IDS 702 HW 1

Eric Ortega Rodriguez

Instructions: Use this template to complete your assignment. When you click "Render," you should get a PDF document that contains both your answers and code. You must show your work/justify your answers to receive credit. Submit your rendered PDF file on Gradescope. Remember to render frequently, as this will help you to catch errors in your code before the last minute.

Add your name in the Author section in the header

Exercise 1

For this question, you should type out the formulas that you use to do the calculations (click Insert > Equation > Inline math). You can read more about technical writing in Quarto here. You can use the provided code chunks to use R as a calculator.

a. $P(A|B) = 0.2$

```
library(dbplyr)
prob_a <- 0.2
prob_b <- 0.4
prob_a_and_b <- 0.08

p_a_given_b <- prob_a_and_b/prob_b

p_a_given_b
```

```
[1] 0.2
```

b. $P(A \cup B) = 0.52$

```
prob_union <- prob_a +prob_b  - prob_a_and_b

prob_union
```

```
[1] 0.52
```

c. A and B are not mutually exclusive. This is because in order for two events to be mutually exclusive, they should not occur together. In other words $P(A \cap B) = 0$, but as mentioned in the problem, $P(A \cap B) = 0.08$. Therefore, A and B are not mutually exclusive.

d. A and B are independent. This is because in order for the two events to be independent, the product of their individual probabilities must be the same as $P(A \cap B)$. Therefore, in this case 0.08 = (0.4) * (0.2). Given, this is the case, indicates that A and B are independent.

Exercise 2

a. The binomial probability distribution best fits this situation. This is because in this scenario there are only two possible outcomes – fraud or no fraud. In addition to this, both outcomes are independent. Furthermore, there will be a fixed number of trials which is another requirement in binomial probabilities. Additionally, for each trial, there is an equal 20% probability that a holder will experience fraud. Therefore, this case fits all the requirements for a binomial probability distribution.

b. The number of customers that the bank should expect to experience fraud are 40 customers.

```
n_customers <- 200
p_fraud <- 0.2

exp_value <- n_customers *p_fraud

exp_value
```

```
[1] 40
```

c. The probability that 50 customers will experience fraud is approximately 0.0149.

```
prob_50_cust <- dbinom(50, size =n_customers, prob = p_fraud)

prob_50_cust
```

```
[1] 0.01485656
```

d. The probability that no more than 50 customers will experience fraud is approximately 0.9655.

```
prob_not_higher_than_50_cust <- pbinom(50,size=n_customers,prob = p_fraud)

prob_not_higher_than_50_cust
```

```
[1] 0.9655032
```

Exercise 3

You are required to show the code you use to complete each part of this exercise. You must also write your narrative answers below the code.

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

a. The mean of Y is 19.97579.

```
mean_of_Y <-mean(population$Y)

mean_of_Y
```

```
[1] 19.97579
```

b.The table below shows the mean of Y for each level and number of observations for X1.

```
library(tidyverse)
```

```
-- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
v dplyr     1.1.4     v readr     2.1.5
v forcats   1.0.0     v stringr   1.5.1
v ggplot2   3.5.1     v tibble    3.2.1
v lubridate 1.9.3     v tidyr     1.3.1
v purrr     1.0.2
-- Conflicts ------------------------------------------ tidyverse_conflicts() --
x dplyr::filter() masks stats::filter()
x dplyr::ident()  masks dbplyr::ident()
x dplyr::lag()    masks stats::lag()
x dplyr::sql()    masks dbplyr::sql()
i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to beco
```

```r
population |>
  group_by(X1) |>
  summarise(mean(Y), count=n())
```

```
# A tibble: 5 x 3
     X1 `mean(Y)` count
  <int>     <dbl> <int>
1     1      20.0 19000
2     2      20.0 20100
3     3      20.0 20200
4     4      20.0 22900
5     5      20.0 17800
```

c. The table below shows the mean of Y for each level and number of observations for X2:

```r
# Calculate mean of Y and count of observations for each level of X2
population |>
  group_by(X2) |>
  summarise(mean(Y), count=n())
```

```
# A tibble: 2 x 3
     X2 `mean(Y)` count
  <int>     <dbl> <int>
1     0      33.7   807
2     1      19.9 99193
```

d. The histogram below shows a normal distribution. This is due to the its shape being close to the traditional bell curve. In addition, the distribution mean is approximately 20.01105. The distribution mean was approximately 0.03526 higher than the population mean which was at 19.97579.

```r
library(dplyr)
library(ggplot2)

SRSmeans <- data.frame(means=NA) #empty dataframe
for(i in 1:1000){
  set.seed(i) # seed creates random sample size
  SRS.sample <- population |> slice(sample(1:n(),size=10))
  SRSmeans[i,1] <- SRS.sample |> summarize(mean=mean(Y))
}
```

```
ggplot(SRSmeans, aes(x=means)) +

  geom_histogram()+
  labs(title="Distribution of Sample Means (SSR- Sample Size of 10)",
       x= "Mean",
       y= "Frequency")
mean_samples = mean(SRSmeans$means)
print(mean_samples)
```

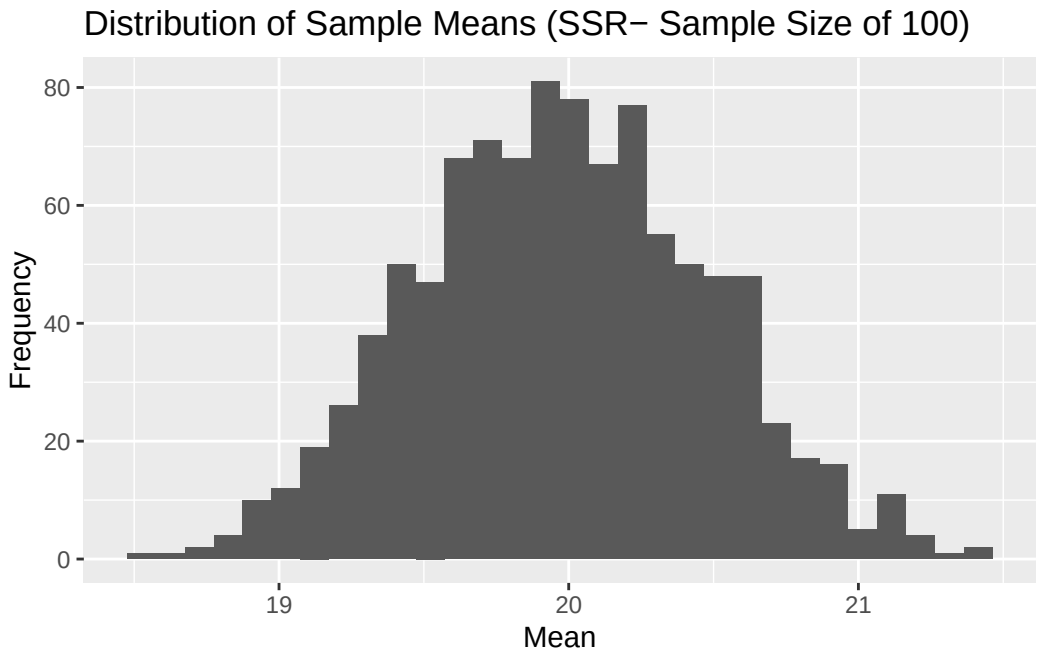
e.When looking at both histograms (from part d and e), the graph looks more like a normal distribution since the larger portion can be seen towards the middle. This ties back to our lesson discussing the central limit theorem which states that "the sampling distribution of the mean will always be normally distributed, as long as the sample size is large enough." This is precisely what can be seen when comparing both histograms.

```
SRSmeans <- data.frame(means=NA) #empty dataframe
for(i in 1:1000){
  set.seed(i) #different sample
  SRS.sample <- population |> slice(sample(1:n(),size=100))
  SRSmeans[i,1] <- SRS.sample |> summarize(mean=mean(Y))
}

ggplot(SRSmeans, aes(x=means)) +

  geom_histogram()+
  labs(title="Distribution of Sample Means (SSR- Sample Size of 100)",
       x= "Mean",
       y= "Frequency")
```

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

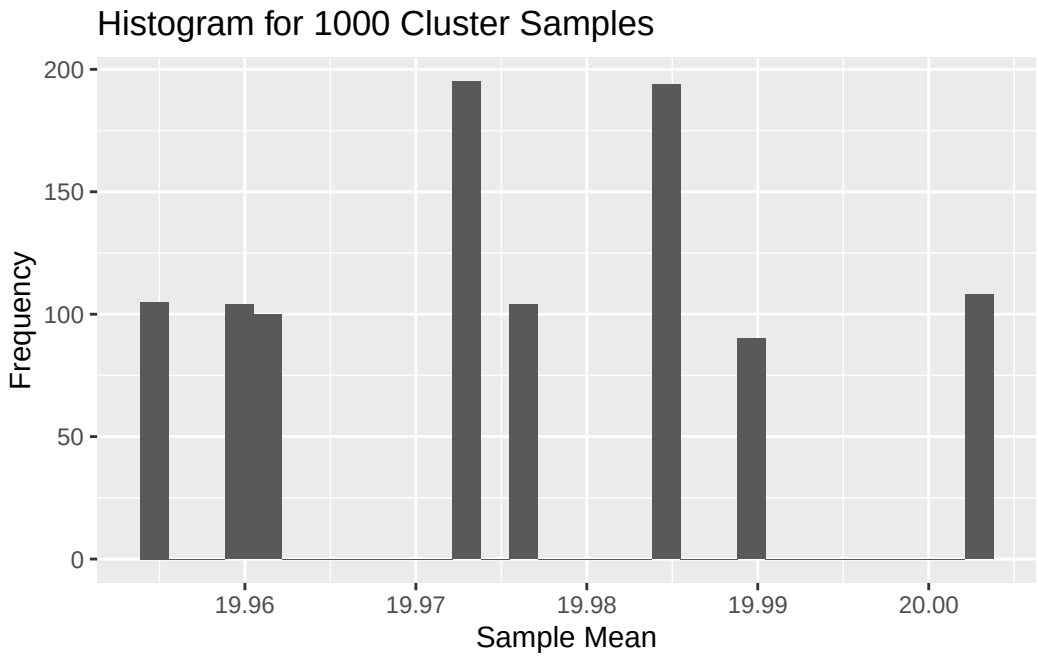


f. In the graph below, what I can observe is that it is close to normal distribution. Regarding the mean of the of the sampling, it is approximately 19.97604.

```
cluster_means <- data.frame(means = NA) # empty data frame
for (i in 1:1000) {
  set.seed(i)
  cluster <- unique(population$X1)
  sel_clusters <- sample(cluster, size = 2)
  sample_cluster <- population[population$X1 %in% sel_clusters,TRUE ]
  mean_of_Y <- mean(sample_cluster$Y)
  cluster_means[i, 1] <- mean_of_Y
}
library(ggplot2)

ggplot(cluster_means, aes(x = means)) +
  geom_histogram() +
  labs(title = "Histogram for 1000 Cluster Samples", x = "Sample Mean", y = "Frequency")
```

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




```
mean_cluster_sampling <- mean(cluster_means$means)
print(mean_cluster_sampling)
```

```
[1] 19.97604
```

g. The sampling distribution is approximately normal. Regarding the sampling distribution, it is about 19.9635.

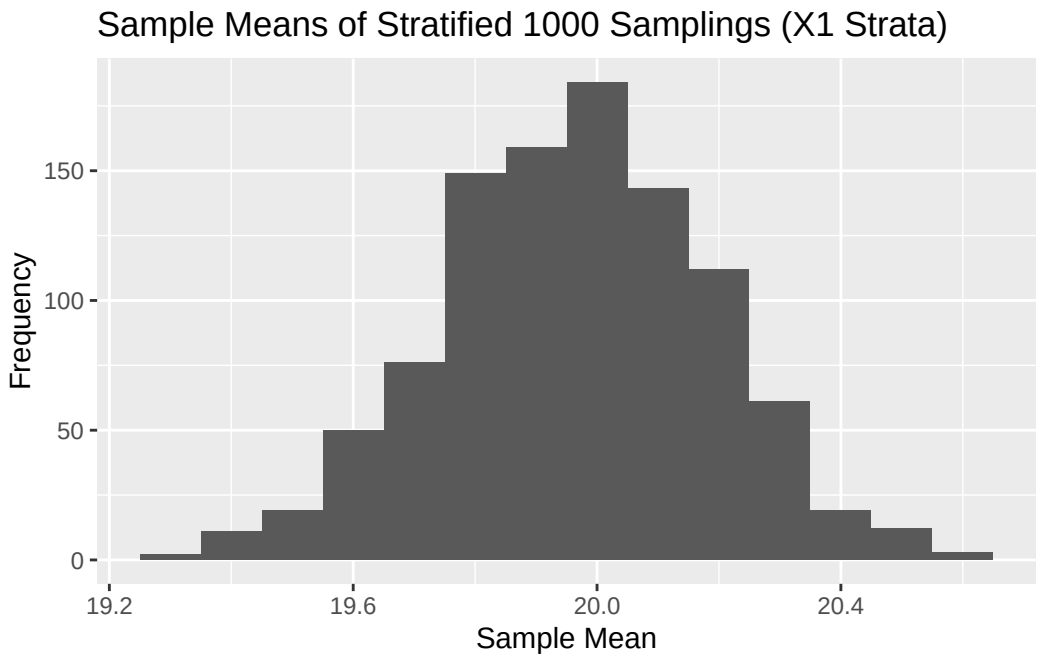
```
sample_means <- numeric(1000)

strata <- unique(population$X1)
for (i in 1:1000) {
  set.seed(i) # empty data frame
  sampled_data <- data.frame() # Stratified sampling
  stratified_sample <- population |>
  group_by(X1) |>
  slice_sample(n=100, replace =FALSE)#

sample_means[i] <- mean(stratified_sample$Y, na.rm = TRUE)}

sample_means_df <- data.frame(SampleMean = sample_means)
  ggplot(sample_means_df, aes(x = SampleMean)) +
```

```
  geom_histogram(binwidth = 0.1)+
labs(title="Sample Means of Stratified 1000 Samplings (X1 Strata)",  x = "Sample Mean", y = "
```




```
mean(sample_means)
```

```
[1] 19.9635
```

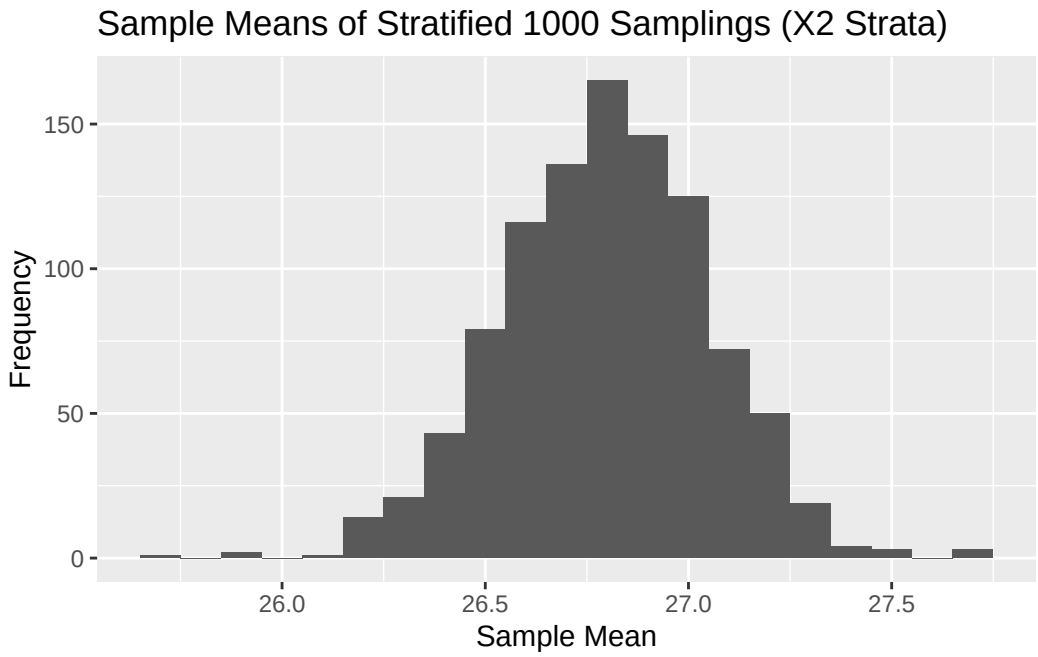
h.The sampling distribution also appears to be approximately normal. Regarding the sampling distribution, it is about 26.79623.

```
sample_means <- numeric(1000)

strata <- unique(population$X2)
for (i in 1:1000) {
  set.seed(i) # empty data frame
  sampled_data <- data.frame() # stratified sampling
  stratified_sample <- population |>
  group_by(X2) |>
  slice_sample(n=100, replace =FALSE)#

sample_means[i] <- mean(stratified_sample$Y, na.rm = TRUE)}
```

```r
sample_means_df <- data.frame(SampleMean = sample_means)
  ggplot(sample_means_df, aes(x = SampleMean)) +
  geom_histogram(binwidth = 0.1)+
labs(title="Sample Means of Stratified 1000 Samplings (X2 Strata)",  x = "Sample Mean", y = "
```




```r
mean(sample_means)
```

```
[1] 26.79623
```

i. Throughout this assignment, the idea of the central limit theorem can be seen. In parts g and h, the sampling distributions appear to have a normal distribution. When we increased our sample size, the normal distribution becomes more apparent. Regarding parts b and c, the means gathered match up closely to g and h. Thus, this indicates and emphasizes that the sample mean serves as an unbiased estimator of the population mean and proving the sample mean is equal to the population mean.

Bonus (optional)

Bonus: Multistage sampling involves taking samples in stages and using smaller and smaller sampling. Having gone through the exercise above, I believe this approach would be more

accurate than cluster sampling since it involves breaking down clusters into smaller sections at various stages. Although multistage it is more precise, it comes at a higher "cost" as it can be more complex. In addition to this, based on the data that was provided, cluster sampling would be appropriate to use. This is because our data is not extensive and further division into groups would not be needed.