6-5-4 (or Ship, Captain, and Crew)

Connor Demorest

11/15/2020

Research Question

- What is the probability of “qualifying”?
- What is the expected score given qualification?
- What is the probability of winning?
- What is the best strategy to play with to win money?

Intro: How to play the game

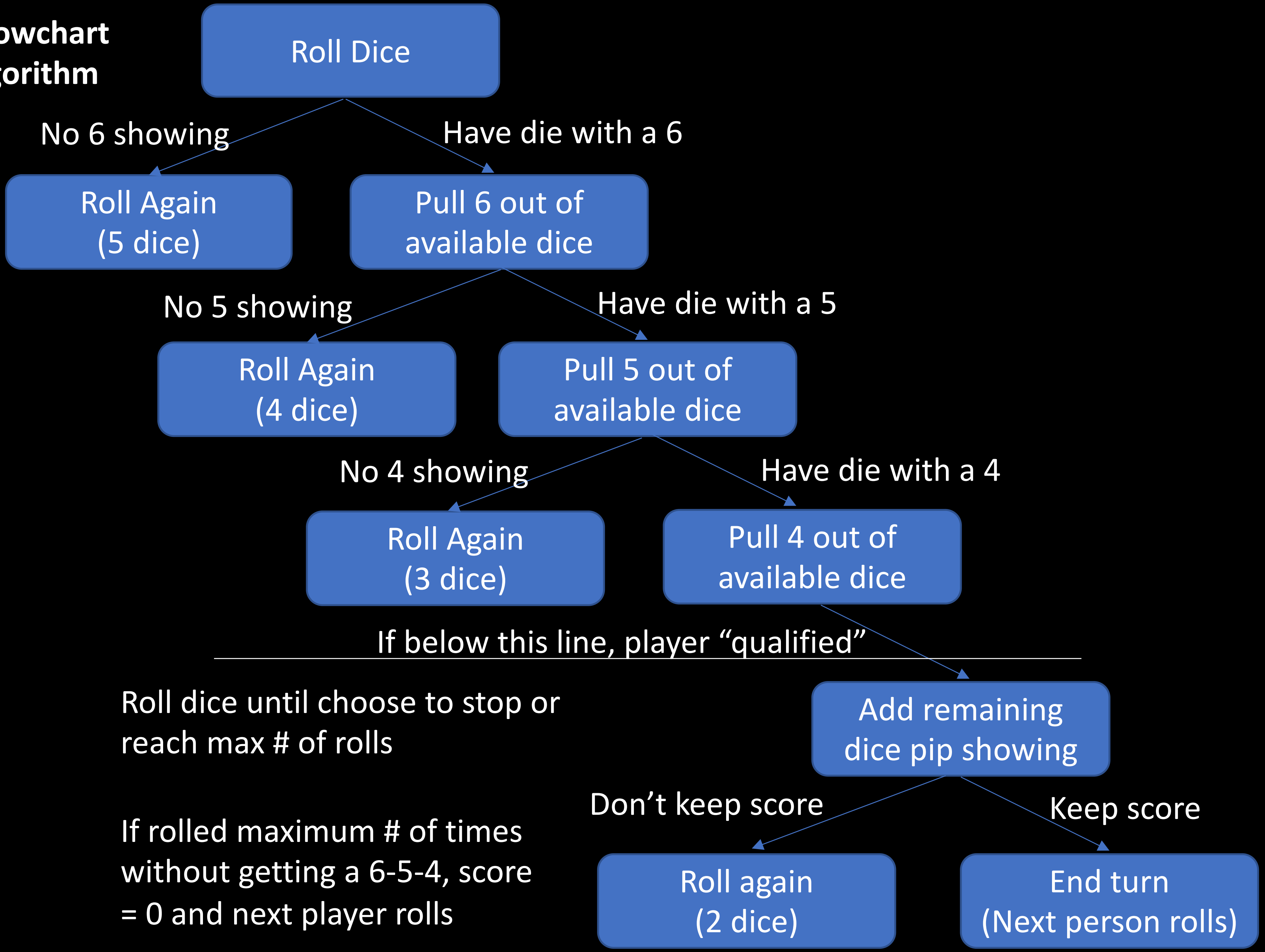


Figure 1: Flowchart of game algorithm

Roll dice until choose to stop or reach max # of rolls

If rolled maximum # of times without getting a 6-5-4, score = 0 and next player rolls

Methods:
How to play the game many times

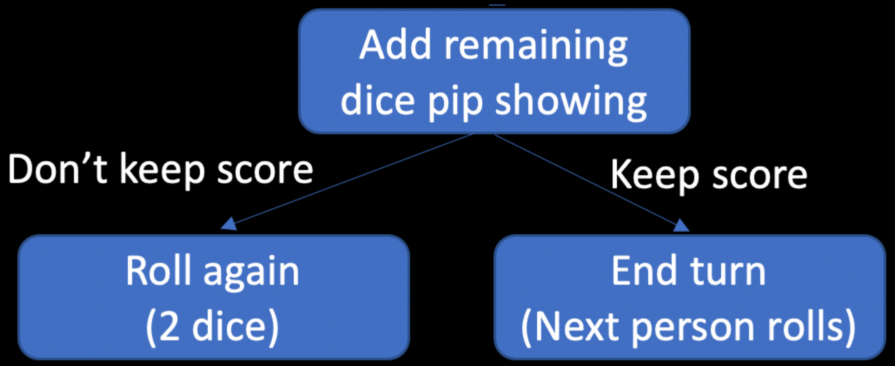
- 2 options: Simulate or solve mathematically
- Simulation method: write an R function that plays the game once, then do it 10,000 times
- Advantages of simulation:
  - Easy to implement basic rules
  - Easy to change rules and run again (e.g. change number of dice, change number of sides on dice)
- Disadvantages:
  - Approximate solution
  - Bugs and errors (foreshadowing!!)

Results:
Simulation of 80,000 total games

- Recall bottom of flowchart: Choice to keep the score you have or roll again, in some situations
- Simulated the game 10,000 times, with different ”rules” to keep a score or not keep a score
  - No rule to keep a score, so a player would roll 3 times every time they play
  - Keep a score that would end turn early if they have a score above that cutoff
  - Plot the differences for each rule

Roll dice until choose to stop or reach max # of rolls

If rolled maximum # of times without getting a 6-5-4, score = 0 and next player rolls



Rules change the distribution of scores

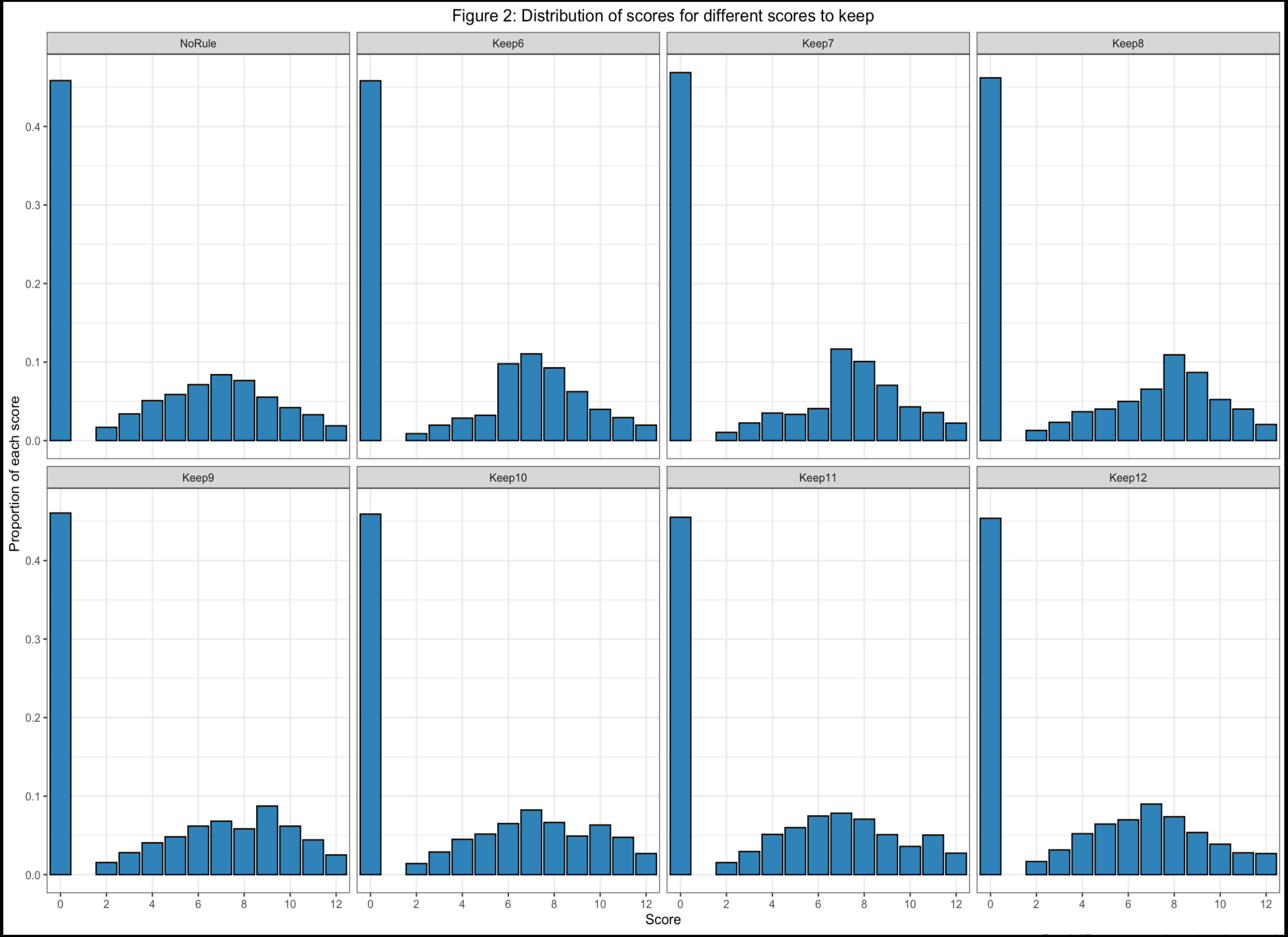
8 rules for keeping scores: No rule, and keep scores that are 6 through 12

Notice differences and similarities in distributions

All distributions on same scale for x and y

- X = Score (0-12)
- Y = Probability of getting that score (0-0.5)

For all rules of keeping scores, there is around 45% chance of scoring 0 (i.e. not qualifying)



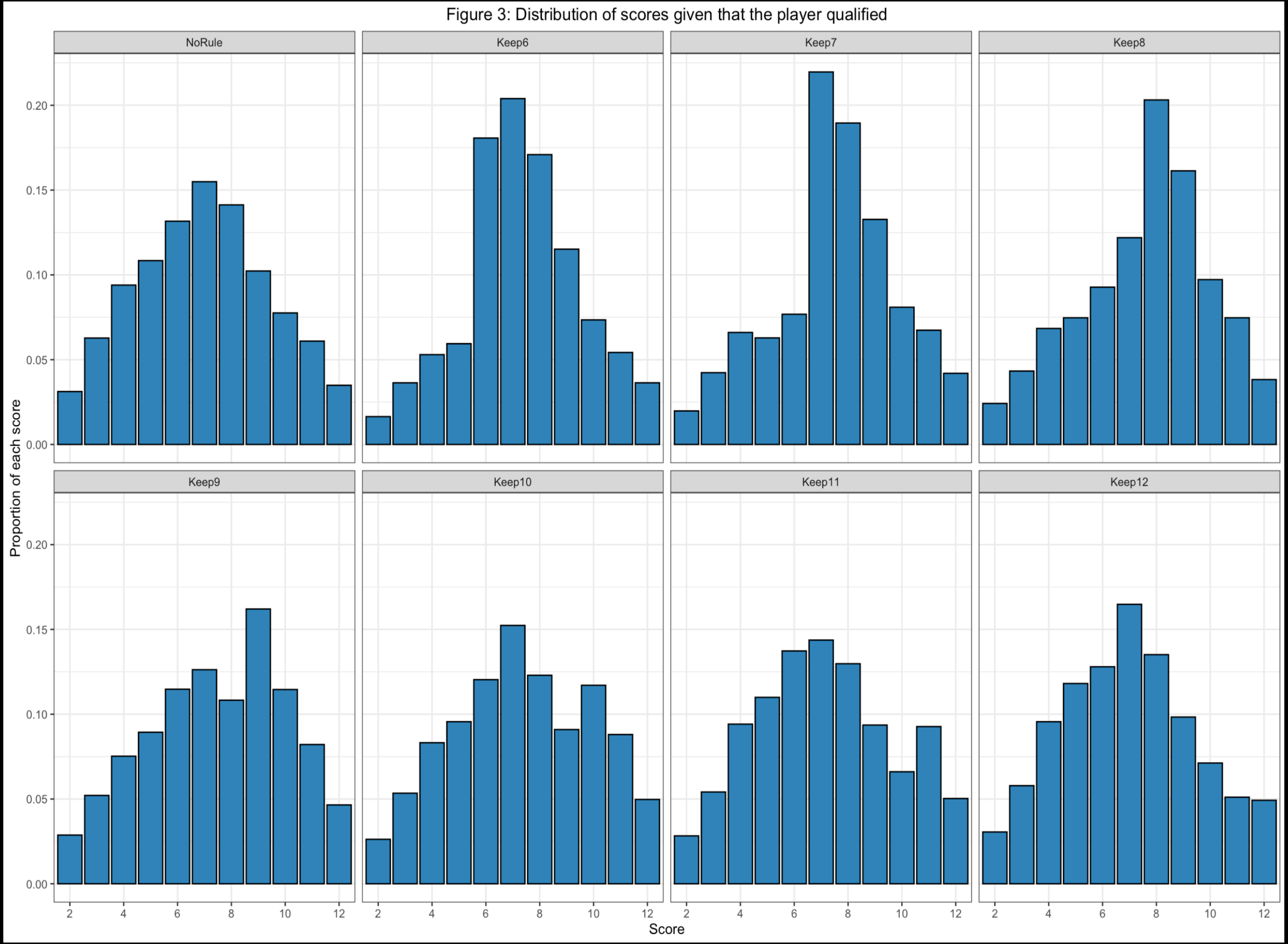
Conditional distributions given qualification

Same plot as before, conditional on not scoring 0

No rule is pretty much triangular

Pretty big shift in probability of scores for different rules

Which of these rules would be best if we wanted to maximize score?



Means and SD of distributions indicate keeping 8 or higher is ideal

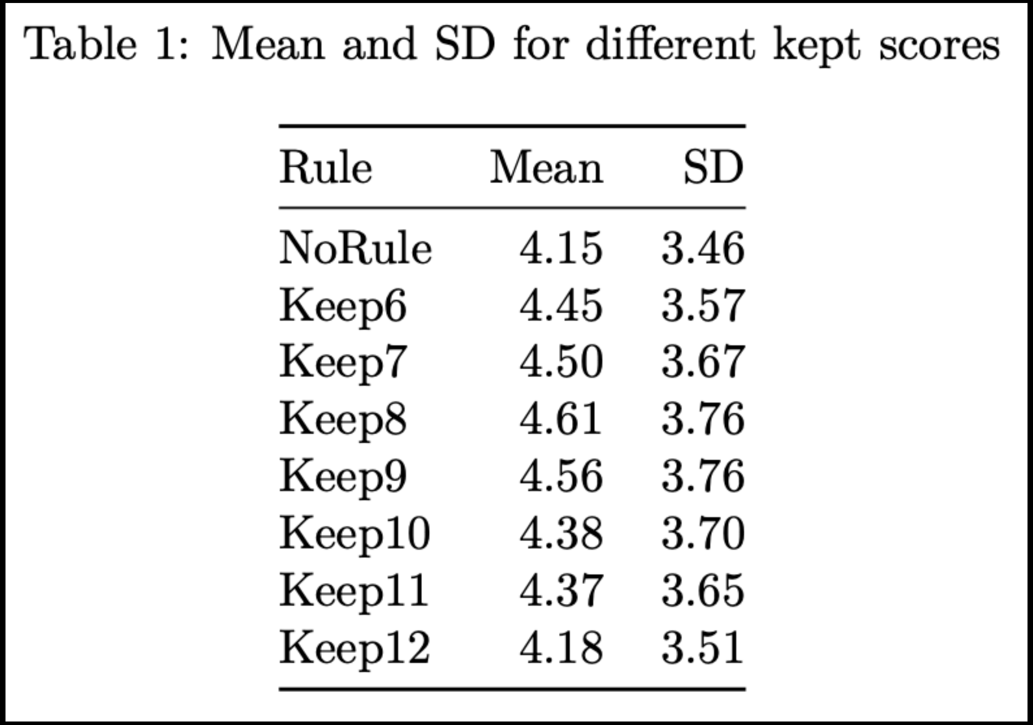


Table 1: Mean and SD for different kept scores

Rule	Mean	SD
NoRule	4.15	3.46
Keep6	4.45	3.57
Keep7	4.50	3.67
Keep8	4.61	3.76
Keep9	4.56	3.76
Keep10	4.38	3.70
Keep11	4.37	3.65
Keep12	4.18	3.51

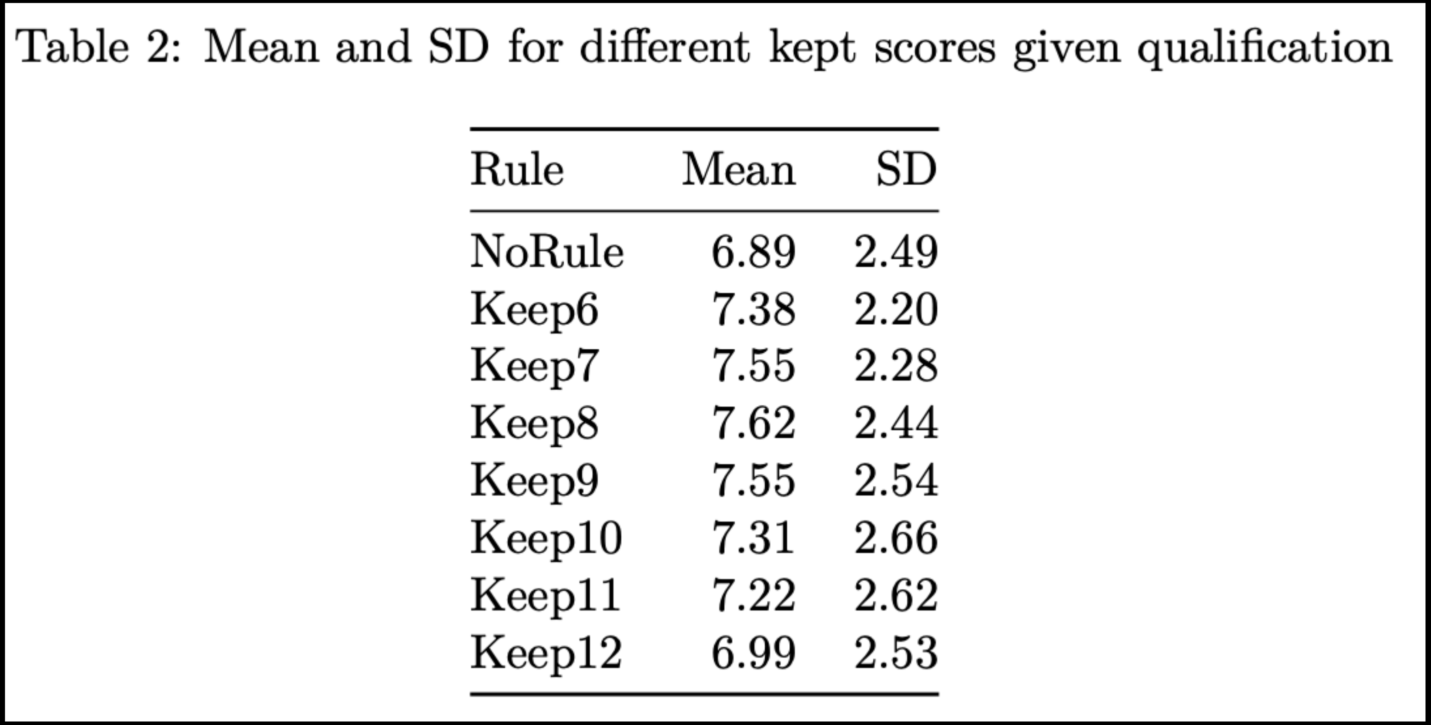


Table 2: Mean and SD for different kept scores given qualification

Rule	Mean	SD
NoRule	6.89	2.49
Keep6	7.38	2.20
Keep7	7.55	2.28
Keep8	7.62	2.44
Keep9	7.55	2.54
Keep10	7.31	2.66
Keep11	7.22	2.62
Keep12	6.99	2.53

Our goal is not to maximize score, but to win!

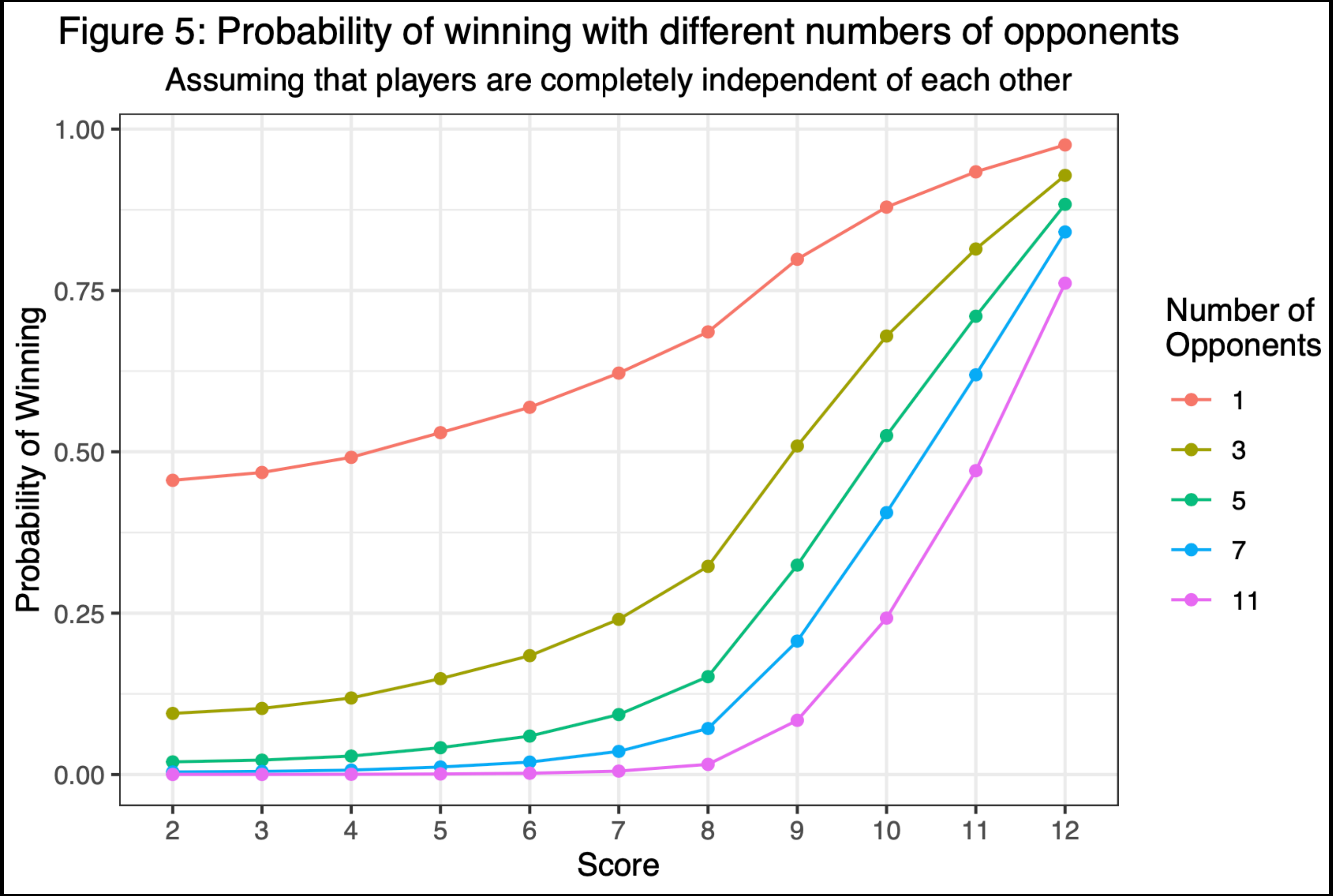
The probability of winning is more important

Shows that the score needed to win with any likelihood is higher for more opponents

With 1 opponent, you only need a 4 or higher to win 50% the time

With 11 opponents, you have around 0% chance of winning with a 6 or lower, so you need to take risks to win

No prize for second place!



Expected return varies depending on number of players

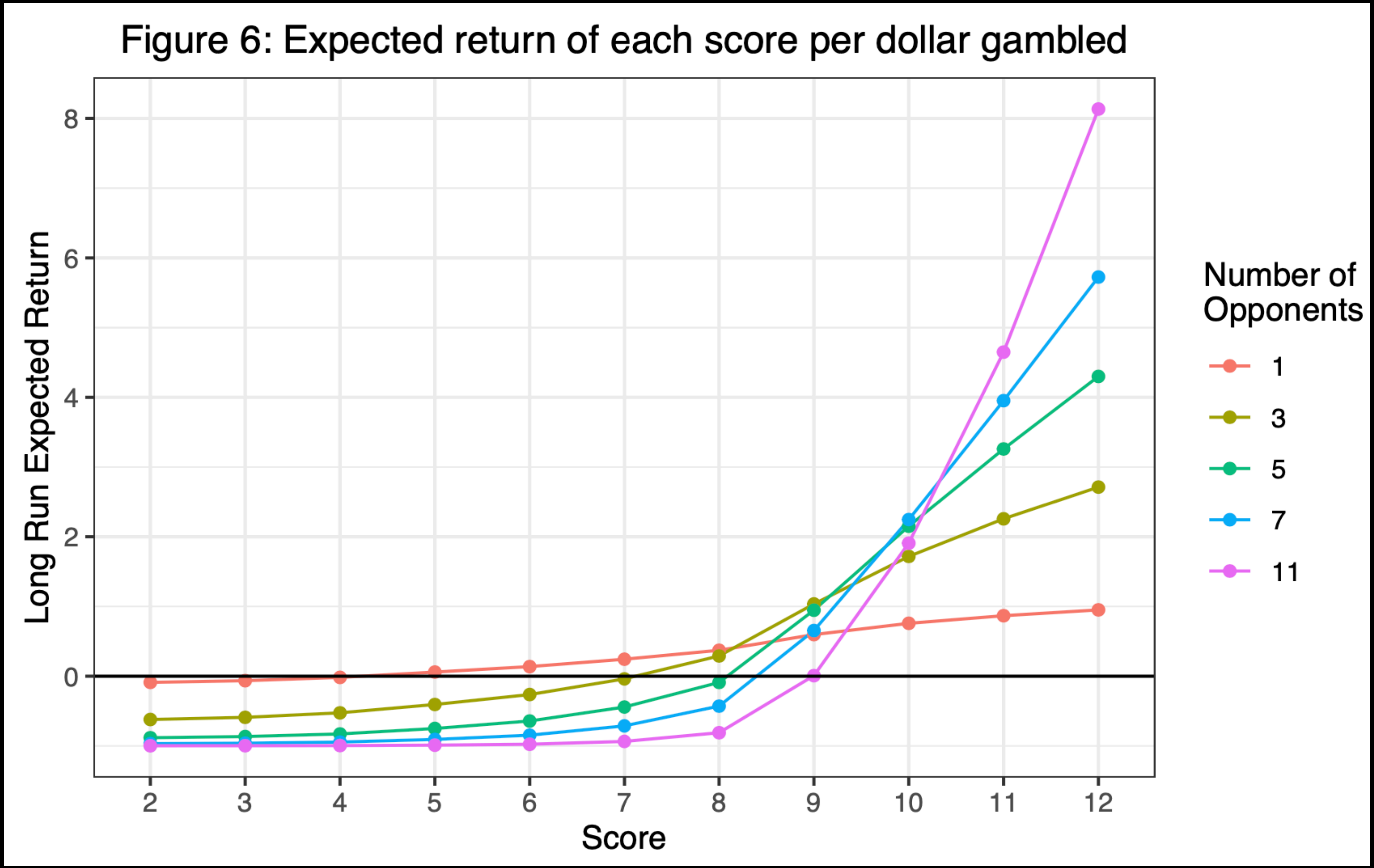
Plot of expected value on a 1$ bet with different scores of different numbers of opponents

If above the black line at y = 0 making money in the long run

When there are few players no need to take risks

When there are many players, you need to be at 10 or higher to have even 25% chance of winning

The low probability of improving on a score of 9 is outweighed by the size of the pot relative to your bet



All models are wrong, but some are useful (decide for you yourself if this one is useful)

- Limitations of the model:
  - Doesn’t capture how a player would actually play
  - Doesn’t handle ties between top scores
  - Situation:
    - You’re going last, score to beat is 11
    - You roll 65455 on your first roll
    - Using ”keep 8 or better” rule, you would keep the 10 and lose.
    - In real life you would roll again to try to get 11 or 12 to try to win or tie

- Future work:
  - More complicated algorithm to simulate how players would actually play
  - Handle ties in that algorithm
  - Exactly solve for distributions of n dice with p points and s sides
  - Other rules for the game
    - More dice
    - Change # of maximum rolls
    - Change # of sides on dice

Appendix A: Algorithm in R

```r
SixFiveFour = function(numDice = 5, numSides = 6, maxRolls = 3, scoreToKeep = 13) {
  sixPresent = F
  fivePresent = F
  fourPresent = F
  score = 0
  for (i in 1:maxRolls) {
    roll_outcome = sample(1:numSides, numDice, replace = T)
    if (6 %in% roll_outcome & !sixPresent) {
      numDice = numDice - 1
      roll_outcome = roll_outcome[-which(roll_outcome == 6)[1]]
      sixPresent = T
    }
    if (5 %in% roll_outcome & sixPresent & !fivePresent) {
      numDice = numDice - 1
      roll_outcome = roll_outcome[-which(roll_outcome == 5)[1]]
      fivePresent = T
    }
    if (4 %in% roll_outcome & sixPresent & fivePresent & !fourPresent) {
      numDice = numDice - 1
      roll_outcome = roll_outcome[-which(roll_outcome == 4)[1]]
      fourPresent = T
    }
    if (fourPresent & fivePresent & sixPresent) {
      score = sum(roll_outcome)
    }
    if (score >= scoreToKeep) {
      break
    }
  }
  return(score)
}
```

Appendix B: Tie handling methods

- Option A: *One-tie-all-tie*
- Option B: *Roll-off*
- Option C: *Split-the-pot*
- In my experience, A and B are most common depending on house rules