```r
library(tidyverse)
library(survival)
library(randomForestSRC)
library(xgboost)
library(survminer)

source("/work/users/y/u/yuukias/BIOS-Material/BIOS992/utils/csv_utils.r")

adjust_type <- ifelse(exists("params"), params$adjust_type, "full") # options: "minimal", "pa
impute_type <- ifelse(exists("params"), params$impute_type, "imputed") # options: "unimputed"
include_statin <- ifelse(exists("params"), params$include_statin, "no") # options: "yes", "no

set.seed(1234)
```

```r
# string of parameters
adjust_type_str <- switch(adjust_type,
    minimal = "minimal",
    partial = "partial",
    full = "full"
)
print(paste0("Model Adjustment Type: ", adjust_type_str))
```

```
[1] "Model Adjustment Type: full"
```

```r
impute_type_str <- switch(impute_type,
    unimputed = "unimputed",
    imputed = "imputed"
)
print(paste0("Data Imputation Type: ", impute_type_str))
```

```
[1] "Data Imputation Type: imputed"
```

Load Models

```r
load(get_data_path("rsf_model", adjust_type_str, impute_type_str, include_statin, model = "rs

load(get_data_path("xgb_var_select_name", adjust_type_str, impute_type_str, include_statin, m
load(get_data_path("xgb_model", adjust_type_str, impute_type_str, include_statin, model = "xg
```

Load Data

All participants will be loaded, including the training set and the test set.

```r
if (include_statin == "yes") {
    data_train <- read.csv(paste0("/work/users/y/u/yuukias/BIOS-Material/BIOS992/data/train_d
        header = TRUE
    )
    data_test <- read.csv(paste0("/work/users/y/u/yuukias/BIOS-Material/BIOS992/data/test_dat
        header = TRUE
    )
} else {
    data_train <- read.csv(paste0("/work/users/y/u/yuukias/BIOS-Material/BIOS992/data/train_d
        header = TRUE
    )
    data_test <- read.csv(paste0("/work/users/y/u/yuukias/BIOS-Material/BIOS992/data/test_dat
        header = TRUE
    )
}

data <- bind_rows(data_train, data_test)
data <- data[, -1] # the first column is the index generated by sklearn
(dim(data))
```

```
[1] 35159   100
```

```r
data <- select_subset(data, type = adjust_type)
(dim(data))
```

```
[1] 35159    89
```

```r
data <- tibble::as_tibble(data)
```

```r
# * There are some imputed ethnicity set to "e". We will exclude them at this time.
data <- data %>%
    filter(ethnicity != "e")

# * We also need to manually relevel the categorical variables
data <- data %>%
    mutate(
        # Set "Never" (0) as baseline for smoking
        smoking = factor(smoking,
            levels = c("0", "1", "2", "-3"),
            labels = c("Never", "Previous", "Current", "Prefer not to answer")
        ),

        # Set "No" (0) as baseline for diabetes
        diabetes = factor(diabetes,
            levels = c("0", "1", "-1", "-3"),
            labels = c("No", "Yes", "Do not know", "Prefer not to answer")
        ),

        # Ensure other categorical variables are properly factored
        ethnicity = factor(ethnicity,
            levels = c("1", "2", "3", "4", "5", "6"),
            labels = c("White", "Mixed", "Asian/Asian British", "Black/Black British", "Chine
        ),
        education = factor(education,
            levels = c("1", "2", "3", "4", "5", "6", "-7", "-3"),
            labels = c(
                "College/University degree", "A levels/AS levels",
                "O levels/GCSEs", "CSEs", "NVQ/HND/HNC",
                "Other professional", "None of the above",
                "Prefer not to answer"
            )
        ),
        activity = factor(activity,
            levels = c("0", "1", "2"),
            labels = c("Low", "Moderate", "High")
        ),
        sex = factor(sex,
            levels = c("0", "1"),
            labels = c("Female", "Male")
        ),
        hypertension_treatment = factor(hypertension_treatment,
            levels = c("0", "1"),
            labels = c("No", "Yes")
        )
    )
```

```
# * It is very hard to compare the HR as different predictors are on different magnitudes, so
time_col <- data$time
```

```r
event_col <- data$event
data <- data %>%
    select(-c(time, event)) %>%
    mutate(across(where(is.numeric), scale)) %>%
    mutate(
        time = time_col,
        event = event_col
    )
```

Note now the interpretation of HR is different! For example, if HR=1.16 for the predictor in the univariate model fitted using scaled data, it means that each standard deviation increase is associated with 16% higher risk of event.

```r
# For Cox model:
data_complete <- na.omit(data)

# For RSF model: We don't need to exclude the missing values
data_complete_rsf <- data_complete %>% select(-c(time, event))
data_complete_rsf <- model.frame(~ . - 1, data = data_complete_rsf, na.action = na.pass)
data_complete_rsf <- model.matrix(~ . - 1, data = data_complete_rsf)
data_complete_rsf <- as.data.frame(data_complete_rsf)


# For XGBoost model:
total_x <- data_complete %>% select(-c(time, event))
total_x_xgb <- model.frame(~ . - 1, data = total_x, na.action = na.pass)
total_x_xgb <- model.matrix(~ . - 1, data = total_x_xgb)
total_y_lower_bound <- data_complete$time
total_y_upper_bound <- ifelse(data_complete$event == 1, data_complete$time, Inf)
# dtotal <- xgb.DMatrix(
#     data = total_x_xgb,
#     label_lower_bound = total_y_lower_bound,
#     label_upper_bound = total_y_upper_bound
# )
dtotal_selected <- xgb.DMatrix(
    data = total_x_xgb[, vars_selected],
    label_lower_bound = total_y_lower_bound,
    label_upper_bound = total_y_upper_bound
)
```

Prediction

Here we will categorize all participants into three risk groups based on their risk scores. As the fully-adjusted model has best performance, we will use the risk scores from these **fully-adjusted model with the selected features** to predict the risk groups.

Predict using RSF Model

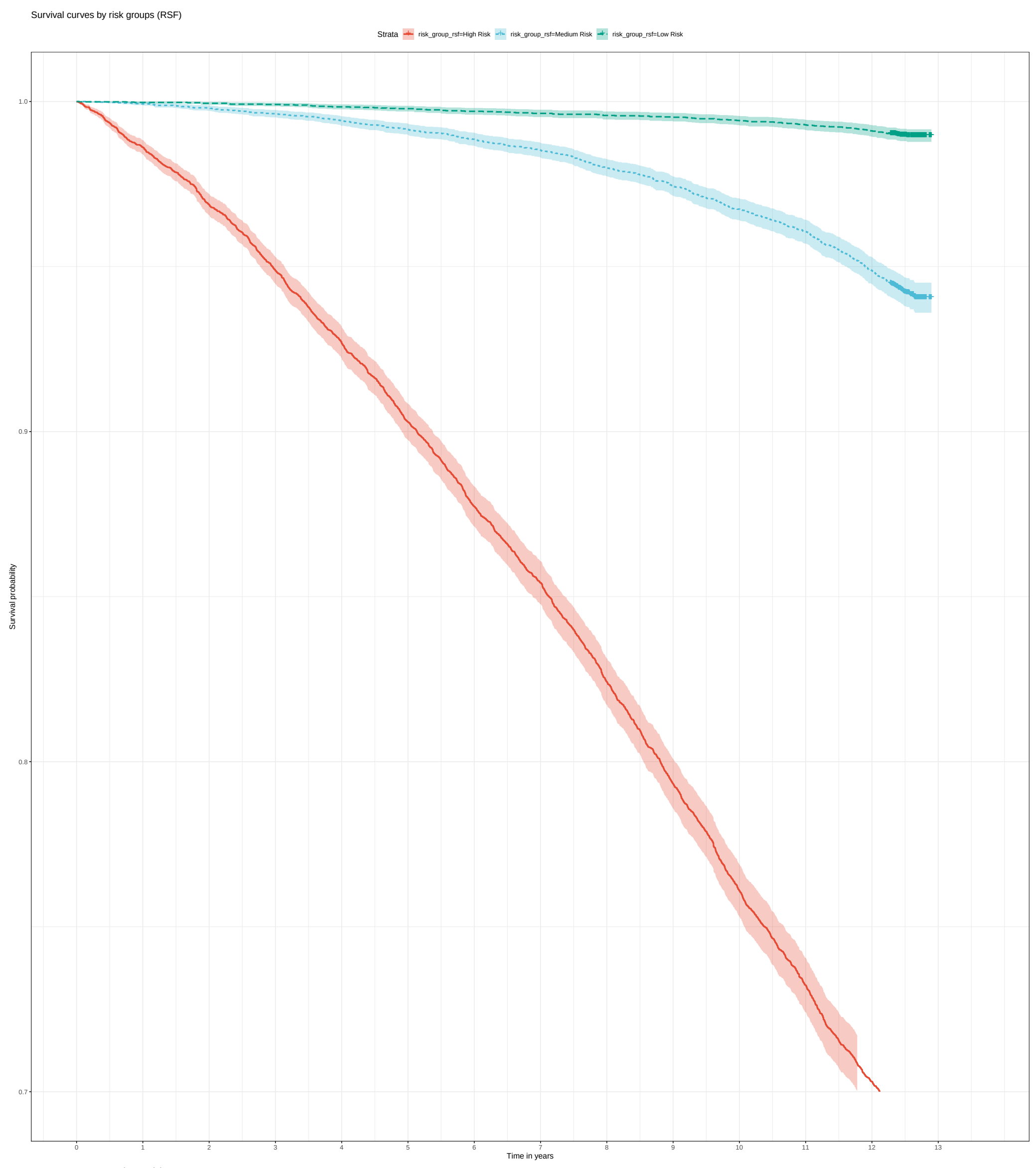
```
pref_rsf <- predict(rsf_model, newdata = data_complete_rsf)$predicted
```

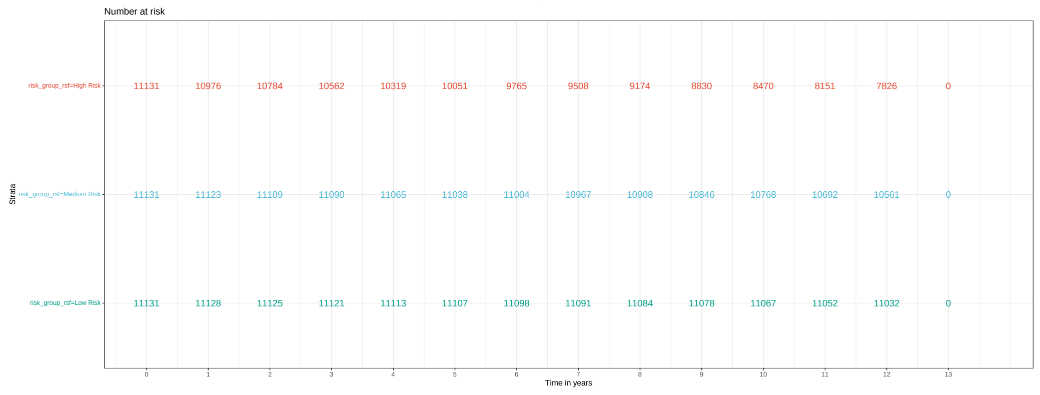
```
data_complete <- data_complete %>%
    mutate(risk_group_rsf = cut(pref_rsf,
        breaks = quantile(pref_rsf, probs = c(0, 1 / 3, 2 / 3, 1)),
        labels = c("Low Risk", "Medium Risk", "High Risk"),
        include.lowest = TRUE
    )) %>%
    mutate(risk_group_rsf = factor(risk_group_rsf,
                                  levels = c("High Risk", "Medium Risk", "Low Risk")))
```

```
ggsurvplot(
    fit = survfit(Surv(time, event) ~ risk_group_rsf, data = data_complete),
    data = data_complete,
    pval = TRUE,
    conf.int = TRUE,
    risk.table = TRUE,
    risk.table.col = "strata",
    linetype = "strata",
    surv.median.line = "hv",
    ggtheme = theme_bw(),
    palette = "npg",
    title = "Survival curves by risk groups (RSF)",
    xlab = "Time in years",
    break.time.by = 365.25,  # tick
    xscale = "d_y",          # convert time scale from day to year
    ylim = c(0.7, 1)
)
```

```
Warning in .add_surv_median(p, fit, type = surv.median.line, fun = fun, :
Median survival not reached.
```

```
Warning: Removed 218 rows containing missing values or values outside the scale range
(`geom_step()`).

Warning: Removed 177 rows containing missing values or values outside the scale range
(`geom_point()`).

Warning: Removed 1 row containing missing values or values outside the scale range
(`geom_text()`).

Warning: Removed 218 rows containing missing values or values outside the scale range
(`geom_step()`).

Warning: Removed 177 rows containing missing values or values outside the scale range
(`geom_point()`).

Warning: Removed 1 row containing missing values or values outside the scale range
(`geom_text()`).
```




Number at risk

Strata	0	1	2	3	4	5	6	7	8	9	10	11	12	13
risk_group_rsf=High Risk	11131	10976	10784	10562	10319	10051	9765	9508	9174	8830	8470	8151	7826	0
risk_group_rsf=Medium Risk	11131	11123	11109	11090	11065	11038	11004	10967	10908	10846	10768	10692	10561	0
risk_group_rsf=Low Risk	11131	11128	11125	11121	11113	11107	11098	11091	11084	11078	11067	11052	11032	0

Time in years

Predict using XGBoost Model

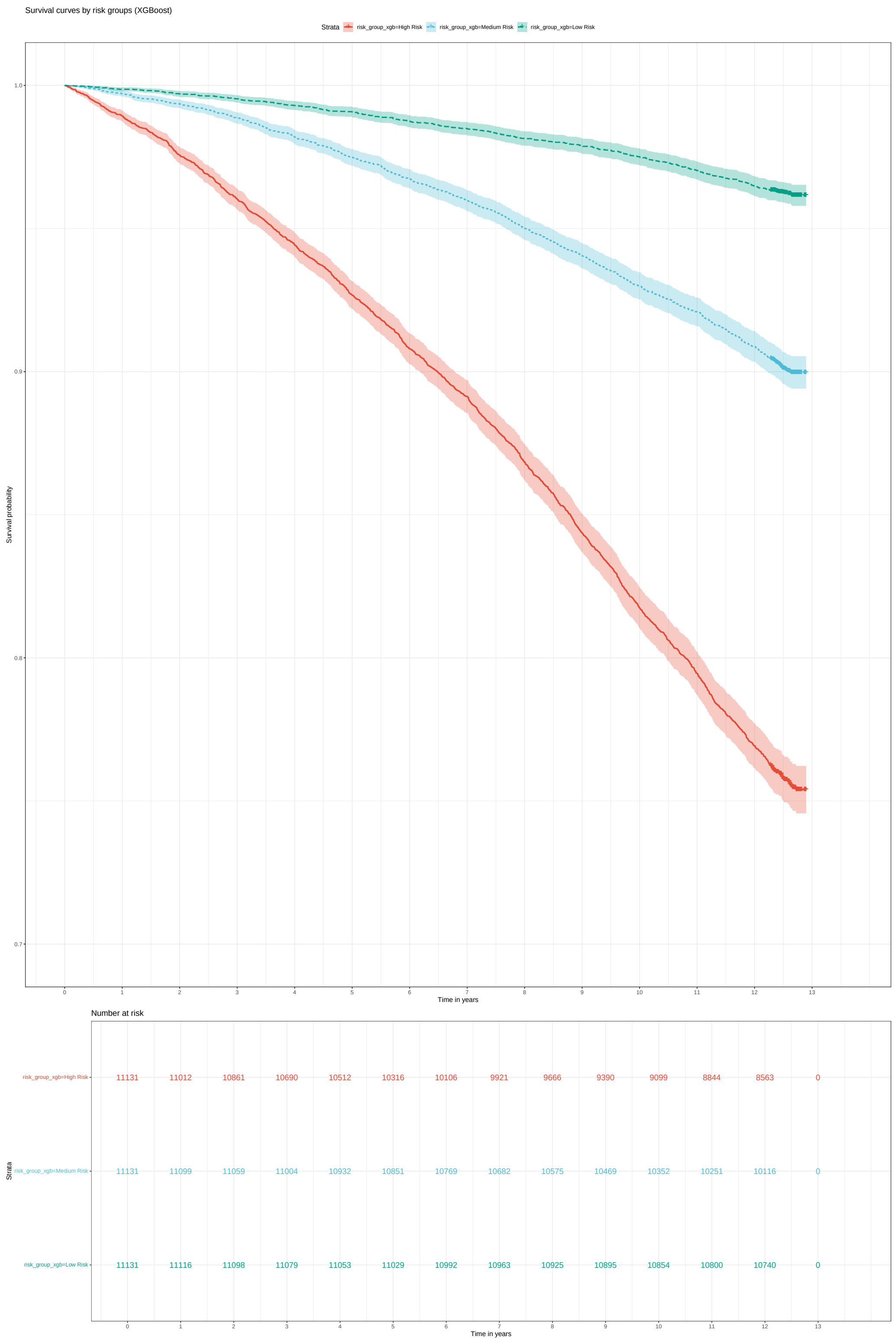
```r
pred_xgb <- predict(xgb_model, newdata = dtotal_selected)
pred_xgb <- -pred_xgb # now larger value means higher risk
```

```r
data_complete <- data_complete %>%
    mutate(risk_group_xgb = cut(pred_xgb,
        breaks = quantile(pred_xgb, probs = c(0, 1 / 3, 2 / 3, 1)),
        labels = c("Low Risk", "Medium Risk", "High Risk"),
        include.lowest = TRUE
    )) %>%
    mutate(risk_group_xgb = factor(risk_group_xgb,
                               levels = c("High Risk", "Medium Risk", "Low Risk")))
```

```r
ggsurvplot(
    fit = survfit(Surv(time, event) ~ risk_group_xgb, data = data_complete),
    data = data_complete,
    pval = TRUE,
    conf.int = TRUE,
    risk.table = TRUE,
    risk.table.col = "strata",
    linetype = "strata",
    surv.median.line = "hv",
    ggtheme = theme_bw(),
    palette = "npg",
    title = "Survival curves by risk groups (XGBoost)",
    xlab = "Time in years",
    break.time.by = 365.25,  # tick
    xscale = "d_y",          # convert time scale from day to year
    ylim = c(0.7, 1)
)
```

```
Warning in .add_surv_median(p, fit, type = surv.median.line, fun = fun, :
Median survival not reached.
```

```
Warning: Removed 1 row containing missing values or values outside the scale range
(`geom_text()`).
Removed 1 row containing missing values or values outside the scale range
(`geom_text()`).
```



Strata	0	1	2	3	4	5	6	7	8	9	10	11	12	13
risk_group_xgb=High Risk	11131	11012	10861	10690	10512	10316	10106	9921	9666	9390	9099	8844	8563	0
risk_group_xgb=Medium Risk	11131	11099	11059	11004	10932	10851	10769	10682	10575	10469	10352	10251	10116	0
risk_group_xgb=Low Risk	11131	11116	11098	11079	11053	11029	10992	10963	10925	10895	10854	10800	10740	0

Time in years