ssnet.iar.results.a1.opt,
  glmnet.results.a05.opt,
  ssnet.results.a05.opt,
  ssnet.iar.results.a05.opt
)
```

```
ggplot(data = results.opt,
       aes(x = auc)) +
  facet_wrap(alpha~model) +
  geom_histogram(fill = cbpg[8], color = cbpg[6]) +
  scale_y_continuous(name = "Frequency") +
  scale_x_continuous(name = "AUC") +
  ggtitle(bquote(paste("Distribution(s) of AUC for ", beta[j] == 0.1))) +
  theme(plot.title = element_text(hjust = 0.5),
```

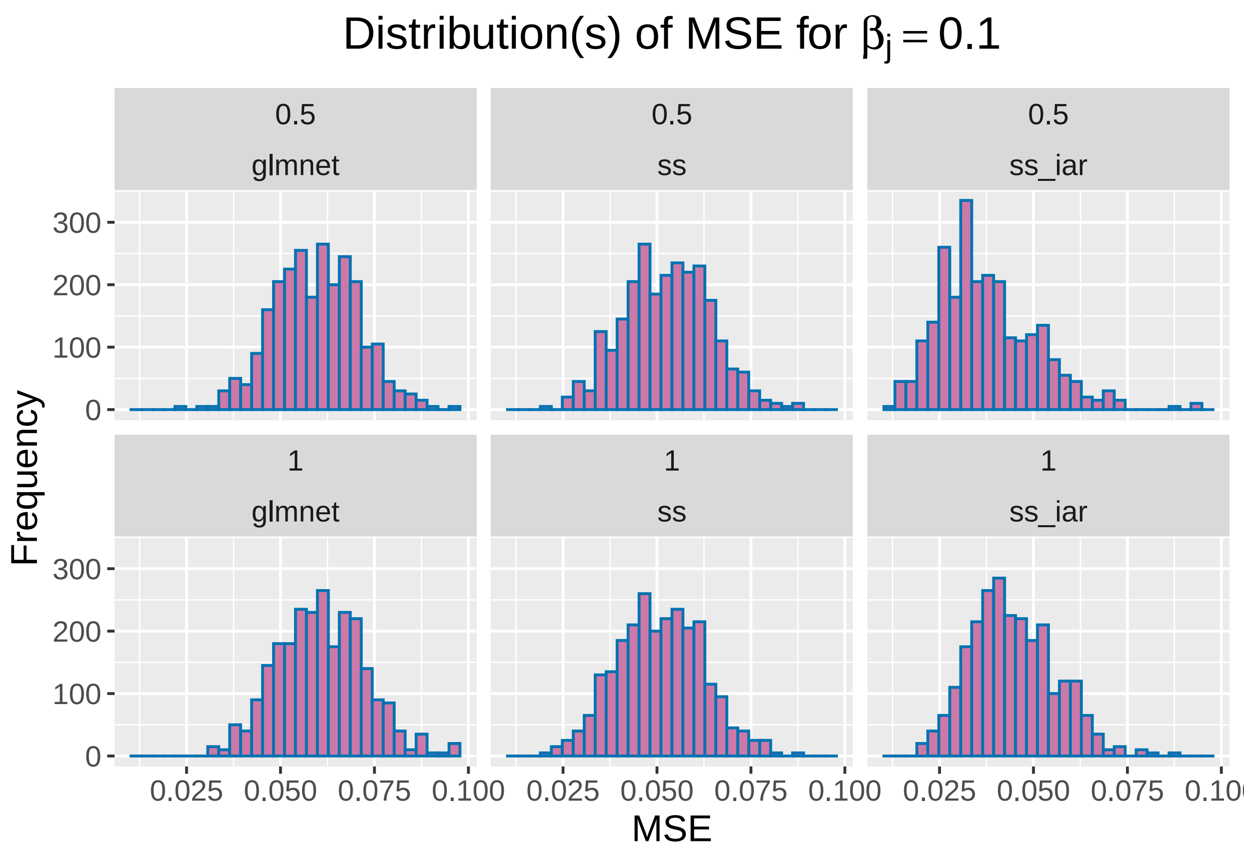
```
        text = element_text(size = 14),
        legend.position = "bottom")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



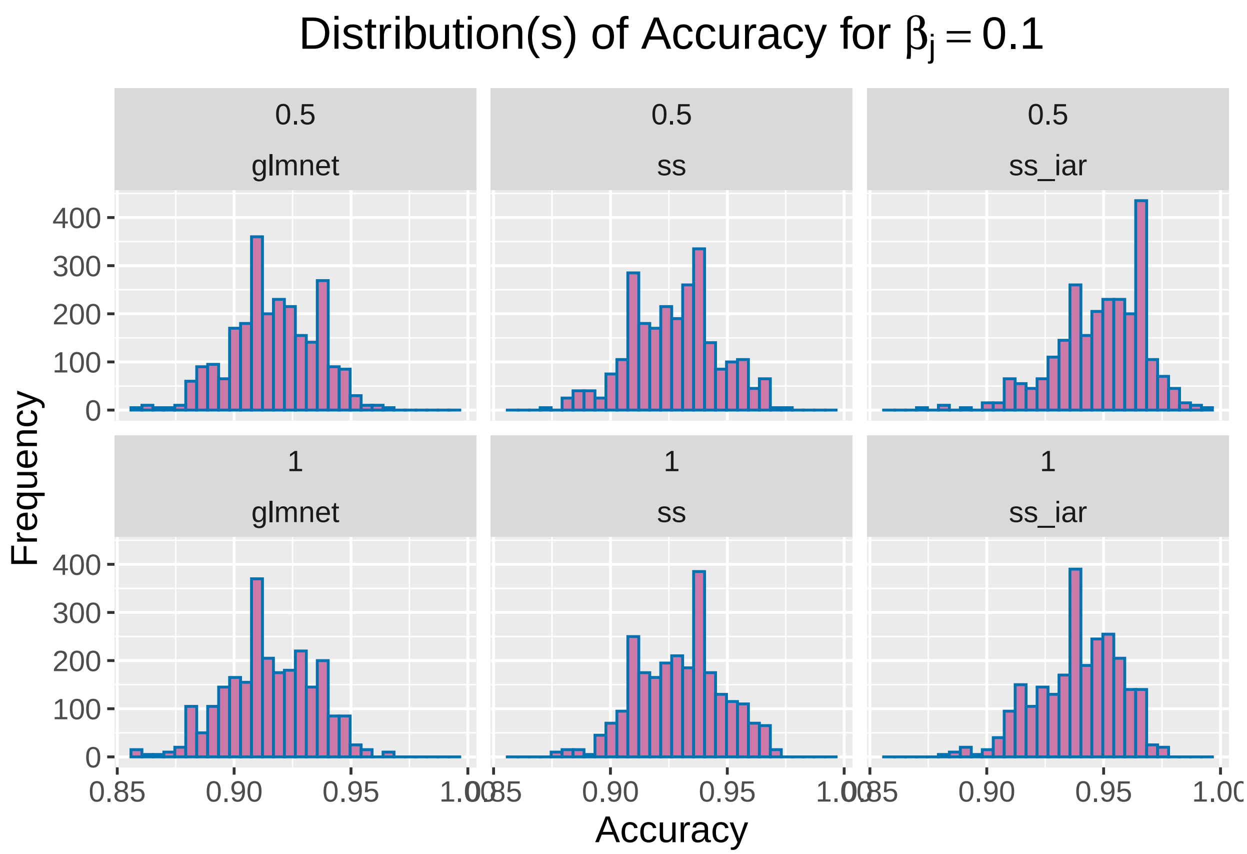

```
ggplot(data = results.opt,
       aes(x = mse)) +
  facet_wrap(alpha~model) +
  geom_histogram(fill = cbpg[8], color = cbpg[6]) +
  scale_y_continuous(name = "Frequency") +
  scale_x_continuous(name = "MSE") +
  ggtitle(bquote(paste("Distribution(s) of MSE for ", beta[j] == 0.1))) +
  theme(plot.title = element_text(hjust = 0.5),
        text = element_text(size = 14),
        legend.position = "bottom")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



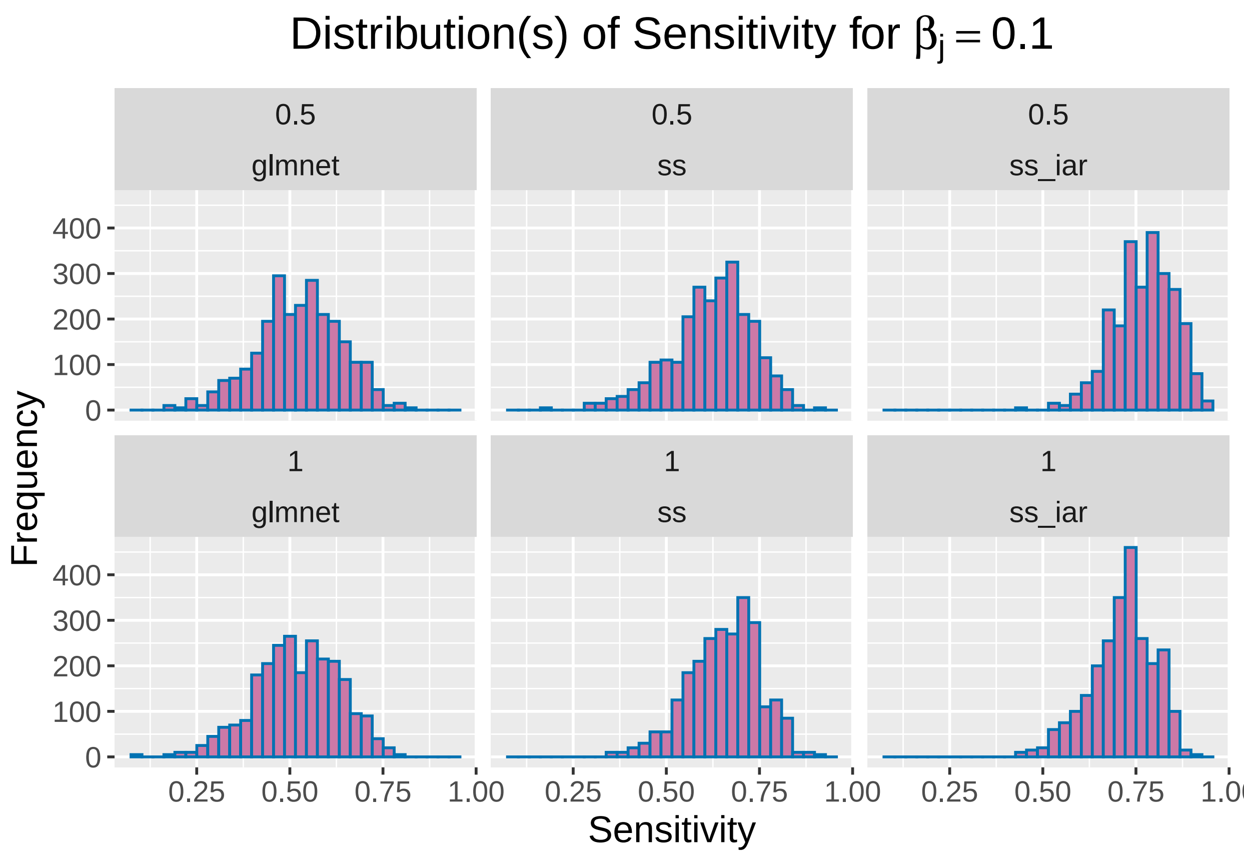

```
ggplot(data = results.opt,
       aes(x = accuracy)) +
  facet_wrap(alpha~model) +
  geom_histogram(fill = cbpg[8], color = cbpg[6]) +
  scale_y_continuous(name = "Frequency") +
  scale_x_continuous(name = "Accuracy") +
  ggtitle(bquote(paste("Distribution(s) of Accuracy for ", beta[j] == 0.1))) +
  theme(plot.title = element_text(hjust = 0.5),
        text = element_text(size = 14),
        legend.position = "bottom")
```

```
## 'stat_bin()' using 'bins = 30'. Pick better value with 'binwidth'.
```



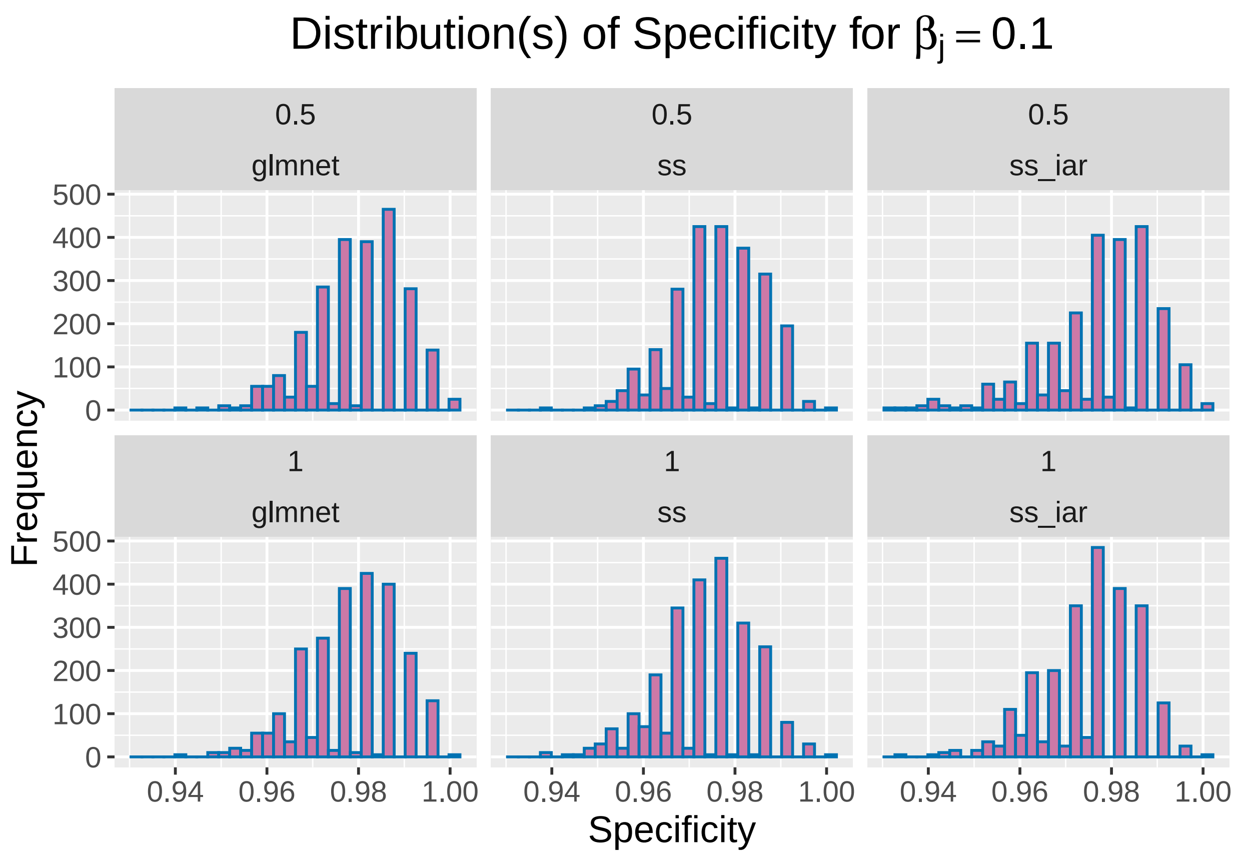

```
ggplot(data = results.opt,
       aes(x = sensitivity)) +
  facet_wrap(alpha~model) +
  geom_histogram(fill = cbpg[8], color = cbpg[6]) +
  scale_y_continuous(name = "Frequency") +
  scale_x_continuous(name = "Sensitivity") +
  ggtitle(bquote(paste("Distribution(s) of Sensitivity for ", beta[j] == 0.1))) +
  theme(plot.title = element_text(hjust = 0.5),
        text = element_text(size = 14),
        legend.position = "bottom")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



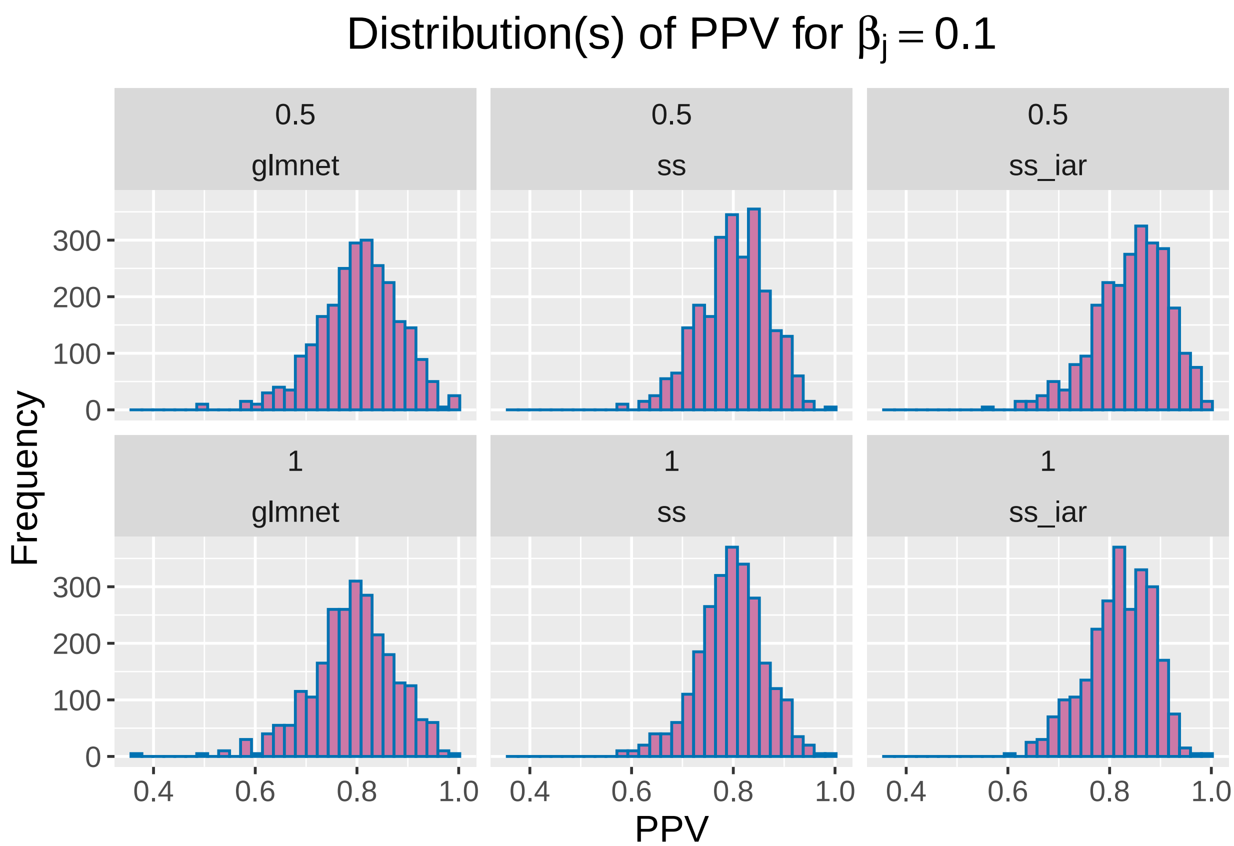

```
ggplot(data = results.opt,
       aes(x = specificity)) +
  facet_wrap(alpha~model) +
  geom_histogram(fill = cbpg[8], color = cbpg[6]) +
  scale_y_continuous(name = "Frequency") +
  scale_x_continuous(name = "Specificity") +
  ggtitle(bquote(paste("Distribution(s) of Specificity for ", beta[j] == 0.1))) +
  theme(plot.title = element_text(hjust = 0.5),
        text = element_text(size = 14),
        legend.position = "bottom")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



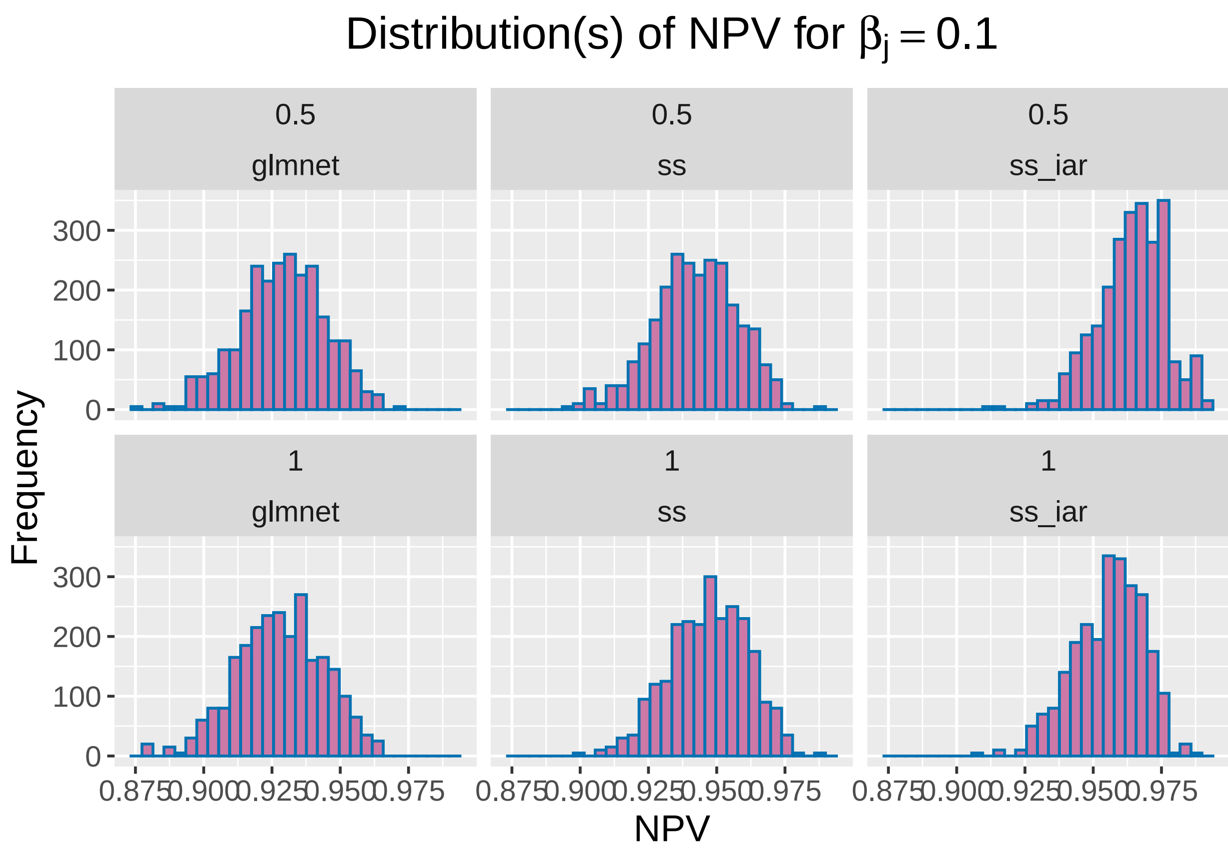

```
ggplot(data = results.opt,
       aes(x = ppv)) +
  facet_wrap(alpha~model) +
  geom_histogram(fill = cbpg[8], color = cbpg[6]) +
  scale_y_continuous(name = "Frequency") +
  scale_x_continuous(name = "PPV") +
  ggtitle(bquote(paste("Distribution(s) of PPV for ", beta[j] == 0.1))) +
  theme(plot.title = element_text(hjust = 0.5),
        text = element_text(size = 14),
        legend.position = "bottom")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



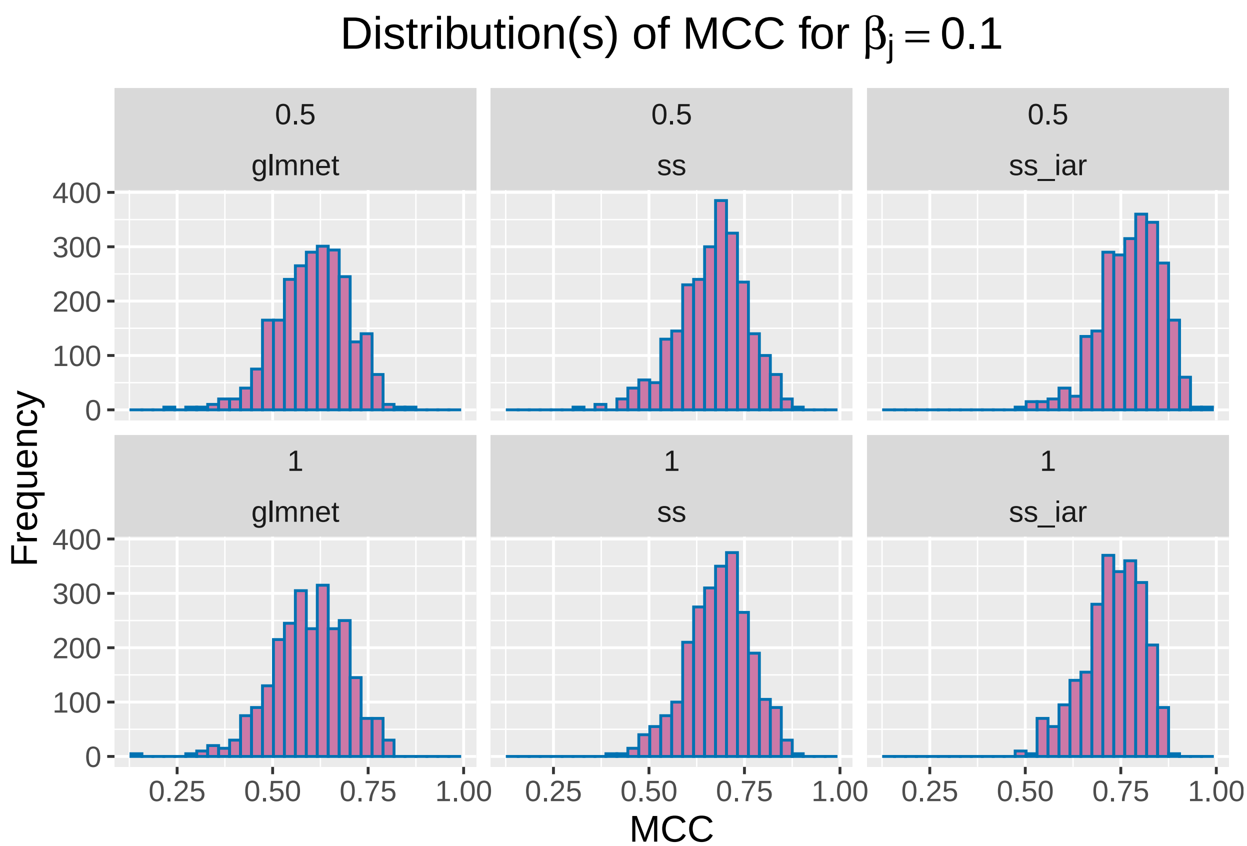

```
ggplot(data = results.opt,
       aes(x = npv)) +
  facet_wrap(alpha~model) +
  geom_histogram(fill = cbpg[8], color = cbpg[6]) +
  scale_y_continuous(name = "Frequency") +
  scale_x_continuous(name = "NPV") +
  ggtitle(bquote(paste("Distribution(s) of NPV for ", beta[j] == 0.1))) +
  theme(plot.title = element_text(hjust = 0.5),
        text = element_text(size = 14),
        legend.position = "bottom")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



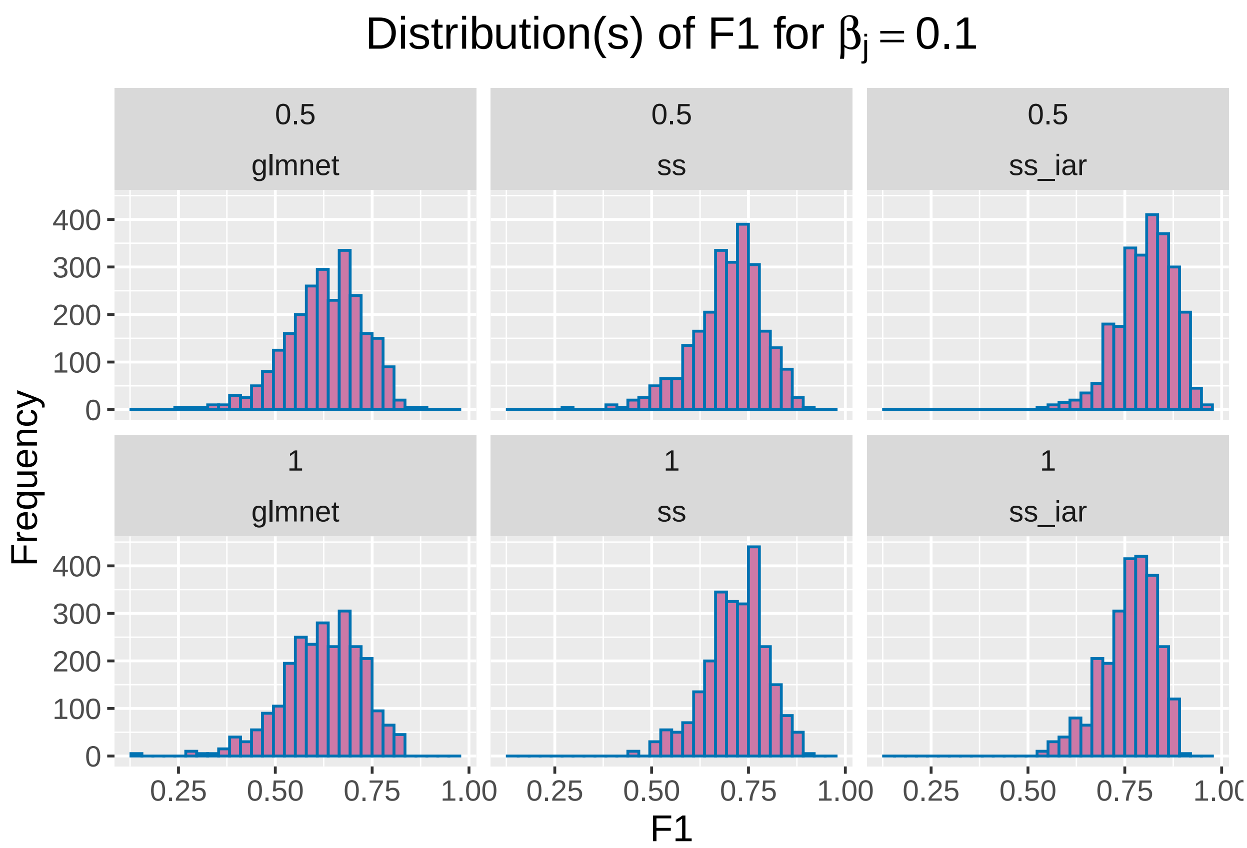

```r
ggplot(data = results.opt,
       aes(x = mcc)) +
  facet_wrap(alpha~model) +
  geom_histogram(fill = cbpg[8], color = cbpg[6]) +
  scale_y_continuous(name = "Frequency") +
  scale_x_continuous(name = "MCC") +
  ggtitle(bquote(paste("Distribution(s) of MCC for ", beta[j] == 0.1))) +
  theme(plot.title = element_text(hjust = 0.5),
        text = element_text(size = 14),
        legend.position = "bottom")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```
ggplot(data = results.opt,
       aes(x = f1)) +
  facet_wrap(alpha~model) +
  geom_histogram(fill = cbpg[8], color = cbpg[6]) +
  scale_y_continuous(name = "Frequency") +
  scale_x_continuous(name = "F1") +
  ggtitle(bquote(paste("Distribution(s) of F1 for ", beta[j] == 0.1))) +
  theme(plot.title = element_text(hjust = 0.5),
        text = element_text(size = 14),
        legend.position = "bottom")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



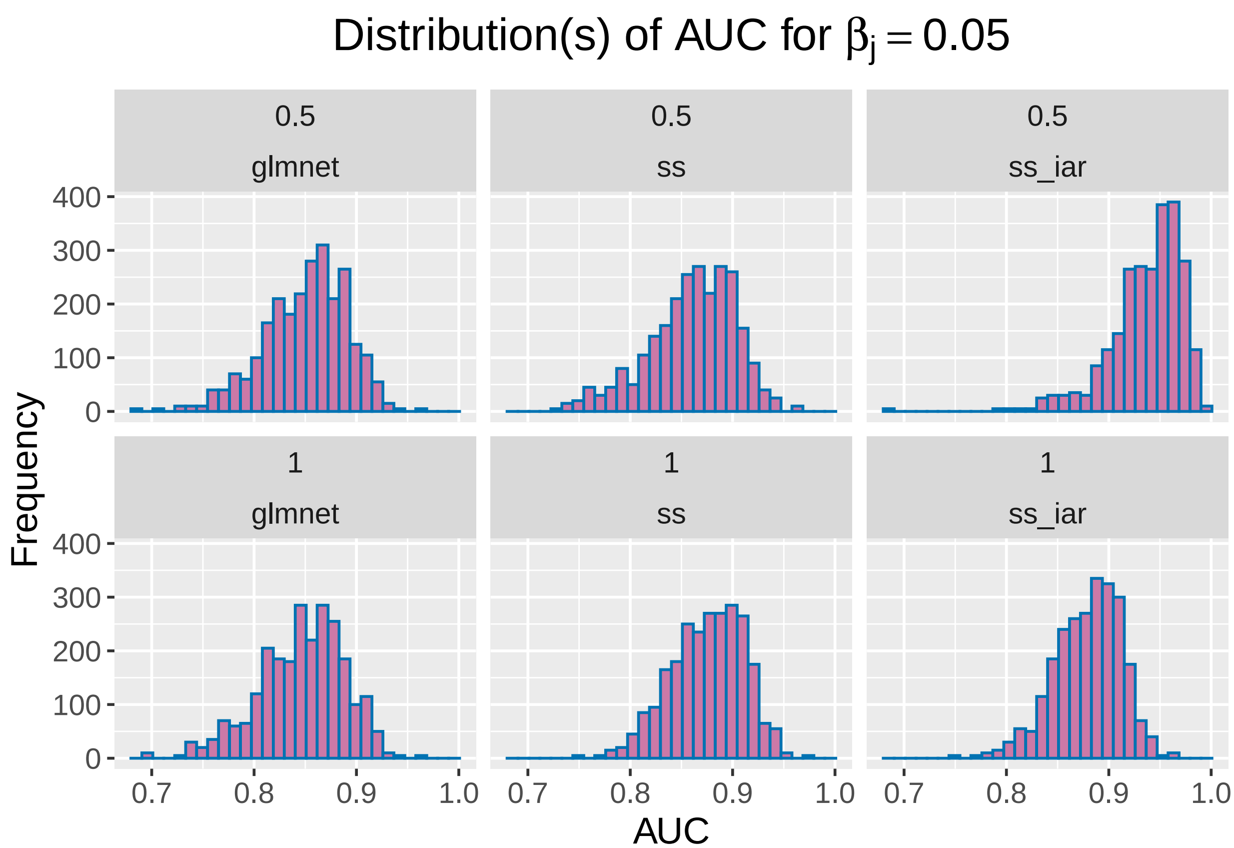
6.3.2 Parameter size 0.05

```
glmnet.results.smallB.a1.opt <- glmnet.results.smallB %>%
  filter(alpha == 1)
glmnet.results.smallB.a05.opt <- glmnet.results.smallB %>%
  filter(alpha == 0.5)

ssnet.results.smallB$s0 <- round(ssnet.results.smallB$s0, digits = 2)
ssnet.results.smallB.a1.opt <- ssnet.results.smallB %>%
  filter(alpha == 1, s0 == 0.06, s1 == 1)
ssnet.results.smallB.a05.opt <- ssnet.results.smallB %>%
  filter(alpha == 0.5, s0 == 0.09, s1 == 1)

ssnet.iar.results.smallB$s0 <- round(ssnet.iar.results.smallB$s0, digits = 2)
ssnet.iar.results.smallB.a1.opt <- ssnet.iar.results.smallB %>%
  filter(alpha == 1, s0 == 0.05, s1 == 1)
ssnet.iar.results.smallB.a05.opt <- ssnet.iar.results.smallB %>%
  filter(alpha == 0.5, s0 == 0.10, s1 == 2)

results.smallB.opt <- bind_rows(
  glmnet.results.smallB.a1.opt,
  ssnet.results.smallB.a1.opt,
  ssnet.iar.results.smallB.a1.opt,
  glmnet.results.smallB.a05.opt,
```

```r
  ssnet.results.smallB.a05.opt,
  ssnet.iar.results.smallB.a05.opt
)
```

```r
ggplot(data = results.smallB.opt,
       aes(x = auc)) +
  facet_wrap(alpha~model) +
  geom_histogram(fill = cbpg[8], color = cbpg[6]) +
  scale_y_continuous(name = "Frequency") +
  scale_x_continuous(name = "AUC") +
  ggtitle(bquote(paste("Distribution(s) of AUC for ", beta[j] == 0.05))) +
  theme(plot.title = element_text(hjust = 0.5),
        text = element_text(size = 14),
        legend.position = "bottom")
```

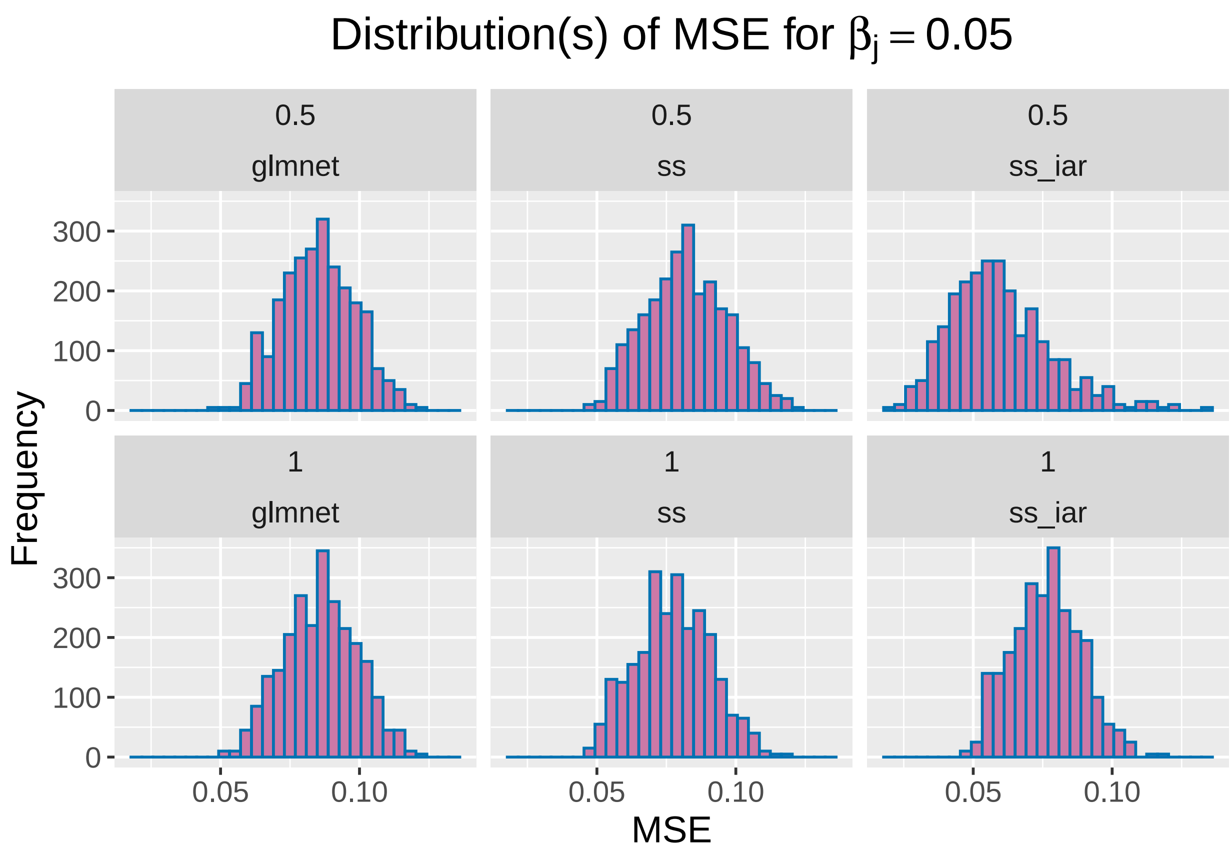
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```r
ggplot(data = results.smallB.opt,
       aes(x = mse)) +
  facet_wrap(alpha~model) +
  geom_histogram(fill = cbpg[8], color = cbpg[6]) +
  scale_y_continuous(name = "Frequency") +
  scale_x_continuous(name = "MSE") +
  ggtitle(bquote(paste("Distribution(s) of MSE for ", beta[j] == 0.05))) +
```

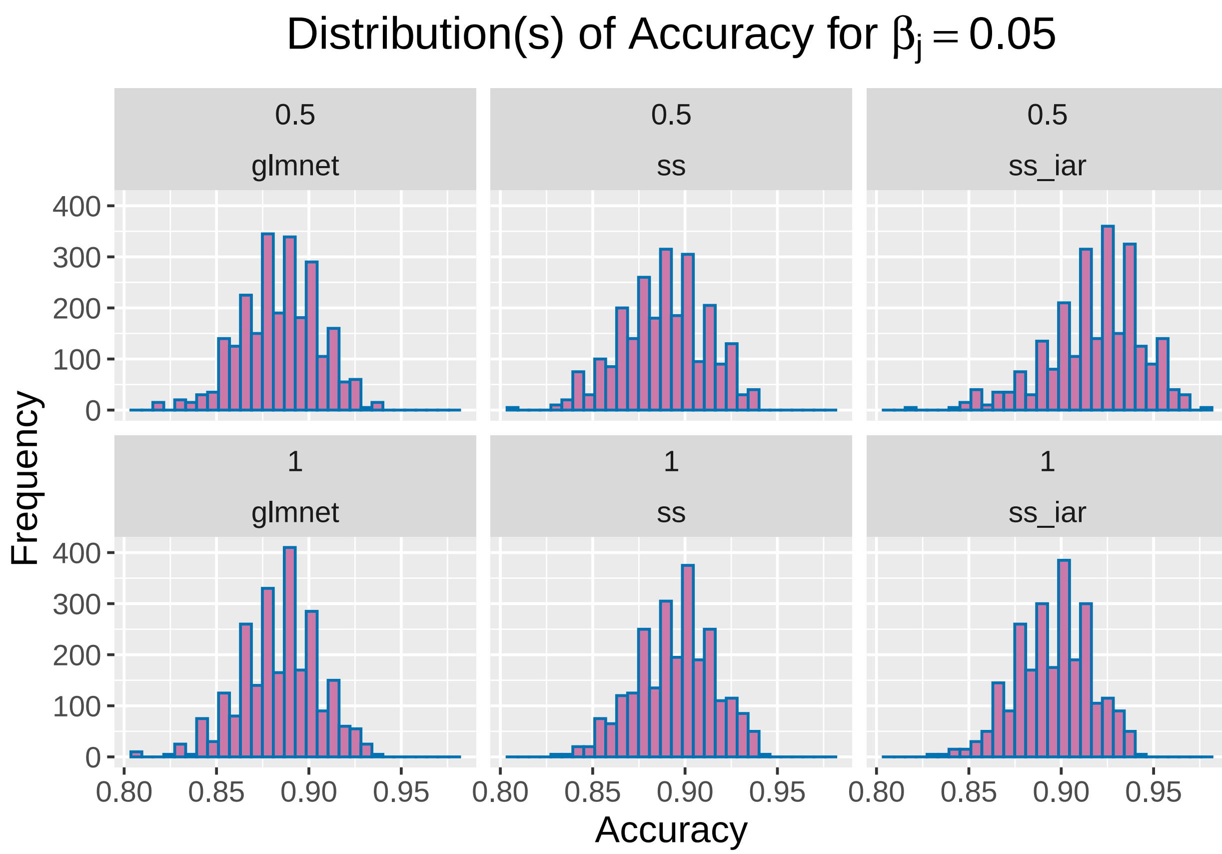
```
  theme(plot.title = element_text(hjust = 0.5),
        text = element_text(size = 14),
        legend.position = "bottom")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



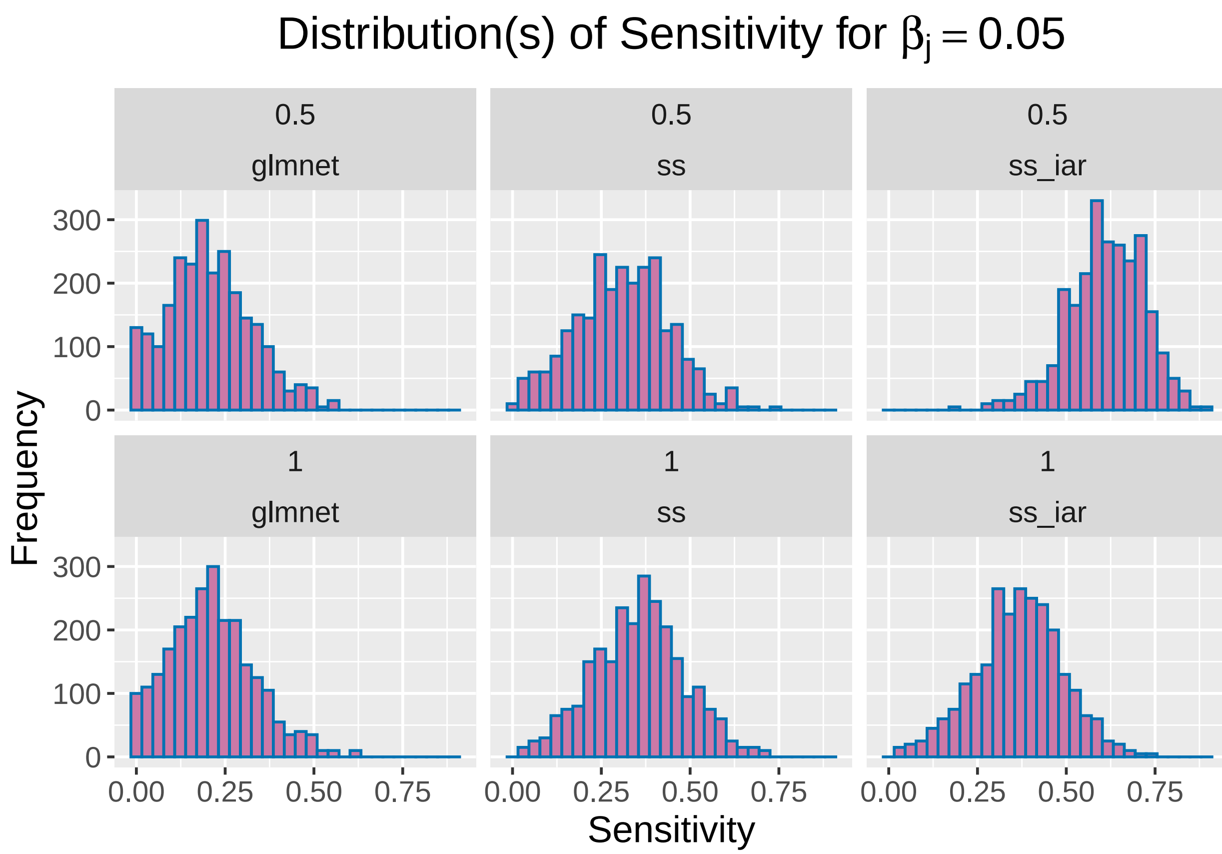

```
ggplot(data = results.smallB.opt,
       aes(x = accuracy)) +
  facet_wrap(alpha~model) +
  geom_histogram(fill = cbpg[8], color = cbpg[6]) +
  scale_y_continuous(name = "Frequency") +
  scale_x_continuous(name = "Accuracy") +
  ggtitle(bquote(paste("Distribution(s) of Accuracy for ", beta[j] == 0.05))) +
  theme(plot.title = element_text(hjust = 0.5),
        text = element_text(size = 14),
        legend.position = "bottom")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



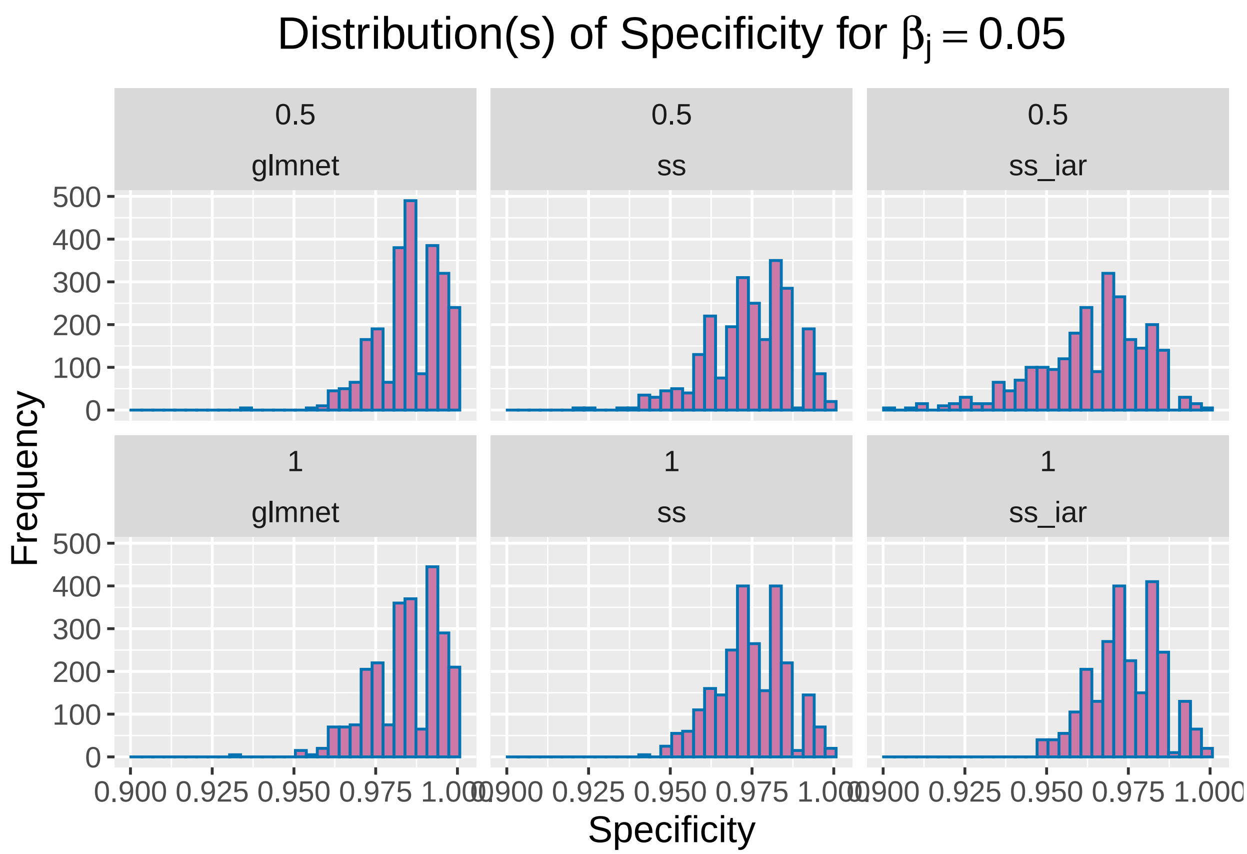

```r
ggplot(data = results.smallB.opt,
       aes(x = sensitivity)) +
  facet_wrap(alpha~model) +
  geom_histogram(fill = cbpg[8], color = cbpg[6]) +
  scale_y_continuous(name = "Frequency") +
  scale_x_continuous(name = "Sensitivity") +
  ggtitle(bquote(paste("Distribution(s) of Sensitivity for ", beta[j] == 0.05))) +
  theme(plot.title = element_text(hjust = 0.5),
        text = element_text(size = 14),
        legend.position = "bottom")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```
ggplot(data = results.smallB.opt,
       aes(x = specificity)) +
  facet_wrap(alpha~model) +
  geom_histogram(fill = cbpg[8], color = cbpg[6]) +
  scale_y_continuous(name = "Frequency") +
  scale_x_continuous(name = "Specificity") +
  ggtitle(bquote(paste("Distribution(s) of Specificity for ", beta[j] == 0.05))) +
  theme(plot.title = element_text(hjust = 0.5),
        text = element_text(size = 14),
        legend.position = "bottom")
```

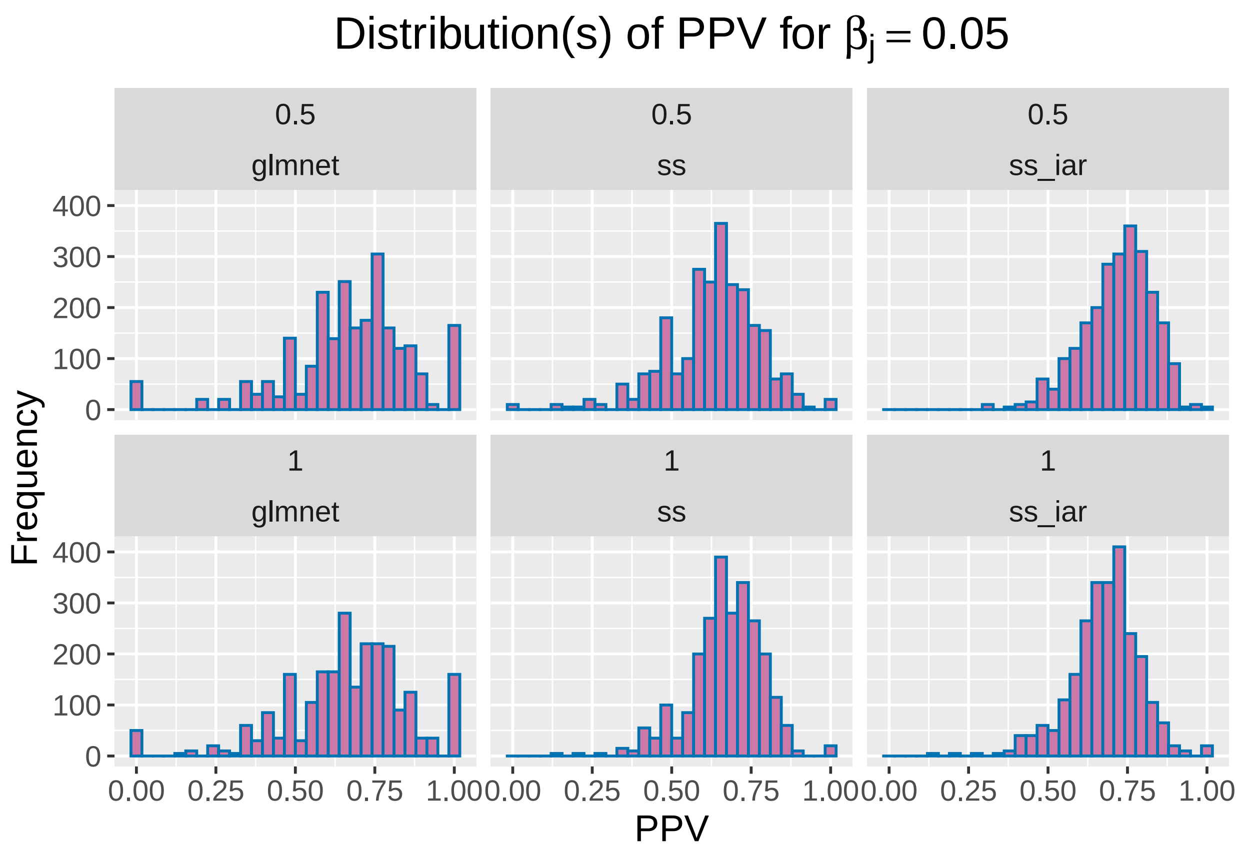
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```
ggplot(data = results.smallB.opt,
       aes(x = ppv)) +
  facet_wrap(alpha~model) +
  geom_histogram(fill = cbpg[8], color = cbpg[6]) +
  scale_y_continuous(name = "Frequency") +
  scale_x_continuous(name = "PPV") +
  ggtitle(bquote(paste("Distribution(s) of PPV for ", beta[j] == 0.05))) +
  theme(plot.title = element_text(hjust = 0.5),
        text = element_text(size = 14),
        legend.position = "bottom")
```

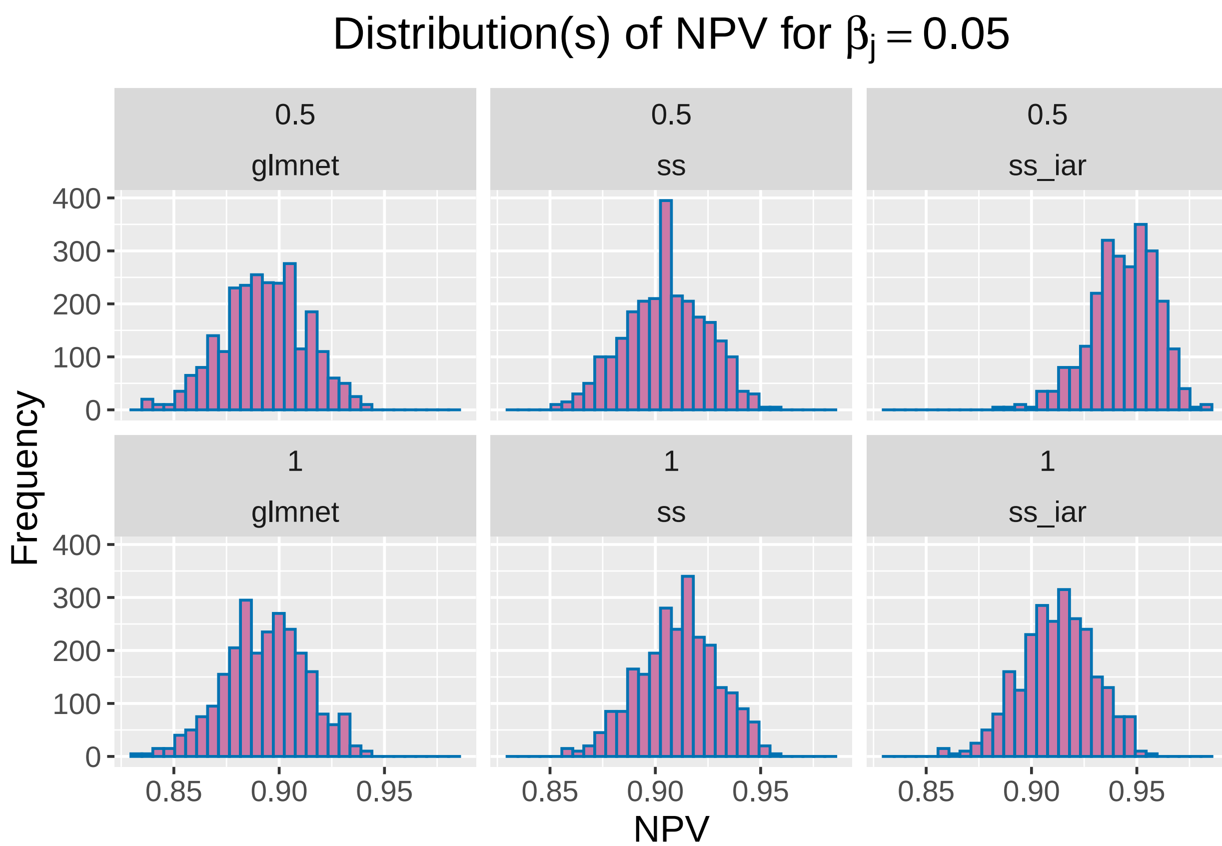
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

```
## Warning: Removed 125 rows containing non-finite values (stat_bin).
```




```
ggplot(data = results.smallB.opt,
       aes(x = npv)) +
  facet_wrap(alpha~model) +
  geom_histogram(fill = cbpg[8], color = cbpg[6]) +
  scale_y_continuous(name = "Frequency") +
  scale_x_continuous(name = "NPV") +
  ggtitle(bquote(paste("Distribution(s) of NPV for ", beta[j] == 0.05))) +
  theme(plot.title = element_text(hjust = 0.5),
        text = element_text(size = 14),
        legend.position = "bottom")
```

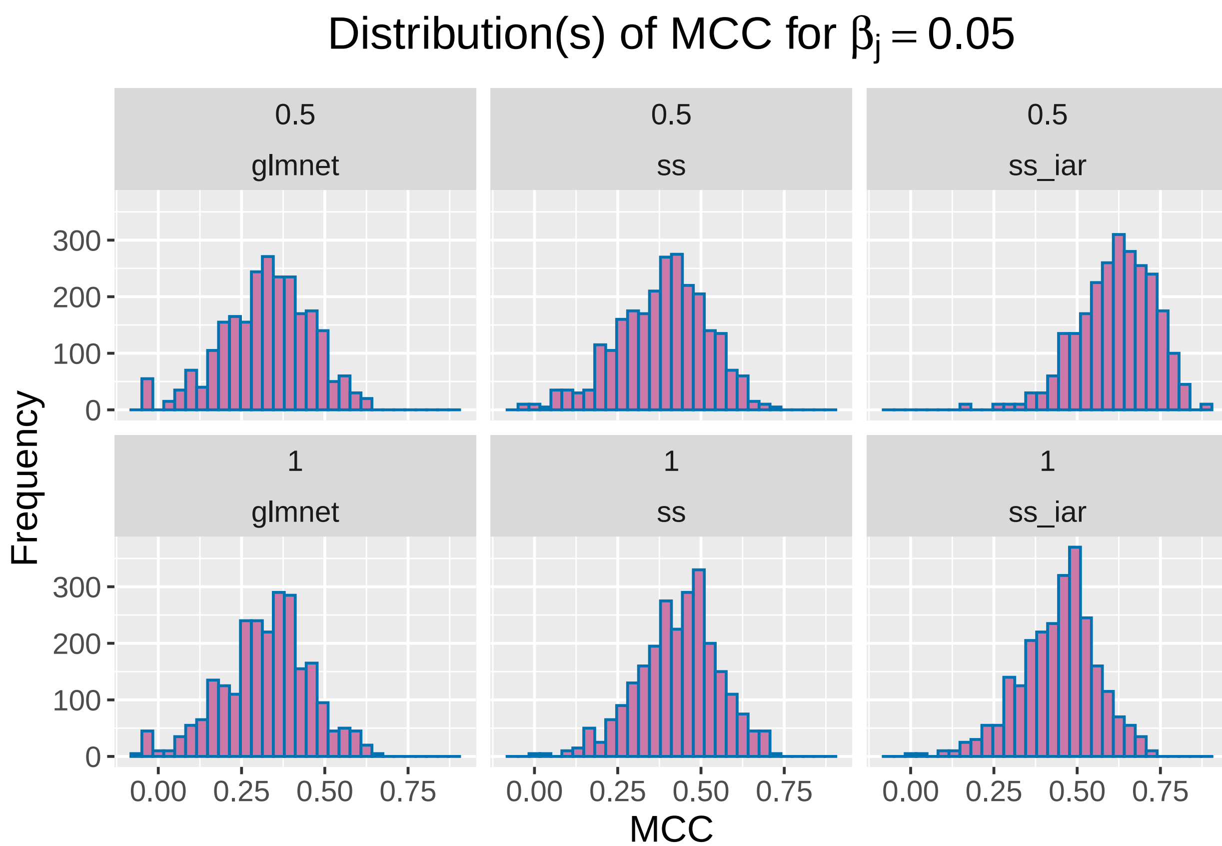
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```
ggplot(data = results.smallB.opt,
       aes(x = mcc)) +
  facet_wrap(alpha~model) +
  geom_histogram(fill = cbpg[8], color = cbpg[6]) +
  scale_y_continuous(name = "Frequency") +
  scale_x_continuous(name = "MCC") +
  ggtitle(bquote(paste("Distribution(s) of MCC for ", beta[j] == 0.05))) +
  theme(plot.title = element_text(hjust = 0.5),
        text = element_text(size = 14),
        legend.position = "bottom")
```

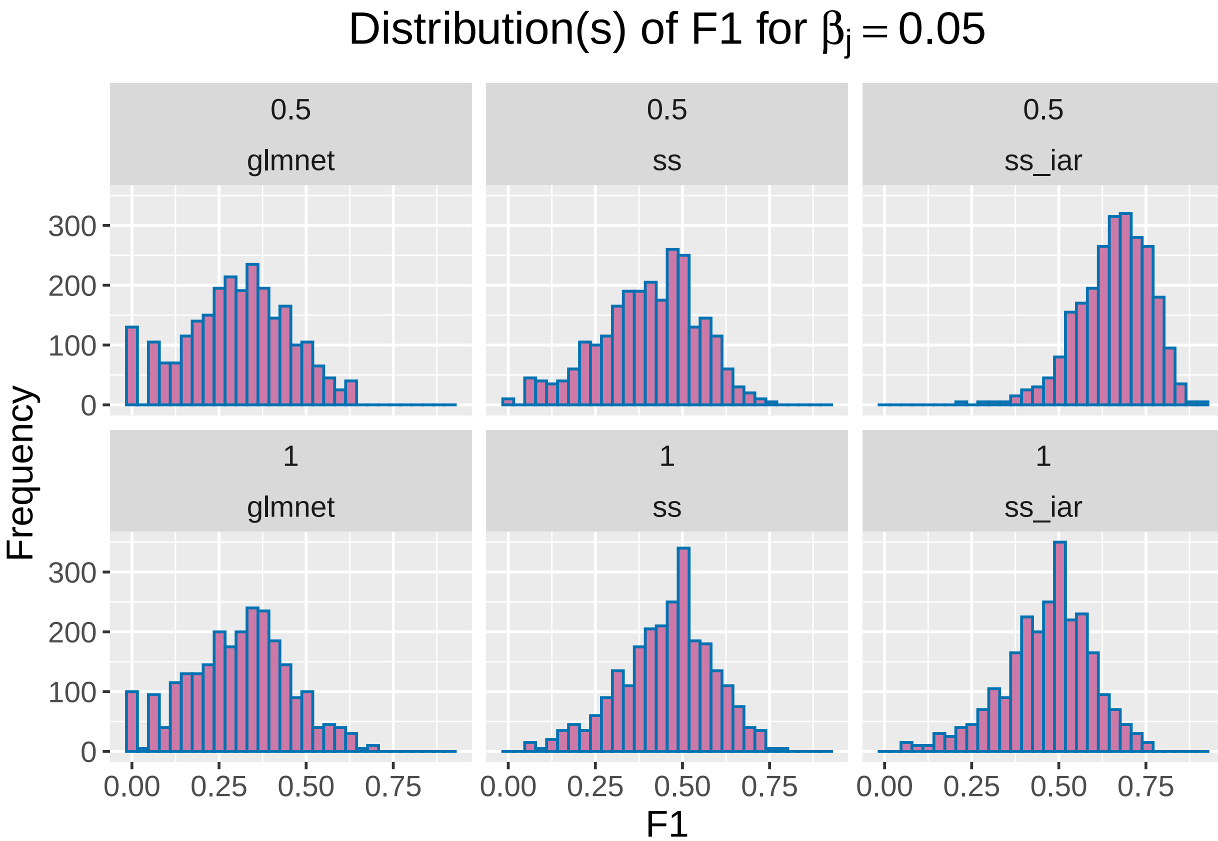
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

```
## Warning: Removed 125 rows containing non-finite values (stat_bin).
```




```
ggplot(data = results.smallB.opt,
       aes(x = f1)) +
  facet_wrap(alpha~model) +
  geom_histogram(fill = cbpg[8], color = cbpg[6]) +
  scale_y_continuous(name = "Frequency") +
  scale_x_continuous(name = "F1") +
  ggtitle(bquote(paste("Distribution(s) of F1 for ", beta[j] == 0.05))) +
  theme(plot.title = element_text(hjust = 0.5),
        text = element_text(size = 14),
        legend.position = "bottom")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Distribution(s) of F1 for $\beta_j = 0.05$
0.5
glmnet
0.5
ss
0.5
ss_iar
1
glmnet
1
ss
1
ss_iar
Frequency
0
100
200
300
0.00
0.25
0.50
0.75
F1