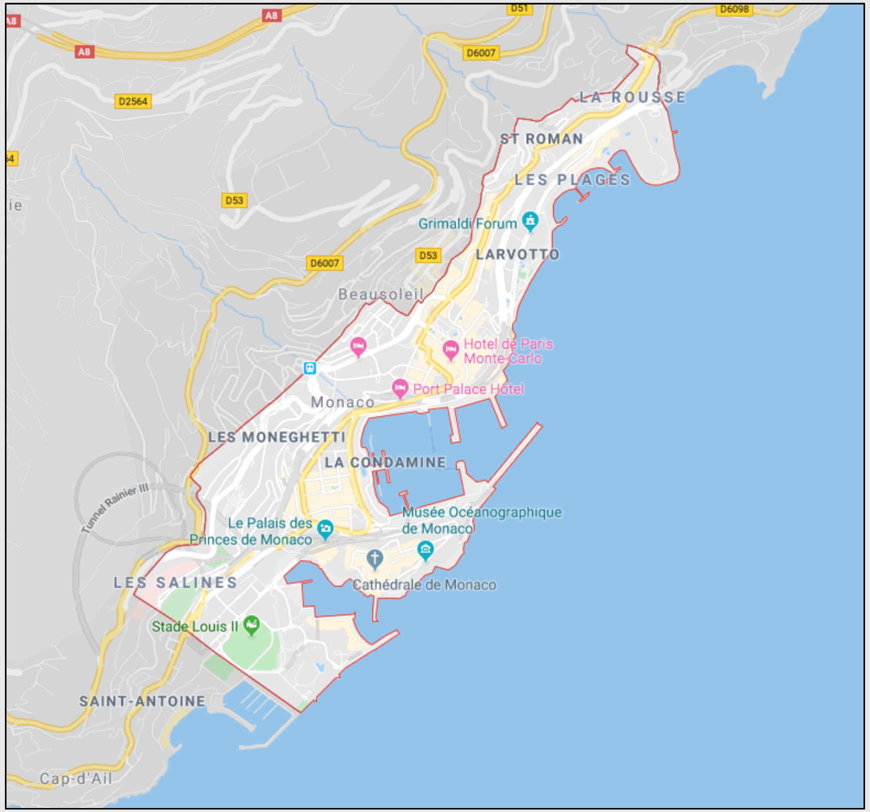
Week 15: *Monte Carlo Methods*

EMSE 4574: Intro to Programming for Analytics

John Paul Helveston

December 08, 2020

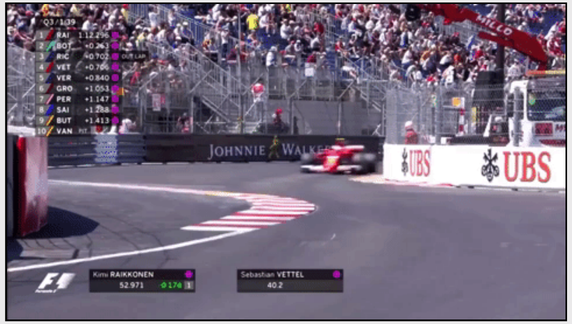
Monte Carlo, Monaco



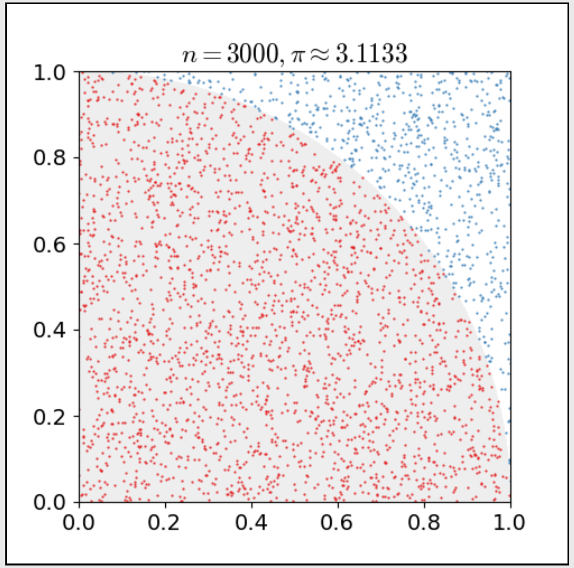
"Monte Carlo" is associated with 3 things

Gambling

Racing



Simulation



Week 15: *Monte Carlo Methods*

1. Monte Carlo Simulation
2. Monte Carlo Integration

Week 15: *Monte Carlo Methods*

1. Monte Carlo Simulation
2. Monte Carlo Integration

Monte Carlo Simulation: *Computing Probability*

General process:

- Run a series of trials.
- In each trial, simulate an event (e.g. a coin toss, a dice roll, etc.).
- Count the number of "successful" trials

$\frac{\#\ \text{Successful Trials}}{\#\ \text{Total Trials}}$ = Observed Odds $\simeq$ Expected Odds

Law of large numbers:
As *N* increases, Observed Odds » Expected Odds

How would you measure if a coin is "fair"?

Run a series of trials and record outcome: "heads" or "tails"

```r
coin <- c("heads", "tails")
N <- 10000
tosses <- sample(x = coin, size = N, replace = TRUE)
head(tosses) # Preview first few tosses
```

```
#> [1] "tails" "tails" "tails" "heads" "heads" "tails"
```

Probability of getting "heads":

```r
sum(tosses == "heads") / N
```

```
#> [1] 0.5053
```

Tossing an unfair coin

Set the `prob` argument to a 40-60 coin

```r
coin <- c("heads", "tails")
N <- 10000
tosses <- sample(x = coin, size = N, replace = TRUE, prob = c(0.4, 0.6))
head(tosses) # Preview first few tosses
```

```
#> [1] "heads" "tails" "tails" "tails" "tails" "tails"
```

Probability of getting "heads":

```r
sum(tosses == "heads") / N
```

```
#> [1] 0.3964
```

A more complex simulation: *dice rolling*

What is the probability of rolling a 6-sided dice 3 times and getting the sequence 1, 3, 5?

```r
library(tidyverse)
dice <- c(1, 2, 3, 4, 5, 6)
N <- 10000
rolls <- tibble(
  roll1 = sample(x = dice, size = N, replace = T),
  roll2 = sample(x = dice, size = N, replace = T),
  roll3 = sample(x = dice, size = N, replace = T)
)
```

```r
dim(rolls)
```

```
#> [1] 10000     3
```

```r
head(rolls)
```

```
#> # A tibble: 6 x 3
#>   roll1 roll2 roll3
#>   <dbl> <dbl> <dbl>
#> 1     2     4     1
#> 2     5     4     3
#> 3     4     6     5
#> 4     3     3     4
#> 5     5     5     4
```

A more complex simulation: *dice rolling*

Simulated probability of getting sequence 1, 3, 5:

```
successes <- rolls %>%
  filter(roll1 == 1, roll2 == 3, roll3 == 5)
nrow(successes) / N
```

```
#> [1] 0.0037
```

Actual probability of getting sequence 1, 3, 5:

```
(1/6)^3
```

```
#> [1] 0.00462963
```

Think pair share: Coins & Dice

15:00

Use the `sample()` function and a monte carlo simulation to estimate the answers to these questions:

- If you flipped a coin 3 times in a row, what is the probability that you'll get three "tails" in a row?
- If you rolled 2 dice, what is the probability that you'll get "snake-eyes" (two 1's)?
- If you rolled 2 dice, what is the probability that you'll get an outcome that sums to 8?

When `replace = FALSE`

Sometimes events cannot be independently simulated

What are the odds that 3 cards drawn from a 52-card deck will sum to 13?
(Aces = 1, Jack, Queen, King = 10)

```
deck <- rep(c(seq(1, 10), 10, 10, 10), 4) # Rep because there are 4 suits
length(deck)
```

```
#> [1] 52
```

Draw 3 cards from the deck *without replacement*:

```
cards <- sample(x = deck, size = 3, replace = FALSE)
cards
```

```
#> [1] 6 2 4
```

When `replace = FALSE`

Note: You can't draw more than 52 cards *without replacement*:

```
cards <- sample(x = deck, size = 53, replace = FALSE)
```

```
#> Error in sample.int(length(x), size, replace, prob): cannot take a sample
larger than the population when 'replace = FALSE'
```

When `replace = FALSE`

What are the odds that 3 cards drawn from a 52-card deck will sum to 13?
(Aces = 1, Jack, Queen, King = 10)

Repeat the 3-card draw *N* times:

```r
N <- 100000
count <- 0
for (i in 1:N) {
   cards <- sample(x = deck, size = 3, replace = FALSE)
   if (sum(cards) == 13) {
      count <- count + 1
   }
}

count / N # Compute the probability
```

```
#> [1] 0.03666
```

Think pair share: Cards

15:00

Use the `sample()` function and a monte carlo simulation to estimate the answers to these questions:

- What are the odds that four cards drawn from a 52-card deck will have the same suit?
- What are the odds that five cards drawn from a 52-card deck will sum to a prime number?
  (Aces = 1, Jack, Queen, King = 10)
  Hint: use `isPrime()` to help.

```
isPrime <- function(n) {
    if (n == 2) { return(TRUE) }
    for (i in seq(2, n-1)) {
        if (n %% i == 0) {
            return(FALSE)
        }
    }
    return(TRUE)
}
```

Break

05:00

Week 15: *Monte Carlo Methods*

1. Monte Carlo Simulation
2. Monte Carlo Integration

Discrete vs. continuous random numbers

Discrete

`sample()`
Takes random samples from vector `x`

```
sample_discrete <- sample(
  x       = c("heads", "tails"),
  size    = 5,
  replace = TRUE
)

sample_discrete
```

```
#> [1] "tails" "heads" "tails" "heads"
"heads"
```

Continuous

`runif()`
Takes random samples between bounds

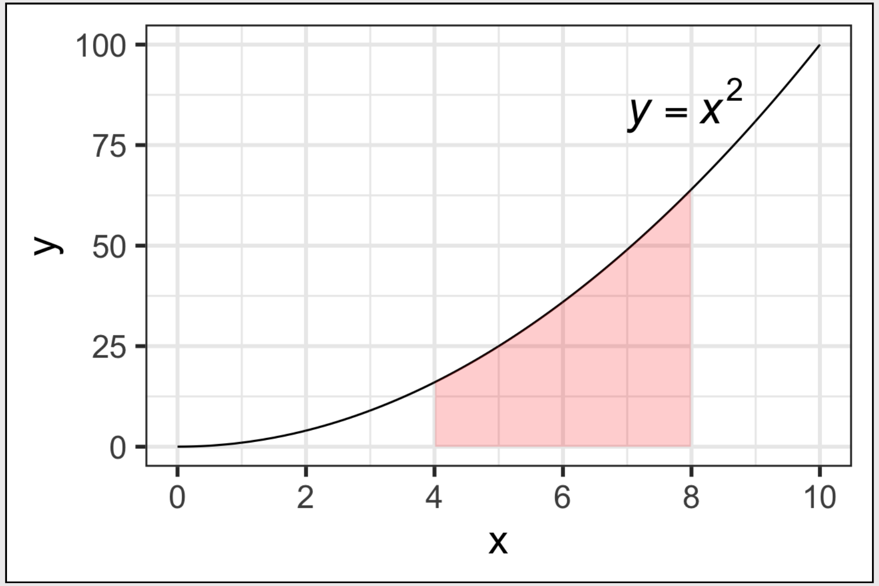
```
sample_continuous <- runif(
  n   = 5,
  min = 0,
  max = 1
)

sample_continuous
```

```
#> [1] 0.03944369 0.37356860
0.78492300 0.70443366 0.56196587
```

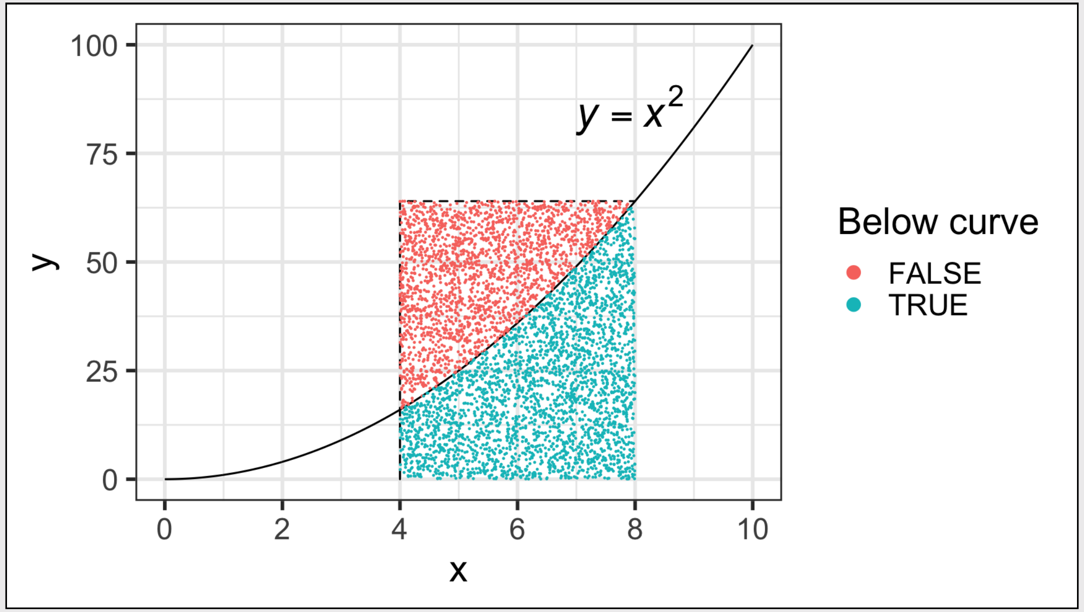
Monte Carlo Integration

Integration = compute the area "under the curve"

Find the area of $y = x^2$ between
$4 < x < 8$

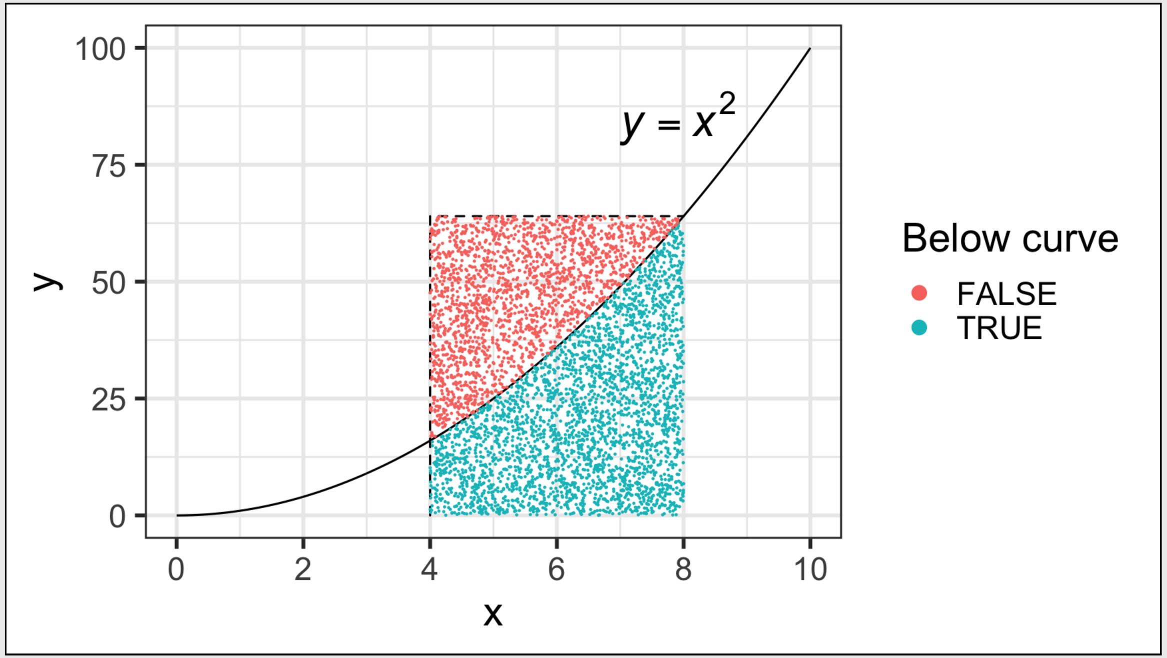


$$\frac{\text{Area Under Curve}}{\text{Area of Rectangle}} = \frac{\text{\# Points Under Curve}}{\text{\# Total Points}}$$



Monte Carlo Integration

$$\frac{\text{Area Under Curve}}{\text{Area of Rectangle}} = \frac{\#\text{ Points Under Curve}}{\#\text{ Total Points}}$$



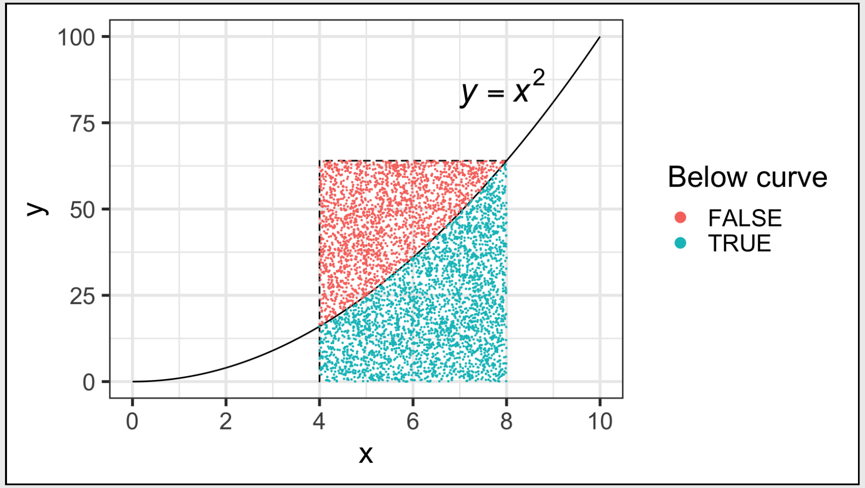
$$\text{Area Under Curve} = \text{Area of Rectangle}\left(\frac{\#\text{ Points Under Curve}}{\#\text{ Total Points}}\right)$$

Monte Carlo Integration

Step 1: Compute area of rectangle

```
area_rectangle <- (8 - 4) * (8^2 - 0)
area_rectangle
```

```
#> [1] 256
```

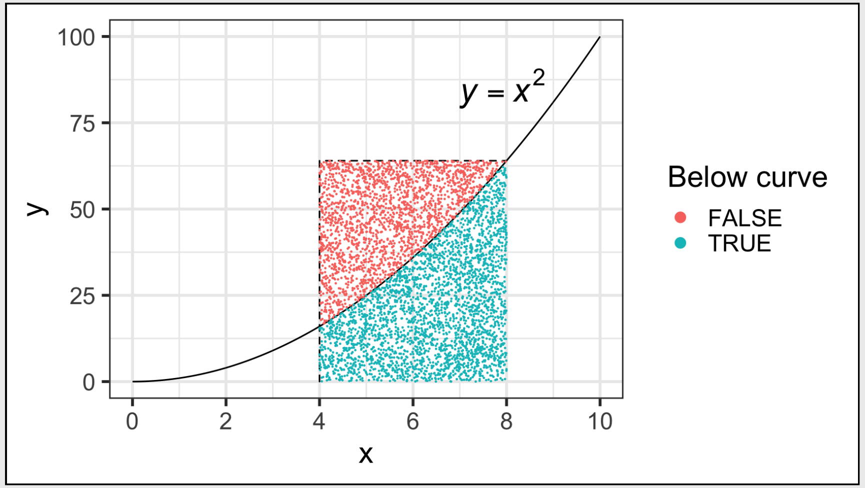


Monte Carlo Integration

Step 2: Simulate points

```r
N <- 100000
points <- tibble(
    x = runif(N, min = 4, max = 8),
    y = runif(N, min = 0, max = 8^2)) %>%
    mutate(belowCurve = y < x^2)

head(points)
```

```
#> # A tibble: 6 x 3
#>         x     y belowCurve
#>     <dbl> <dbl> <lgl>
#> 1  7.36 37.2  TRUE
#> 2  4.63 53.4  FALSE
#> 3  6.10 58.1  FALSE
#> 4  7.28 20.9  TRUE
#> 5  6.58 30.1  TRUE
#> 6  7.03  9.88 TRUE
```



Monte Carlo Integration

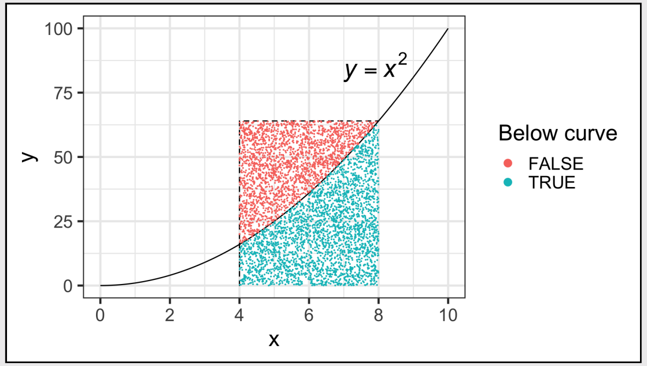
Step 3: Compute area under curve

```
N <- 100000
points <- tibble(
    x = runif(N, min = 4, max = 8),
    y = runif(N, min = 0, max = 8^2)) %>%
    mutate(belowCurve = y < x^2)

points_ratio <- sum(points$belowCurve) / N
points_ratio
```

```
#> [1] 0.57984
```

```
area_under_curve <- area_rectangle * points_ratio
area_under_curve
```

```
#> [1] 148.439
```



How did we do?

Simulated area under curve:

```
area_under_curve
```

```
#> [1] 148.439
```

Actual area under curve:

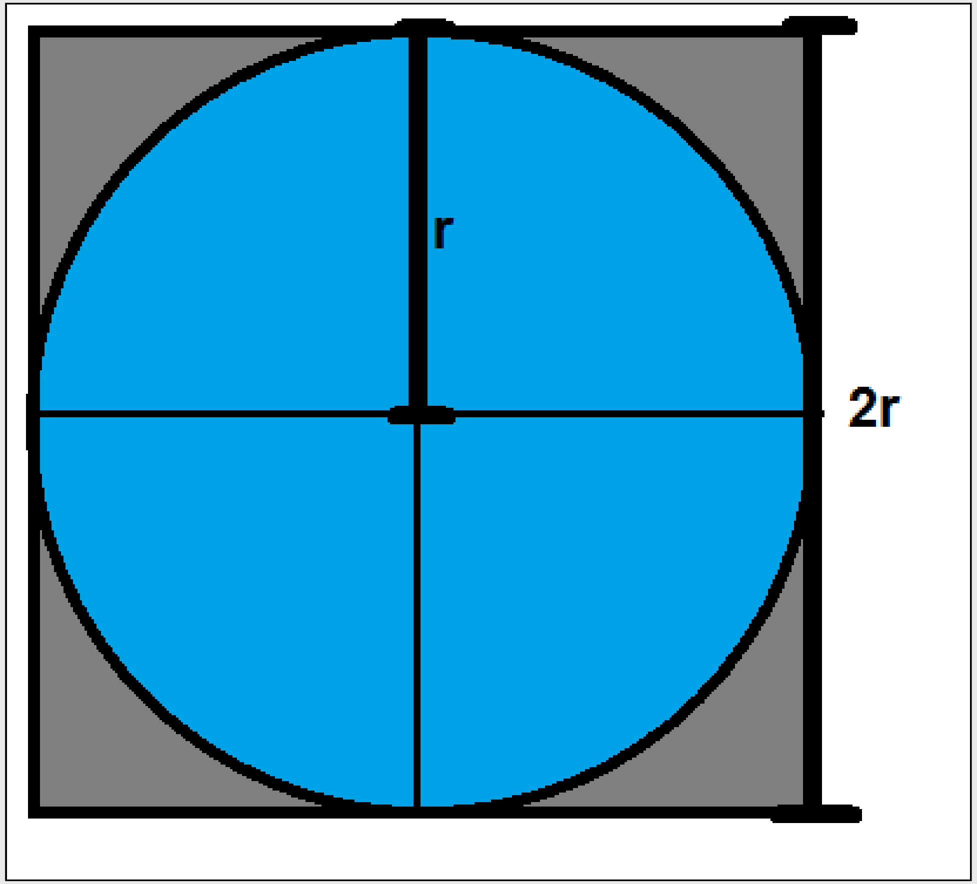
$$\int_4^8 x^2 \text{dx} = \left(\frac{x^3}{3} \right) \Big|_4^8 = \frac{8^3}{3} - \frac{4^3}{3} = 149.33\bar{3}$$

% Error:

```
true_area <- ((8^3 / 3) - (4^3 / 3))
100*((area_under_curve - true_area) / true_area)
```

```
#> [1] -0.5988571
```

Monte Carlo π



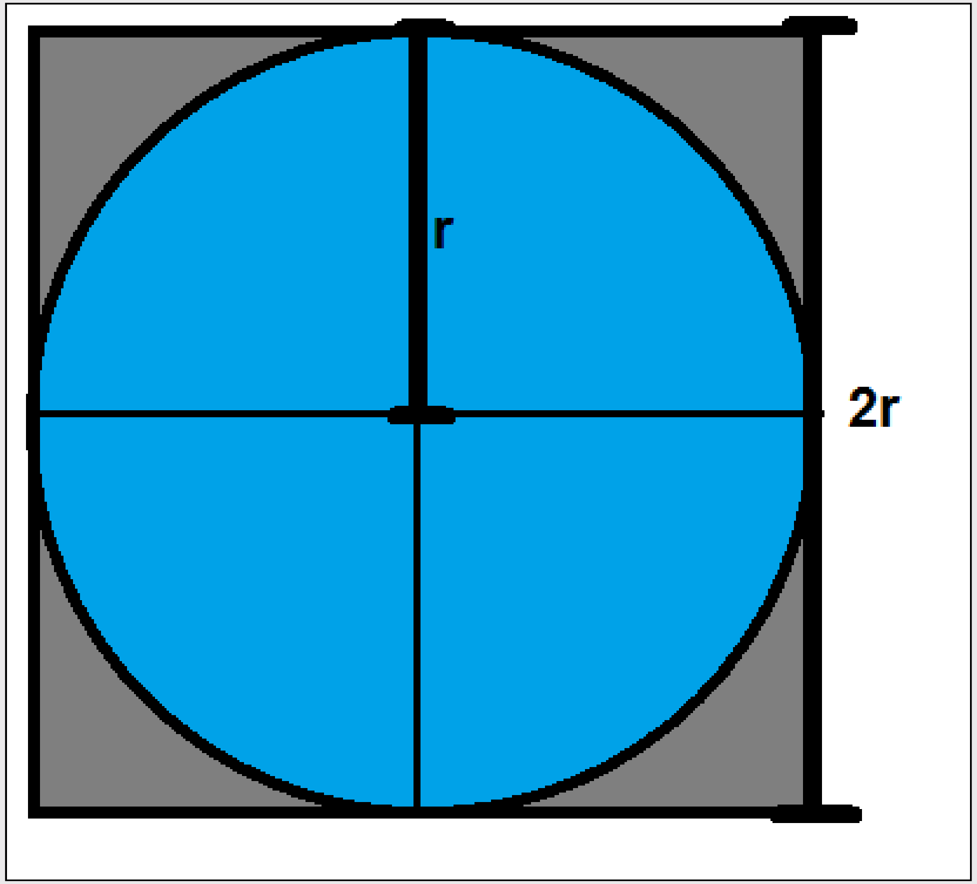
Area of a circle:

$$A_{circle} = \pi r^2$$

Area of square containing circle:

$$A_{square} = 4r^2$$

Monte Carlo π



Area of a circle:

$$A_{circle} = \pi r^2$$

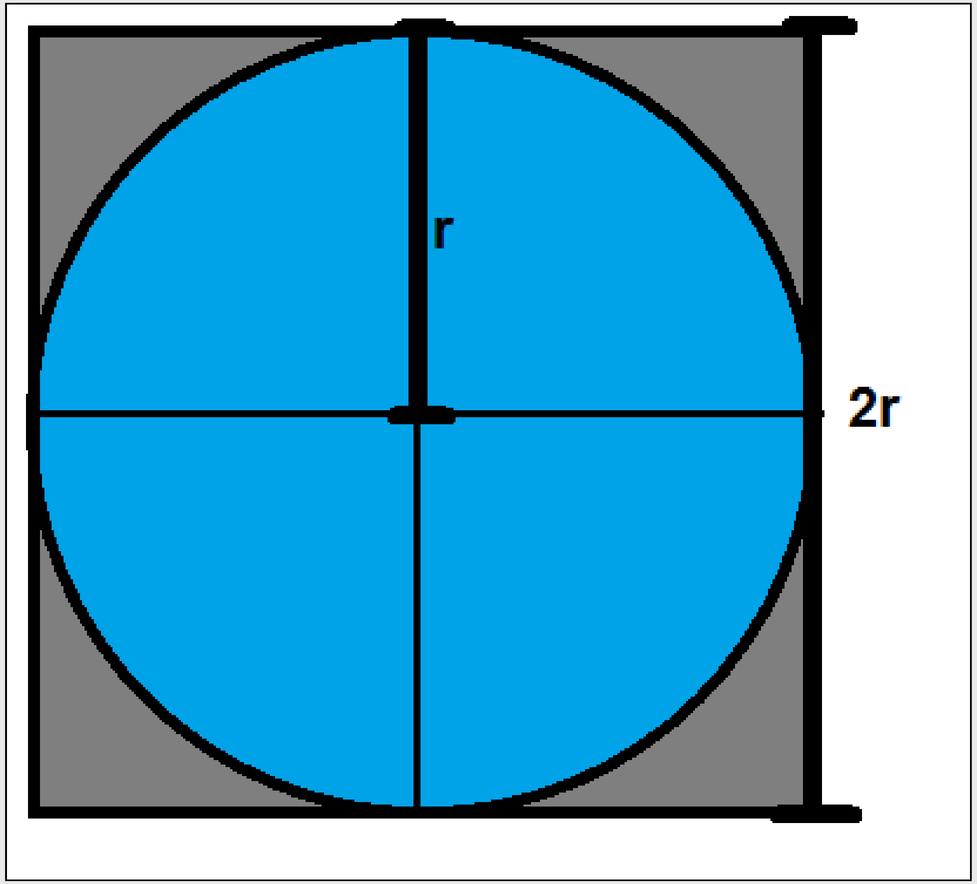
Area of square containing circle:

$$A_{square} = 4r^2$$

Ratio of areas = $\pi/4$:

$$\frac{A_{circle}}{A_{square}} = \frac{\pi r^2}{4r^2} = \frac{\pi}{4}$$

Monte Carlo π



Area of a circle:

$$A_{circle} = \pi r^2$$

Area of square containing circle:

$$A_{square} = 4r^2$$

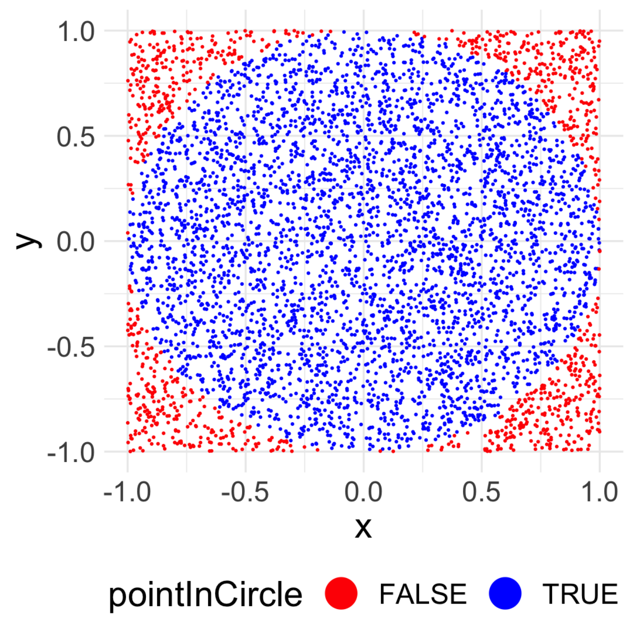
Ratio of areas = $\pi/4$:

$$\frac{A_{circle}}{A_{square}} = \frac{\pi r^2}{4r^2} = \frac{\pi}{4}$$

$$\pi = 4\left(\frac{A_{circle}}{A_{square}}\right)$$

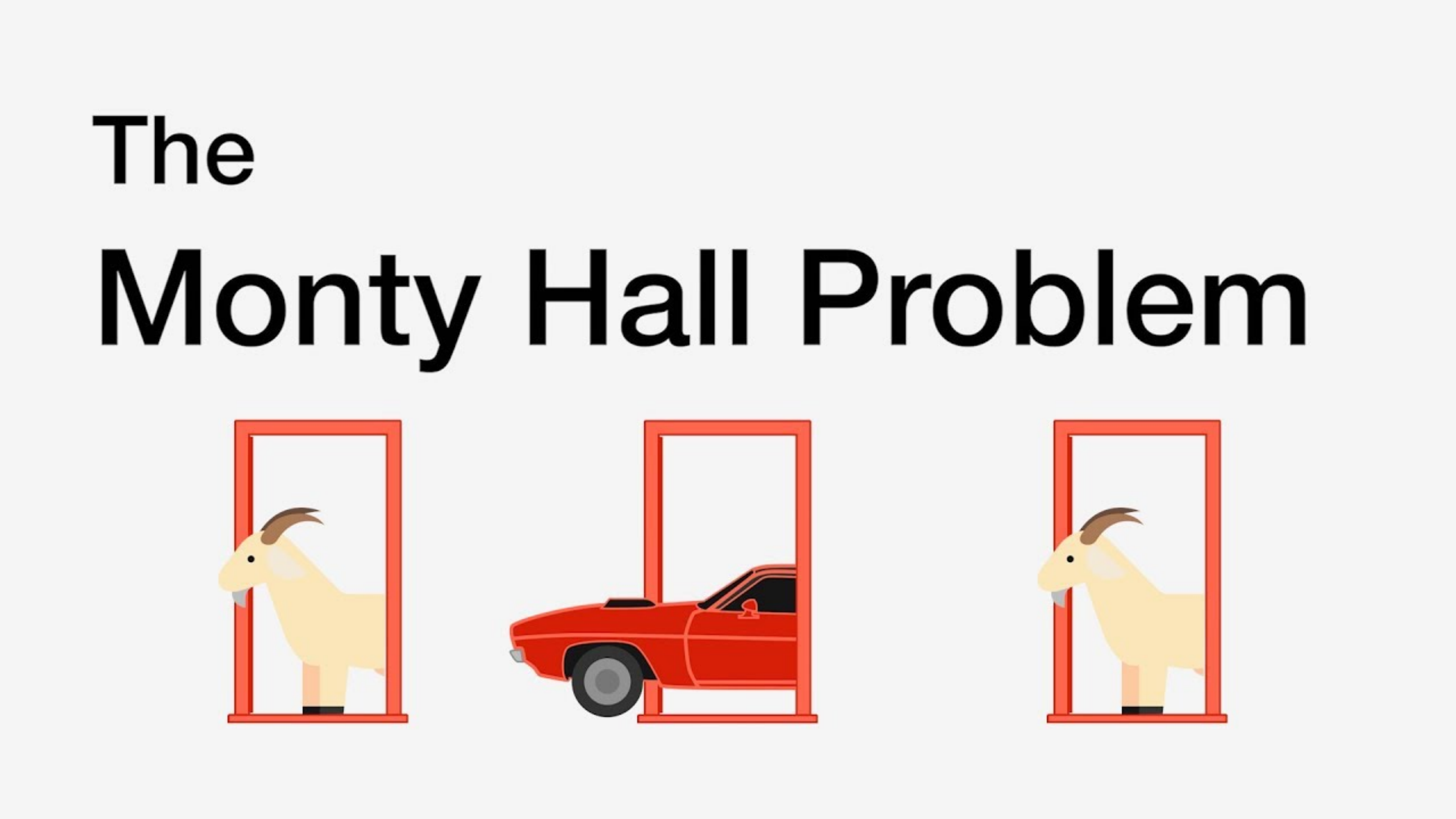
Think pair share: Estimate π

15:00



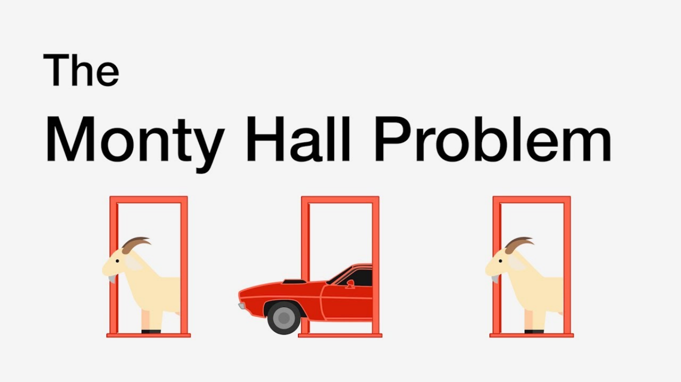
$$\pi = 4\left(\frac{A_{circle}}{A_{square}}\right)$$

1. Create a tibble with variables `x` and `y` that each contain 10,000 random points between -1 and 1, representing the (x, y) coordinates to a random point inside a square of side length 2 centered at `(x, y) = (0, 0)`. **Hint**: use `runif()`
2. Create a new column, `radius`, that is equal to the distance to each `(x, y)` point from the center of the square.
3. Create a new column, `pointInCircle`, that is `TRUE` if the point lies *within* the circle inscribed in the square, and `FALSE` otherwise.
4. Create the scatterplot on the left (don't worry about the precise colors, dimensions, etc.).
5. Estimate π by multiplying 4 times the ratio of points inside the circle to the total number of points


The
Monty Hall Problem

Think pair share: Monte Hall Problem

15:00



1. You choose door 1, 2, or 3
2. One door is removed
3. Should you swap doors?

In this simulation, the prize is always behind door #1:

- If you choose door #1, you must KEEP it to win.
- If you choose door #2 or #3, you must SWAP to win.

1) Create the tibble, `choices`, with two variables:

- `door` contains the first door chosen (`1`, `2`, or `3`)
- `swap` contains a logical (`TRUE` or `FALSE`) for whether the contestant swaps doors. **Hint**: use `sample()`

2) Create a new tibble, `wins`, which contains only the rows from `choices` that resulted in a win.

3) Compute the percentage of times the contestant won after swapping doors.

Reminders

1) Please fill the GW course feedback (see slack announcement)

2) I'll hold a final review sometime Thursday or Friday (check slack)

3) Final is Tuesday, December 15, 2020 12:45pm-2:45pm