Mass Spec Aging Data Analysis

Samuel Leung

2023-11-04

```
data <- read.csv("data/cardio_data_processed.csv")
head(data)
```

```
##   id   age gender height weight ap_hi ap_lo cholesterol gluc smoke alco active
## 1  0 18393      2    168     62   110    80           1    1     0    0      1
## 2  1 20228      1    156     85   140    90           3    1     0    0      1
## 3  2 18857      1    165     64   130    70           3    1     0    0      0
## 4  3 17623      2    169     82   150   100           1    1     0    0      1
## 5  4 17474      1    156     56   100    60           1    1     0    0      0
## 6  8 21914      1    151     67   120    80           2    2     0    0      0
##   cardio age_years      bmi          bp_category  bp_category_encoded
## 1      0        50 21.96712 Hypertension Stage 1 Hypertension Stage 1
## 2      1        55 34.92768 Hypertension Stage 2 Hypertension Stage 2
## 3      1        51 23.50781 Hypertension Stage 1 Hypertension Stage 1
## 4      1        48 28.71048 Hypertension Stage 2 Hypertension Stage 2
## 5      0        47 23.01118               Normal               Normal
## 6      0        60 29.38468 Hypertension Stage 1 Hypertension Stage 1
```

```
nrow(data)
```

```
## [1] 68205
```

Goal

- To estimate the total number of people with cardiovascular disease: $\hat{t}_{\text{cardio}}$
- To estimate the average blood pressure of the population: $\bar{u}_{\text{ap\_hi}}$, $\bar{u}_{\text{ap\_lo}}$

Sample Size Calculations:

We can't find the maximum variance from a continuous variable, we go through sample size calculations with the maximum variance possible from a binary variable: the total number of people with cardiovascular disease. We want the margin of error δ of our estimate to be within $\pm$ 1000 people, 19 times out of 20. Given the size of our population is $\sim$ 68000, we believe that an estimate within this margin of error range is reasonable.

Mathematical steps

Let X be the binary random variable that models the number of people with cardiovascular disease, with mean p and variance $p(1-p)$. Let $\bar{y}_S = \hat{p}$ be the estimate of the average number people with cardiovascular

disease. of We want to estimate the total number of people with cardiovascular disease, $\hat{t}_{p,\text{cardio}} = N\bar{y}_s$. The variance of this estimator will be $var(\hat{t}_{p,\text{cardio}}) = N^2\hat{p}(1-\hat{p})$. Going through our sample size calculations, we want to maximize variance, so we take $\hat{p} = 0.5$. Therefore, ignoring the finite population correction factor for now, we want to solve the following:

$$\begin{aligned} 1000 &\geq 1.96\sqrt{0.25N^2/n_0} \\ \implies \left(\frac{1000}{1.96}\right)^2 &\geq \frac{N^2}{4n_0} \\ \implies n_0 &\geq \left(\frac{N}{2} \cdot \frac{1.96}{1000}\right)^2 \end{aligned}$$

Converting this to code:

```
# add reasoning for why we chose MOE to be 1000

N <- nrow(data)
target <- 1000
n_0 <- (1.96 * N / (2*target))^2
print(paste0("sample size ignoring FPC: ", n_0))
```

```
## [1] "sample size ignoring FPC: 4467.70591281"
```

```
print(paste0("n_0 / N: ", n_0 / N))
```

```
## [1] "n_0 / N: 0.065504082"
```

Comparing the sample size calculated above with the total population size, we see that $\frac{n_0}{N} = 0.066 \geq 0.05$. So we cannot ignore the finite population correction factor in our sample size calculations. Factoring in the FPC, we would want to solve the following:

$$n = \frac{n_0}{1 + n_0/N}$$

Doing the calculations again with the FPC in mind with code:

```
n <- ceiling(n_0 / (1 + n_0 / N))
print(paste0("Sample Size with FPC: ", n))
```

```
## [1] "Sample Size with FPC: 4194"
```

So we use a sample size of 4194.

```
# set seed for reproducable results
set.seed(124)

# generate random sample
srs <- data[sample(1:nrow(data), n), ]

# double check size of sample
dim(srs)
```

```
## [1] 4194   17
```

```
fpc <- 1 - n / N
target_lst <- c("ap_hi", "ap_lo", "cardio")

ap_hi.est <- round(mean(srs$ap_hi), 3)
ap_hi.var <- fpc*var(srs$ap_hi) / n
ap_hi.se <- sqrt(ap_hi.var)
ap_hi.moe <- round(1.96*sqrt(ap_hi.var), 3)

ap_lo.est <- round(mean(srs$ap_lo), 3)
ap_lo.var <- fpc*var(srs$ap_lo) / n
ap_lo.se <- sqrt(ap_lo.var)
ap_lo.moe <- round(1.96*sqrt(ap_lo.var), 3)

cardio.p <- mean(srs$cardio)
cardio.est <- round(N*cardio.p, 3)
cardio.var <- fpc*(cardio.p)*(1-cardio.p) * N
cardio.se <- sqrt(cardio.var)
cardio.moe <- round(1.96*sqrt(cardio.var), 3)

vanilla.summary <- rbind(
  c(ap_hi.est, ap_hi.se),
  c(ap_lo.est, ap_lo.se),
  c(cardio.est, cardio.se)
)

colnames(vanilla.summary) <- c("est", "se")
rownames(vanilla.summary) <- paste("vanilla.", target_lst, sep="")

vanilla.summary
```

```
##                      est          se
## vanilla.ap_hi    126.314   0.2366057
## vanilla.ap_lo     81.373   0.1358974
## vanilla.cardio 33630.887 126.4898790
```

Regression and Ratio Estimates

To calculate regression and ratio estimates, we want to use an auxiliary variable with strong correlation. We do so by first calculating the correlation values for each of our target variables against the other numeric columns in our data set:

```
pop.cor <- cor(data[,2:15])
pop.cor[target_lst, !(colnames(pop.cor)%in%target_lst)]
```

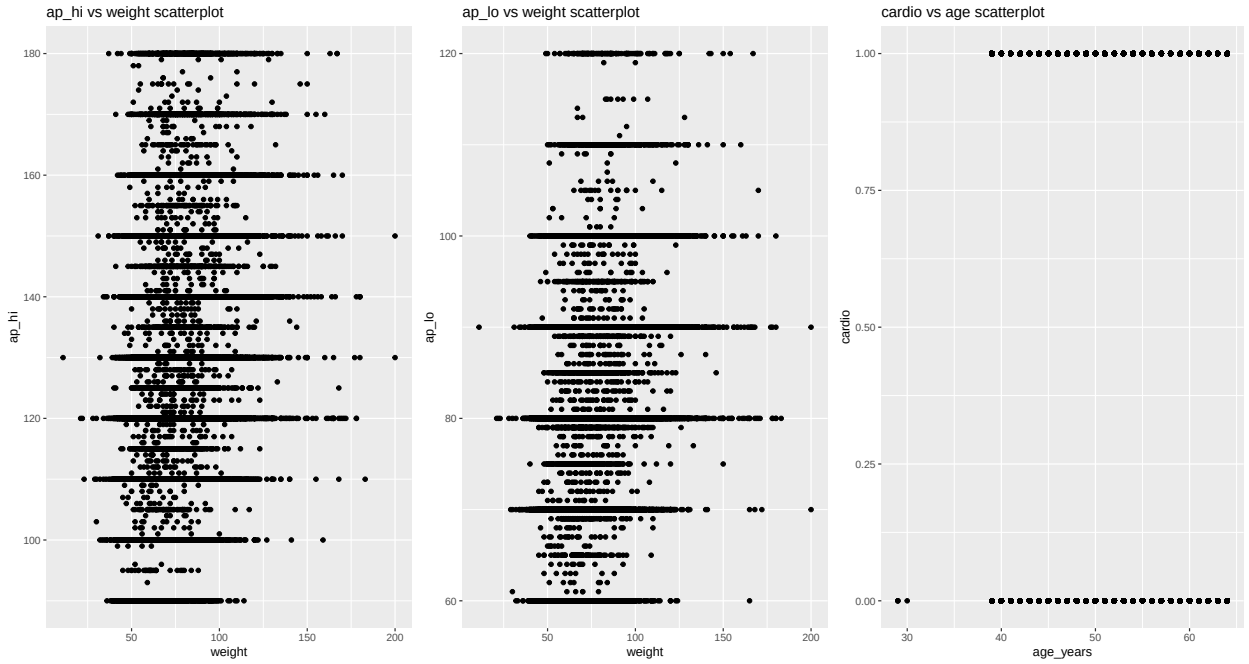
```
##              age      gender      height    weight cholesterol       gluc
## ap_hi  0.2116064 0.060721542  0.01854418 0.2682888   0.1953296 0.09315051
## ap_lo  0.1559992 0.066125957  0.03554988 0.2501841   0.1616365 0.07331896
## cardio 0.2390321 0.006097527 -0.01127589 0.1778293   0.2207782 0.08890543
##              smoke         alco       active age_years       bmi
## ap_hi   0.02603188  0.032535902 -0.001409008 0.2113136 0.2301632
```

```
## ap_lo   0.02383578  0.036211509 -0.001233588 0.1557770 0.2069436
## cardio -0.01656671 -0.009037713 -0.037943619 0.2388754 0.1628798
```

The highest positive-correlating variable for `ap_hi` and `ap_lo` are `weight`. The highest positively-correlating variable for `cardio` is `age`, however it is difficult to interpret the value in this column, as the age is not counted by year. So we opt to use the `age_years` column, which has similar correlation values to `age`. We note that for all these correlations, none of them seem particularly strong, which suggests that the ratio and regression estimates may not be that useful in prediction. To verify their usefulness, we make scatter plots for each target variable against the proposed auxiliary variables above:

```
aphi_vs_weight <- ggplot(data = data, aes(x=weight, y=ap_hi)) +
  geom_point() +
  labs(title = "ap_hi vs weight scatterplot")
aplo_vs_weight <- ggplot(data = data, aes(x=weight, y=ap_lo)) +
  geom_point() +
  labs(title = "ap_lo vs weight scatterplot")
cardio_vs_age <- ggplot(data = data, aes(x=age_years, y=cardio)) +
  geom_point() +
  labs(title = "cardio vs age scatterplot")

plot_grid(aphi_vs_weight, aplo_vs_weight, cardio_vs_age, ncol = 3)
```



For the scatter plots, we can visually confirm that there is little to no linear correlation for all plots. In the case of the `cardio` vs `age_years` plot, an estimate based on linear regression does not seem appropriate given the `cardio` variable is binary.

```
# ratio estimate:
ratio.ap_hi.ratio <- mean(srs$ap_hi) / mean(srs$weight)
ratio.ap_lo.ratio <- mean(srs$ap_lo) / mean(srs$weight)
ratio.cardio.ratio <- mean(srs$cardio) / mean(srs$age_years)

ratio.ap_hi.se <- sqrt(fpc*sum(var(srs$ap_hi-ratio.ap_hi.ratio*srs$weight))/n)
```

```r
ratio.ap_lo.se <- sqrt(fpc*sum(var(srs$ap_lo-ratio.ap_lo.ratio*srs$weight))/n)
ratio.cardio.se <- N*sqrt(fpc*sum(var(
  srs$cardio-ratio.cardio.ratio*srs$age_years))/n)

ratio.ap_hi.est <- ratio.ap_hi.ratio * mean(data$weight)
ratio.ap_lo.est <- ratio.ap_lo.ratio * mean(data$weight)
ratio.cardio.est <- ratio.cardio.ratio * mean(data$age_years) * N

ratio.summary <- rbind(
  c(ratio.ap_hi.est, ratio.ap_hi.se),
  c(ratio.ap_lo.est, ratio.ap_lo.se),
  c(ratio.cardio.est, ratio.cardio.se)
)

colnames(ratio.summary) <- c("est", "se")
rownames(ratio.summary) <- paste("ratio.", target_lst, sep="")
ratio.summary
```

```
##                       est          se
## ratio.ap_hi    126.55994   0.3805196
## ratio.ap_lo     81.53094   0.2398783
## ratio.cardio 33582.42893 499.5363974
```

```r
# regression estimate:

# helper function for regression estimate. X, y should be matrices of proper size.
# Don't ask me why i'm not just using lm() for this; idk either
regression.est_and_se <- function(X, y, x.pop) {
  X <- as.matrix(X); y <- as.matrix(y)
  X <- cbind(rep(1, dim(X)[1]), X)
  w <- solve(t(X)%*%X)%*%t(X)%*%y
  y.hat <- X%*%w
  se <- sqrt(fpc*sum(var(y - y.hat))/nrow(X))
  est <- cbind(1, x.pop) %*% w

  ret <- c()
  ret$est <- est; ret$se <- se
  return(ret)
}

reg.ap_hi <- regression.est_and_se(srs$weight, srs$ap_hi, mean(data$weight))
reg.ap_lo <- regression.est_and_se(srs$weight, srs$ap_lo, mean(data$weight))
reg.cardio <- regression.est_and_se(srs$age_years, srs$cardio, mean(data$age_years))

# reg.cardio estimates population mean, so we need to multiply estimate and SE by N
reg.cardio$est <- N*reg.cardio$est
reg.cardio$se <- N*reg.cardio$se

reg.summary <- rbind(
  reg.ap_hi, reg.ap_lo, reg.cardio
)

reg.summary
```

```
##            est      se
## reg.ap_hi  126.353  0.2296917
## reg.ap_lo  81.39437 0.1321236
## reg.cardio 33543.01 496.8257
```

Comparing the performance of each estimator in one table:

```
rbind(vanilla.summary, ratio.summary, reg.summary)
```

```
##                est      se
## vanilla.ap_hi  126.314  0.2366057
## vanilla.ap_lo  81.373   0.1358974
## vanilla.cardio 33630.89 126.4899
## ratio.ap_hi    126.5599 0.3805196
## ratio.ap_lo    81.53094 0.2398783
## ratio.cardio   33582.43 499.5364
## reg.ap_hi      126.353  0.2296917
## reg.ap_lo      81.39437 0.1321236
## reg.cardio     33543.01 496.8257
```

We observe that the estimated values for each target variable across all estimating methods are roughly the same. When comparing the standard errors of each, we see that ratio estimator has the largest SE out of all three methods. This is expected as from the correlation table and scatter plots above, there is little to no positive correlation between the target and response variable, making the ratio estimator a poor choice. The standard errors for `ap_hi` and `ap_lo` are similar for vanilla and regression estimates, with the vanilla estimate having slightly lower standard errors. Given the low correlation values shown above, the regression line formed will probably be similar to the constant prediction line that the vanilla estimate has, resulting in similar estimates and standard error. However, for the `cardio` response variable, given that the response is binary, linear regression would not be a suitable method to model the data, which is why we see such a higher standard error using the regression method when compared to the vanilla estimate.

Stratification

To determine which variable to stratify by, we base our decision on the variable that would produce the lowest within-strata variance. Equivalently, from a chosen subset of potential stratifying candidates $s_i \in \{s_1 \cdots s_n\}$, we want to choose to stratify on s_i if:

$$\text{argmax}\big(s^2_{\text{between strata}}/s^2_{\text{total}}\big) = s_i$$

$s^2_{\text{between strata}}$ and s^2_{total} are between-strata and total variances respectively. Calculating this ratio for the potential stratifying variables `gender`, `cholesterol`, `gluc`, `smoke`, `alco`, and `active`:

```
# should stratify based on variable that produces smallest within-strata variance

# candidate stratifiying variables:
# gender
# cholesterol
# gluc
# smoke
# alco
# active
```

```r
# Calculate within-strata variance of target based off of levels of variables in var_lst

strata.compare_var <- function(df, target_df) {
  var_lst <- colnames(df)
  target_lst <- colnames(target_df)
  ret <- NULL

  for (t in 1:length(target_df)) {
    target <- target_df[,t]
    ratios <- rep(NA, length(var_lst))
    for (i in 1:length(var_lst)) {
      v <- df[,i]
      var.ws <- sum(prop.table(table(v)) * tapply(target, v, var))
      var.bs <- sum(prop.table(table(v)) * (tapply(target, v, mean) - mean(target))^2)
      var.tot <- var.ws + var.bs
      ratios[i] <-  var.ws / var.tot
    }
    ret <- rbind(ret, ratios)
  }
  rownames(ret) <- target_lst
  colnames(ret) <- var_lst
  return(ret)
}

pred <- data[,c("gender", "cholesterol", "gluc",
                "smoke", "alco", "active")]
target <- data[,c("cardio", "ap_hi", "ap_lo")]
strata.compare_var(pred, target)
```

```
##           gender cholesterol      gluc     smoke      alco    active
## cardio 0.9999628   0.9512554 0.9917109 0.9997256 0.9999183 0.9985603
## ap_hi  0.9963130   0.9610327 0.9896462 0.9993224 0.9989414 0.9999980
## ap_lo  0.9956275   0.9736067 0.9934043 0.9994319 0.9986888 0.9999985
```

From the table above, we see that variables `smoke`, `alco`, and `active` have the highest between-strata to overall variance ratios, indicating that these variables are the best to stratify on. These three variables all have similar performance, so we arbitrarily choose one (go with `alco` for now).

We now want to see how we much from each strata we should sample. In our case, cost of sampling in our case is constant, as our sampling process is simulated. We want to see if we can use proportional allocation, so we check to see if we can assume the variances of each strata are equal through the stratum-specific population variances:

```r
pop.alco_one.ap_hi.var <- var(data[data$alco==1,"ap_hi"])
pop.alco_one.ap_lo.var <- var(data[data$alco==1,"ap_lo"])
pop.alco_one.cardio.var <- var(data[data$alco==1,"cardio"])

pop.alco_zero.ap_hi.var <- var(data[data$alco==0,"ap_hi"])
pop.alco_zero.ap_lo.var <- var(data[data$alco==0,"ap_lo"])
pop.alco_zero.cardio.var <- var(data[data$alco==0,"cardio"])

pop.var.summary <- rbind(
  c(pop.alco_zero.ap_hi.var, pop.alco_one.ap_hi.var),
```

```
  c(pop.alco_zero.ap_lo.var, pop.alco_one.ap_lo.var),
  c(pop.alco_zero.cardio.var, pop.alco_one.cardio.var)
)

rownames(pop.var.summary) <- target_lst
colnames(pop.var.summary) <- c("alco==0", "alco==1")
pop.var.summary
```

```
##            alco==0     alco==1
## ap_hi  253.1225819 279.2297426
## ap_lo   82.8501864  95.1590516
## cardio   0.2499764   0.2494244
```

From above, the variances are roughly equal for each strata for each target variable, so we assume equal variance assumption holds. Therefore, optimal allocation is equal to proportional allocation:

$$\frac{n_h}{n} = \frac{N_h}{N}$$

Calculating the stratified estimate and standard error under proportional allocation:

```
set.seed(124)

# generate stratified sample
prop.alco_one <- mean(data$alco)
n1 <- round((1-prop.alco_one) * n, 0); n2 <- round(prop.alco_one * n, 0)
pop.alco_zero <- data[data$alco==0,]; pop.alco_one <- data[data$alco==1,]
strata.sample <- rbind(pop.alco_zero[sample(1:nrow(pop.alco_zero), n1),],
                       pop.alco_one[sample(1:nrow(pop.alco_one), n2), ])


# given population data, sample and calculate the stratified estimate + SE
stratify.est_and_se <- function(pop, sample, var_str, target_str, total=FALSE) {
  pop.var.count <- table(data[,var_str])
  sample.var.count <- table(sample[,var_str])
  # population proportion for each strata
  pop.prop <- prop.table(pop.var.count)

  sample.means <- tapply(sample[,target_str], sample[,var_str], mean)
  sample.vars <- tapply(sample[,target_str], sample[,var_str], var) / pop.var.count
  fpc <- 1 - sample.var.count / pop.var.count

  est <- round(sum(pop.prop * sample.means), 3)
  se <- round(sqrt(sum(pop.prop^2 * fpc * sample.vars)), 3)
  if (total==TRUE) {est <- N*est; se <- N*se}
  # print(paste0("Strata Estimate for ", target_str, ": ", est, " +/- ", 1.96*se))
  ret <- c()
  ret$est <- est
  ret$se <- se
  return(ret)
}

strat.ap_hi <- stratify.est_and_se(data, strata.sample,
```

```
                                    var_str = "alco", target_str = "ap_hi")
strat.ap_lo <- stratify.est_and_se(data, strata.sample,
                                    var_str = "alco", target_str = "ap_lo")
strat.cardio <- stratify.est_and_se(data, strata.sample, var_str = "alco",
                     target_str = "cardio", total=TRUE)
strat.summary <- rbind(strat.ap_hi, strat.ap_lo, strat.cardio)
strat.summary
```

```
##              est     se
## strat.ap_hi  126.691 0.06
## strat.ap_lo  81.228  0.034
## strat.cardio 34102.5 136.41
```

Comparing all estimates and standard errors from all methods of estimation:

```
rbind(vanilla.summary, ratio.summary, reg.summary, strat.summary)
```

```
##                est      se
## vanilla.ap_hi  126.314  0.2366057
## vanilla.ap_lo  81.373   0.1358974
## vanilla.cardio 33630.89 126.4899
## ratio.ap_hi    126.5599 0.3805196
## ratio.ap_lo    81.53094 0.2398783
## ratio.cardio   33582.43 499.5364
## reg.ap_hi      126.353  0.2296917
## reg.ap_lo      81.39437 0.1321236
## reg.cardio     33543.01 496.8257
## strat.ap_hi    126.691  0.06
## strat.ap_lo    81.228   0.034
## strat.cardio   34102.5  136.41
```

Estimates from stratified estimate are similar to the ones from vanilla, ratio, and regression, but standard error for `ap_hi` and `ap_lo` are much lower. The standard error for `cardio` is worse, but why?

```
# FROM TA:
# - stratifying variable based on intuition
# - equal variance assumption -> proportional allocation, but determine equal variance from population

# FROM WU LANG:
# - just use intuition
```