

Ride the Bus

2025-11-09

A - Outline

A - Black or Red? B - Higher or Lower? C - In-between or Outside? D - Suit? E - Joint Probability

```
set.seed(052)
N <- 200

simulate <- function () {

  # Build and shuffle a standard 52 card deck.
  ranks <- rep(1:13, times = 4)
  suits <- rep(c("S", "H", "D", "C"), each = 13)
  colors <- ifelse(suits %in% c("S", "C"), "black", "red")
  deck_idx <- sample(52) # shuffle

  #By structure, leverage deck_idx as the index
  r1 <- ranks[deck_idx[1]]
  s1 <- suits[deck_idx[1]]
  c1 <- colors[deck_idx[1]]

  # A: Red or Black (26/52 = 50% chance of correct)
  guess_A <- sample(c("red", "black"), 1)
  correct_A <- as.integer(guess_A == c1)

  # B: Higher or Lower, P(B) = P(B|A), optimal guess
  # ^ If ranks equal, it's not correct
  # ^ and is an incorrect guess

  # number of higher/lower cards in the deck
  n_higher <- 4 * (13 - r1)
  n_lower <- 4 * (r1 - 1)

  if (n_higher > n_lower) {
    guess_B <- "higher"
  } else if (n_lower > n_higher) {
    guess_B <- "lower"
  } else {
    # To handle a tie from a rank of 7
    guess_B <- sample(c("higher", "lower"), 1)
  }

  # Sample the next card and compare to guess
  r2 <- ranks[deck_idx[2]]
```

```

correct_B <- as.integer((guess_B == "higher" &&
                        r2 > r1) ||
                      (guess_B == "lower" &&
                        r2 < r1))

#C: In-between or Outside,  $P(C | B, A)$ 

#define min/max, make the remaining deck
rmin <- min(r1, r2)
rmax <- max(r1, r2)
rem_idx <- deck_idx[-c(1,2)]
rem_ranks <- ranks[rem_idx]

#calc. number of cards with the inside and num outside, apply naive def of probability
between_count <- sum(rem_ranks > rmin &
                    rem_ranks < rmax)
outside_count <- 50 - between_count #covers both equal and outside cases

if (between_count > outside_count) {
  guess_C <- "between"
} else if (outside_count > between_count) {
  guess_C <- "outside"
} else {
  guess_C <- sample(c("between", "outside"), 1)
}

# Sample the next card and compare to guess
r3 <- ranks[deck_idx[3]]
is_between <- (r3 > rmin & r3 < rmax)
correct_C <- as.integer((guess_C == "between" &&
                        is_between) ||
                      (guess_C == "outside" &&
                        !is_between))

# D: Suit (guess modal remaining suit)
rem_idx <- deck_idx[-c(1,2,3)]
rem_suits <- suits[rem_idx]
counts <- table(rem_suits)
max_count <- max(counts)
best_suits <- names(counts)[counts == max_count]
guess_D <- sample(best_suits, 1)

s4 <- suits[deck_idx[4]]
correct_D <- as.integer(guess_D == s4)

# E: Joint Probability of Events
probs <- c(correct_A, correct_B, correct_C, correct_D)
probs
}

# ---- run the simulation ----

```

```

set.seed(52)
N <- 10000

res <- replicate(N, simulate())

# Marginal probabilities of each correct guess
probs <- rowMeans(res)
names(probs) <- c("A: Red/Black",
                  "B: Higher/Lower",
                  "C: Between/Outside",
                  "D: Suit")

# Joint probability: all four correct in the same trial
joint_prob <- mean(res[1,] * res[2,] * res[3,] * res[4,])
print(joint_prob)

## [1] 0.0689

simulate()

## [1] 1 1 1 1

}}

```