HW 2

KEY

Instructions

First, fill in your name in the author line in the header above. Then, use the template below to complete the assignment. For question 1, you can choose to upload an image of your work rather than typing out the math (if you do want to type it, here is a helpful resource to get started). To upload an image, click "Insert > Figure/Image." For question 2, use the provided code boxes and then write the narrative answers underneath.

Once you have completed the exercises, render your document and submit the pdf on Gradescope. Be sure to assign the appropriate pages to the exercise numbers when you submit.

Exercise answers

1

a. $L(\mu) = \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi}} e^{\frac{-(x_i-\mu)^2}{2}}$ (take the product for the joint likelihood, plug in $\sigma^2 = 1$, and add $_i$ to x)

b. $l(\mu) = \sum_{i=1}^{n} \log \frac{1}{\sqrt{2\pi}} - \frac{(x_i-\mu)^2}{2} = -\frac{n}{2} \log 2\pi - \frac{(x_i-\mu)^2}{2}$ (not required to simplify to the second expression)

c. $\frac{d}{d\mu}(-\frac{n}{2} \log 2\pi - \frac{(x_i-\mu)^2}{2}) = \sum_{i=1}^{n}(x_i - \mu) = \sum_{i=1}^{n} x_i - n\mu$

Setting this to 0 and solving for μ, we get $\hat{\mu} = \frac{\sum_{i=1}^{n} x_i}{n}$

d. $E[\hat{\mu}] = E[\frac{\sum_{i=1}^{n} x_i}{n}] = \frac{\sum_{i=1}^{n} E[x_i]}{n} = \frac{n\mu}{n} = \mu$, so $\hat{\mu}$ is unbiased.

2

```
library(tidyverse)
```

```
-- Attaching packages --------------------------------------- tidyverse 1.3.1 --

v ggplot2 3.4.0     v purrr   0.3.4
v tibble  3.1.8     v dplyr   1.1.0
v tidyr   1.2.0     v stringr 1.4.0
v readr   2.1.2     v forcats 0.5.1

-- Conflicts ------------------------------------------ tidyverse_conflicts() --
x dplyr::filter() masks stats::filter()
x dplyr::lag()    masks stats::lag()
```

```
population <- read.csv("https://raw.githubusercontent.com/anlane611/datasets/main/population
```

a.

```
mean(population$Y)
```

```
[1] 19.97579
```

```
#OR

population |>
  summarise(mean(Y))
```

```
   mean(Y)
1 19.97579
```

b.

```
population |>
  group_by(X2) |>
  summarise(mean(Y))
```

```
# A tibble: 2 x 2
     X2 `mean(Y)`
  <int>     <dbl>
1     0      33.7
2     1      19.9
```

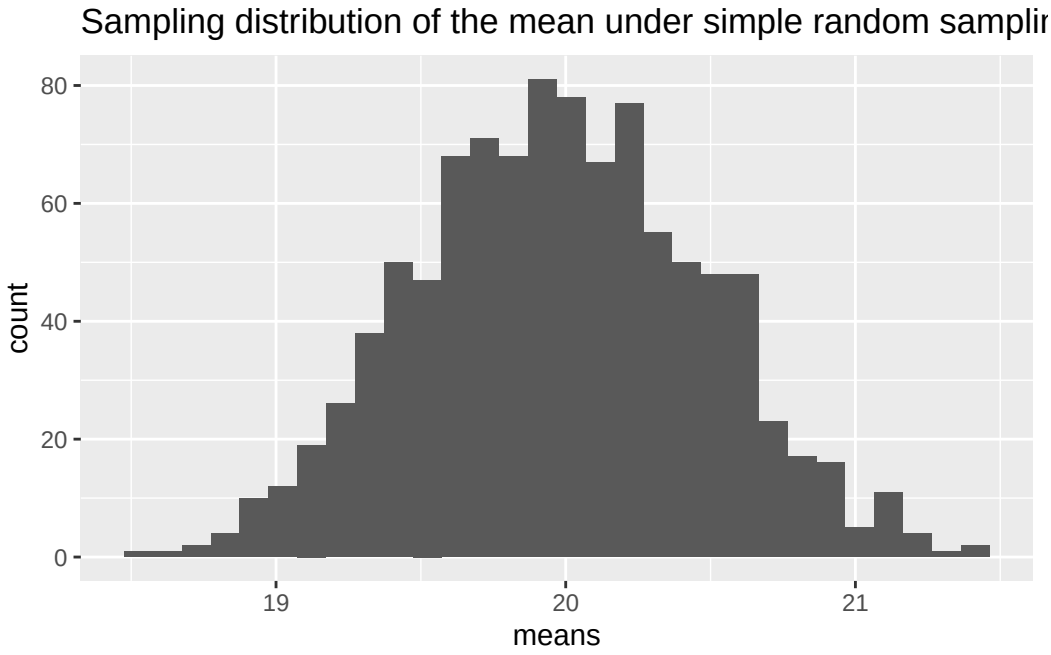
```
population |>
  count(X2)
```

```
  X2     n
1  0   807
2  1 99193
```

```
SRSmeans <- data.frame(means=NA) #create an empty dataframe to store the sample means
for(i in 1:1000){
  set.seed(i) #ensure that we have a different sample each time
  SRS.sample <- population |> slice(sample(1:n(),size=100))
  #the slice function selects rows of the specified data (population),
  #and the sample function randomly selects 100 numbers out of the
  #sequence 1:100000
  SRSmeans[i,1] <- SRS.sample |> summarise(mean(Y))
  #calculate the mean of each sample
}

ggplot(SRSmeans, aes(x=means)) +
  geom_histogram()+
  labs(title="Sampling distribution of the mean under simple random sampling (N=100)")
```

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



c.

```
mean(SRSmeans$means)
```

```
[1] 19.97874
```

The distribution looks approximately normal and the mean of the sampling distribution is nearly identical to the true population mean.

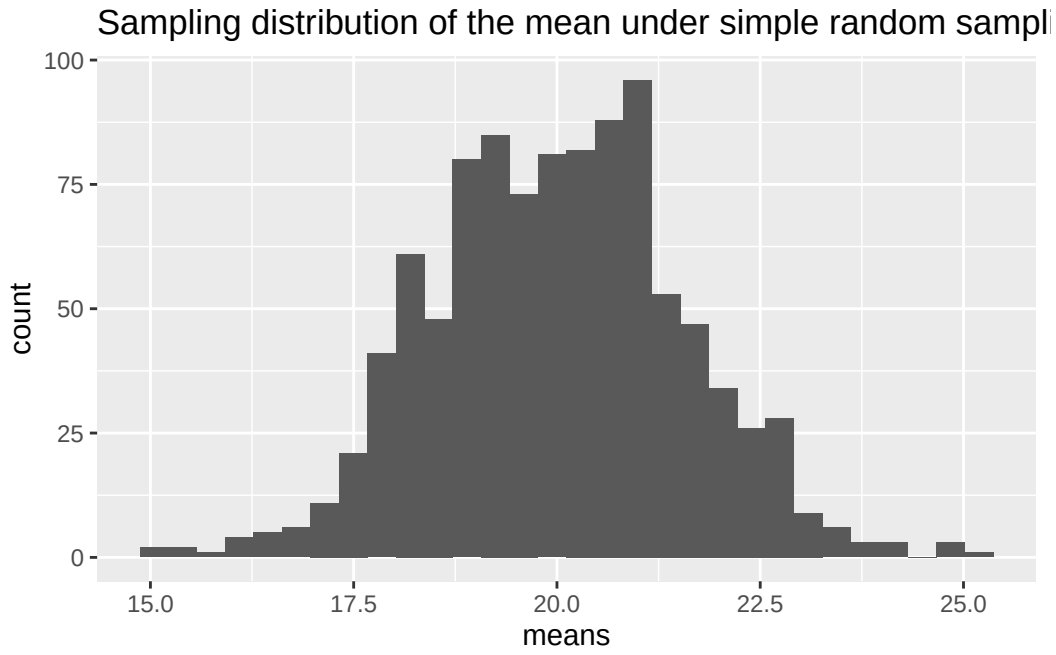
d.

```
SRSmeans_10 <- data.frame(means=NA)

for(i in 1:1000){
  set.seed(i)
  SRS.sample <- population |> slice(sample(1:n(),size=10))
  SRSmeans_10[i,1] <- SRS.sample |> summarise(mean(Y))
}

ggplot(SRSmeans_10, aes(x=means)) +
  geom_histogram()+
  labs(title="Sampling distribution of the mean under simple random sampling (N=10)")
```

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




```
mean(SRSmeans_10$means)
```

```
[1] 20.01105
```

The shape is similar and the mean is still close to the population mean (though not as accurate as the previous part), and the distribution is more spread out (i.e., the standard error is higher).

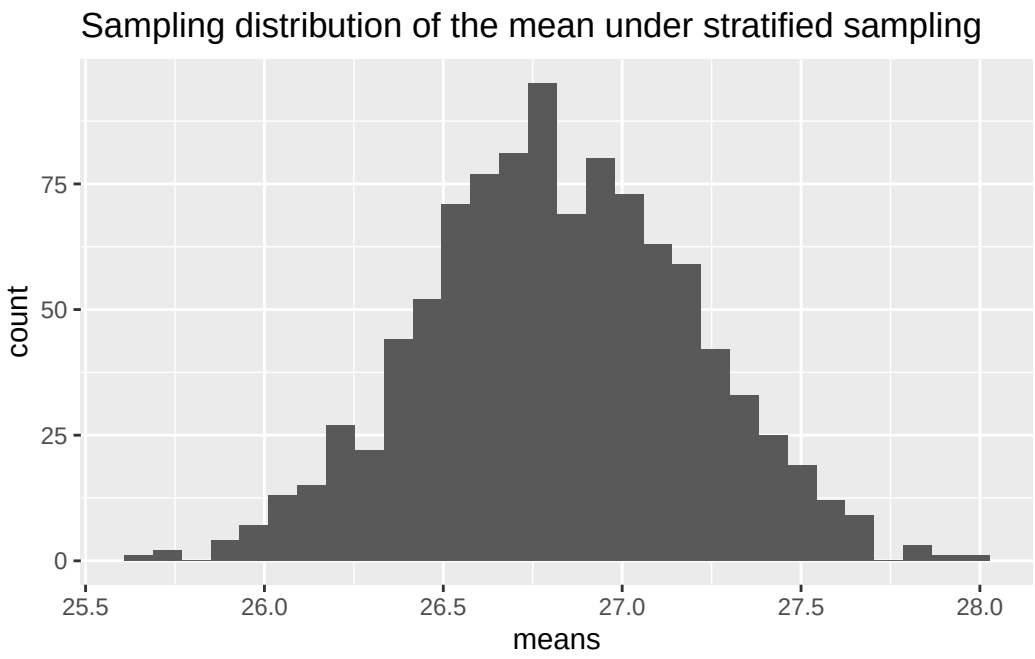
e.

```
Stratmeans <- data.frame(means=NA)

for(i in 1:1000){
  set.seed(i)
  Stratified.sample <- population |>
                       group_by(X2) |>
                       slice(sample(1:n(),size=50)) |>
                       ungroup()
  Stratmeans[i,1] <- Stratified.sample |> summarise(mean(Y))
}
```

```
ggplot(Stratmeans, aes(x=means)) +
  geom_histogram()+
  labs(title="Sampling distribution of the mean under stratified sampling")
```

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



f.

```
mean(Stratmeans$means)
```

```
[1] 26.82094
```

The distribution is approximately normal, but the mean is higher than the population mean.

g. We oversample from the group that has a higher mean, which pulls the mean of the sampling distribution up. If observations do not have an equal probability of being sampled, the sampling distribution may not be accurate.

Bonus:

```r
population |>
  group_by(X1) |>
  summarise(mean(Y))
```

```
# A tibble: 5 x 2
     X1 `mean(Y)`
  <int>     <dbl>
1     1      20.0
2     2      20.0
3     3      20.0
4     4      20.0
5     5      20.0
```

```r
population |>
  count(X1)
```

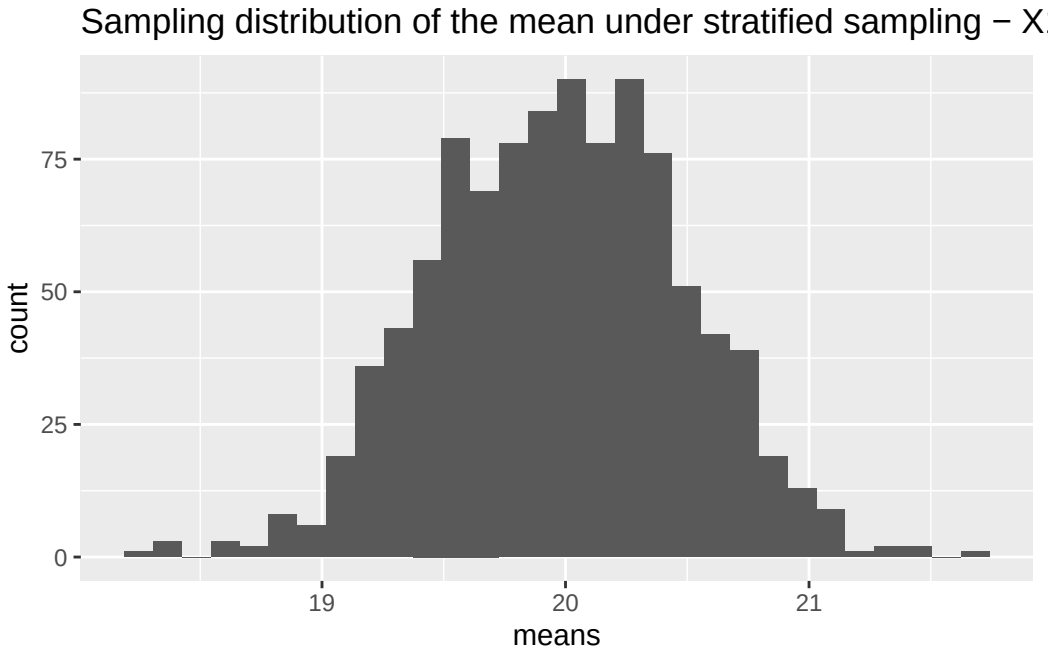
```
  X1     n
1  1 19000
2  2 20100
3  3 20200
4  4 22900
5  5 17800
```

```r
Stratmeans_x1 <- data.frame(means=NA)

for(i in 1:1000){
  set.seed(i)
  Stratified.sample <- population |>
                       group_by(X1) |>
                       slice(sample(1:n(),size=20)) |>
                       ungroup()
  Stratmeans_x1[i,1] <- Stratified.sample |> summarise(mean(Y))
}

ggplot(Stratmeans_x1, aes(x=means)) +
  geom_histogram()+
  labs(title="Sampling distribution of the mean under stratified sampling - X1")
```

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




```
mean(Stratmeans_x1$means)
```

```
[1] 19.97158
```

This sampling distribution looks similar to the one we generated for simple random sampling. The distribution is approximately normal and the mean is close to the population mean. In this case, the strata are balanced, and the true mean in each stratum is close to the population mean. We can conclude that regardless of the sampling scheme, the sample must reflect the population for the sampling distribution to be accurate.