Sampling

El Mex

Contents


```
library(tidyverse)
library(moderndive)
```

1 Sampling

1.1 Sampling bowl activity

1.1.1 What proportion of this bowl's balls are red?

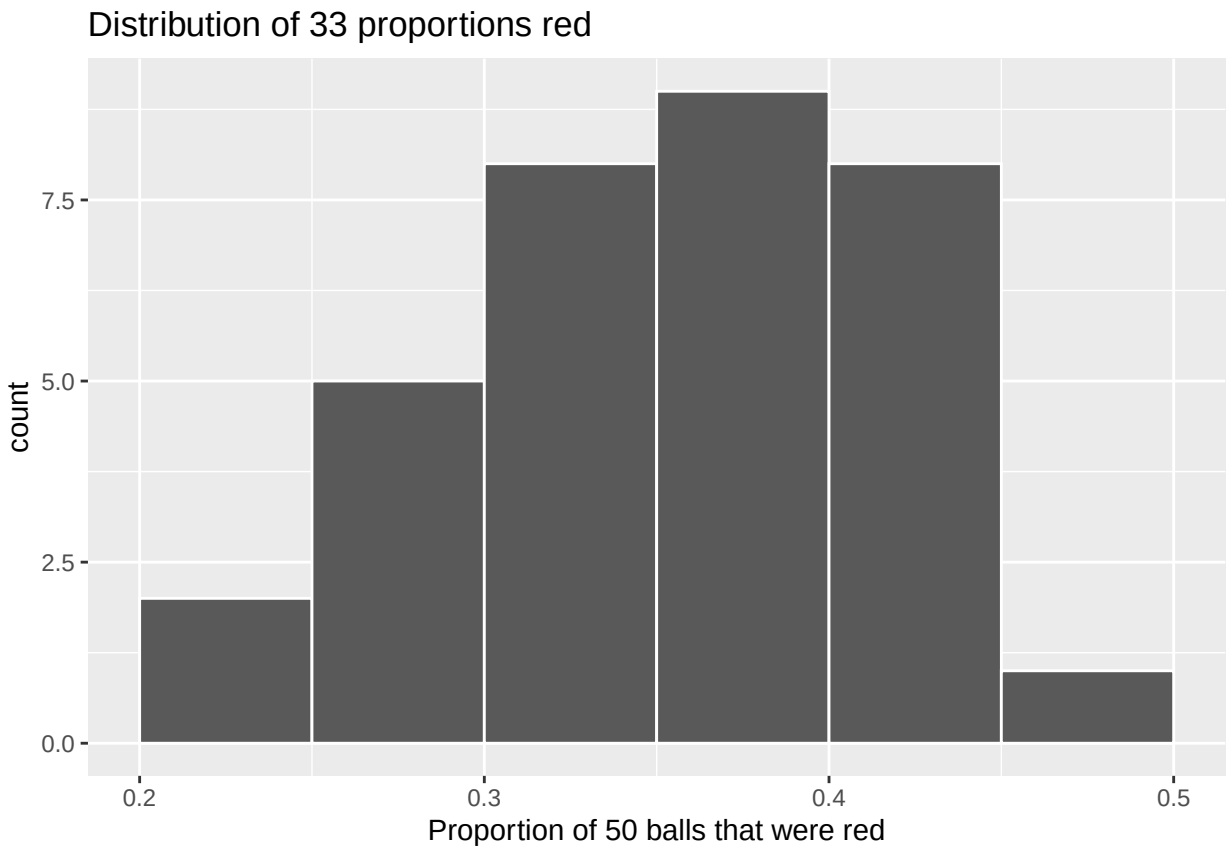
After finishing the getting-several-samples experiment, we saved our 33 groups of friends' results in the `tactile_prop_red` data frame included in the `moderndive` package. Run the following to display the first 10 of 33 rows:

```
head(tactile_prop_red, 10)
```

```
## # A tibble: 10 x 4
##    group            replicate red_balls prop_red
##    <chr>                <int>     <int>    <dbl>
##  1 Ilyas, Yohan             1        21     0.42
##  2 Morgan, Terrance         2        17     0.34
##  3 Martin, Thomas           3        21     0.42
##  4 Clark, Frank             4        21     0.42
##  5 Riddhi, Karina           5        18     0.36
##  6 Andrew, Tyler            6        19     0.38
##  7 Julia                    7        19     0.38
##  8 Rachel, Lauren           8        11     0.22
##  9 Daniel, Caroline         9        15     0.3
## 10 Josh, Maeve             10        17     0.34
```

Let's visualize the distribution of these 33 proportions using `geom_histogram()` with `binwidth = 0.05`. Note that setting `boundary = 0.4` indicates that we want a binning scheme such that one of the bins' boundary is at 0.4

```
ggplot(tactile_prop_red, aes(x = prop_red)) +
  geom_histogram(binwidth = 0.05, boundary = 0.4, color = "white") +
  labs(x = "Proportion of 50 balls that were red",
       title = "Distribution of 33 proportions red")
```



1.2 Virtual sampling

1.2.1 Using the virtual shovel once

Explore the data frame named `bowl` in the `moderndive` package

```
glimpse(bowl, 10)
```

```
## Rows: 2,400
## Columns: 2
## $ ball_ID <int> ...
## $ color   <chr> ...
```

```
head(bowl, 10)
```

```
## # A tibble: 10 x 2
##    ball_ID color
##      <int> <chr>
```

```
##  1       1 white
##  2       2 white
##  3       3 white
##  4       4 red
##  5       5 white
##  6       6 white
##  7       7 red
##  8       8 white
##  9       9 red
## 10      10 white
```

It has 2400 rows, telling us that the bowl contains 2400 equally sized balls, white and red. Now we're going to use the `rep_sample_n()` function included in the `moderndive` package. This function allows us to take `repeated`, or `replicated`, `samples` of size `n`

```
# assign the result to "virtual_shovel"
virtual_shovel <- bowl %>%
  rep_sample_n(size = 50)

virtual_shovel
```

```
## # A tibble: 50 x 3
## # Groups:   replicate [1]
##    replicate ball_ID color
##        <int>   <int> <chr>
##  1         1     811 white
##  2         1     227 red
##  3         1     943 white
##  4         1    1511 white
##  5         1    1174 white
##  6         1     867 white
##  7         1    2133 white
##  8         1    1457 white
##  9         1     502 white
## 10         1    2083 red
## # ... with 40 more rows
```

What does the replicate variable indicate? In `virtual_shovel`'s case, `replicate` is equal to 1 for all 50 rows. This is telling us that these 50 rows correspond to the first repeated/replicated use of the shovel, in our case our first sample

For each of our 50 sampled balls, let's identify if it is red or not using a test for equality with `==`. Let's create a new Boolean variable `is_red` using the `mutate()` function

```
virtual_shovel %>%
  mutate(is_red = (color == "red"))
```

```
## # A tibble: 50 x 4
## # Groups:   replicate [1]
##    replicate ball_ID color is_red
##        <int>   <int> <chr> <lgl>
##  1         1     811 white FALSE
##  2         1     227 red   TRUE
```

```
##  3         1     943 white FALSE
##  4         1    1511 white FALSE
##  5         1    1174 white FALSE
##  6         1     867 white FALSE
##  7         1    2133 white FALSE
##  8         1    1457 white FALSE
##  9         1     502 white FALSE
## 10         1    2083 red   TRUE
## # ... with 40 more rows
```

Let's compute the number of balls out of 50 that are red using the `summarize()` function

```
virtual_shovel %>%
  mutate(is_red = (color == "red")) %>%
  summarize(num_red = sum(is_red))
```

```
## # A tibble: 1 x 2
##   replicate num_red
##       <int>   <int>
## 1         1      17
```

Finally, let's compute the proportion of the 50 sampled balls that are red by dividing `num_red` by 50:

```
virtual_shovel %>%
  mutate(is_red = color == "red") %>%
  summarize(num_red = sum(is_red)) %>%
  mutate(prop_red = num_red / 50)
```

```
## # A tibble: 1 x 3
##   replicate num_red prop_red
##       <int>   <int>    <dbl>
## 1         1      17     0.34
```

In other words, 36% of this virtual sample of 50 balls were red. Let's make this code a little more compact and succinct by combining the first `mutate()` and the `summarize()` as follows:

```
virtual_shovel %>%
  summarize(num_red = sum(color == "red")) %>%
  mutate(prop_red = num_red / 50)
```

```
## # A tibble: 1 x 3
##   replicate num_red prop_red
##       <int>   <int>    <dbl>
## 1         1      17     0.34
```

1.2.2 Using the virtual shovel 33 times

Remember we had 33 groups of students each use the shovel, yielding 33 samples of size 50 balls. We can perform this repeated/replicated sampling virtually by once again using our virtual shovel function `rep_sample_n()`, but by adding the `reps = 33` argument

```r
virtual_samples <- bowl %>%
  rep_sample_n(size = 50, reps = 33)

glimpse(virtual_samples)
```

```
## Rows: 1,650
## Columns: 3
## Groups: replicate [33]
## $ replicate <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ...
## $ ball_ID   <int> 2374, 1908, 620, 39, 979, 526, 2016, 2376, 292, 440, 1459...
## $ color     <chr> "white", "white", "red", "white", "white", "white", "red"...
```

```r
head(virtual_samples, 10)
```

```
## # A tibble: 10 x 3
## # Groups:   replicate [1]
##    replicate ball_ID color
##        <int>   <int> <chr>
##  1         1    2374 white
##  2         1    1908 white
##  3         1     620 red
##  4         1      39 white
##  5         1     979 white
##  6         1     526 white
##  7         1    2016 red
##  8         1    2376 white
##  9         1     292 red
## 10         1     440 white
```

Let's now take `virtual_samples` and compute the resulting 33 proportions red by using additionally `group_by()` of the `replicate` variable

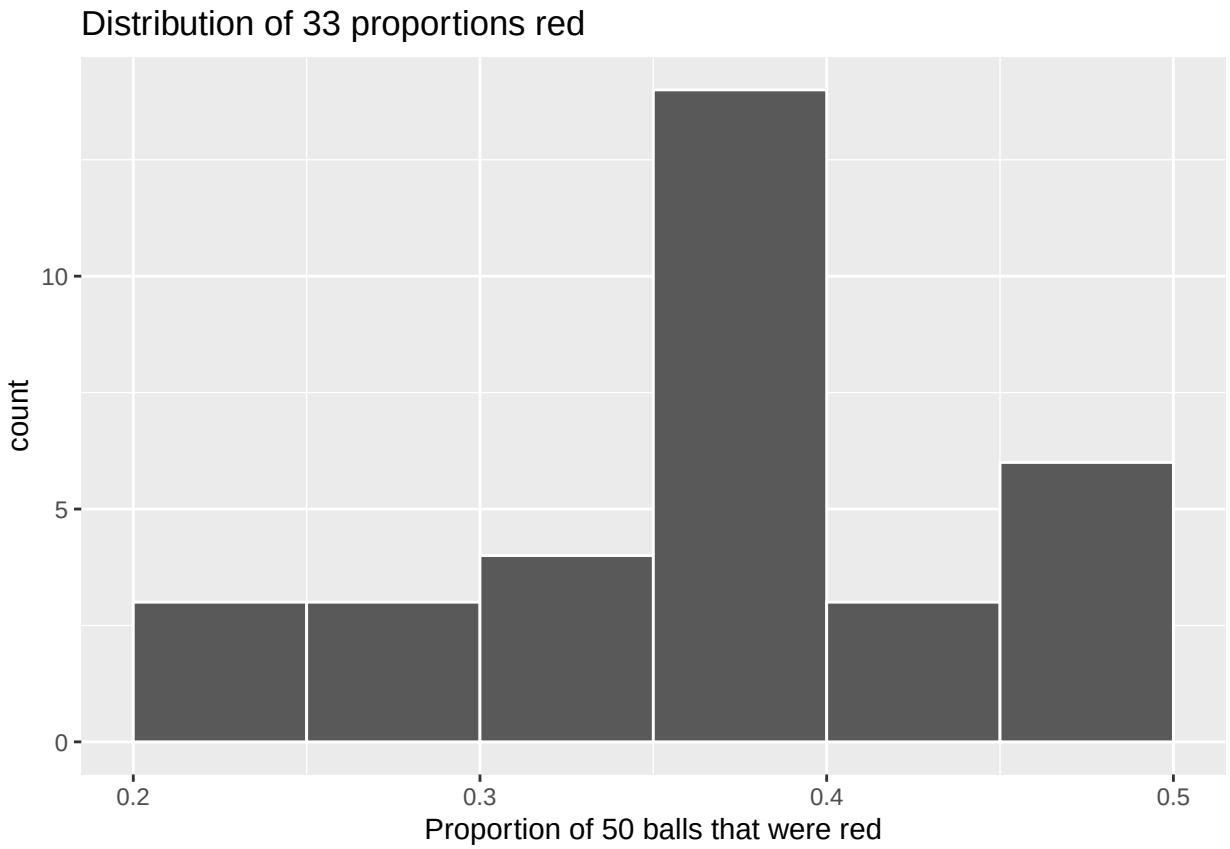
```r
# assign the result to a variable called "virtual_prop_red"
virtual_prop_red <- virtual_samples %>%
  group_by(replicate) %>%
  summarize(red = sum(color == "red")) %>%
  mutate(prop_red = red / 50)

virtual_prop_red
```

```
## # A tibble: 33 x 3
##    replicate   red prop_red
##        <int> <int>    <dbl>
##  1         1    19     0.38
##  2         2    23     0.46
##  3         3    19     0.38
##  4         4    21     0.42
##  5         5    16     0.32
##  6         6    17     0.34
##  7         7    22     0.44
##  8         8    19     0.38
```

```
##  9         9    20     0.4
## 10        10    18     0.36
## # ... with 23 more rows
```

Let's visualize this variation in a histogram in Figure 7.8. Note that we add `binwidth = 0.05` and `boundary = 0.4` arguments as well

```
ggplot(virtual_prop_red, aes(x = prop_red)) +
  geom_histogram(binwidth = 0.05, boundary = 0.4, color = "white") +
  labs(x = "Proportion of 50 balls that were red",
       title = "Distribution of 33 proportions red")
```



The most frequently occurring proportions were between 35% and 40%. Why do we have these differences in proportions red? Because of sampling variation

Let's now compare our virtual results with our tactile results from the previous section in Figure 7.9. Observe that both histograms are somewhat similar in their center and variation, although not identical. These slight differences are again due to random sampling variation. Furthermore, observe that both distributions are somewhat bell-shaped.

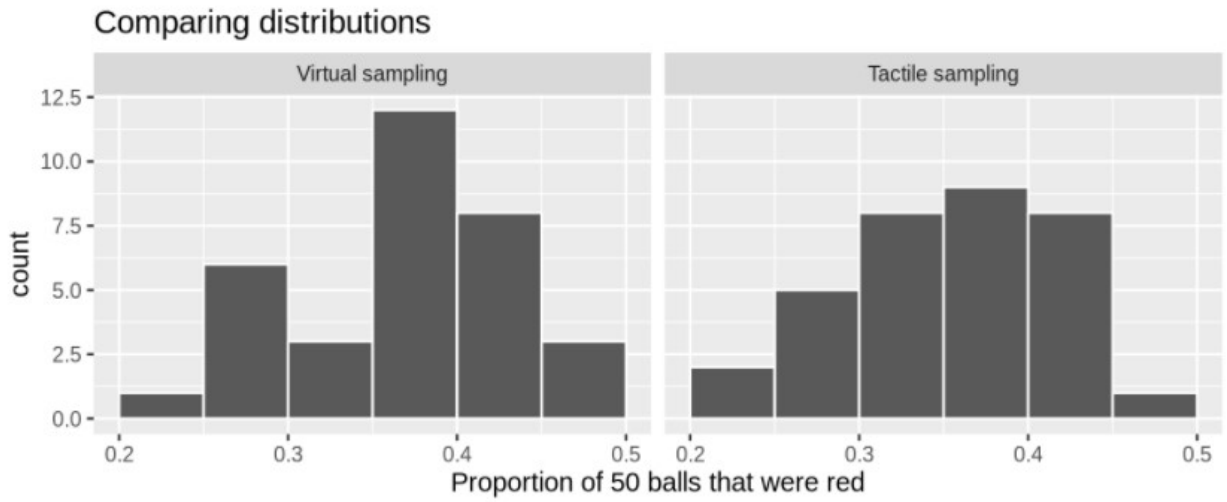


FIGURE 7.9: Comparing 33 virtual and 33 tactile proportions red.

1.2.3 Using the virtual shovel 1000 times

Let's once again use the `rep_sample_n()` function with sample `size` set to be 50 once again, but this time with the number of replicates `reps` set to `1000`

```
virtual_samples <- bowl %>%
  rep_sample_n(size = 50, reps = 1000)

glimpse(virtual_samples)
```

```
## Rows: 50,000
## Columns: 3
## Groups: replicate [1,000]
## $ replicate <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ...
## $ ball_ID   <int> 2018, 1009, 1269, 1549, 1299, 1244, 1134, 1542, 214, 1670...
## $ color     <chr> "white", "white", "red", "white", "red", "red", "white", ...
```

```
head(virtual_samples, 10)
```

```
## # A tibble: 10 x 3
## # Groups:   replicate [1]
##    replicate ball_ID color
##        <int>   <int> <chr>
##  1         1    2018 white
##  2         1    1009 white
##  3         1    1269 red
##  4         1    1549 white
```

```
##  5         1    1299 red
##  6         1    1244 red
##  7         1    1134 white
##  8         1    1542 white
##  9         1     214 white
## 10         1    1670 white
```

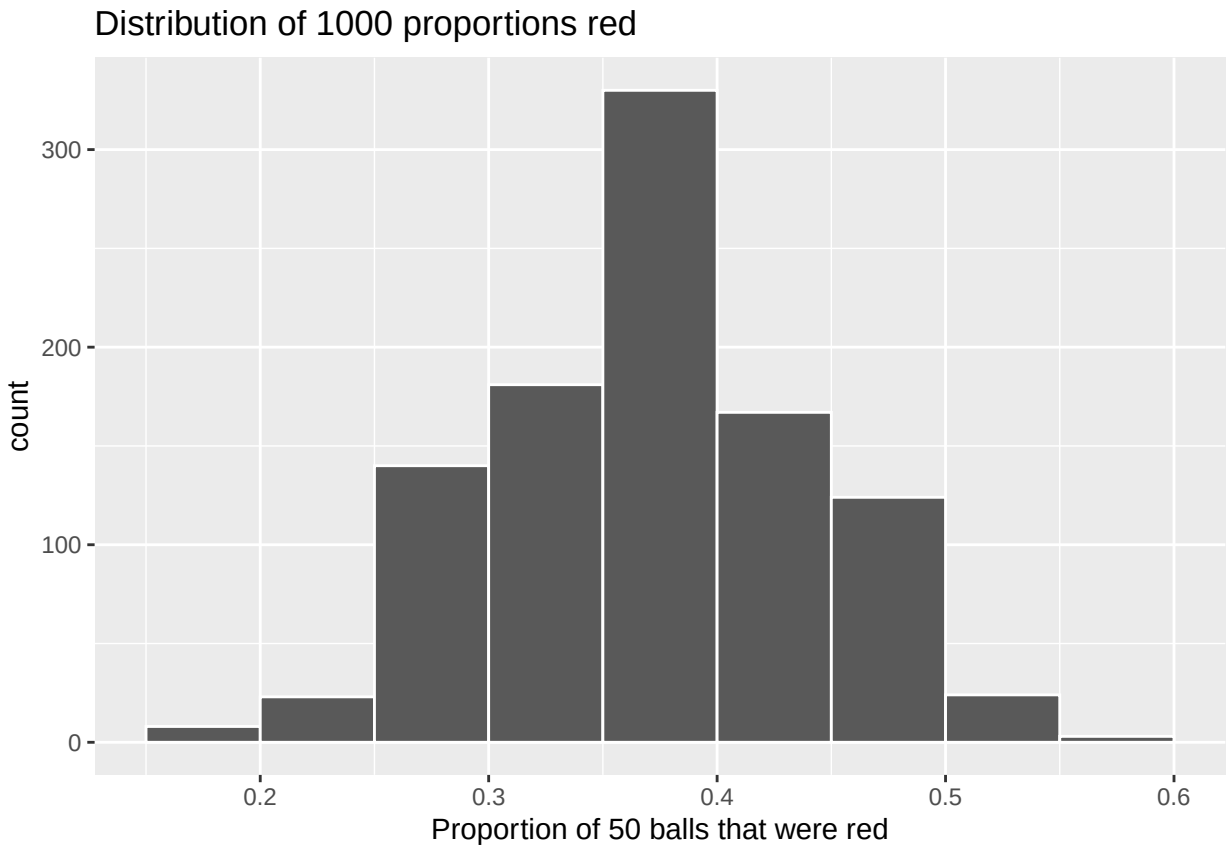
Compute the resulting 1000 proportions of red balls

```
virtual_prop_red <- virtual_samples %>%
  group_by(replicate) %>%
  summarize(red = sum(color == "red")) %>%
  mutate(prop_red = red / 50)

head(virtual_prop_red, 10)
```

```
## # A tibble: 10 x 3
##    replicate   red prop_red
##        <int> <int>    <dbl>
##  1         1    18     0.36
##  2         2    17     0.34
##  3         3    23     0.46
##  4         4    20     0.4
##  5         5    15     0.3
##  6         6    25     0.5
##  7         7    18     0.36
##  8         8    19     0.38
##  9         9    17     0.34
## 10        10    17     0.34
```

Let's now visualize the distribution of these 1000 replicates of `prop_red` in a histogram

```
ggplot(virtual_prop_red, aes(x = prop_red)) +
  geom_histogram(binwidth = 0.05, boundary = 0.4, color = "white") +
  labs(x = "Proportion of 50 balls that were red",
       title = "Distribution of 1000 proportions red")
```



The most frequently occurring proportions of red balls occur between 35% and 40%. Every now and then, we obtain proportions as low as between 20% and 25%, and others as high as between 55% and 60%. These are rare, however. Furthermore, observe that we now have a much more symmetric and smoother bell-shaped distribution

1.2.4 Using different shovels

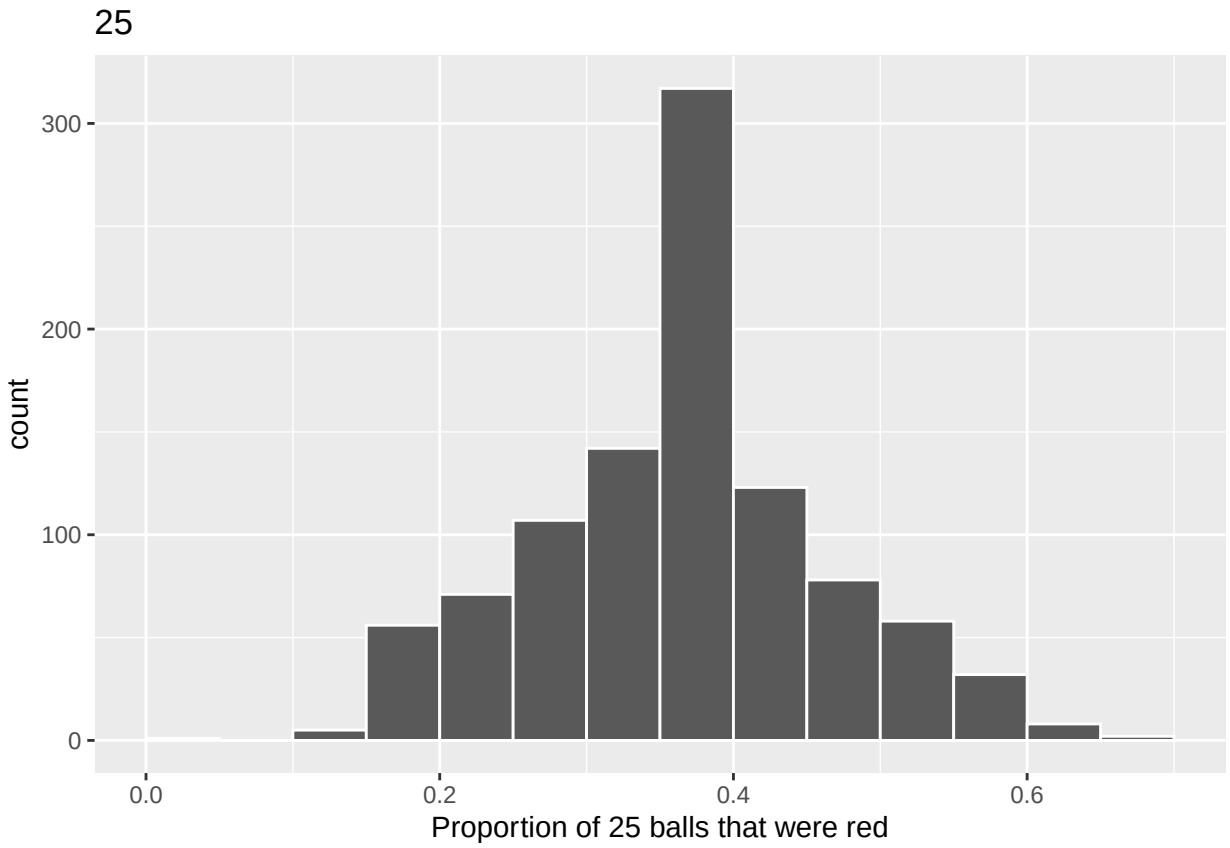
Now say instead of just one shovel, you have three choices of shovels to extract a sample of balls with: shovels of size 25, 50, and 100

Let's use `rep_sample_n()` with `size` set to `25`, `50`, and `100`, respectively, while keeping the number of repeated/replicated samples at 1000:

1. Virtually use the appropriate shovel to generate 1000 samples with size balls
2. Compute the resulting 1000 replicates of the proportion of the shovel's balls that are red
3. Visualize the distribution of these 1000 proportions red using a histogram

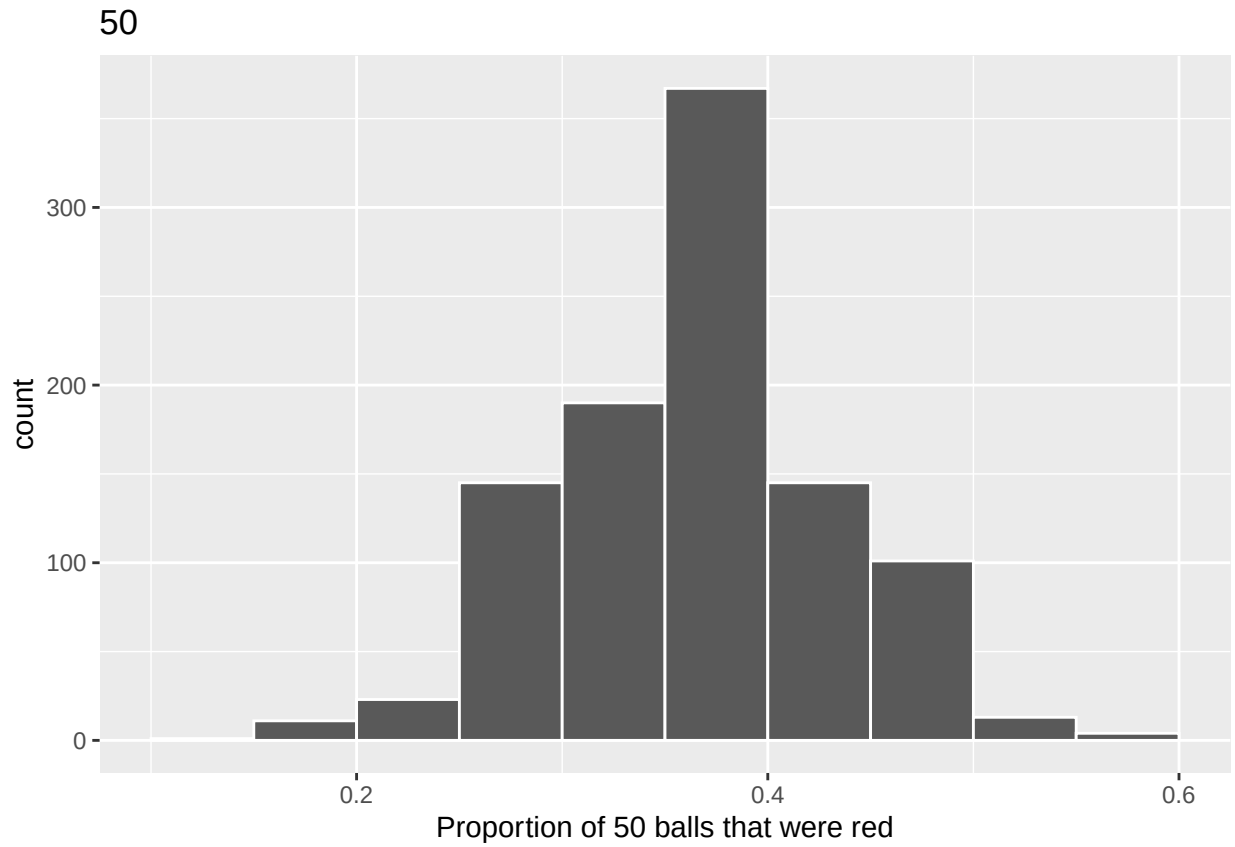
```
# Segment 1: sample size = 25------------------------------
#  1.a) Virtually use shovel 1000 times
virtual_samples_25 <- bowl %>%
  rep_sample_n(size = 25, reps = 1000)

# 1.b) Compute resulting 1000 replicates of proportion red
virtual_prop_red_25 <- virtual_samples_25 %>%
```

```
  group_by(replicate) %>%
  summarize(red = sum(color == "red")) %>%
  mutate(prop_red = red / 25)

# 1.c) Plot distribution via a histogram
ggplot(virtual_prop_red_25, aes(x = prop_red)) +
  geom_histogram(binwidth = 0.05, boundary = 0.4, color = "white") +
  labs(x = "Proportion of 25 balls that were red",
       title = "25")
```




```
# Segment 2: sample size = 50 ------------------------------
# 2.a) Virtually use shovel 1000 times
virtual_samples_50 <- bowl %>%
  rep_sample_n(size = 50, reps = 1000)

# 2.b) Compute resulting 1000 replicates of proportion red
virtual_prop_red_50 <- virtual_samples_50 %>%
  group_by(replicate) %>%
  summarize(red = sum(color == "red")) %>%
  mutate(prop_red = red / 50)

# 2.c) Plot distribution via a histogram
ggplot(virtual_prop_red_50, aes(x = prop_red)) +
  geom_histogram(binwidth = 0.05, boundary = 0.4, color = "white") +
  labs(x = "Proportion of 50 balls that were red",
```

```
       title = "50")
```




```
# Segment 3: sample size = 100 ------------------------------
# 3.a) Virtually using shovel with 100 slots 1000 times
virtual_samples_100 <- bowl %>%
  rep_sample_n(size = 100, reps = 1000)

# 3.b) Compute resulting 1000 replicates of proportion red
virtual_prop_red_100 <- virtual_samples_100 %>%
  group_by(replicate) %>%
  summarize(red = sum(color == "red")) %>%
  mutate(prop_red = red / 100)

# 3.c) Plot distribution via a histogram
ggplot(virtual_prop_red_100, aes(x = prop_red)) +
  geom_histogram(binwidth = 0.05, boundary = 0.4, color = "white") +
  labs(x = "Proportion of 100 balls that were red",
       title = "100")
```

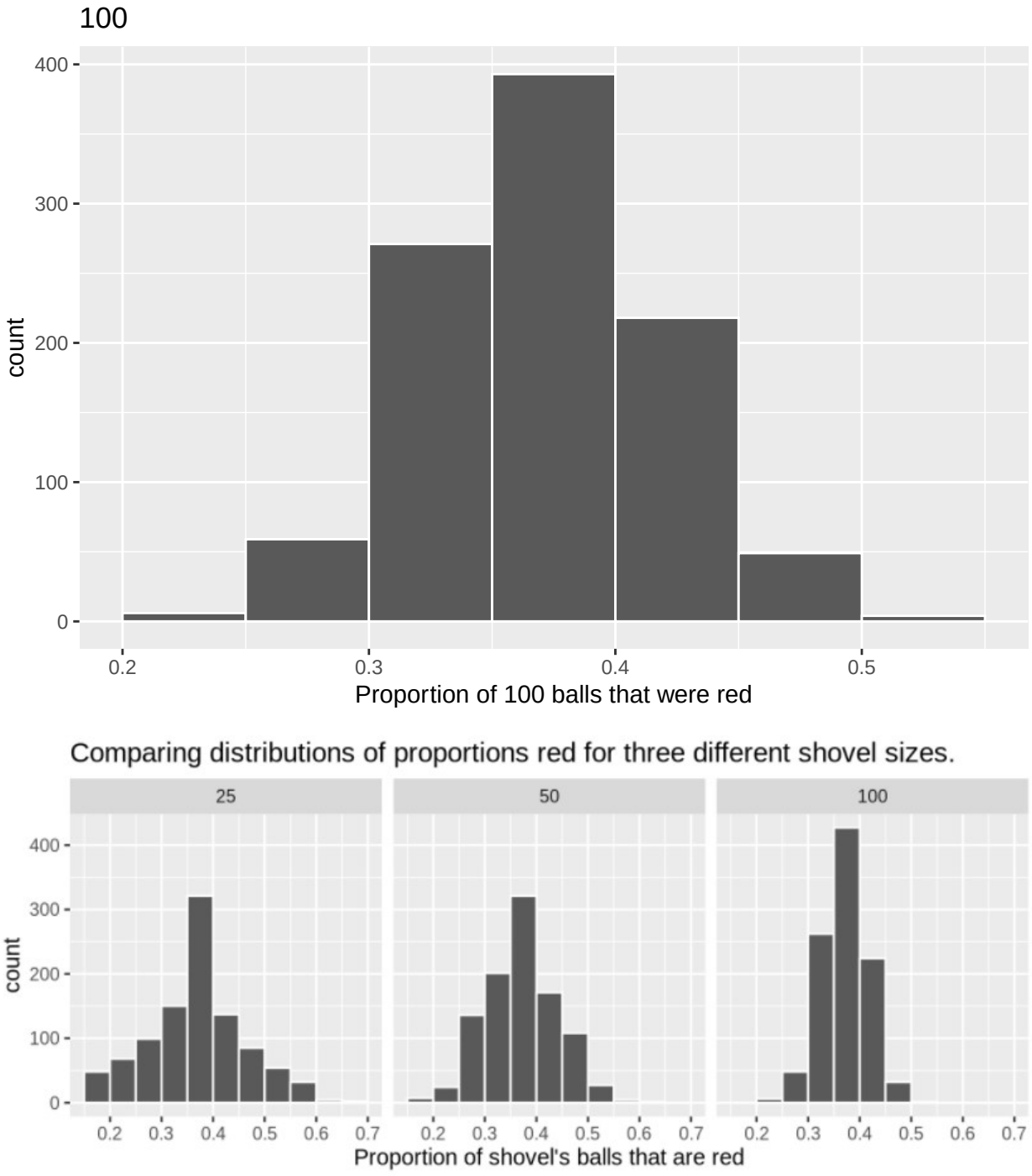


FIGURE 7.12: Comparing the distributions of proportion red for different sample sizes.

Observe that as the sample size increases, the variation of the 1000 replicates of the proportion of red decreases. In other words, as the sample size increases, there are fewer differences due to sampling variation and the distribution centers more tightly around the same value. Eyeballing Figure 7.12, all three histograms appear to center around roughly 40%

We can be numerically explicit about the amount of variation in our three sets of 1000 values of `prop_red` using the *standard deviation*. A standard deviation is a summary statistic that measures the amount of variation within a numerical variable

```r
# n = 25
virtual_prop_red_25 %>%
  summarize(sd = sd(prop_red))
```

```
## # A tibble: 1 x 1
##       sd
##    <dbl>
## 1 0.0994
```

```r
# n = 50
virtual_prop_red_50 %>%
  summarize(sd = sd(prop_red))
```

```
## # A tibble: 1 x 1
##       sd
##    <dbl>
## 1 0.0663
```

```r
# n = 100
virtual_prop_red_100 %>%
  summarize(sd = sd(prop_red))
```

```
## # A tibble: 1 x 1
##       sd
##    <dbl>
## 1 0.0474
```

Table 1: Comparing standard deviations of proportions red for three different shovels

Number of slots in shovel	Standard Deviation of proportions red
25	0.096
50	0.069
100	0.047

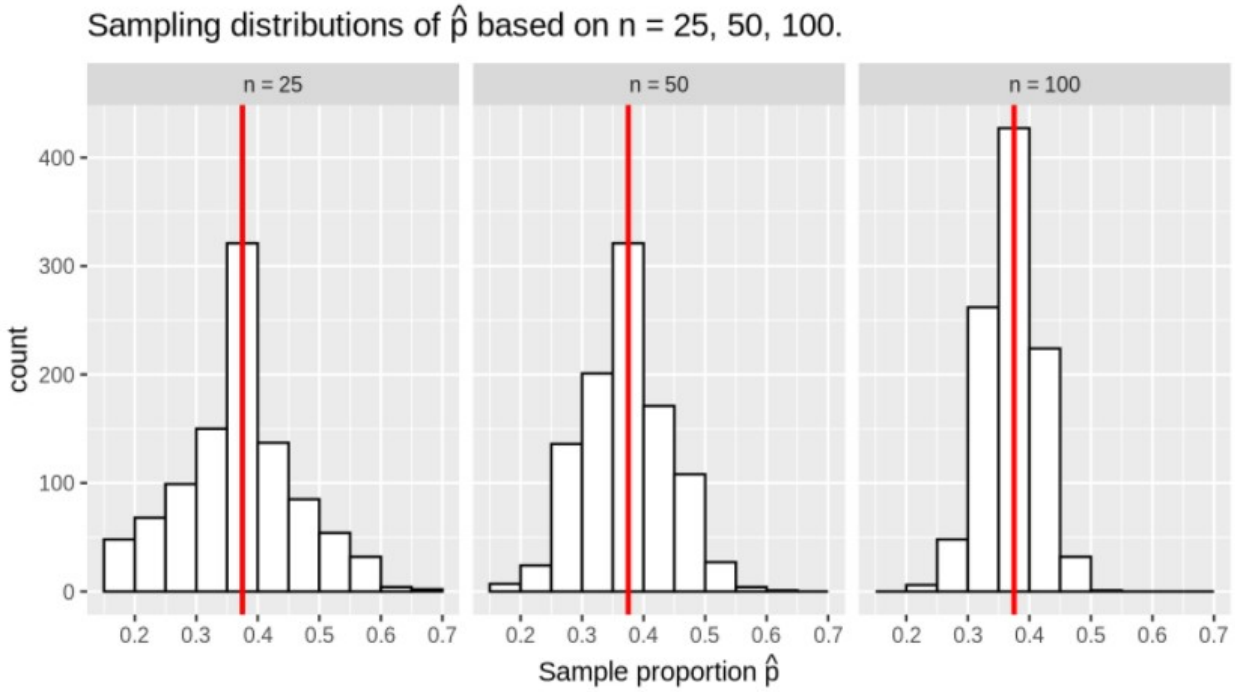
As we observed in Figure 7.12 & Table 1, as the sample size increases, the variation decreases. In other words, there is less variation in the 1000 values of the proportion red. So as the sample size increases, our guesses at the true proportion of the bowl's balls that are red get more precise

What was the value of the population proportion p of the $N = 2400$ balls in the actual bowl that were red? There were 900 red balls, for a proportion red of $900/2400 = 0.375 = 37.5\%$! How do we know this? Did the authors do an exhaustive count of all the balls? No! They were listed in the contents of the box that the bowl came in! Hence we were able to make the contents of the virtual `bowl` match the tactile bowl:

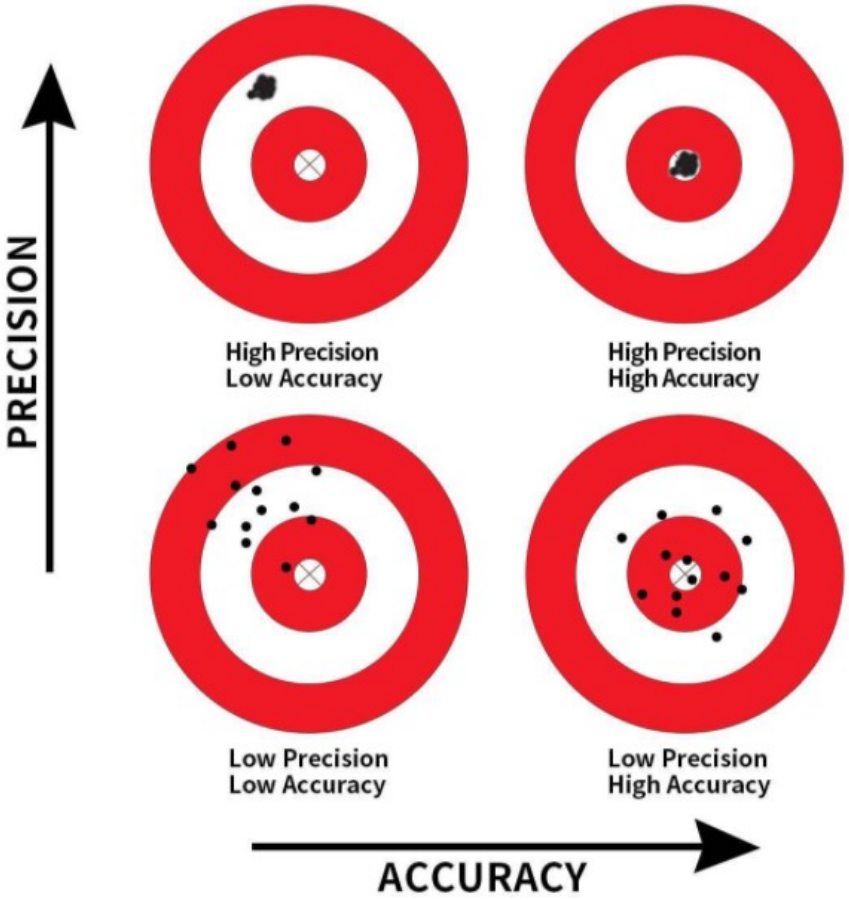
```r
bowl %>%
  summarize(sum_red = sum(color == "red"),
            sum_not_red = sum(color != "red"))
```

```
## # A tibble: 1 x 2
##   sum_red sum_not_red
##     <int>       <int>
## 1     900        1500
```

Let's re-display our sampling distributions from Figures 7.12 and 7.14, but now with a vertical red line marking the true population proportion p of balls that are red $= 37.5\%$ in Figure 7.15. We see that while there is a certain amount of error in the sample proportions $\hat{p}$ for all three sampling distributions, on average the $\hat{p}$ are centered at the true population proportion red p.



So random sampling ensures our point estimates are *accurate*, while on the other hand having a large sample size ensures our point estimates are *precise*. While the terms "accuracy" and "precision" may sound like they mean the same thing, there is a subtle difference. Accuracy describes how "on target" our estimates are, whereas precision describes how "consistent" our estimates are. Figure 7.16 illustrates the difference.



The sampling activity involving the bowl is merely an idealized version of how sampling is done in real life. We performed this exercise only to study and understand:

1. The effect of sampling variation.
2. The effect of sample size on sampling variation.

This is not how sampling is done in real life. In a real-life scenario, we won't know what the true value of the population parameter is. Furthermore, we wouldn't take 1000 repeated/replicated samples, but rather a single sample that's as large as we can afford.