

Relative error of point estimates

SAMPLING IN R



Richie Cotton

Data Evangelist at DataCamp

Sample is number of rows

```
coffee_ratings %>%  
  slice_sample(n = 300) %>%  
  nrow()
```

300

```
coffee_ratings %>%  
  slice_sample(prop = 0.25) %>%  
  nrow()
```

334

Various sample sizes

```
coffee_ratings %>%  
  summarize(mean_points = mean(total_cup_points)) %>%  
  pull(mean_points)
```

82.15

```
coffee_ratings %>%  
  slice_sample(n = 10) %>%  
  summarize(mean_points = mean(total_cup_points)) %>%  
  pull(mean_points)
```

82.82

```
coffee_ratings %>%  
  slice_sample(n = 100) %>%  
  summarize(mean_points = mean(total_cup_points)) %>%  
  pull(mean_points)
```

82.02

```
coffee_ratings %>%  
  slice_sample(n = 1000) %>%  
  summarize(mean_points = mean(total_cup_points)) %>%  
  pull(mean_points)
```

82.16

Relative errors

Population parameter

```
population_mean <- coffee_ratings %>%  
  summarize(mean_points = mean(total_cup_points)) %>%  
  pull(mean_points)
```

Point estimate

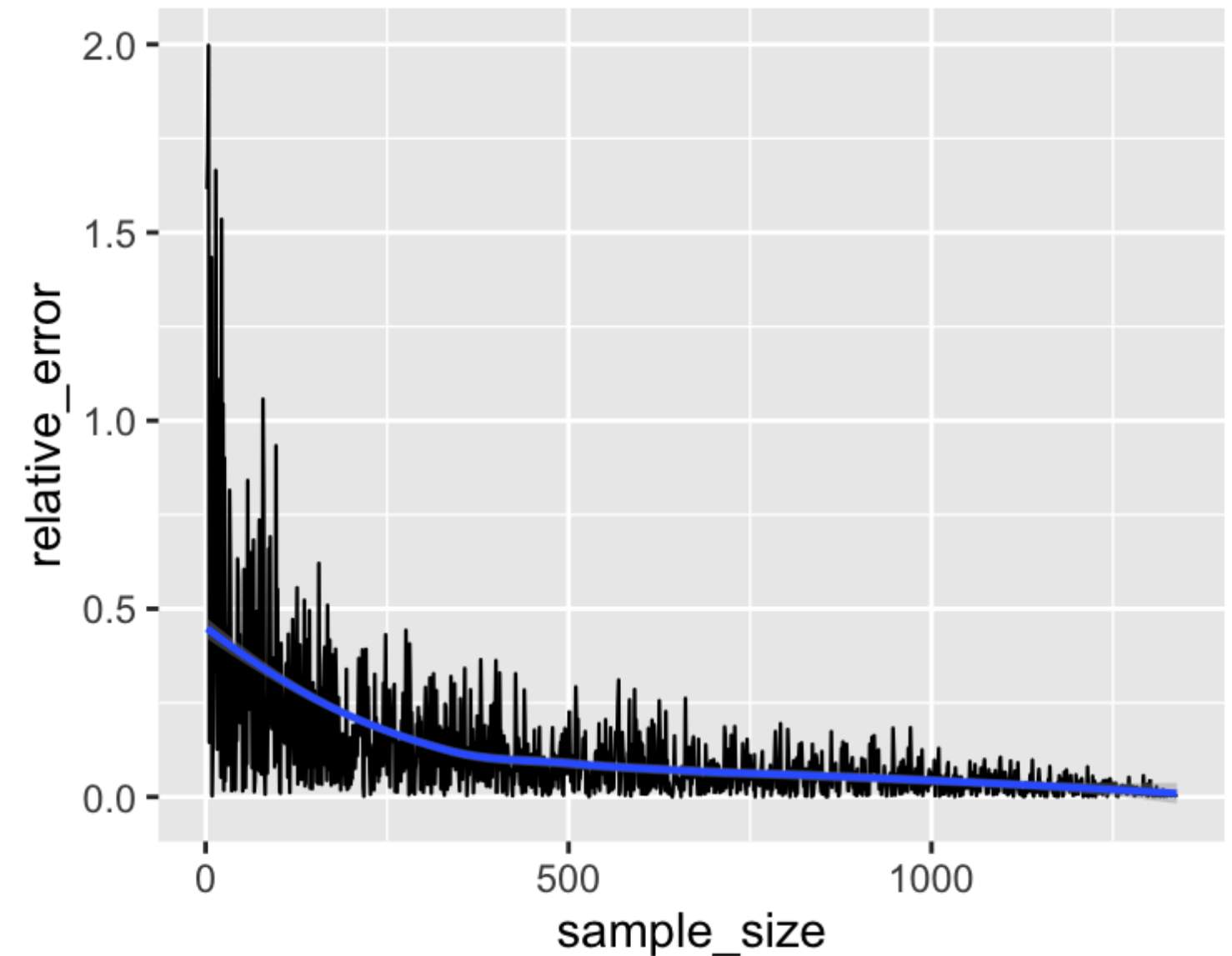
```
sample_mean <- coffee_ratings %>%  
  slice_sample(n = sample_size) %>%  
  summarize(mean_points = mean(total_cup_points)) %>%  
  pull(mean_points)
```

Relative error as a percentage

```
100 * abs(population_mean - sample_mean) / population_mean
```

Relative error vs. sample size

```
ggplot(errors, aes(sample_size, relative_error)) +  
  geom_line() +  
  geom_smooth(method = "loess")
```



Let's practice!
SAMPLING IN R

Creating a sampling distribution

SAMPLING IN R



Richie Cotton

Data Evangelist at DataCamp

Same code, different answer

```
coffee_ratings %>%  
  slice_sample(n = 30) %>%  
  summarize(mean_cup_points = mean(total_cup_points)) %>%  
  pull(mean_cup_points)
```

83.33

```
coffee_ratings %>%  
  slice_sample(n = 30) %>%  
  summarize(mean_cup_points = mean(total_cup_points)) %>%  
  pull(mean_cup_points)
```

82.16

```
coffee_ratings %>%  
  slice_sample(n = 30) %>%  
  summarize(mean_cup_points = mean(total_cup_points)) %>%  
  pull(mean_cup_points)
```

82.59

```
coffee_ratings %>%  
  slice_sample(n = 30) %>%  
  summarize(mean_cup_points = mean(total_cup_points)) %>%  
  pull(mean_cup_points)
```

82.25

Same code, 1000 times

```
mean_cup_points_1000 <- replicate(  
  n = 1000,  
  expr = coffee_ratings %>%  
    slice_sample(n = 30) %>%  
    summarize(  
      mean_cup_points = mean(total_cup_points)  
    ) %>%  
    pull(mean_cup_points)  
)
```

```
[1] 81.65 81.57 82.66 82.27 81.76 81.74 82.71  
[8] 82.20 80.43 82.45 82.29 82.63 82.28 82.11  
[15] 82.14 81.72 81.97 82.58 81.78 82.47 81.73  
[22] 82.78 82.14 82.39 81.69 82.36 82.64 82.68  
[29] 82.56 82.14 82.72 82.43 81.68 82.74 82.80  
[36] 82.12 82.31 81.02 82.83 81.71 82.25 82.11  
[43] 82.76 82.26 81.57 82.00 81.75 81.47 81.99  
[50] 82.68 82.05 82.43 82.40 82.66 80.78 82.43  
...  
[967] 81.84 83.12 81.54 81.83 82.24 82.36 82.49  
[974] 82.05 82.08 81.98 82.45 82.04 81.42 83.06  
[981] 81.97 82.65 81.12 82.48 81.64 81.92 81.96  
[988] 81.71 81.96 81.78 82.30 81.76 82.46 82.43  
[995] 81.95 82.60 81.84 82.78 82.23 82.56
```

Preparing for plotting

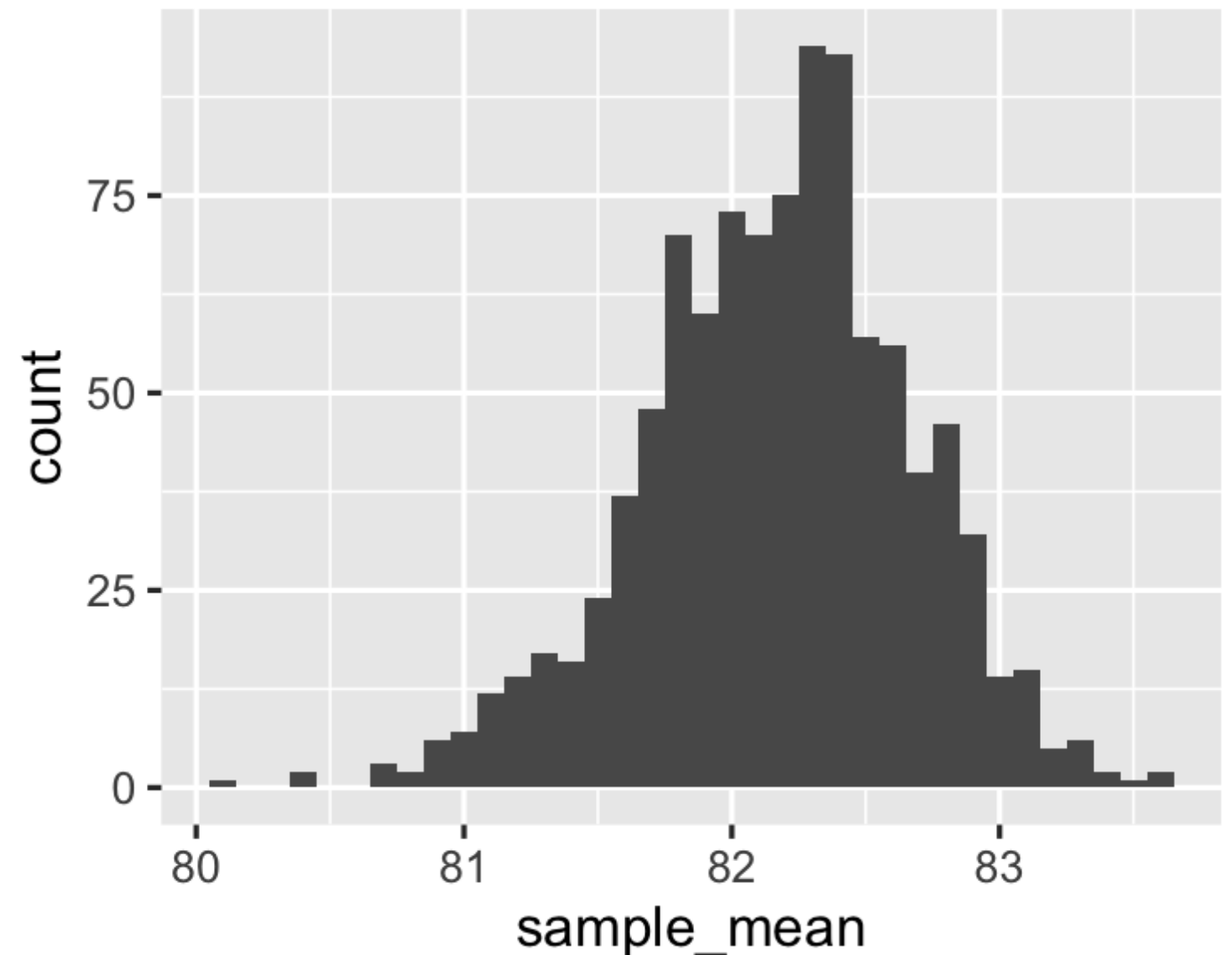
```
library(tibble)
sample_means <- tibble(
  sample_mean = mean_cup_points_1000
)
```

```
# A tibble: 1,000 x 1
  sample_mean
      <dbl>
1      83.3
2      82.6
3      82.2
4      82.2
5      81.7
6      81.6
7      82.7
8      82.3
9      81.8
10     81.7
# ... with 990 more rows
```

Distribution of sample means for size 30

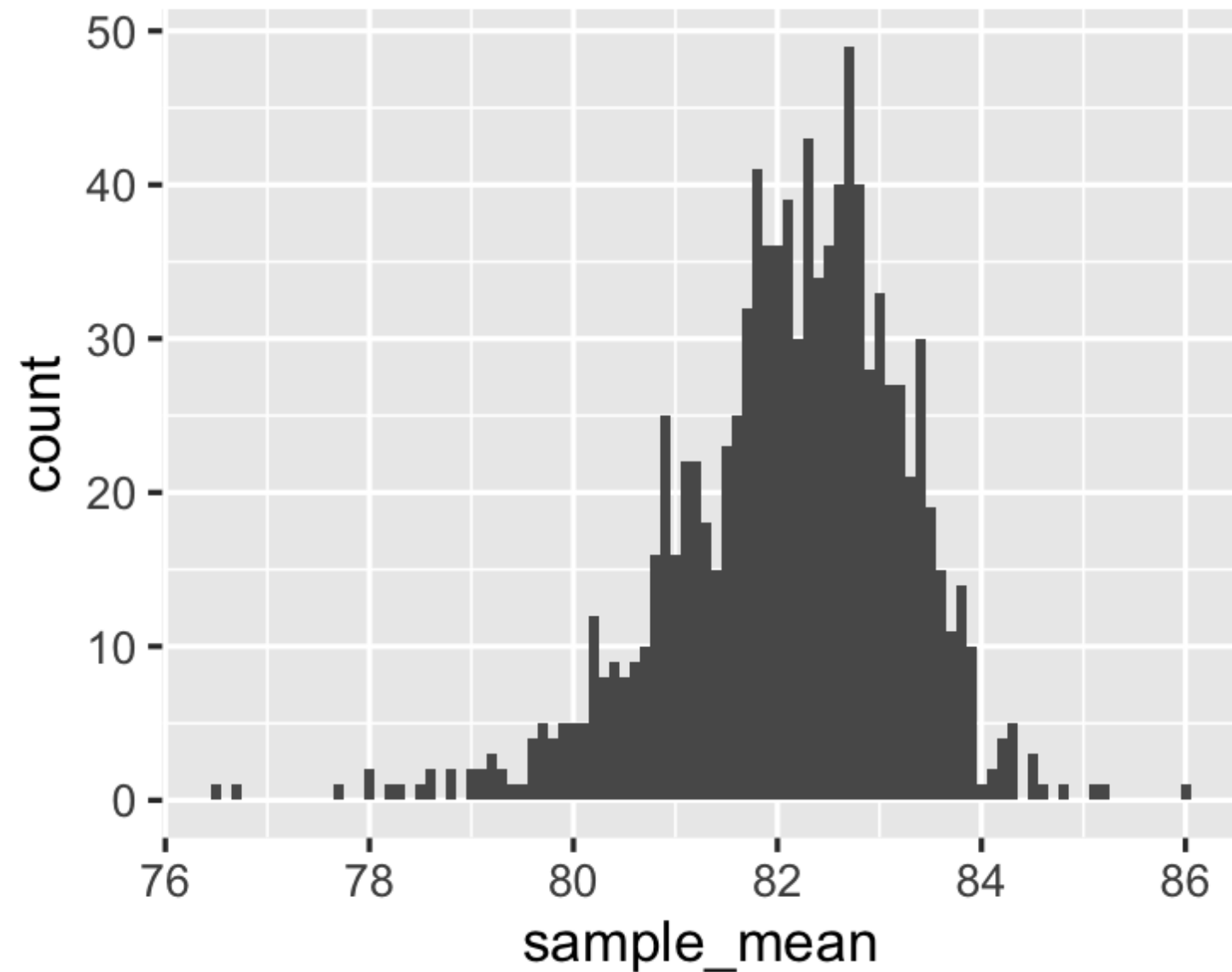
```
ggplot(sample_means, aes(sample_mean)) +  
  geom_histogram(binwidth = 0.1)
```

A *sampling distribution* is a distribution of several replicates of point estimates.

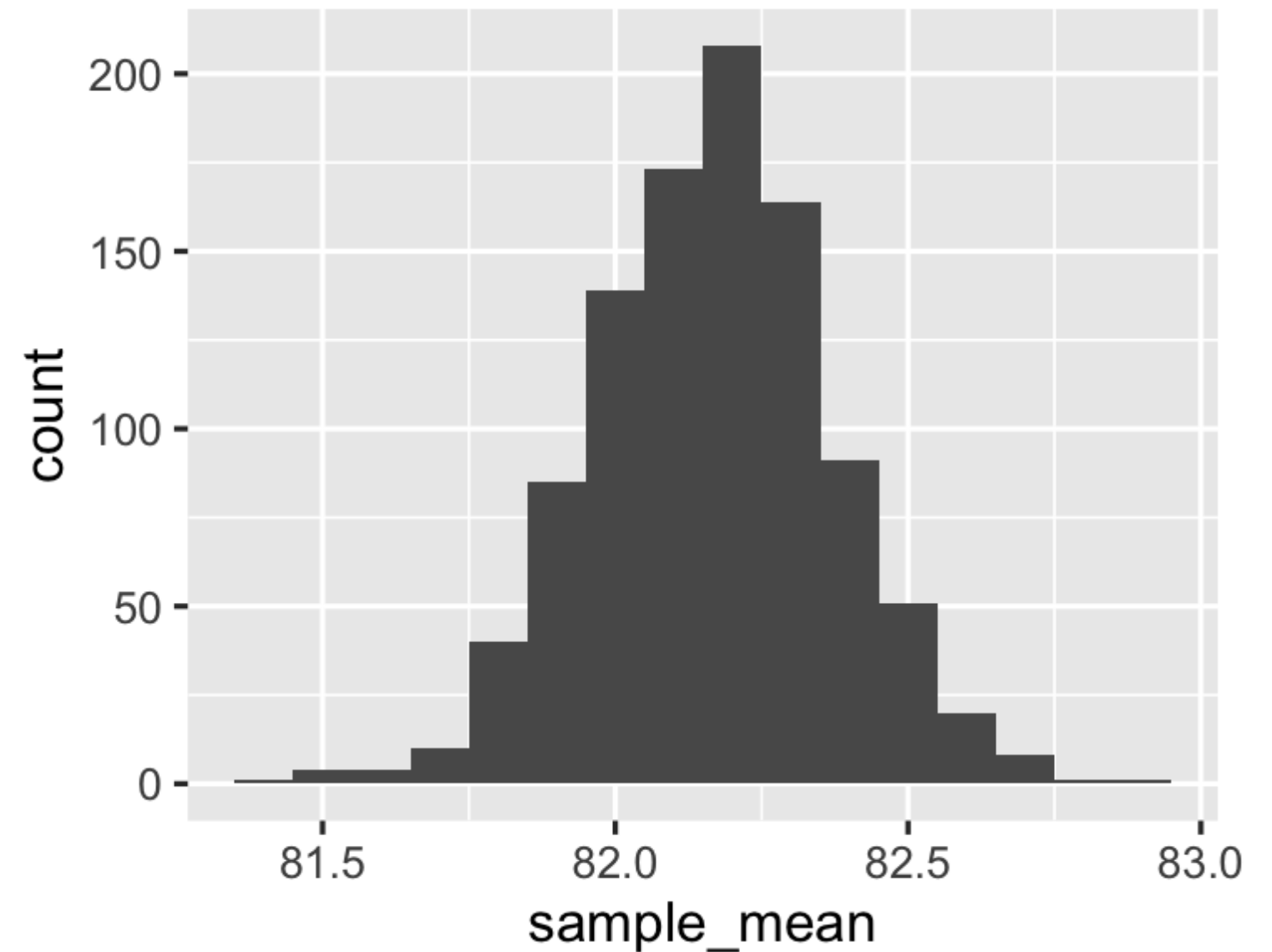


Different sample sizes

Sample size 6



Sample size 150



Let's practice!
SAMPLING IN R

Approximate sampling distributions

SAMPLING IN R



Richie Cotton

Data Evangelist at DataCamp

4 dice



```
library(tidyr)
dice <- expand_grid(
  die1 = 1:6,
  die2 = 1:6,
  die3 = 1:6,
  die4 = 1:6
)
```

```
# A tibble: 1,296 x 4
  die1 die2 die3 die4
  <int> <int> <int> <int>
1     1     1     1     1
2     1     1     1     2
3     1     1     1     3
4     1     1     1     4
5     1     1     1     5
6     1     1     1     6
7     1     1     2     1
8     1     1     2     2
9     1     1     2     3
10    1     1     2     4
# ... with 1,286 more rows
```

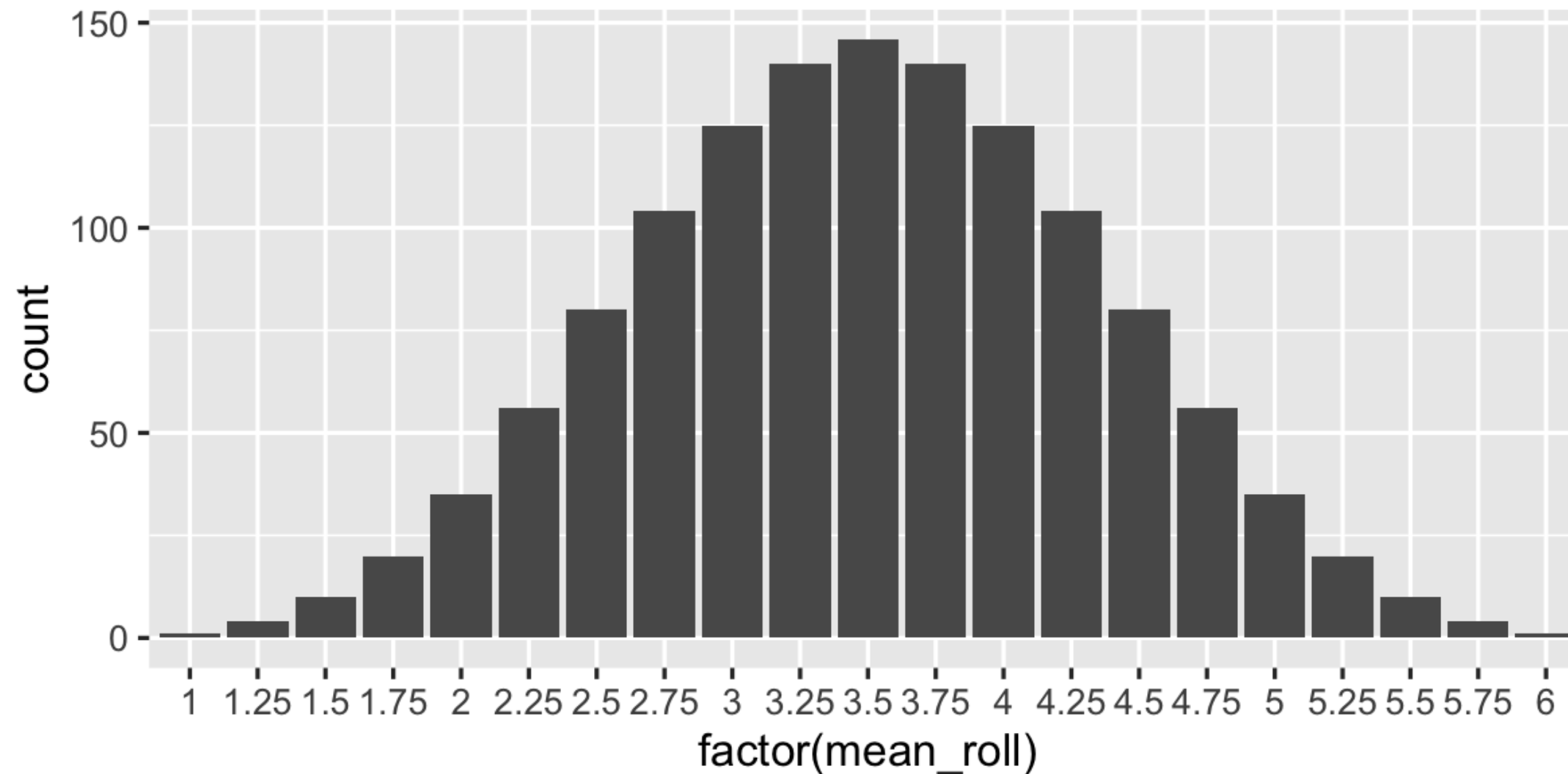
Mean roll

```
dice <- expand_grid(
  die1 = 1:6,
  die2 = 1:6,
  die3 = 1:6,
  die4 = 1:6
) %>%
  mutate(
    mean_roll = (die1 + die2 + die3 + die4) / 4
  )
```

```
# A tibble: 1,296 x 5
   die1 die2 die3 die4 mean_roll
   <int> <int> <int> <int>     <dbl>
1     1     1     1     1         1
2     1     1     1     2        1.25
3     1     1     1     3        1.5
4     1     1     1     4        1.75
5     1     1     1     5         2
6     1     1     1     6        2.25
7     1     1     2     1        1.25
8     1     1     2     2        1.5
9     1     1     2     3        1.75
10    1     1     2     4         2
# ... with 1,286 more rows
```


Exact sampling distribution

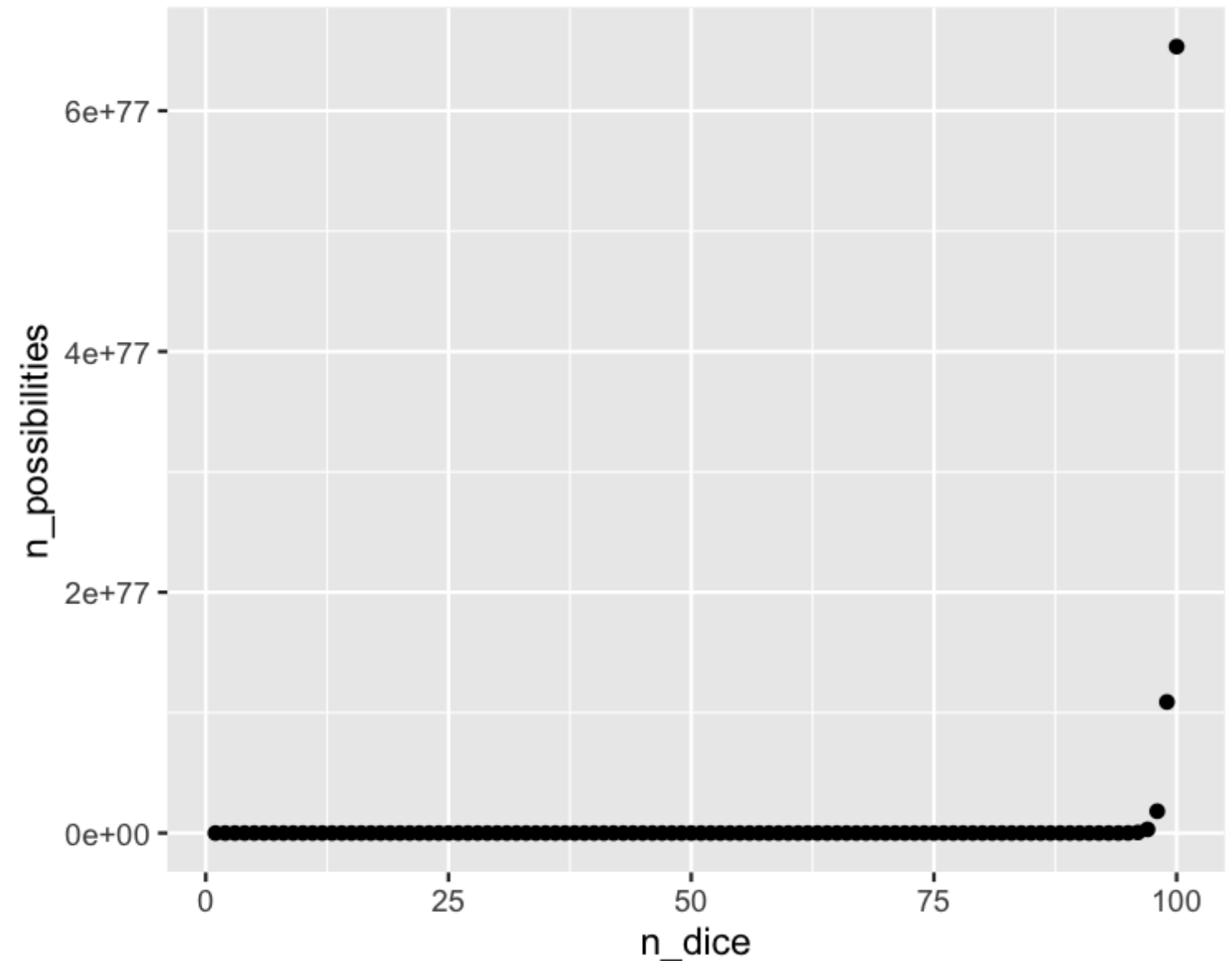
```
ggplot(dice, aes(factor(mean_roll))) +  
  geom_bar()
```



The number of outcomes increases fast

```
outcomes <- tibble(  
  n_dice = 1:100,  
  n_outcomes = 6 ^ n_dice  
)
```

```
ggplot(outcomes, aes(n_dice, n_outcomes)) +  
  geom_point()
```



Simulating the mean of four dice rolls

```
four_rolls <- sample(  
  1:6, size = 4, replace = TRUE  
)  
mean(four_rolls)
```

Simulating the mean of four dice rolls

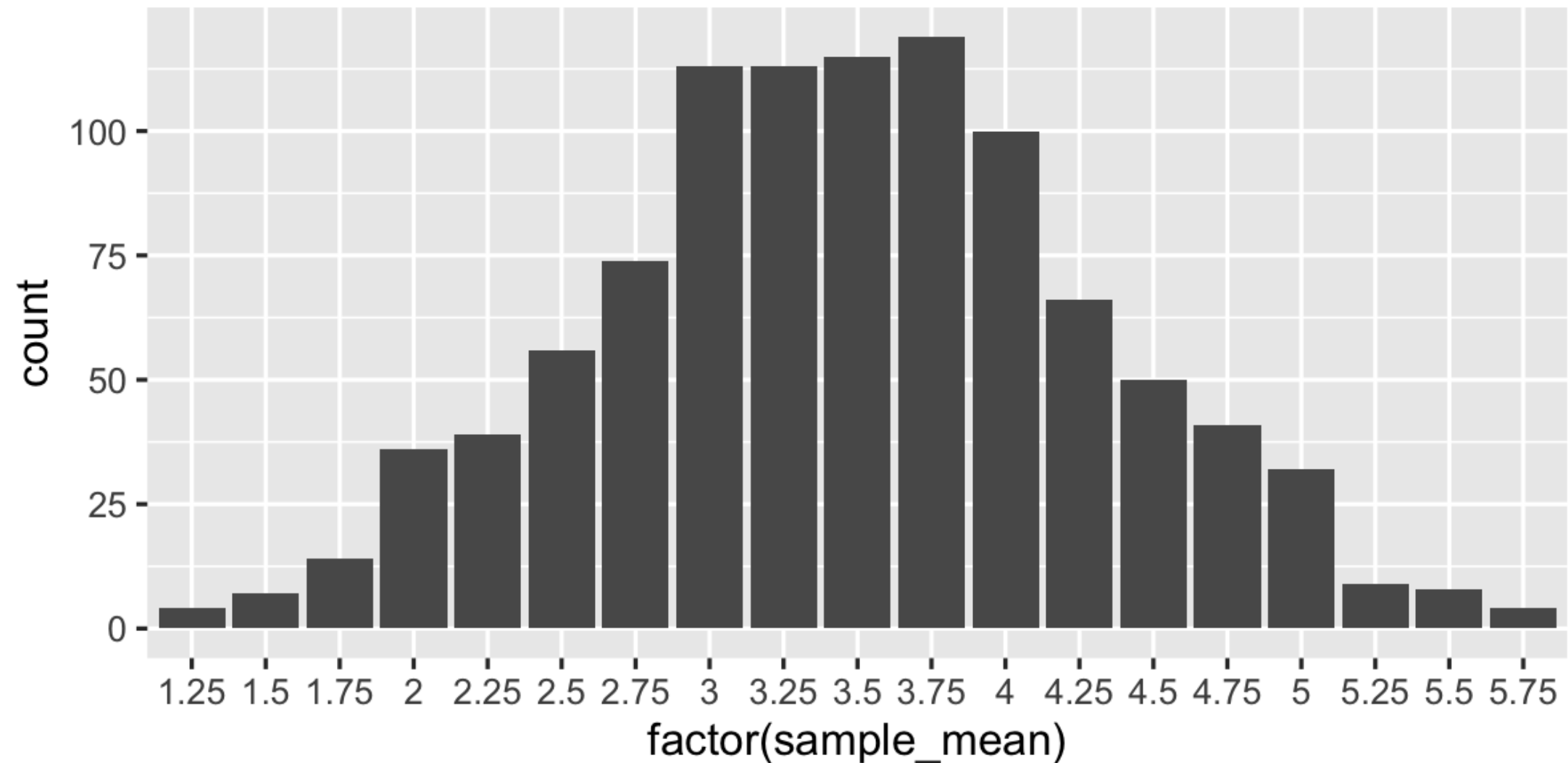
```
sample_means_1000 <- replicate(  
  n = 1000,  
  expr = {  
    four_rolls <- sample(  
      1:6, size = 4, replace = TRUE  
    )  
    mean(four_rolls)  
  }  
)
```

```
sample_means <- tibble(  
  sample_mean = sample_means_1000  
)
```

```
# A tibble: 1,000 x 1  
  sample_mean  
      <dbl>  
1         4  
2        4.5  
3        2.5  
4        3.75  
5        3.75  
6         4  
7         3  
8        4.75  
9        3.75  
10       4.25  
# ... with 990 more rows
```

Approximate sampling distribution

```
ggplot(sample_means, aes(factor(sample_mean))) +  
  geom_bar()
```



Let's practice!
SAMPLING IN R

Standard errors and the Central Limit Theorem

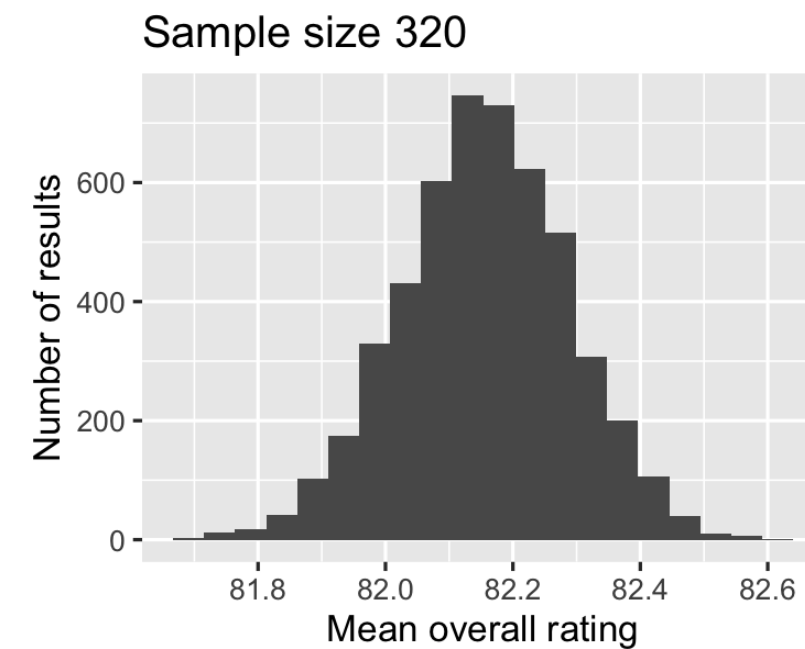
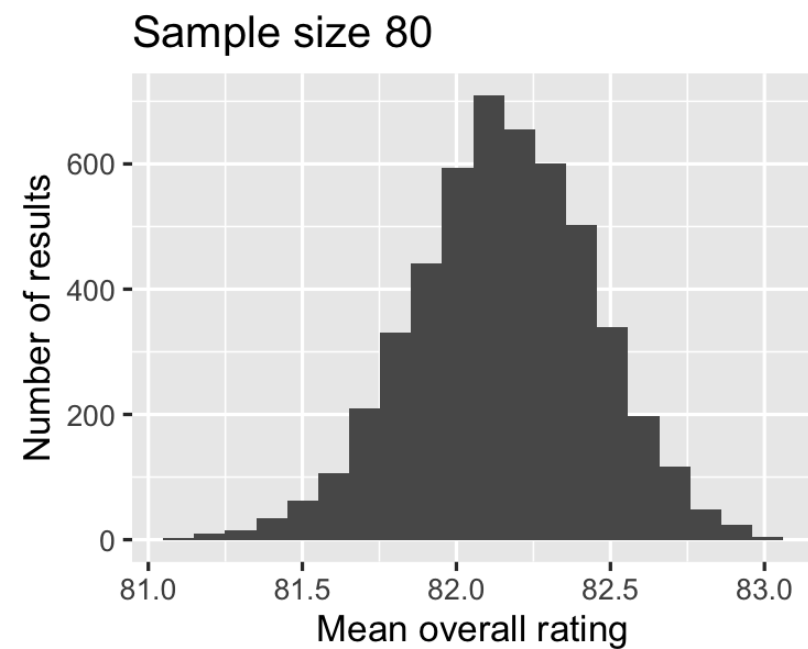
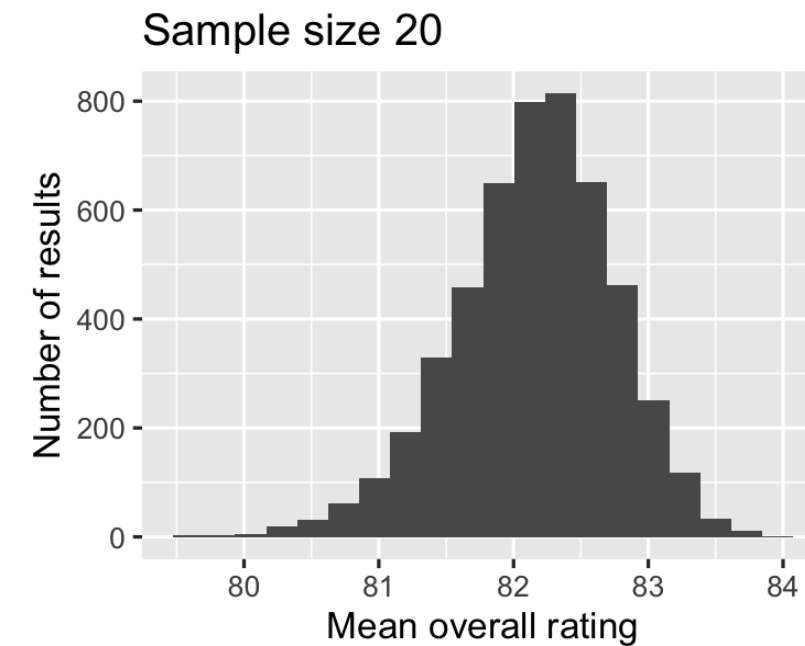
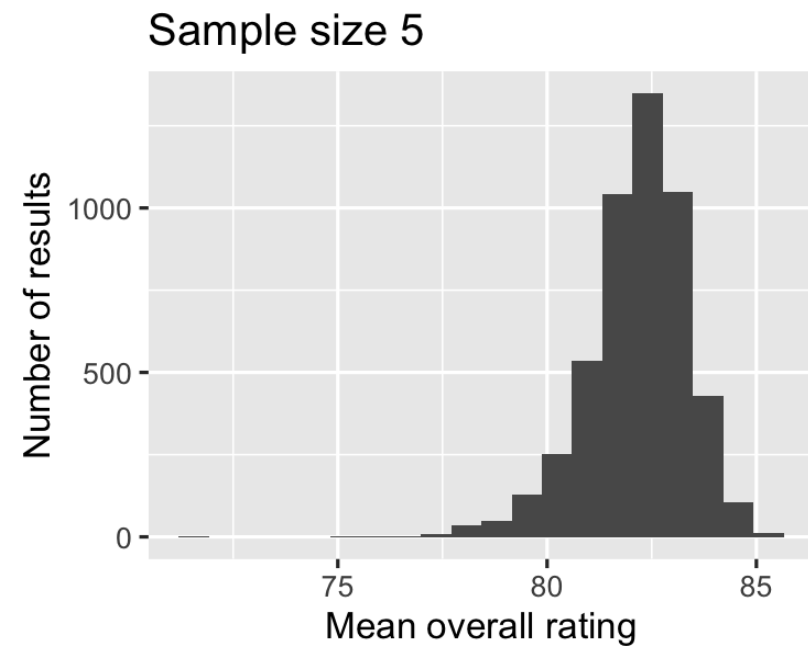
SAMPLING IN R



Richie Cotton

Data Evangelist at DataCamp

Sampling distribution of mean cup points



Consequences of the central limit theorem

- Averages of independent samples have approximately normal distributions.

As the sample size increases,

- the distribution of the averages gets closer to being normally distributed, and
- the width of the sampling distribution gets narrower.

Population & sampling distribution means

```
coffee_ratings %>%  
  summarize(  
    mean_cup_points = mean(total_cup_points)  
  ) %>%  
  pull(mean_cup_points)
```

82.1512

Sample size	Mean sample mean
5	82.1496
20	82.1610
80	82.1496
320	82.1521

Population & sampling distribution standard deviations

```
coffee_ratings %>%  
  summarize(  
    sd_cup_points = sd(total_cup_points)  
  ) %>%  
  pull(sd_cup_points)
```

2.68686

Sample size	Std dev sample mean
5	1.1929
20	0.6028
80	0.2865
320	0.1304

Population mean over square root sample size

Sample size	Std dev sample mean	Calculation	Result
5	1.1929	<code>2.68686 / sqrt(5)</code>	1.2016
20	0.6028	<code>2.68686 / sqrt(20)</code>	0.6008
80	0.2865	<code>2.68686 / sqrt(80)</code>	0.3004
320	0.1304	<code>2.68686 / sqrt(320)</code>	0.1502

Let's practice!
SAMPLING IN R