NHISS Example

Contents

Chapter 1

Baseline characteristics table

Baseline tables show the characteristics of research subjects included in a study. A table characterizing baseline characteristics is so important that it's typically the first table that appears in any observational epidemiology (or clinical trial) manuscript, so it's commonly referred to as a "Table 1". The "Table 1" contain information about the mean and standard deviation(or median and IQR) for continue/scale variable, and proportion for categorical variable.

Baseline characteristic table should be created before imputaion, matching, or weighting.

Using data **final_db**
Outcome variable : **HTN**
Follow-up period : **DATEDIFF**
Exposure variable : **DM**
Covariates : **Age, Sex, SES, Region, BMI, CCI, Comorbidities(Dyslipidemia, Ischemic heart disease)**

```
## load library
library(moonBook)
library(dplyr)
```

```
## load data
final_db <- read.csv('Data/final_db.csv', header=T)
```

```
## formula
formula.bc <- formula(DM ~ HTN + DATEDIFF + AGE + SEX + SES + REGION + BMI + CCI + DYS
```

- Use **mytable()** function in **moonBook** package to create baseline characteristic tables.
  - method=1 : forces analysis as normal-distributed
  - method=3 : performs a Shapiro-Wilk test to decide between normal or non-normal

1.1 Baseline characteristics between groups

```
mytable(formula.bc, data=final_db, method=3)
```

```
##
##               Descriptive Statistics by 'DM'
## ------------------------------------------------------------
##                     0                    1             p
##                  (N=2356)             (N=118)
## ------------------------------------------------------------
##  HTN                                                  0.000
##    - 0        2215 (94.0%)           69 (58.5%)
##    - 1         141 ( 6.0%)           49 (41.5%)
##  DATEDIFF 1685.0 [835.5;2460.5] 963.5 [324.0;1690.0] 0.000
##  AGE        36.0 [22.0;48.0]      58.0 [50.0;68.0]   0.000
##  SEX                                                  0.903
##    - 1        1182 (50.2%)           58 (49.2%)
##    - 2        1174 (49.8%)           60 (50.8%)
##  SES                                                  0.393
##    - 1         668 (29.6%)           29 (25.2%)
##    - 2         709 (31.4%)           34 (29.6%)
##    - 3         883 (39.1%)           52 (45.2%)
##  REGION                                               0.996
##    - 1        1160 (49.5%)           58 (49.2%)
##    - 2         489 (20.9%)           25 (21.2%)
##    - 3         694 (29.6%)           35 (29.7%)
##  BMI        23.1 [21.0;25.2]      24.3 [22.6;26.1]   0.013
##  CCI                                                  0.032
##    - 0        1810 (76.8%)           80 (67.8%)
```

```
##    - 1         546 (23.2%)           38 (32.2%)
##  DYS                                              0.000
##    - 0        2285 (97.0%)          100 (84.7%)
##    - 1          71 ( 3.0%)           18 (15.3%)
##  IHD                                              0.476
##    - 0        2340 (99.3%)          116 (98.3%)
##    - 1          16 ( 0.7%)            2 ( 1.7%)
## ------------------------------------------------------------
```

1.2 Baseline characteristics (total)

```
tot1 <- final_db %>% mutate(tmp=1)
tot2 <- final_db %>% mutate(tmp=2)
tot3 <- rbind(tot1,tot2)

mytable(tmp ~ HTN + DATEDIFF + AGE + SEX + SES + REGION + BMI + CCI + DYS + IHD, data=tot3, metho
```

```
##
##               Descriptive Statistics by 'tmp'
## ------------------------------------------------------------
##                     1                    2            p
##                  (N=2474)             (N=2474)
## ------------------------------------------------------------
##  HTN                                                1.000
##    - 0        2284 (92.3%)          2284 (92.3%)
##    - 1         190 ( 7.7%)           190 ( 7.7%)
##  DATEDIFF 1656.0 [811.0;2458.0] 1656.0 [811.0;2458.0] 1.000
##  AGE        36.0 [22.0;50.0]      36.0 [22.0;50.0]    1.000
##  SEX                                                1.000
##    - 1        1240 (50.1%)          1240 (50.1%)
##    - 2        1234 (49.9%)          1234 (49.9%)
##  SES                                                1.000
##    - 1         697 (29.3%)           697 (29.3%)
##    - 2         743 (31.3%)           743 (31.3%)
##    - 3         935 (39.4%)           935 (39.4%)
##  REGION                                             1.000
##    - 1        1218 (49.5%)          1218 (49.5%)
##    - 2         514 (20.9%)           514 (20.9%)
##    - 3         729 (29.6%)           729 (29.6%)
##  BMI        23.2 [21.0;25.3]      23.2 [21.0;25.3]    1.000
##  CCI                                                1.000
##    - 0        1890 (76.4%)          1890 (76.4%)
##    - 1         584 (23.6%)           584 (23.6%)
```

```
##  DYS                                                          1.000
##    - 0        2385 (96.4%)          2385 (96.4%)
##    - 1         89 ( 3.6%)            89 ( 3.6%)
##  IHD                                                          1.000
##    - 0        2456 (99.3%)          2456 (99.3%)
##    - 1         18 ( 0.7%)            18 ( 0.7%)
## ------------------------------------------------------------------
```

Chapter 2

Multiple imputation

Multiple imputation is a general approach to the problem of missing data. It aims to allow for the uncertainty about the missing data by creating several different plausible imputed data sets and appropriately combining results obtained from each of them.

Multiple imputation using chained equations (MICE) were performed to generate 10 imputed datasets. For the imputation model, predictive mean matching was used for continuous data and logistic regression was used for binary data.

Using data **final_db**
Outcome variable : **HTN**
Follow-up period : **DATEDIFF**
Exposure variable : **DM**
Covariates : **Age, Sex, SES, Region, BMI, CCI, Comorbidities(Dyslipidemia, Ischemic heart disease)**

```
## load library
library(mice)
library(dplyr)
```

```
## load data
final_db <- read.csv('Data/final_db.csv', header=T)
```

2.1 The number of missing values

```
na_count <- function(data){
  num.na <- colSums(is.na(data))
  per.na <- paste0(round(colSums(is.na(data))/nrow(data) *100,2),"%")

  return(data.frame(missing=paste0(num.na,"(",per.na,")"),row.names = names(num.na)))
}

na_count(final_db)
```

```
##               missing
## RN_INDI         0(0%)
## DM              0(0%)
## INDEX_DT        0(0%)
## HTN             0(0%)
## FU_DT           0(0%)
## AGE             0(0%)
## SEX             0(0%)
## SES            99(4%)
## REGION      13(0.53%)
## BMI       1565(63.26%)
## CCI             0(0%)
## DYS             0(0%)
## IHD             0(0%)
## DATEDIFF        0(0%)
```

- Use **mice()** function in **mice** package to deal with missing data.
  - m=10 refers to the number of imputed datasets. Five is the default value.
  - Extract imputed data sets using **compleate()** function

2.2 Imputation for missing values

```
## Exclude subject ID, index date before imputation
dat_mice <- final_db %>% select(-RN_INDI, -INDEX_DT, -FU_DT)
dat_imp <- mice(dat_mice, m=10, seed=1)
```

```
##
##  iter imp variable
```

```
##   1   1  SES  REGION  BMI
##   1   2  SES  REGION  BMI
##   1   3  SES  REGION  BMI
##   1   4  SES  REGION  BMI
##   1   5  SES  REGION  BMI
##   1   6  SES  REGION  BMI
##   1   7  SES  REGION  BMI
##   1   8  SES  REGION  BMI
##   1   9  SES  REGION  BMI
##   1   10  SES  REGION  BMI
##   2   1  SES  REGION  BMI
##   2   2  SES  REGION  BMI
##   2   3  SES  REGION  BMI
##   2   4  SES  REGION  BMI
##   2   5  SES  REGION  BMI
##   2   6  SES  REGION  BMI
##   2   7  SES  REGION  BMI
##   2   8  SES  REGION  BMI
##   2   9  SES  REGION  BMI
##   2   10  SES  REGION  BMI
##   3   1  SES  REGION  BMI
##   3   2  SES  REGION  BMI
##   3   3  SES  REGION  BMI
##   3   4  SES  REGION  BMI
##   3   5  SES  REGION  BMI
##   3   6  SES  REGION  BMI
##   3   7  SES  REGION  BMI
##   3   8  SES  REGION  BMI
##   3   9  SES  REGION  BMI
##   3   10  SES  REGION  BMI
##   4   1  SES  REGION  BMI
##   4   2  SES  REGION  BMI
##   4   3  SES  REGION  BMI
##   4   4  SES  REGION  BMI
##   4   5  SES  REGION  BMI
##   4   6  SES  REGION  BMI
##   4   7  SES  REGION  BMI
##   4   8  SES  REGION  BMI
##   4   9  SES  REGION  BMI
##   4   10  SES  REGION  BMI
##   5   1  SES  REGION  BMI
##   5   2  SES  REGION  BMI
##   5   3  SES  REGION  BMI
##   5   4  SES  REGION  BMI
##   5   5  SES  REGION  BMI
##   5   6  SES  REGION  BMI
```

```
##   5   7  SES  REGION  BMI
##   5   8  SES  REGION  BMI
##   5   9  SES  REGION  BMI
##   5   10  SES  REGION  BMI
```

```
## Create 10 imputed data
for (i in 1:dat_imp$m){
  z <- assign(paste0('dat_imp',i),complete(dat_imp,i))
  assign(paste0('dat_imp',i),cbind(z,final_db %>% select(RN_INDI)))
}

## list of 10 imputed data
dat_imp_list <- list(dat_imp1,dat_imp2,dat_imp3,dat_imp4,dat_imp5,dat_imp6,dat_imp7,dat
```

```
## Save multiple imputation result
save(dat_imp,file="Data/dat_imp.RData")
## Save list for imputed data
save(dat_imp_list,file="Data/dat_imp_list.RData")
```

Chapter 3

Propensity Score Matching

Covariate balance

Covariate balance is the degree to which the distribution of covariates is similar across levels of the treatment.
SMD(Standardized Mean Difference) is the most widely used statistic for the assessment of balance after PSM.

SMD for continuous variables :

$$SMD = \frac{\bar{X}_1 - \bar{X}_2}{\sqrt{(S_1^2 + S_2^2)/2}}$$

- $\bar{X}_1$ and $\bar{X}_2$ are sample mean for the treated and control groups.
- S_1^2 and S_2^2 are sample variance for the treated and control groups.

SMD for binary variables :

$$SMD = \frac{\hat{p}_1 - \hat{p}_2}{\sqrt{[\hat{p}_1(1-\hat{p}_1) + \hat{p}_2(1-\hat{p}_2)]/2}}$$

- $\hat{p}_1$ and $\hat{p}_2$ are prevalence of binary variables in the treated and control groups.

If the SMD after matching is less than **0.1**, it is determined that the difference by the covariates between the two groups is negligible.

Using list **dat_imp_list**
Outcome variable : **HTN**
Follow-up period : **DATEDIFF**
Exposure variable : **DM**
Covariates : **Age, Sex, SES, Region, BMI, CCI, Comorbidities(Dyslipidemia, Ischemic hear**

```
## load library
library(MatchIt)
library(dplyr)
source("cobalt_3.9.0.R")
```

```
## load data
load("Data/dat_imp_list.RData")
final_com <- read.csv('Data/final_com.csv', header=T)
```

```
## Formula
formula.mat <- formula(DM ~ AGE + SEX + SES + REGION + BMI + CCI + DYS + IHD)
```

- Use **matchit()** function in **MatchIt** package to create treatment and control groups balanced on included covariates.
  - method='nearest' : nearest neighbor matching on the propensity score
  - ratio=k : the number of controls matched to each treated unit for k:1 matching
  - caliper : Units whose propensity score difference is larger than the caliper will not be paired, and some treated units may therefore not receive a match.

3.1 Complete data version

3.1.1 1:5 nearest matching

caliper: 0.4

```
## Optimal caliper
ps <- glm(formula.mat,data=final_com, family = 'binomial')
ps$pscore<- predict(ps, type='link')
0.2*sd(ps$pscore)
```

```
## [1] 0.2132231
```

```
set.seed(1)
mat <- matchit(formula.mat, method = 'nearest', data=final_com, ratio=5, caliper=0.4)
matdat <- match.data(mat) %>% select(-subclass) #   R   subclass

# adding an matching index as a subclass to matdat and store as dat_mat
# as.numeric(tmp[,1:ratio]), rep(c(1:nrow(tmp)),ratio+1)
tmp <- na.omit(mat$match.matrix)
matid <- data.frame(rowid=c(as.numeric(rownames(tmp)), as.numeric(tmp[,1:5])), subclass=rep(c(1:r
matid$RN_INDI <- final_com$RN_INDI[matid$row]
dat_mat <- matdat %>% left_join(matid %>% select(RN_INDI,subclass),by = 'RN_INDI') %>% filter(is.
```

```
## Save matching data
save(dat_mat,file="Data/dat_mat.RData")
```

3.1.2 Balance check

```
bal.ch <- function(before_data, after_data, group){

  group<-deparse(substitute(group))

  # SMD before matching
  bal_check_un <- bal.tab.data.frame(before_data[covariates], treat=before_data[,group], binary='
  un <- abs(bal_check_un$Balance$Diff.Un)

  # SMD after matching
  bal_check_adj <- bal.tab.data.frame(after_data[covariates], treat=after_data[,group], binary="s
  adj <- abs(bal_check_adj$Balance$Diff.Un)

  bal.res <- data.frame(un=round(un,3),adj=round(adj,3))
  rownames(bal.res) <- rownames(bal_check_un$Balance)
  return(bal.res)
}

covariates <- c("AGE","SEX","SES","REGION","BMI","CCI","DYS","IHD")
```

bal.ch(before_matching_data, after_matching_data, group variable)

```
bal.ch(final_com, dat_mat, DM)
```

```
##           un   adj
## AGE    0.997 0.001
## SEX_2  0.029 0.053
## SES    0.159 0.210
## REGION 0.087 0.058
## BMI    0.273 0.176
## CCI    0.168 0.194
## DYS    0.519 0.172
## IHD    0.036 0.093
```

3.2 Missing data version

3.2.1 1:3 nearest matching

caliper: 0.3, 0.35, 0.4

```
## Optimal caliper
opt.clp <- c()
for (i in 1:length(dat_imp_list)){
  ps <- glm(formula.mat,data=dat_imp_list[[i]], family = 'binomial')
  ps$pscore<- predict(ps, type='link')
  opt.clp <- c(opt.clp, 0.2*sd(ps$pscore))
}
opt.clp; mean(opt.clp)
```

```
##  [1] 0.2772334 0.2871682 0.2815519 0.2695950 0.2773369 0.2727781 0.2734195
##  [8] 0.2794713 0.2822619 0.2712372
```

```
## [1] 0.2772053
```

```
caliper <- c(0.3,0.35,0.4)

for (i in 1:length(dat_imp_list)){
  for (j in caliper){
    set.seed(1)
    mat <- matchit(formula.mat, method = 'nearest', data=dat_imp_list[[i]], ratio=3, ca
    matdat <- match.data(mat) %>% select(-subclass) #   R   subclass

    # adding an matching index as a subclass to matdat and store as dat_mati_j
    # as.numeric(tmp[,1:ratio]), rep(c(1:nrow(tmp)),ratio+1)
    tmp <- na.omit(mat$match.matrix)
    matid <- data.frame(rowid=c(as.numeric(rownames(tmp)), as.numeric(tmp[,1:3])), subc
    matid$RN_INDI <- dat_imp_list[[i]]$RN_INDI[matid$row]
```

```
    assign(paste0('dat_mat',i,"_",j), matdat %>% left_join(matid %>% select(RN_INDI,subclass),by
  }
}
```

```
## list of 10 matched data
dat_mat_list_0.3 <- list(dat_mat1_0.3,dat_mat2_0.3,dat_mat3_0.3,dat_mat4_0.3,dat_mat5_0.3,dat_mat
dat_mat_list_0.35 <- list(dat_mat1_0.35,dat_mat2_0.35,dat_mat3_0.35,dat_mat4_0.35,dat_mat5_0.35,d
dat_mat_list_0.4 <- list(dat_mat1_0.4,dat_mat2_0.4,dat_mat3_0.4,dat_mat4_0.4,dat_mat5_0.4,dat_mat

## Save list for matched data
save(dat_mat_list_0.3,file="Data/dat_mat_list_0.3.RData")
save(dat_mat_list_0.35,file="Data/dat_mat_list_0.35.RData")
save(dat_mat_list_0.4,file="Data/dat_mat_list_0.4.RData")
```

3.2.2 Balance check

```
bal.ch <- function(dat_imp_list, dat_mat_list, group){

  group<-deparse(substitute(group))

  # SMD before matching
  bal_check_un <- dat_imp_list %>% lapply(function(x){
    bal.tab.data.frame(x[covariates],
                       treat=x[,group], binary="std", s.d.denom = "pooled")})
  un <- sapply(bal_check_un, function(x) (abs(x$Balance$Diff.Un)))
  rownames(un) <- rownames(bal_check_un[[1]]$Balance)

  # SMD after matching
  bal_check_adj <- dat_mat_list %>% lapply(function(x){
    bal.tab.data.frame(x[covariates],
                       treat=x[,group], binary="std", s.d.denom = "pooled")})
  adj <- sapply(bal_check_adj, function(x) (abs(x$Balance$Diff.Un)))
  rownames(adj) <- rownames(bal_check_adj[[1]]$Balance)

  bal.res <- list(un=apply(un, 1, summary), adj=apply(adj, 1, summary))
  return(data.frame(un=round(bal.res$un[6,],3),adj=round(bal.res$adj[6,],3)))
}

covariates <- c("AGE","SEX","SES","REGION","BMI","CCI","DYS","IHD")
```

bal.ch(before_matching_list, after_matching_list, group variable)

```
bal.ch(dat_imp_list, dat_mat_list_0.3, DM)
```

```
##           un   adj
## AGE    1.450 0.040
## SEX_2  0.020 0.112
## SES    0.153 0.089
## REGION 0.009 0.095
## BMI    0.458 0.157
## CCI    0.203 0.080
## DYS    0.435 0.071
## IHD    0.094 0.123
```

```
bal.ch(dat_imp_list, dat_mat_list_0.35, DM)
```

```
##           un   adj
## AGE    1.450 0.040
## SEX_2  0.020 0.105
## SES    0.153 0.089
## REGION 0.009 0.094
## BMI    0.458 0.144
## CCI    0.203 0.082
## DYS    0.435 0.106
## IHD    0.094 0.123
```

```
bal.ch(dat_imp_list, dat_mat_list_0.4, DM)
```

```
##           un   adj
## AGE    1.450 0.040
## SEX_2  0.020 0.110
## SES    0.153 0.089
## REGION 0.009 0.102
## BMI    0.458 0.121
## CCI    0.203 0.092
## DYS    0.435 0.105
## IHD    0.094 0.121
```

Chapter 4

Propensity score weighting

Using data **final_com** (complete data version)
Using object **dat_imp** (missing data version, imputaion result)
Outcome variable : **HTN**
Follow-up period : **DATEDIFF**
Exposure variable : **DM**
Covariates : **Age, Sex, SES, Region, BMI, CCI, Comorbidities(Dyslipidemia, Ischemic heart disease)**

```
## load library
library(dplyr)
source("cobalt_3.9.0.R")
```

```
## load data
final_com <- read.csv('Data/final_com.csv', header=T)
load("Data/dat_imp.RData")
```

```
## Formula
formula.wt <- formula(DM ~ AGE + SEX + SES + REGION + BMI + CCI + DYS + IHD)
```

4.1 Complete data version

4.1.1 Inverse Probability Weighting

- s.weights : Stabilized weights

- trim.weights : Trimming weights

```
## propensity score function
ps_wt <- function(data, ps.formula, group, trt, psweight){

  modellist <- glm(formula=ps.formula, data=data, family="binomial", x=TRUE, y=TRUE)
  cov = data.frame(modellist$x[,-1])
  p.score <- modellist$fitted.values
  weights <- (data[,group]==trt)/p.score + (data[,group]!=trt)/(1-p.score)
  pt <- sum(data[,group]==trt)/nrow(cov)
  s.weights <- pt*(data[,group]==trt)/p.score + (1-pt)*(data[,group]!=trt)/(1-p.score)
  trim.p.score <-ifelse(p.score<0.01, 0.01,p.score)
  trim.p.score <- ifelse(p.score>0.99, 0.99,p.score)
  trim.weights <- (data[,group]==trt)/trim.p.score + (data[,group]!=trt)/(1-trim.p.sco

  ballist <- bal.tab.data.frame(cov, treat=modellist$y,
                                weights=get(paste0(psweight)), s.d.denom = "pooled", b
                                continuous="std", disp.means=TRUE, disp.sds=TRUE)

  return(list(modellist = modellist, weights=weights, s.weights=s.weights, trim.weights
}
```

```
## Propensity score for imputed data sets
ps1 <- ps_wt(final_com, formula.wt, "DM", "1", "s.weights")
```

```
## Assuming "weighting". If not, specify with an argument to method.
```

```
## Complete weighted data
dat_wt <- final_com
dat_wt$weights <- ps1$s.weights
head(dat_wt)
```

```
##   RN_INDI DM   INDEX_DT HTN      FU_DT AGE SEX SES REGION  BMI CCI DYS IHD
## 1 1011725  0 2006-03-27   0 2013-01-01  36   1   1      2 23.7   0   0   0
## 2 1042143  0 2006-11-04   0 2013-01-01  56   1   3      2 21.9   0   0   0
## 3 1049401  1 2006-01-19   0 2013-01-01  66   1   2      3 27.1   0   0   0
## 4 1049841  0 2008-07-04   0 2013-01-01  48   2   3      3 21.0   0   0   0
## 5 1050697  0 2010-03-31   0 2013-01-01  50   1   3      3 24.9   1   0   0
## 6 1063538  0 2010-12-14   0 2013-01-01  50   2   1      1 19.3   0   0   0
##   DATEDIFF   weights
## 1     2472 0.9535391
## 2     2250 1.0108264
## 3     2539 0.3654591
## 4     1642 0.9688670
```

```
## 5     1007 0.9716872
## 6      749 0.9704260
```

```
## Save propensity score
save(dat_wt,file="Data/dat_wt.RData")
```

4.1.2 Balance check

```
## Banlance check function
bal_wt<- function(obj){
  un <- round(abs(obj$ballist$Balance$Diff.Un),3)
  adj <- round(abs(obj$ballist$Balance$Diff.Adj),3)
  bal_df <- data.frame(un=un,adj=adj)
  rownames(bal_df) <- rownames(obj$ballist$Balance)

  return(bal_df)
}
```

```
## Balance check across imputed datasets
bal_wt(ps1)
```

```
##           un   adj
## AGE    0.997 0.332
## SEX_2  0.029 0.159
## SES    0.159 0.222
## REGION 0.087 0.006
## BMI    0.273 0.208
## CCI    0.168 0.347
## DYS    0.519 0.038
## IHD    0.036 0.074
```

4.2 Missing data version

4.2.1 Inverse Probability Weighting

- s.weights : Stabilized weights
- trim.weights : Trimming weights

```
## propensity score function
ps_impute <- function(datasets, ps.formula, group, trt, psweight){
  modellist <- vector("list", 10)
  ballist <- vector("list", 10)
  weights <- s.weights <- trim.weights <- vector("list", 10)
  for(i in 1:10){
    tmp.dat <- mice::complete(datasets, i)

    ### propensity score
    modellist[[i]] <- glm(formula=ps.formula, data=tmp.dat, family="binomial", x=TRUE,
    cov = data.frame(modellist[[i]]$x[, -1])
    p.score <- modellist[[i]]$fitted.values
    weights[[i]] <- (tmp.dat[,group]==trt)/p.score + (tmp.dat[,group]!=trt)/(1-p.score)
    pt <- sum(tmp.dat[,group]==trt)/nrow(cov)
    s.weights[[i]] <- pt*(tmp.dat[,group]==trt)/p.score + (1-pt)*(tmp.dat[,group]!=trt)
    trim.p.score <-ifelse(p.score<0.01, 0.01,p.score)
    trim.p.score <- ifelse(p.score>0.99, 0.99,p.score)
    trim.weights[[i]] <- (tmp.dat[,group]==trt)/trim.p.score + (tmp.dat[,group]!=trt)/

    ballist[[i]] <- bal.tab.data.frame(cov, treat=modellist[[i]]$y,
                                       weights=get(paste0(psweight))[[i]], s.d.denom =
                                       continuous="std", disp.means=TRUE, disp.sds=TRU
  }

  return(list(modellist = modellist, weights=weights, s.weights=s.weights, trim.weights
}
```

```
## Complete weighted data function
complete.wdata <- function(object, data, weights) {

  lapply(seq_len(data$m), function(j, object, data, weights) {
    out <- mice::complete(data, j)
    modelvars <- weights
    for( v in modelvars)
      out[[v]] <- object[[v]][[j]]

    out}, object = object, data=data, weights=weights)
}
```

```
## Propensity score for imputed data sets
ps1 <- ps_impute(dat_imp, formula.wt, "DM", "1", "s.weights")
```

```
## Assuming "weighting". If not, specify with an argument to method.
## Assuming "weighting". If not, specify with an argument to method.
## Assuming "weighting". If not, specify with an argument to method.
```

```
## Assuming "weighting". If not, specify with an argument to method.
## Assuming "weighting". If not, specify with an argument to method.
## Assuming "weighting". If not, specify with an argument to method.
## Assuming "weighting". If not, specify with an argument to method.
## Assuming "weighting". If not, specify with an argument to method.
## Assuming "weighting". If not, specify with an argument to method.
## Assuming "weighting". If not, specify with an argument to method.
```

```
## Complete weighted data
data_wt <- complete.wdata(ps1, dat_imp, "s.weights")
head(data_wt[[1]])
```

```
##   DM HTN AGE SEX SES REGION  BMI CCI DYS IHD DATEDIFF s.weights
## 1  0   0  20   1   2      3 25.2   1   0   0     1001 0.9587127
## 2  0   0  46   1   2      1 20.0   0   0   0     2542 0.9805237
## 3  0   0  16   1   2      3 22.1   0   0   0     2542 0.9559439
## 4  0   0  36   1   1      2 23.7   0   0   0     2472 0.9688151
## 5  0   1  56   2   3      1 20.5   0   0   0     2449 1.0216980
## 6  0   0  20   2   2      1 16.4   1   0   0     1095 0.9555949
```

```
## Save propensity score
save(data_wt,file="Data/data_wt.RData")
```

4.2.2 Balance check

```
## Banlance check function
bal.comb <- function(obj){
  unadj <- sapply(obj$ballist, function(x) (abs(x$Balance$Diff.Un)))
  adj <- sapply(obj$ballist, function(x) (abs(x$Balance$Diff.Adj)))
  rownames(unadj) <- rownames(adj) <- rownames(obj$ballist[[1]]$Balance)

  bal.res <- list(un=round(apply(unadj, 1, summary),3), adj=round(apply(adj, 1, summary),3))
  return(data.frame(un=round(bal.res$un[6,],3),adj=round(bal.res$adj[6,],3)))
}
```

```
## Balance check across imputed datasets
bal.comb(ps1)
```

```
##           un   adj
## AGE    1.450 0.636
## SEX_2  0.020 0.161
## SES    0.153 0.195
```

```
## REGION 0.009 0.058
## BMI    0.458 0.180
## CCI    0.203 0.104
## DYS    0.435 0.064
## IHD    0.094 0.050
```

Chapter 5

Incidence rate

Using data **final_com, dat_mat** (complete data version)
Using list **dat_mat_list, dat_imp_list** (missing data version)
Outcome variable : **HTN**
Follow-up period : **DATEDIFF**
Exposure variable : **DM**
Covariates : **Age, Sex, SES, Region, BMI, CCI, Comorbidities(Dyslipidemia, Ischemic heart disease)**

```
## load library
library(dplyr)
```

```
## load data
final_com <- read.csv('Data/final_com.csv', header=T)
load("Data/dat_imp_list.RData")
load("Data/dat_mat_list_0.4.RData")
load("Data/dat_mat.RData")
```

5.1 Incidence rate function

```
IR <- function(data, group, outcome, time, tconv, unit){
  attach(data)
  n <- length(group)
  nrisk <- c(table(group),sum(table(group)))
  names(nrisk) <- c("ctr","trt","total")
```

```
  nevent <- c(table(group, outcome)[,2],sum(table(group, outcome)[,2]))
  ptime <- aggregate(time, list(group), FUN=sum)
  ptime[3,] <- sum(ptime$x)

  if (tconv %in% c(1,2,3)){
    if (tconv==1) ptime.conv = ptime$x
    else if (tconv==2) ptime.conv = ptime$x/12
    else ptime.conv = ptime$x/365.25
  }else {
    stop(paste(tconv, "is not an acceptable entry to tconv"), call. = FALSE)
  }

  ir <- nevent/ptime.conv*unit
  res <- data.frame(nrisk, nevent, ptime2=ptime.conv/unit, py=round(ptime.conv,2), ir=r
  ci <- t(sapply(1:nrow(res), function(i) exp(confint.default(glm(nevent~offset(log(pti
  colnames(ci) <- c("lir", "uir")
  res <- cbind(res, round(ci,2))
  detach(data)
  return(res)
}
```

- IR(data, group, status, time, tconv, unit)
- tconv: Used time criteria in dataset
  - 1: year / 2: month / 3: day
- unit: Person year criteria (ex. unit = 1000 for incidence rate per 1000PY)

5.1.1 Complete data _ General version

```
IR(final_com, DM, HTN, DATEDIFF, 3, 1000)
```

```
##       nrisk nevent    ptime2      py     ir   lir    uir
## ctr     825     80 3.4564709 3456.47  23.14 18.59  28.82
## trt      61     23 0.1906311  190.63 120.65 80.18 181.56
## total   886    103 3.6471020 3647.10  28.24 23.28  34.26
```

5.1.2 Complete data _ Matching version

```
IR(dat_mat, DM, HTN, DATEDIFF, 3, 1000)
```

```
##       nrisk nevent   ptime2      py     ir   lir    uir
## ctr     230     39 0.886883  886.88  43.97 32.13  60.19
## trt      46     16 0.159436  159.44 100.35 61.48 163.81
## total   276     55 1.046319 1046.32  52.57 40.36  68.47
```

5.1.3 Missing data _ Before Matching

```
IR(dat_imp_list[[1]], DM, HTN, DATEDIFF, 3, 1000)
```

```
##       nrisk nevent     ptime2       py     ir    lir    uir
## ctr    2356    141 10.0459849 10045.98  14.04  11.90  16.55
## trt     118     49  0.3551129   355.11 137.98 104.29 182.57
## total  2474    190 10.4010979 10401.10  18.27  15.85  21.06
```

5.1.4 Missing data _ Matching version

```
IR(dat_mat_list_0.4[[1]], DM, HTN, DATEDIFF, 3, 1000)
```

```
##       nrisk nevent    ptime2      py     ir   lir    uir
## ctr     336     51 1.2416099 1241.61  41.08 31.22  54.05
## trt     112     46 0.3449254  344.93 133.36 99.89 178.05
## total   448     97 1.5865352 1586.54  61.14 50.11  74.60
```

Chapter 6

Kaplan-Meier curve

6.1 The Kaplan Meier estimator

The Kaplan-Meier estimator is used to estimate the survival function. The visual representation of this function is usually called the Kaplan-Meier curve, and it shows what the probability of an event (for example, survival) is at a certain time interval.
We compared two groups in this study. : DM(case) vs NonDM(control)

- Survival function :

$$S(t) = P(T > t)$$

- Kaplan-Meier estimator :

$$\hat{S}(t) = \prod_{j:t_j \leq t} \left(1 - \frac{d_j}{n_j}\right) = \prod_{s \leq t} \left(1 - \frac{\triangle N(s)}{Y(s)}\right)$$

d_j : the number of individuals who experience the event at t_j.
n_j : the number individuals at risk at t_j.

* Using data **final_com, dat_mat, dat_wt** (complete data version)
Using list **dat_mat_list, dat_imp_list** (missing data version)
Outcome variable : **HTN**
Follow-up period : **DATEDIFF**
Exposure variable : **DM**
Covariates : **Age, Sex, SES, Region, BMI, CCI, Comorbidities(Dyslipidemia, Ischemic hear**

```
## load library
library(survival)
library(survminer)
library(ggsci)
library(RISCA) #     library(IPWsurvival)
```

```
## load data
final_com <- read.csv('Data/final_com.csv', header=T)
load("Data/dat_mat.RData")
load("Data/dat_wt.RData")
load("Data/dat_mat_list_0.4.RData")
load("Data/dat_imp_list.RData")
```

```
## Formula
formula.wt <- formula(DM ~ AGE + SEX + SES + REGION + BMI + CCI + DYS + IHD)
```

6.2 Complete data version

6.2.1 Kaplan-Meier curve function

- Use **survfit()**, **ggsurvplot()** function in **survival, survminer** package to plot survival curve.

```
gg_km <- function(data){
  p <- ggsurvplot(fit = fit_km,
                  title = "Kaplan-Meier Curve",
                  ggtheme = theme(axis.line = element_line(color="black"),
                                  panel.background = element_blank(),
                                  plot.title = element_text(hjust = 0, size=16, face =
                                  plot.margin = unit(c(5,5,5,5), "mm"),
                                  axis.title.y = element_text(margin=margin(r=13)),
                                  axis.title.x = element_text(margin=margin(t=10)),
```

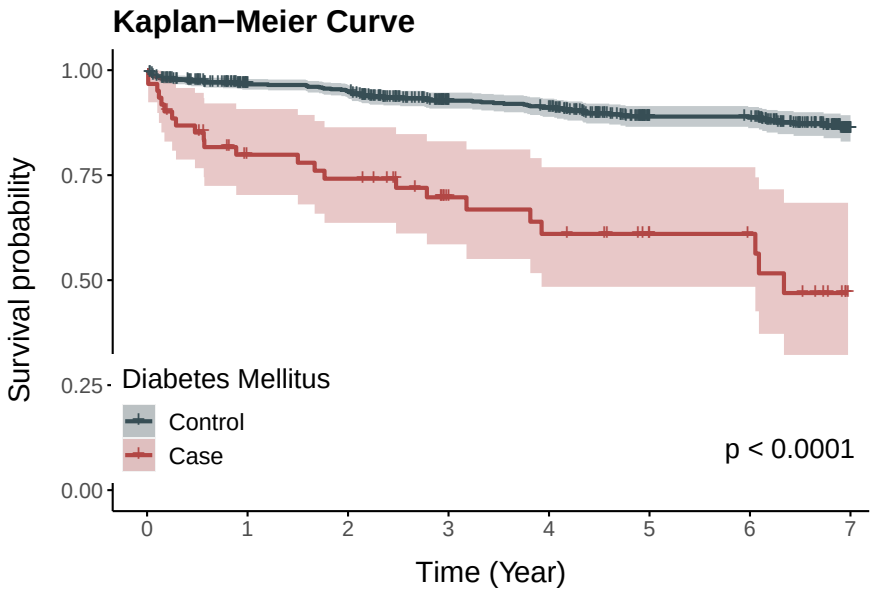
```
                                    legend.text = element_text(size = 11.5, color="black"),
                                    legend.title = element_text(size=13.5)),
                  palette = c("#374E55FF", "#B24745FF"),
                  xlim = c(0,2500),
                  ylim=c(0,1),
                  ### Censor Details
                  censor = TRUE, # logical value. If TRUE, censors will be drawn
                  censor.shape="+", # Default value is "+", a sensible choice is "|".
                  censor.size = 4,
                  ### Confidence Interval
                  conf.int = TRUE, # To Remove conf intervals use FALSE
                  ### Format Axes (changes x,y axis label)
                  xlab = "Time (Year)",
                  ylab = "Survival probability",
                  font.x=c(15),
                  font.y=c(15),
                  font.xtickslab=c(11,"plain"),
                  font.ytickslab=c(11,"plain"),
                  ### Format Legend
                  legend.title = "Diabetes Mellitus",
                  legend.labs = c("Control", "Case"), # Change the Strata Legend
                  legend = c(0.15,0.2), # c(0,0) corresponds to the "bottom left" and c(1,1) cor
                  ### Risk Table
                  risk.table = F, # To Remove risk table use FALSE
                  tables.height = 0.25, # Adjusts the height of the risk table
                  tables.col = "black",
                  tables.y.text = FALSE,
                  tables.y.text.col = TRUE,
                  risk.table.fontsize = 4.5,
                  tables.theme = theme_cleantable(font.y=12),
                  ### p-value details
                  pval = TRUE,
                  pval.size = 5,
                  pval.coord = c(2100,0.1)
  )

  p$plot <- p$plot + scale_x_continuous(breaks=seq(0,2600,365.25), labels=0:7)
  return(p)
}
```

6.2.2 General version

```
fit_km <- survfit(Surv(DATEDIFF, HTN==1)~ DM, data=final_com)
gg_km(fit_km)
```

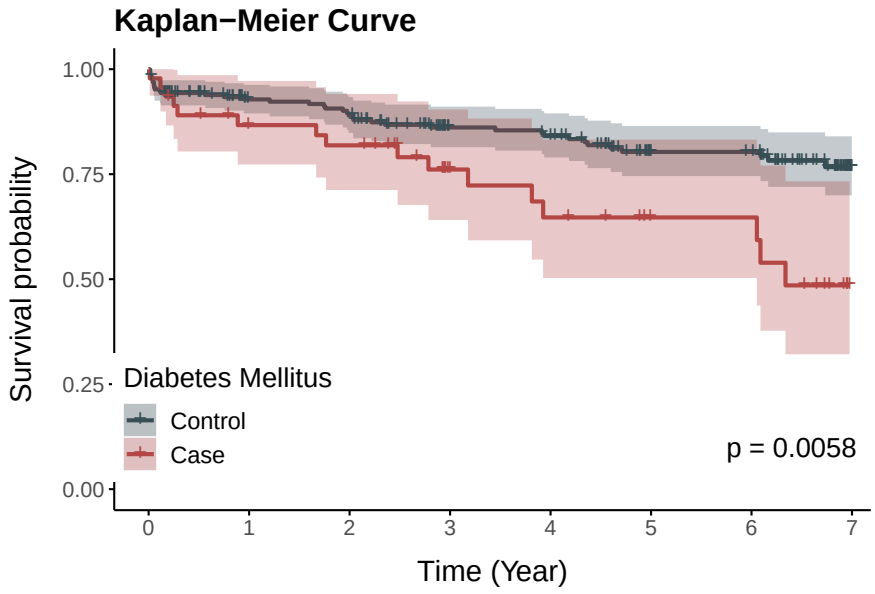
```
## Scale for x is already present.
## Adding another scale for x, which will replace the existing scale.
```



6.2.3 Matching version

```
fit_km <- survfit(Surv(DATEDIFF, HTN==1)~ DM, data=dat_mat, weight = weights)
gg_km(fit_km)
```

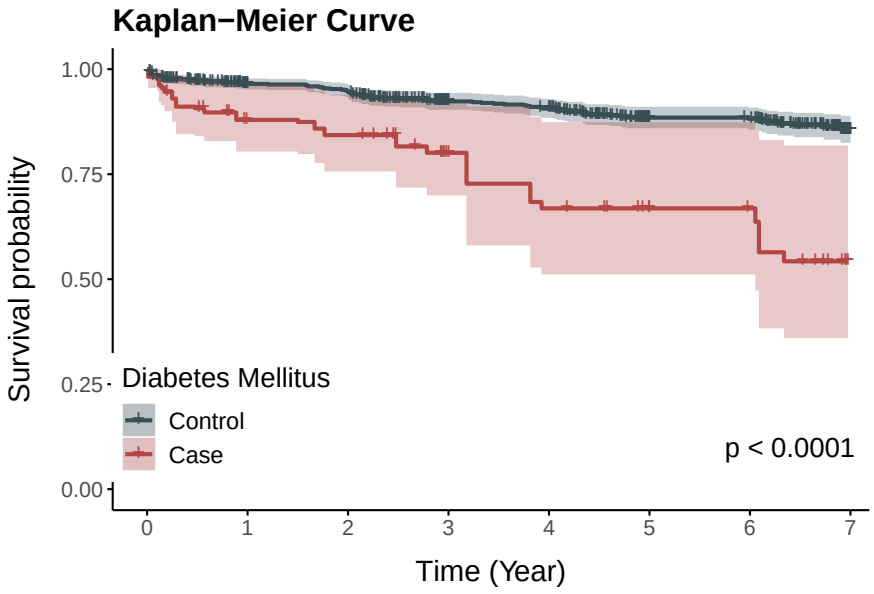
```
## Scale for x is already present.
## Adding another scale for x, which will replace the existing scale.
```



6.2.4 Weighting version

```
fit_km <- survfit(Surv(DATEDIFF, HTN==1)~ DM, data=dat_wt, weight = weights)
gg_km(fit_km)
```

```
## Scale for x is already present.
## Adding another scale for x, which will replace the existing scale.
```

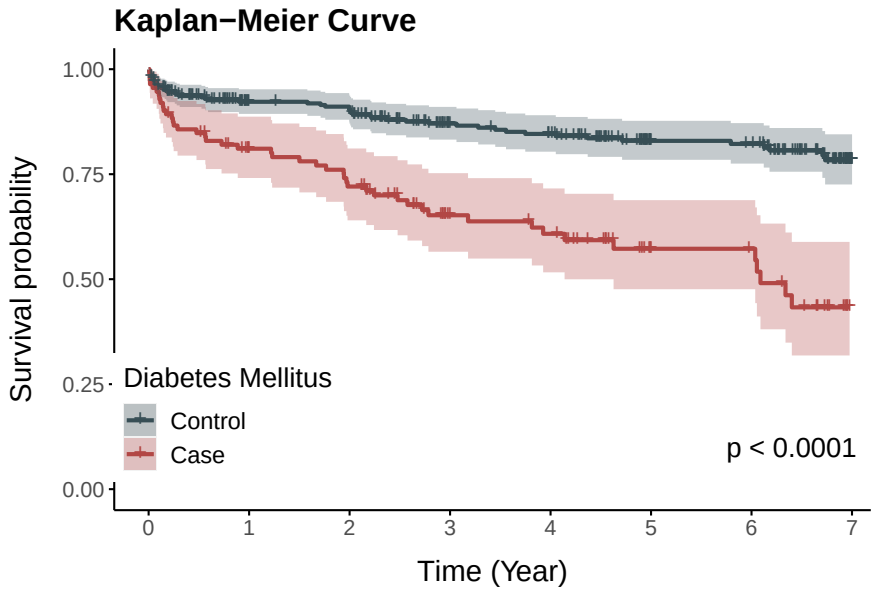


6.3 Missing data version

6.3.1 Matching version

```
fit_km <- survfit(Surv(DATEDIFF, HTN==1)~ DM, data=dat_mat_list_0.4[[1]])
gg_km(fit_km)
```

```
## Scale for x is already present.
## Adding another scale for x, which will replace the existing scale.
```



6.3.2 Weighting version

6.3.2.1 Weighted kaplan-meier estimates

- Use **ipw.survival()** function in **RISCA** package to estimate adjusted survival curves by weighting the individual contributions by the inverse of the probability to be in the group (IPW).

```
surv_res <- function(data, time, status, group, formula){
  p.score <- glm(formula=formula, data=data, family="binomial", x=TRUE, y=TRUE)$fitted.values
  wt <- (data[,group]=="1")/p.score + (data[,group]=="0")/(1-p.score)
  pt <- sum(data[,group]==1)/nrow(data)
  sw <- pt*(data[,group]==1) /p.score + (1-pt)*(data[,group]==0)/(1-p.score)
  res.akm <- ipw.survival(times=data[,time], failures=data[,status], variable=data[,group], weigh
  return(res.akm)
}

res.akm <- dat_imp_list %>% lapply(function(x) surv_res(x,"DATEDIFF","HTN","DM", formula.wt)$tabl
risk <- do.call("cbind",res.akm %>% lapply(function(x) x$n.risk))
event <- do.call("cbind",res.akm %>% lapply(function(x) x$n.event))
surv <- do.call("cbind",res.akm %>% lapply(function(x) x$survival))
km_res <- data.frame(time=res.akm[[1]]$times, strata=res.akm[[1]]$variable,
                     n.risk=apply(risk,1,mean),
```

```
                        n.event=apply(event,1,mean),
                        surv=apply(surv,1,mean))
```

```
head(km_res)
```

```
##   time strata   n.risk  n.event      surv
## 1    0      0 2360.020 0.000000 1.0000000
## 2    2      0 2360.020 2.099422 0.9991104
## 3    5      0 2357.921 1.062941 0.9986600
## 4    6      0 2353.958 3.045573 0.9973680
## 5    7      0 2350.912 1.035237 0.9969288
## 6    9      0 2349.877 4.227905 0.9951351
```

6.3.2.2 Kaplan-Meier curve by Using IPW

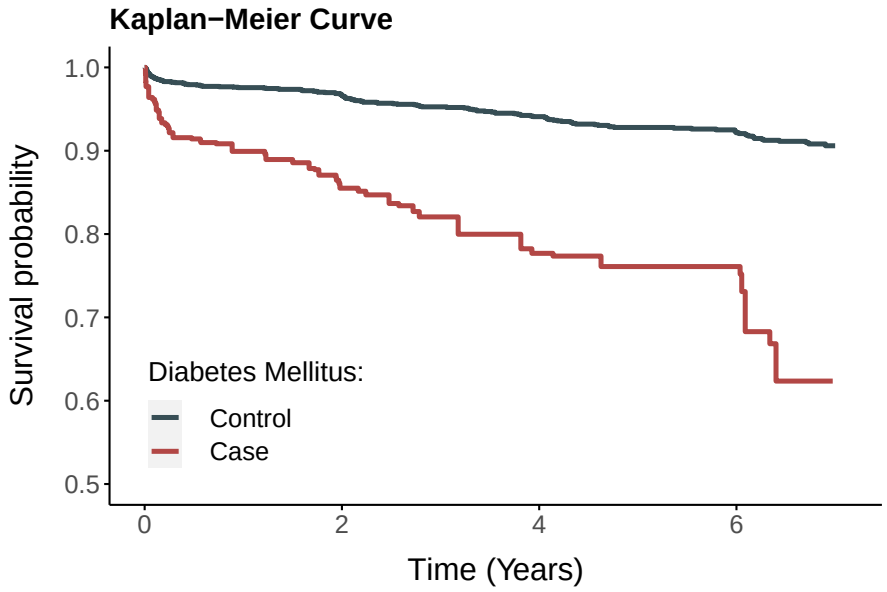
- Use **ggsurvplot_df()** function in **survminer** package to plot survival curve from any data frame containing the summary of survival curves.

```
ggsurv_plot <- ggsurvplot_df(km_res,
                             title = "Kaplan-Meier Curve",
                             ggtheme = theme(axis.line = element_line(color="black"),
                                             panel.background = element_blank(),
                                             plot.title = element_text(hjust = 0, size=
                                             plot.margin = unit(c(5,3,5,5), "mm"),
                                             axis.title.y = element_text(margin=margin
                                             axis.title.x = element_text(margin=margin
                                             legend.background =element_blank(),
                                             legend.text = element_text(size = 13, colo
                                             legend.title = element_text( size=13.5),
                                             legend.spacing.x = unit(0.5, 'cm'),
                                             legend.spacing.y = unit(0.3, 'cm')),
                             palette = c("#374E55FF", "#B24745FF"),
                             xlim = c(0,2600),
                             ylim = c(0.5,1),
                             size = 1.2,
                             ### Format Axes (changes x,y axis label)
                             xlab="Time (Years)",
                             font.x = c(16),
                             font.y = c(16),
                             font.tickslab = c(13),
                             ### Format Legend
                             legend.title = "Diabetes Mellitus:  ",
                             legend.labs = c("Control", "Case"), # Change the Strata L
```

```
                                legend = c(0.2,0.2)
                                ) + scale_x_continuous(breaks = c(0,730.5,1461,2191.5,2922,3652.5),
```

```
## Scale for x is already present.
## Adding another scale for x, which will replace the existing scale.
```

```
ggsurv_plot
```



Chapter 7

Logistic regression

Using data **final_com, dat_mat, dat_wt** (complete data version)
Using list **dat_mat_list, dat_imp_list** (missing data version)
Outcome variable : **HTN**
Follow-up period : **DATEDIFF**
Exposure variable : **DM**
Covariates : **Age, Sex, SES, Region, BMI, CCI, Comorbidities(Dyslipidemia, Ischemic heart disease)**

```
## load library
library(survival)
library(survminer)
library(ggsci)
library(RISCA) #     library(IPWsurvival)
```

```
## load data
final_com <- read.csv('Data/final_com.csv', header=T)
load("Data/dat_mat.RData")
load("Data/dat_wt.RData")
load("Data/dat_mat_list_0.4.RData")
load("Data/dat_imp_list.RData")
```

```
## Formula
formula.wt <- formula(DM ~ AGE + SEX + SES + REGION + BMI + CCI + DYS + IHD)
```
