Propensity Score Weighting using Machine Learning

Young Geun Kim
ygeunkim.github.io
2019711358, Department of Statistics

18 Dec, 2020

Abstract

Generally, we estimate propensity score using logistic regression model. In this report, we try to implement machine learning methods - random forests and SVM. In some simulation scheme, we evaluate the result with averate standardized absolute mean distance and empirical distribution of average treatment effect. We provide an R package for this experiment in this link.[1]

Contents

[1] https://github.com/ygeunkim/propensityml

1 Introduction

Write propensity score $e(\mathbf{x})$ by

$$e(\mathbf{x}) := P(Z = 1 \mid \mathbf{X} = \mathbf{x})$$

In general, we estimate propensity scores via logistic regression model.

$$\log \frac{e(\mathbf{x})}{1 - e(\mathbf{x})} = X\boldsymbol{\beta} \tag{1}$$

Observe that covariates are linear. If this parametric model is wrong, the estimation can work bad. On the other hand, machine learning models sometimes can explain nonlinear or nonparametric situations. In this sense, we try to compare propensity score weighting from logistic with machine learning models.

- Random forests (Liaw and Wiener 2002): default values of the function
- SVM (Meyer et al. 2020): linear kernel and radial kernel

In this report, we conduct Monte Carlo simulation. Section 2 presents the structure of the simulation and evaluation. In Section @ref{discuss}, we see the results of the simulation and discuss about them.

For this work, we made an `R` package called `propensityml`. In each step, we try to introduce some function in this package.

```
# remotes::install_github("ygeunkim/propensityml")
library(propensityml)
```

2 Monte Carlo Simulation

2.1 Setting

We implement the Monte Carlo simulation setting of Lee, Lessler, and Stuart (2010). They changed the outcome part of Setoguchi et al. (2008). See Figure 1. There are 10 covariates - 4 confounders, 3 exposure predictors, and 3 outcome predictors. Now we generate true propensity score. Since we want to see whether logistic regression model properly works, we consider 4 scenarios. Lee, Lessler, and Stuart (2010) and Setoguchi et al. (2008) had actually tried 7, but we choose 4 due to computation limit. These 4 scenarios are similar to choice of Pirracchio, Petersen, and Laan (2014).

A Additivity and linearity:

$$P(Z = 1 \mid X_i) = \frac{1}{1 + \exp(-(\beta_0 + \beta_1 X_1 + \cdots + \beta_7 X_7))}$$

B Moderate non-linearity: *3 quadratic term*

$$P(Z = 1 \mid X_i) = \frac{1}{1 + \exp(-(\beta_0 + \beta_1 X_1 + \cdots + \beta_7 X_7 + \beta_2 X_2^2))}$$

F Moderate non-linearity: *10 two-way interaction terms*

G Moderate non-additivity and non-linearity: *10 two-way interaction terms and 3 quadratic terms*

Here, true parameters are $(\beta_0, \beta_1, \ldots, \beta_7)^T = (0, 0.8, -0.25, 0.6, -0.4, -0.8, -0.5, 0.7)^T$. Next, Lee, Lessler, and Stuart (2010) generate continuous outcome by

$$Y = \alpha_0 + \alpha_1 X_1 + \cdots + \alpha_4 X_4 + \alpha_5 X_8 + \cdots + \alpha_7 X_{10} + \gamma Z$$

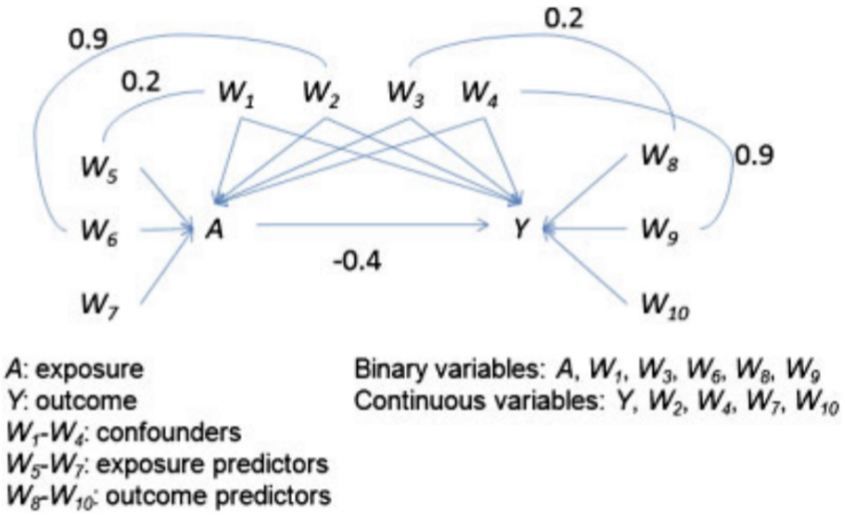

Figure 1: Simulation Data - Each W and A can be as X and Z in the course, respectively

where $(\alpha_0, \alpha_1, \ldots, \alpha_7)^T = (-3.85, 0.3, -0.36, -73, -0.2, 0.71, -0.19, 0.26)^T$ and the true effect is $\gamma = -0.4$.

In `propensityml` package, `sim_outcome()` can reproduce dataset. Without this step. The following function generates dataset ready for MC simulation.

```
doMC::registerDoMC(cores = 4)
mc_list <- mc_setoguchi(
  N = 1000, n_dat = 1000, scenario = scen,
  parallel = TRUE
)
```

Based on this setting, we generate 1000 replicates of datasets of which sample size is 1000.

2.2 Evaluation

Now we can estimate propensity score for each MC set. Here we introduce methods of evaluation.

Average standardized absolute mean distance (ASAM)

Recall that covariate balancing is computed by standardized mean difference. Average standardized abosulute mean distance (ASAM) is its average across all covariates. The lower, the more similar treatment and control groups are given covariates. We provide the code as follows:

```
doMC::registerDoMC(cores = 8)
logit_asam <-
  mc_list %>%
  compute_asam(
    treatment = "exposure", outcome = "y", exclude = "exposure_prob",
    formula = exposure ~ . - y - exposure_prob, method = "logit",
    mc_col = "mcname", sc_col = "scenario", parallel = TRUE
  )
```

Effect Estimator

Lee, Lessler, and Stuart (2010) saw both ATE and ATC estimator. For ATE,

Table 1: ASAM results

Scenarios	Model			
	Logistic	RF	SVM (Linear)	SVM (Radial)
A	0.011	0.011	0.009	0.009
B	0.033	0.029	0.041	0.042
F	0.035	0.033	0.042	0.042
G	0.077	0.075	0.081	0.081

$$\frac{Z_i}{\hat{e}_i} - \frac{1 - Z_i}{1 - \hat{e}_i} \tag{2}$$

For ATC,

$$Z_i - \frac{\hat{e}_i(1 - Z_i)}{1 - \hat{e}_i} \tag{3}$$

Empirical distribution of these esitmators can be the evaluation.

```
doMC::registerDoMC(cores = 8)
wt_logit <-
  mc_list %>%
  add_weighting(
    treatment = "exposure",
    formula = exposure ~ . - y - exposure_prob, method = "logit",
    mc_col = "mcname", sc_col = "scenario", parallel = TRUE
  )
```

3 Results

3.1 ASAM

Lee, Lessler, and Stuart (2010) condered under 0.2 as balanced scenario. Table 1 represents the result of ASAM computation. In every scenario, logistic regression shows the lowest value. Lee, Lessler, and Stuart (2010) mentioned about the skewed distribution despite about the low value.

3.2 Effect Estimator

Propensity Score

In Figure 2, empirical distribution of propensity score is prented. In every scenario, random forest leads to extreme estimates (0 or 1) of propensity scores. In scenario G, exceptionally, logistic regression also shows similar pattern.

ATE

The pattern of propensity score affects estimation of effect.

ATC

The pattern of propensity score affects estimation of effect. See Figure 3 and Figure 4. Appendix A gives each performance metric.

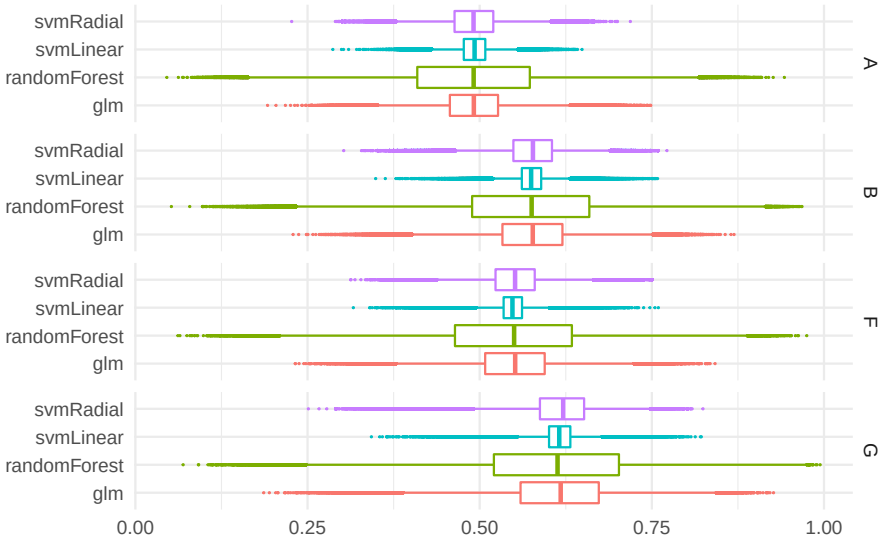

Figure 2: Empirical Distribution of Propensity Scores

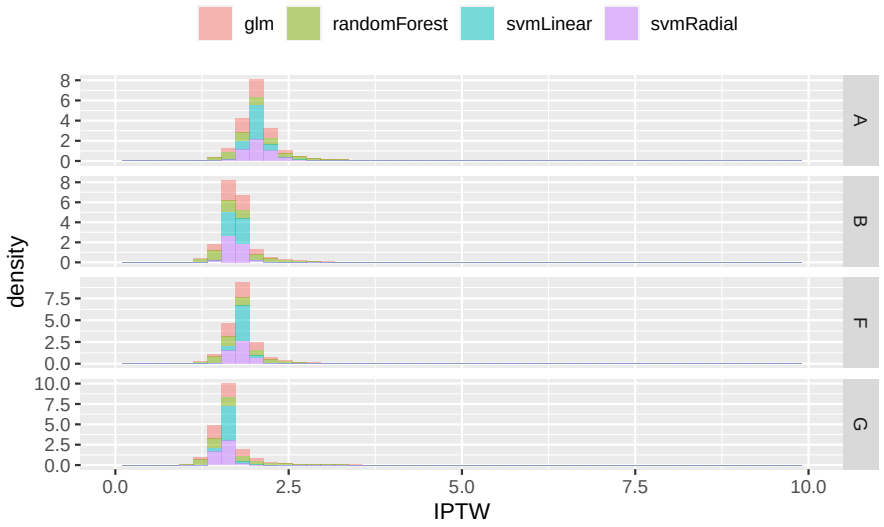

Figure 3: Empirical Distribution of IPTW

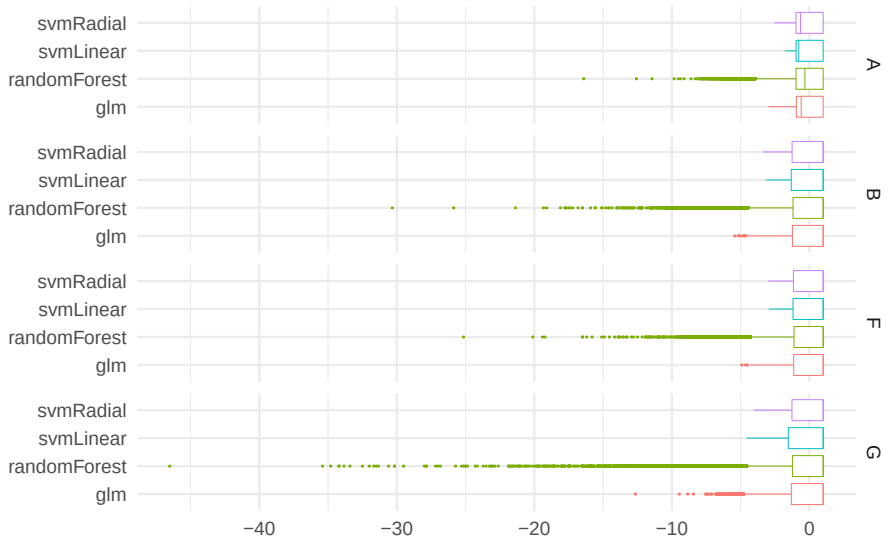

Figure 4: Empirical Distribution of ATC Estimation

4 Conclusion

In this work, we compare propensity score weighting based on logistic regression with random forests or SVM. Random forests gave too extreme propensity score. We cannot give responsibility for random forest method of this because we skipped model selection step. On the other hand, SVM worked quite well.

As future studies, we need to try parameter selection and IPW-SIPW comparison. As mentioned in Section 3.1, empirical distribution of ASAM might be good insight for this subject.

References

Lee, Brian K., Justin Lessler, and Elizabeth A. Stuart. 2010. "Improving propensity score weighting using machine learning." *Statistics in Medicine* 29 (3): 337–46. https://doi.org/10.1002/sim.3782.

Liaw, Andy, and Matthew Wiener. 2002. "Classification and Regression by randomForest." *R News* 2 (3): 18–22. https://CRAN.R-project.org/doc/Rnews/.

Meyer, David, Evgenia Dimitriadou, Kurt Hornik, Andreas Weingessel, and Friedrich Leisch. 2020. "E1071: Misc Functions of the Department of Statistics, Probability Theory Group (Formerly: E1071), Tu Wien." https://CRAN.R-project.org/package=e1071.

Pirracchio, Romain, Maya L. Petersen, and Mark van der Laan. 2014. "Improving Propensity Score Estimators' Robustness to Model Misspecification Using Super Learner." *American Journal of Epidemiology* 181 (2): 108–19. https://doi.org/10.1093/aje/kwu253.

Setoguchi, Soko, Sebastian Schneeweiss, M. Alan Brookhart, Robert J. Glynn, and E. Francis Cook. 2008. "Evaluating uses of data mining techniques in propensity score estimation: a simulation study." *Pharmacoepidemiology and Drug Safety* 17 (6): 546–55. https://doi.org/10.1002/pds.1555.

A Appendix: Tables

Large tables for effect estimators.

Table 2: Performance metic of IPTW

Metric	Scenarios	Model			
		Logistic regression	Random forests	SVM (Linear)	SVM (Radial)
bias	A	2.400	2.534	2.434	2.403
	B	2.400	2.538	2.376	2.400
	F	2.400	2.534	2.385	2.399
	G	2.399	2.540	2.289	2.400
estimate	A	0.000	0.007	0.001	-0.003
	B	0.000	-0.026	0.004	0.001
	F	0.000	-0.015	0.001	0.012
	G	0.003	-0.040	0.041	-0.006
mse	A	4.207	5.096	4.342	4.183
	B	4.336	5.331	4.199	4.266
	F	4.276	5.204	4.183	4.210
	G	4.534	5.768	3.977	4.400
sd	A	2.012	2.220	2.045	2.006
	B	2.044	2.278	2.009	2.026
	F	2.029	2.248	2.006	2.010
	G	2.091	2.375	1.945	2.060

Table 3: Performance Metric of ATC Estimation

Metric	Scenarios	Model			
		Logistic regression	Random forests	SVM (Linear)	SVM (Radial)
bias	A	1.383	1.447	1.400	1.387
	B	1.552	1.634	1.538	1.551
	F	1.501	1.576	1.494	1.495
	G	1.627	1.719	1.553	1.632
estimate	A	0.000	-0.064	-0.017	-0.003
	B	0.000	-0.082	0.014	0.000
	F	0.000	-0.075	0.008	0.006
	G	0.002	-0.090	0.076	-0.003
mse	A	1.149	1.382	1.166	1.136
	B	1.576	2.056	1.525	1.525
	F	1.433	1.817	1.403	1.381
	G	1.872	2.633	1.635	1.775
sd	A	0.995	1.126	1.009	0.989
	B	1.190	1.398	1.164	1.168
	F	1.128	1.308	1.112	1.103
	G	1.308	1.593	1.187	1.272

B Appendix: Codes

B.1 Loading Packages

```
# tidyverse family----------------------
library(tidyverse)
# large data frame----------------------
library(data.table)
# parallel-----------------------------
library(foreach)
```

```r
library(parallel)
# custom packages----------------------
library(rmdtool) # install_github("ygeunkim/rmdtool")
# kable--------------------------------
library(knitr)
library(kableExtra)
# set seed for report -----------------
set.seed(1)
```

B.2 De-echoed Codes

```r
# rf-----------------------------
doMC::registerDoMC(cores = 8)
rf_asam <-
  mc_list %>%
  compute_asam(
    treatment = "exposure", outcome = "y", exclude = "exposure_prob",
    formula = exposure ~ . - y - exposure_prob, method = "rf",
    mc_col = "mcname", sc_col = "scenario", parallel = TRUE
  )
# svm----------------------------
doMC::registerDoMC(cores = 8)
svm_asam <-
  mc_list %>%
  compute_asam(
    treatment = "exposure", outcome = "y", exclude = "exposure_prob",
    formula = exposure ~ . - y - exposure_prob,
    method = "SVM", kernel = "radial",
    mc_col = "mcname", sc_col = "scenario", parallel = TRUE
  )
# svm (linear)-------------------------
doMC::registerDoMC(cores = 8)
svm_asam_lin <-
  mc_list %>%
  compute_asam(
    treatment = "exposure", outcome = "y", exclude = "exposure_prob",
    formula = exposure ~ . - y - exposure_prob,
    method = "SVM", kernel = "linear",
    mc_col = "mcname", sc_col = "scenario", parallel = TRUE
  )
# rf-------------------------
doMC::registerDoMC(cores = 8)
wt_rf <-
  mc_list %>%
  add_weighting(
    treatment = "exposure",
    formula = exposure ~ . - y - exposure_prob, method = "rf",
    mc_col = "mcname", sc_col = "scenario", parallel = TRUE
  )
# SVM (radial)-------------------------
doMC::registerDoMC(cores = 8)
wt_svm <-
  mc_list %>%
```

```
  add_weighting(
    treatment = "exposure",
    formula = exposure ~ . - y - exposure_prob, method = "SVM",
    mc_col = "mcname", sc_col = "scenario", parallel = TRUE
  )
# SVM (linear)-------------------------
doMC::registerDoMC(cores = 8)
wt_svm_lin <-
  mc_list %>%
  add_weighting(
    treatment = "exposure",
    formula = exposure ~ . - y - exposure_prob,
    method = "SVM", kernel = "linear",
    mc_col = "mcname", sc_col = "scenario", parallel = TRUE
  )
```

B.3 Knitting Figures

```
col_name <- names(mc_list)
name_logit <-
  names(wt_logit[,-c("iptw", "propwt")]) %>%
  str_replace_all(pattern = "propensity", replacement = "glm")
name_rf <-
  names(wt_rf[,-c("iptw", "propwt")]) %>%
  str_replace_all(pattern = "propensity", replacement = "randomForest")
name_svm_lin <-
  names(wt_svm_lin[,-c("iptw", "propwt")]) %>%
  str_replace_all(pattern = "propensity", replacement = "svmLinear")
name_svm <-
  names(wt_svm[,-c("iptw", "propwt")]) %>%
  str_replace_all(pattern = "propensity", replacement = "svmRadial")
ps_dat <-
  wt_logit[,-c("iptw", "propwt")] %>%
  setNames(name_logit) %>%
  merge(wt_rf[,-c("iptw", "propwt")] %>% setNames(name_rf), by = col_name) %>%
  merge(wt_svm_lin[,-c("iptw", "propwt")] %>% setNames(name_svm_lin), by = col_name) %>%
  merge(wt_svm[,-c("iptw", "propwt")] %>% setNames(name_svm), by = col_name) %>%
  melt(id.vars = col_name, variable.name = "model", value.name = "PS")
ps_dat %>%
  ggplot() +
  geom_boxplot(aes(x = model, y = PS, colour = model), outlier.size = .05, show.legend = FALSE) +
  theme_minimal() +
  labs(
    x = element_blank(),
    y = element_blank()
  ) +
  facet_grid(scenario ~ ., scales = "free_y") +
  coord_flip()
col_name <- names(mc_list)
name_logit <-
  names(wt_logit[,-c("propensity", "propwt")]) %>%
  str_replace_all(pattern = "iptw", replacement = "glm")
name_rf <-
```

```
  names(wt_rf[,-c("propensity", "propwt")]) %>%
  str_replace_all(pattern = "iptw", replacement = "randomForest")
name_svm_lin <-
  names(wt_svm_lin[,-c("propensity", "propwt")]) %>%
  str_replace_all(pattern = "iptw", replacement = "svmLinear")
name_svm <-
  names(wt_svm[,-c("propensity", "propwt")]) %>%
  str_replace_all(pattern = "iptw", replacement = "svmRadial")
iptw_dat <-
  wt_logit[,-c("propensity", "propwt")] %>%
  setNames(name_logit) %>%
  merge(wt_rf[,-c("propensity", "propwt")] %>% setNames(name_rf), by = col_name) %>%
  merge(wt_svm_lin[,-c("propensity", "propwt")] %>% setNames(name_svm_lin), by = col_name) %>%
  merge(wt_svm[,-c("propensity", "propwt")] %>% setNames(name_svm), by = col_name) %>%
  melt(id.vars = col_name, variable.name = "model", value.name = "IPTW")
iptw_dat %>%
  ggplot() +
  geom_histogram(aes(x = IPTW, y = ..density.., fill = model), alpha = .5, bins = 50) +
  theme(legend.position = "top", legend.title = element_blank()) +
  # labs(
  #   fill = element_blank()
  # ) +
  xlim(0, 10) +
  facet_grid(scenario ~ ., scales = "free_y")
col_name <- names(mc_list)
name_logit <-
  names(wt_logit[,-c("propensity", "iptw")]) %>%
  str_replace_all(pattern = "propwt", replacement = "glm")
name_rf <-
  names(wt_rf[,-c("propensity", "iptw")]) %>%
  str_replace_all(pattern = "propwt", replacement = "randomForest")
name_svm_lin <-
  names(wt_svm_lin[,-c("propensity", "iptw")]) %>%
  str_replace_all(pattern = "propwt", replacement = "svmLinear")
name_svm <-
  names(wt_svm[,-c("propensity", "iptw")]) %>%
  str_replace_all(pattern = "propwt", replacement = "svmRadial")
propwt_dat <-
  wt_logit[,-c("propensity", "iptw")] %>%
  setNames(name_logit) %>%
  merge(wt_rf[,-c("propensity", "iptw")] %>% setNames(name_rf), by = col_name) %>%
  merge(wt_svm_lin[,-c("propensity", "iptw")] %>% setNames(name_svm_lin), by = col_name) %>%
  merge(wt_svm[,-c("propensity", "iptw")] %>% setNames(name_svm), by = col_name) %>%
  melt(id.vars = col_name, variable.name = "model", value.name = "weight")
propwt_dat %>%
  ggplot() +
  geom_boxplot(aes(x = model, y = weight, colour = model), size = .1, outlier.size = .01, show.legend =
  theme_minimal() +
  labs(
    x = element_blank(),
    y = element_blank()
  ) +
  facet_grid(rows = vars(scenario)) +
```

```
  coord_flip()
```

B.4 Knitting Tables

```
logit_asam %>%
  setNames(c("scenario", "glm")) %>%
  merge(rf_asam %>% setNames(c("scenario", "rf")), by = "scenario") %>%
  merge(svm_asam_lin %>% setNames(c("scenario", "svmLin")), by = "scenario") %>%
  merge(svm_asam %>% setNames(c("scenario", "svmRad")), by = "scenario") %>%
  kable(
    format = "latex",
    col.names = c("Scenarios", "Logistic", "RF", "SVM (Linear)", "SVM (Radial)"),
    escape = FALSE,
    caption = "ASAM results"
  ) %>%
  add_header_above(c(" " = 1, "Model" = 4))
emp_tab_iptw %>%
  kable(
    format = "latex",
    col.names = c("Metric", "Scenarios", "Logistic regression", "Random forests", "SVM (Linear)", "SVM
    escape = FALSE,
    caption = "Performance metic of IPTW"
  ) %>%
  kable_styling("striped", full_width = FALSE, latex_options = c("HOLD_position", "scale_down"), font_s
  add_header_above(c(" " = 1, " " = 1, "Model" = 4)) %>%
  collapse_rows(columns = 1, valign = "top")
emp_tab_wt %>%
  kable(
    format = "latex",
    col.names = c("Metric", "Scenarios", "Logistic regression", "Random forests", "SVM (Linear)", "SVM
    escape = FALSE,
    caption = "Performance Metric of ATC Estimation"
  ) %>%
  kable_styling("striped", full_width = FALSE, latex_options = c("HOLD_position", "scale_down"), font_s
  add_header_above(c(" " = 1, " " = 1, "Model" = 4)) %>%
  collapse_rows(columns = 1, valign = "top")
```