Simulating Yahtzee

Abstract: In this paper, I use simulation to evaluate various Yahtzee strategies. Among the strategies I implemented, a greedy strategy was the most performant, yielding an average final score of 214.37, while a rules-based strategy generated an average score of 212.28. The Kim-Nelson indifference zone procedure confirms the greedy strategy as the winning strategy, with the rules-based strategy as a consistent runner-up. Since a greedy strategy is difficult while playing Yahtzee in real life, the rules-based strategy is the best "real-life" approach I tried.

Other literature has estimated the final score generated by a truly optimal Yahtzee strategy as 254.59, which means that there is still room for improvement with the rules-based strategy. Specifically, the current rules-based strategy tends to overemphasize earning Yahtzees and underemphasize achieving the upper section bonus, something that could potentially be addressed by modifying existing rules or introducing new rules.

I. Introduction

Yahtzee is a popular multiplayer dice game sold by Hasbro. Originally invented in the 1950s and sold by Milton Bradley, Yahtzee is now played worldwide by a 100 million people [2].

Given the set of outcomes for a Yahtzee game is extremely large and there is a significant degree of chance involved in success, simulation is a useful tool for finding performant strategies that are accessible to an actual player. In this paper, I discuss my attempt at simulating various Yahtzee algorithms and what my results imply about the best approaches to winning Yahtzee.

II. Rules

A Yahtzee set contains five standard six-sided dice and a scoring card. The objective of the game is to maximize score over 13 rounds. Within each round, a player can roll the dice up to three times. After each of the first two rolls, the player can decide whether to end the round or to roll again. If they choose to roll again, they can choose any number of dice to reroll. Dice selections between rolls are independent — the number of dice and the specific dice selected after the current roll has no bearing on how many and which dice can be selected after the next roll [2].

Once a round is over, either after the dice have been rolled three times or the player chooses to keep all five dice, the player decides which of the 13 categories on the scorecard to apply the current dice values to and is awarded the corresponding point value. See Table 1 below for a summary of the categories and their corresponding scores. Some categories offer a flat point bonus — e.g. full house is always 25 points — while other categories are dependent on the dice values. Each category can only be selected once. If a player chooses a category that is not fulfilled by the current dice (e.g. the player selects Yahtzee even though they don't have five-of-a-kind), their score is zero for that category.

Table 1. Scores by category

Category	Score	Section
1s	Sum of 1s	Upper
2s	Sum of 2s	Upper
3s	Sum of 3s	Upper
4s	Sum of 4s	Upper
5s	Sum of 5s	Upper
6s	Sum of 6s	Upper
3 of a kind	Sum of all dice	Lower
4 of a kind	Sum of all dice	Lower
Full house	25	Lower

Small straight	30	Lower
Large straight	40	Lower
Yahtzee	50	Lower
Chance	Sum of all dice	Lower

The rules above cover the most basic version of Yahtzee. The standard versions of Yahtzee also include two rule extensions. First, the player earns an extra 35 points at the end of the game if they score at least 63 points in the first six sections ("upper section bonus"). Second, if a player has already applied a non-zero score to the Yahtzee category and they roll another Yahtzee, they earn an additional 100 points. The player can then apply the Yahtzee roll as a "joker" in the following order ("forced joker rule"):

1. If the corresponding upper section category is open, apply the Yahtzee to that category (e.g. a set of five fours will be applied to the fours category).
2. If the corresponding upper section category is not open, apply the Yahtzee to three-of-a-kind or four-of-a-kind, using the standard scoring rules for these categories.
3. If #1 and #2 are not options, the player can "steal" full house, small straight, and large straight, earning full points for that category even without having rolled the corresponding dice values. The player can also apply the roll to chance per normal scoring rules
4. If the only remaining category or categories available are non-corresponding upper section categories, select one of these categories and earn zero points.

The standard Yahtzee Classic game is played by at least two players, where the winner is the player with the highest score after all rounds have finished [2]. However, much of the literature focuses on solitaire Yahtzee, where a single player seeks to maximize their overall score based on the rules outlined above [1] [4] [7] [8]. Given each player's rolls and scoring are independent from those of other players and the winner is determined solely determined by which player has the highest overall score, a strategy that maximizes solitaire Yahtzee score should also maximize win probability for multiplayer Yahtzee on average (though not necessarily within an individual game). For these reasons, I also choose to focus on optimizing solitaire Yahtzee score.

III. Background

Woodward summarizes the problem of “solving” Yahtzee well:

> “In principle, it is relatively straightforward to solve the game of Yahtzee: there are a finite number of decisions (which dice to re-roll and scoring group to use), all the probabilities associated with the random events (dice rolling) are known and the utility associated with each decision is well defined (ultimately, points are obtained)… The main problem is that there are literally billions of calculations that need to be undertaken” (19) [8].

To deal with this large solution set, mathematicians and computer scientists have used dynamic programming and, to a lesser extent, simulation to uncover performant Yahtzee strategies.

Verhoeff defines a Yahtzee game as a graph with nodes that correspond to decisions the player must make; specifically, the player must decide which dice to keep and which category to score at the end of a turn. Under this framework, a strategy is a choice that dictates how a player travels along the edges between nodes. He then uses dynamic programming to implement an “optimal strategy”, whereby the option selected at every node is the option that maximizes expected final score. Verhoeff finds that the mean final score for the optimal strategy is 254.59. Verhoeff has developed a web app that adapts this optimal strategy to show a player the optimal decision to make in any scenario [7].

Woodward, Glenn, and Kelly & Liese similarly develop programs to implement this optimal strategy, with all three sets of authors reaching the same estimated final score as Verhoeff [1] [4] [8]. Woodward’s approach is fairly similar to Verhoeff in that he uses dynamic programming to implement an optimal strategy. He also compares performance of the optimal strategy against a few of his own attempts at playing the game manually and finds statistically significant evidence that the optimal strategy outperforms his own attempts [8]. Kelly and Liese define a Yahtzee game with a probability generating polynomial that models the probability that the sum of random variables equals a given final score. They input this probability generating function into Mathematica, which optimizes the value of the probability generating polynomial for each potential decision. They use the outcomes of simulated games to make conclusions about how human players should approach Yahtzee strategy. Among other things, they conclude that achieving the upper section bonus is important to optimizing final score, given it is achieved in 68% of optimal games, but that it is not worthwhile pursuing bonus Yahtzees beyond the first Yahtzee since this is a highly unlikely outcome (they estimate the probability to be at most 1%). They also advise that players not score four-of-a-kind in the first round and that players not score a zero in the Yahtzee category in early rounds given it is often achieved later in the game [4].

Glenn's analysis is perhaps the most thorough; along with introducing refinements to Verhoeff's graph framework that reduces the number of edges to explore and thus reduces computation power needed, he also experiments with several sub-optimal but human interpretable strategies via simulation, including a greedy strategy, several rules-based strategies, and genetic algorithms built upon these rules-based strategies. He also uses simulation to estimate the standard deviation of the optimal strategy as 59.61 points. He still finds that the optimal strategy is the highest performing strategy but is able to achieve relatively comparable performance (mean scores within 10-30 points) with the genetic algorithms [1].

For the purposes of this project, I felt that coding an optimal strategy was out of scope, due to constraints on time and computing power. Instead, I chose to follow Glenn's lead and implement several imperfect strategies that might be more readily accessible to a human player. I then evaluate each strategy's efficacy via repeated simulations.

IV. Implementation

a. Background

All of the programming for this project was done in Python. See the linked GitHub repository, with a README file that explains how to set up and run my analysis workbook. For each potential strategy, I generate and run 100,000 complete games.

I created Dice and Game classes, which include various gameplay and scoring functions. The Game class also has a *get_game_state()* function, which returns the current dice values, the potential category scores of those dice, score by category for current dice values, which categories have been used, number of Yahtzees used, whether the upper section bonus has been earned, and some tracking information for analysis (e.g. order of categories scored, rolls per turn, etc.). This method was inspired by Kelly & Liese, who define a game state as the categories currently available for scoring, number of Yahtzees earned, and number of points needed to achieve upper section bonus at a specific decision [4].

The Dice class accepts an integer as input that it uses as a seed for a NumPy random number generator that decides rolls. The Game object also accepts an integer n as input, which it uses to initiate 5 Dice with seeds n, n+1, …, n+4. I exploit common random number seeds to do pairwise comparison between games with the same initial seed played with a different strategy.

Given my Dice and Game classes were designed to enable repeated simulated replications, I did not create functionality that would enable a human player to interact directly with the program to play an individual game. This could be an extension in a further iteration, especially to compare a real player's strategy to algorithmic strategies.

Each strategy is implemented as its own class, which includes methods that decide which dice to keep and which category to score at a given node, based on a specified set of rules. I can think of my approach to simulation as a discrete event model – the events being deciding which dice to keep and which categories to score – with a finite horizon.

b. Random strategy

As a first step, I developed an algorithm that randomly selects both which dice to keep and which category to apply scores to. My goal was to set a simple unskilled baseline that other strategies could be compared against. In addition to the standard random number generators implemented in the Dice class, I also create a separate random number generator with a different, deterministic seed to randomly select dice to keep and categories to score.

I implemented two versions of this strategy – what I refer to as a “purely random strategy” and a “tiered random strategy”. In the purely random strategy, I independently decide whether to keep each individual dice by generating a simple binary random variable for each dice. The expected number of dice kept is thus 2.5 and it is unlikely (~3% each) that the strategy will choose to end any turn early by keeping all five dice or will reroll all five dice. In the tiered strategy, I first randomly select the number of dice to keep (0-5) and then randomly select a combination of dice of the specified size. Under this approach, the strategy will either choose to end a turn early or reroll all dice ~17% (⅙) of the time each.

With both strategies, I choose the category to score in at random among all categories with non-zero scores, though I do default to scoring a Yahtzee if five-of-a-kind is rolled and the Yahtzee category is available.

c. Greedy strategy

Next, I developed a greedy strategy that selects the next action based solely on which outcome has the highest expected score. I precompute all 252 potential combinations of 5 dice (6 choose 5 with replacement) and the value that would be assigned to the combination if it were applied to each of the 13 categories. While there are rolls remaining (so after the first and second roll of a turn), I enumerate all potential combinations of current dice that could be kept (“keep combinations”), identify the potential combinations of five dice that would result if those dice were kept and the remaining rerolled, retrieve the maximum score limiting to available categories for each resulting five-dice combination (assuming the user always chooses the category with the highest score), and then calculate the expected value of each kept combination based on the scores of the potential five-dice combinations. Additionally, if the player has not yet earned the upper section bonus and a five-dice combination would raise the user's upper section above 63, I also include the additional 35 point bonus in the expected value calculation. The algorithm then selects the combination of kept dice that maximizes expected score.

See below for an example of how the algorithm would calculate expected values for an opening roll of [3, 2, 4, 4, 5]. For brevity, I only show scores for if the user keeps all five dice and if they keep all but one of the 4s.

Current dice values: **[3, 2, 4, 4, 5]**

Categories available: All 13

Rolls left in turn: 2

Turns left in game: 13

1. Keep all five dice [**3, 2, 4, 4, 5**] and conclude turn
    a. *Outcomes*:
        i. [**3, 2, 4, 4, 5**]
            1. Probability: 100%
            2. Best score = 30 (small straight)
    b. *Expected value*: 100% * 30 = 30
2. Keep [**3, 2, 4, 5**]
    a. Outcomes:
        i. [**3, 2, 4, 5,** 1]
            1. *Probability*: 16.67%
            2. Score: 40 (large straight)
        ii. [**3, 2, 4, 5,** 2]
            1. *Probability*: 16.67%
            2. *Score*: 30 (small straight)
        iii. [**3, 2, 4, 5,** 3]
            1. *Probability*: 16.67%

        2. *Score*: 30 (small straight)
    iv. [**3, 2, 4, 5,** 4]
        1. *Probability*: 16.67%
        2. *Score*: 30 (small straight)
    v. [**3, 2, 4, 5,** 5]
        1. *Probability*: 16.67%
        2. *Score*: 30 (small straight)
    vi. [**3, 2, 4, 5,** 6]
        1. *Probability*: 16.67%
        2. *Score*: 40 (large straight)

b. *Expected value:* 16.67% * 4 * 30 + 16.67% * 2 * 40 = 33.33

etc. for remaining kept combinations

In this scenario, the algorithm opts to keep [3, 2, 4, 5] and reroll for the possibility of achieving a large straight.

When a turn ends – either because the player elects to keep all five dice or they exhaust all three rolls – the algorithm scores the remaining dice and then selects which category of the remaining categories to assign that score to, based on which category results in the highest score. If multiple categories result in the same score, I input a "tie break order" of priority, which can also be adjusted. The same logic is also applied when following the forced joker rule.

d. Rules-based strategy

Lastly, I develop a set of simple heuristics to see if these strategies can produce "human intelligible" rules that are still performant. For my initial implementation, I primarily follow the rules laid out by Glenn (6) [1]. See Figure XX below for the set of rules suggested by Glenn for keeping dice and scoring categories, which he estimates produce an average score of 218.18. Following Kelly and Liese, I make a modest modification to the category scoring rules and require that four-of-a-kind not be scored in the first round [4]. I also input a priority list for the order in which categories should be "thrown away" (i.e. scored as zero), with Yahtzee as the last option.

TABLE V

SAMPLE RULES FOR KEEPING DICE

Rule
Yahtzee if Y is unused or Yahtzee Joker is applicable
Large Straight if LS or SS is unused
Small straight if SS is unused or both LS and C are unused
a tripleton if the corresponding upper category is unused
any tripleton if one of 3K, 4K, FH, or C is unused
a doubleton (high preferred) if the corresponding upper category is unused
[2 3 4] or [3 4 5] if SS unused or both LS and C are unused
any doubleton if 3K or C is unused
any tripleton (high preferred) if *Yahtzee* is unused or non-zero
a singleton (low preferred) if the corresponding upper category is unused, unless more than four upper categories are unused
any doubleton (high preferred)
a singleton 4, 5, or 6 (high preferred) if 3K, 4K, or C unused
nothing

TABLE VI

SAMPLE RULES FOR SCORING ROLLS

Rule
Yahtzee
Large Straight
Small Straight
a tripleton in an upper category if it earns the bonus
four 5's or four 6's in the upper category
Four of a Kind
three 5's or three 6's in the upper category
Full House
Three of a Kind if the total is at least 22
a tripleton in an upper category
Three of a Kind
Chance if the total is at least 22
doubletons in an upper category (lower preferred)

As extensions to Glenn's rules, I devise two additional strategies: one that seeks to maximize the probability of achieving the upper section bonus and one that seeks to maximize the probability of rolling a Yahtzee. I chose to implement these strategies to test how pursuing these higher variance, lower probability events might affect the final score. I was especially interested in the outcome of the upper section bonus maximization strategy, given Kelly and Liese's findings about the importance of this category to the optimal strategy [4].

For the upper section bonus strategy, if the bonus has not yet been earned, the strategy will select whichever keep combination would maximize the number of upper section points earned on the next roll. For example, if the current dice values are [3, 2, 3, 2, 3], the 1s, 2s, and 6s categories are available, and the player has 20 points left to achieve the upper section bonus, the strategy will choose to keep [2, 2] since the expected contribution to upper section points for the next roll is 5 points (4 + 3 * 2 * ⅙), vs. 0 for [3, 3, 3] (3s are already taken). The Yahtzee-maximizing strategy will keep whichever dice appears the most often – in this case, [3, 3, 3] – until one Yahtzee has been earned, prioritizing unused upper categories if there is a tie.

The upper section bonus strategy will always score Yahtzees first but then chooses will score in whatever upper section category makes the largest contribution toward the bonus, if any categories are non-zero. Once the upper section bonus is achieved, it defaults to the standard rules-based strategy scoring approach. The Yahtzee-maximizing strategy uses the same approach as the original rules-based strategy, since Yahtzees are already preferred.

V. Results and Discussion

a. Highest scoring strategies

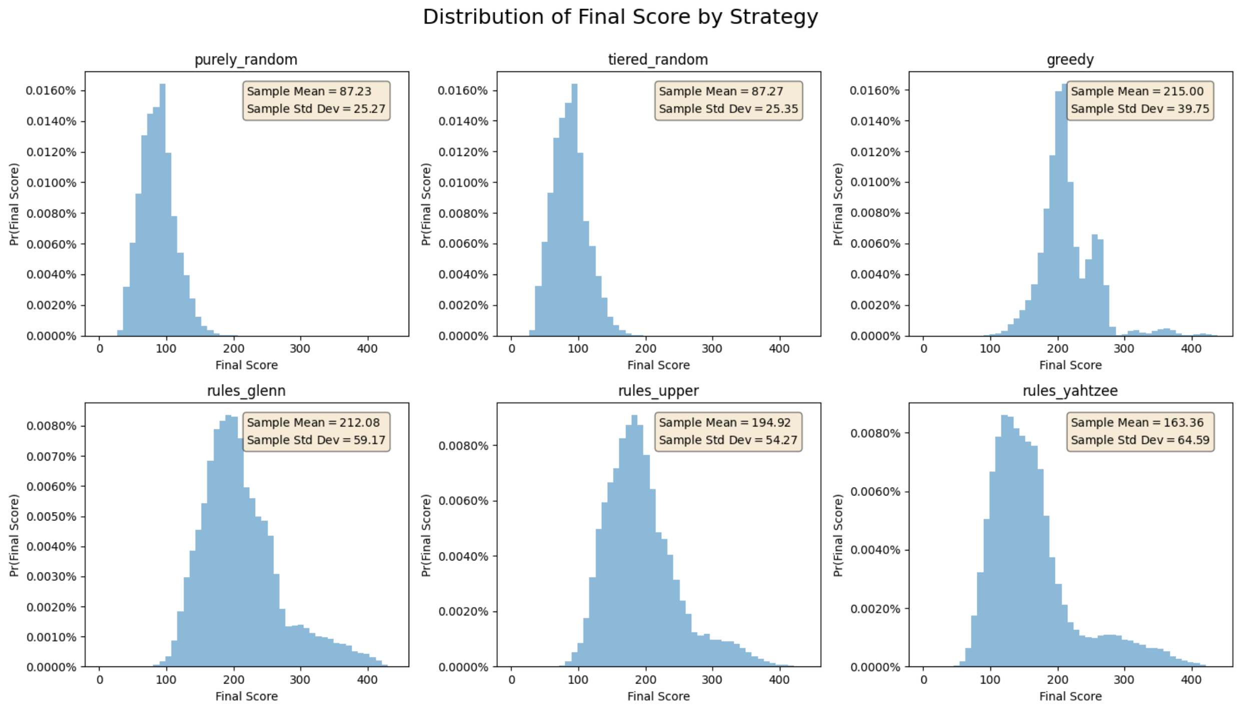
Output analysis is remarkably straightforward for this application. Nearly all parameters for analysis – e.g. final score, number of Yahtzees, % of games earning final bonus, etc. – have only one observation per simulated run, meaning there are few issues with correlated within-run metrics. Given replications are generated using non-overlapping seeds, fed into the NumPy's high permuted congruential random number generator (which has a full period of 2^{128}), the final scores of the simulations should be independent and identically distributed [6]. The Central Limit Theorem implies that the distribution of the sample mean approaches normality as the number of independent replications increases, which means that I can use both the sample mean and sample variance to estimate the population mean for the final score and provide a 95% confidence interval.

Table 2 shows sample mean, standard deviation, and 95% confidence interval for final score by strategy while figure 1 shows the underlying distributions. I also include the implied confidence intervals for the true optimal Yahtzee strategy, as calculated by Glenn [1].

Table 2. Sample means and standard deviations for final score

	Final Score			
Strategy	**Sample Mean**	**Sample Standard Deviation**	**95% Confidence Interval, Lower Bound**	**95% Confidence Interval, Upper Bound**
Purely Random	87.23	25.27	87.07	87.38
Tiered Random	87.27	25.35	87.11	87.43
Greedy	214.37	39.64	214.13	214.62
Rules-Based: Glenn 2007	212.18	59.38	211.81	212.55
Rules-Based: Prioritize Upper Section	192.92	53.82	192.58	193.25
Rules-Based: Prioritize Yahtzee	163.26	64.47	162.86	163.66
Optimal Strategy (Glenn)	*254.59*	*59.61*	*254.22*	*254.96*

Figure 1. Distribution of final score by strategy



The greedy strategy results in the highest average final score (214.37 +/- 0.25), followed by my implementation of Glenn's original rules (212.18 +/- 0.37), though neither approach is close to the score attainable with the optimal Yahtzee strategy. Interestingly, my implementation of Glenn's rules-based strategy results in a slightly lower mean score than his (212 vs. 218). This could be due to a variety of factors, such as unspecified assumptions about tie break order or programming errors on my part. Neither of the other two rules-based strategies are as successful at generating an overall higher score, though I explore in the next section if they achieve their respective goals of earning the upper section bonus and generating Yahtzees. As expected, the two random strategies perform poorly, though there is little difference in performance between the two strategies.

Between the two most promising strategies (greedy and Glenn's rules-based strategy), the greedy strategy tends to produce a tighter final score distribution with lower variance than the rules-based strategy. The greedy strategy score distribution is also bimodal, with peaks around 207 points and 259 points, while the rules-based distribution is smooth. From closer examination, I find that the second peak in the greedy distribution scores largely comes from players rolling either one or two Yahtzees (see appendix figures 1 and 2).

To more rigorously identify a winning strategy, I implement Kim and Nelson's sequential procedure for indifference zone selection [5]. This procedure sequentially eliminates strategies to select the best strategy or best set of strategies whose performance falls within a specified indifference zone parameter (δ), the smallest difference between means "worth" detecting. Unlike normal means selection, Kim and Nelson's sequential approach does not assume known and common variances and allows for common random numbers; pairwise sample variances are calculated in an initial stage, prior to iterative selection. Selection stops once *m* strategies remain.

Table 3 below shows the winning strategies under a variety of choices of δ, as well as if *m* = 1 and *m* = 2 (i.e. one best strategy vs. two best strategies). Alpha is 0.05, meaning the probability of selecting the correct strategy is 95%. I also include the number of observations sampled before the decision was made. Under all assumptions, the greedy strategy is the winning strategy, with the rules-based strategy from Glenn (2007) as a runner-up.

Table 3. Best strategies by final score, based on Kim & Nelson (2001)

Final Score			
δ: Indifference Zone Parameter	**m: Winners Allowed**	**Winners**	**Observations Needed**
0.1	1	Greedy	84,859
0.1	2	Greedy, Rules-Based: Glenn (2007)	10,364
0.2	1	Greedy	30,803
0.2	2	Greedy, Rules-Based: Glenn (2007)	5,359
0.5	1	Greedy	14,958
0.5	2	Greedy, Rules-Based: Glenn (2007)	2,021
1	1	Greedy	3,429
1	2	Greedy, Rules-Based: Glenn (2007)	931
2	1	Greedy	1,337

2	2	Greedy, Rules-Based: Glenn (2007)	461
3	1	Greedy	1,114
3	2	Greedy, Rules-Based: Glenn (2007)	324
4	1	Greedy	850
4	2	Greedy, Rules-Based: Glenn (2007)	217
5	1	Greedy	593
5	2	Greedy, Rules-Based: Glenn (2007)	181
6	1	Greedy	450
6	2	Greedy, Rules-Based: Glenn (2007)	137

b. Number of Yahtzees and upper section bonus

Along with examining the final score, I also investigate how the strategies differ by number of Yahtzees earned and games earning the upper section bonus. See tables 4 and 5 below for sample means and standard deviations for number of Yahtzees earned and % earning upper section bonus. I also repeat Kim and Nelson's sequential procedure for these two metrics, results of which are in tables 6 and 7.

Table 4. Sample means and standard deviations for number of Yahtzees

	Number of Yahtzees			
Strategy	**Sample Mean**	**Sample Standard Deviation**	**95% Confidence Interval, Lower Bound**	**95% Confidence Interval, Upper Bound**
Purely Random	0.00	0.06	0.00	0.00
Tiered Random	0.00	0.06	0.00	0.00
Greedy	0.30	0.51	0.30	0.31
Rules-Based: Glenn 2007	0.51	0.67	0.50	0.51
Rules-Based: Prioritize Upper Section	0.45	0.64	0.45	0.46
Rules-Based: Prioritize Yahtzee	0.57	0.69	0.57	0.58

Table 5. Sample means and standard deviations for % earning upper section bonus

	Has Earned Upper Bonus			
Strategy	**Sample Mean**	**Sample Standard Deviation**	**95% Confidence Interval, Lower Bound**	**95% Confidence Interval, Upper Bound**
Purely Random	0.0%	0.3%	0.0%	0.0%
Tiered Random	0.0%	0.4%	0.0%	0.0%
Greedy	3.3%	17.7%	3.1%	3.4%
Rules-Based: Glenn 2007	23.6%	42.4%	23.3%	23.8%
Rules-Based: Prioritize Upper Section	14.4%	35.1%	14.2%	14.6%
Rules-Based: Prioritize Yahtzee	16.2%	36.9%	16.0%	16.5%

Table 6. Best strategies by number of Yahtzees, based on Kim & Nelson (2001)

Number of Yahtzees			
δ: Indifference Zone Parameter	**m: Winners Allowed**	**Winners**	**Observations Needed**
0.01	1	Rules-Based: Prioritize Yahtzee	14,067
0.01	2	Rules-Based: Glenn (2007), Rules-Based: Prioritize Yahtzee	5,912
0.05	1	Rules-Based: Prioritize Yahtzee	1,563
0.05	2	Rules-Based: Glenn (2007), Rules-Based: Prioritize Yahtzee	921
0.1	1	Rules-Based: Prioritize Yahtzee	730
0.1	2	Rules-Based: Glenn (2007), Rules-Based: Prioritize Yahtzee	366
0.2	1	Rules-Based: Prioritize Yahtzee	228

0.2	2	Rules-Based: Glenn (2007), Rules-Based: Prioritize Yahtzee	162

Table 7. Best strategies by % earning upper section bonus, based on Kim & Nelson (2001)

Has Earned Upper Section Bonus			
δ: Indifference Zone Parameter	**m: Winners Allowed**	**Winners**	**Observations Needed**
0.01	1	Rules-Based: Glenn (2007)	726
0.01	2	Rules-Based: Glenn (2007), Rules-Based: Prioritize Yahtzee	550
0.05	1	Rules-Based: Glenn (2007)	73
0.05	2	Rules-Based: Glenn (2007), Rules-Based: Prioritize Yahtzee	68
0.1	1	Rules-Based: Glenn (2007)	43
0.1	2	Rules-Based: Glenn (2007), Rules-Based: Prioritize Yahtzee	35
0.2	1	Rules-Based: Prioritize Yahtzee	50
0.2	2	Rules-Based: Prioritize Upper Section, Rules-Based: Prioritize Yahtzee	12

The rules-based approach that prioritizes Yahtzees does generate the most Yahtzees of any strategy – 0.57 +/- 0.0043 – followed by the rules-based approach based on Glenn (2007) (0.51 +/- 0.0041). Interestingly, the strategy that optimizes the upper section bonus actually does not achieve the upper section bonus as often as other strategies – specifically, the rules-based strategy based on Glenn (2007) and the strategy that prioritizes Yahtzee. Looking at the distribution of scores by category as well as the distribution of order scored, I find that the upper section bonus strategy tends to score 5s and 6s very early in the game, at the expense of a higher upper section score. On average, this strategy scores 6s on turn 1.86 (+/- 0.01) and 5s on turn 2.45 (+/- 0.01). The average 6s score for this strategy is 14.9 (+/- 0.03) and for 5s is 12.9 (+/- 0.03), implying that the average number of dice of the same value recorded is roughly 2.5, while the standard rules-based strategy is closer to 3. The greedy strategy also only rarely achieves either a Yahtzee or an upper section bonus and is not selected by the Kim and Nelson process for either of these two attributes, despite being the best contender overall for final score.

c. Strengths and weaknesses

Focusing solely on the two best strategies – greedy strategy and my adapted version of Glenn's rules-based approach – I investigate the distributions of scores by category, the rate at which categories are scored as zero, and the order in which categories are scored to see if there are further areas for improvement within these strategies. See tables 8, 9, and 10 for confidence intervals for these metrics. I also juxtapose the category score sample means against the reported category score sample means from Glenn in table 11, an approach he suggests for identifying areas of improvement for a rules-based Yahtzee strategy [1].

Table 8. Sample means and standard deviations for category-level scores

		Category Score			
Category	**Strategy**	**Sample Mean**	**Sample Standard Deviation**	**95% Confidence Interval, Lower Bound**	**95% Confidence Interval, Upper Bound**
1s	Greedy	1.66	1.25	1.65	1.67
1s	Rules-Based: Glenn 2007	2.38	0.98	2.37	2.38
2s	Greedy	4.24	2.02	4.22	4.25
2s	Rules-Based: Glenn 2007	4.78	1.94	4.77	4.79
3s	Greedy	6.66	2.74	6.64	6.68
3s	Rules-Based: Glenn 2007	7.25	2.94	7.23	7.27

4s	Greedy	8.81	3.43	8.79	8.83
4s	Rules-Based: Glenn 2007	9.70	3.97	9.67	9.72
5s	Greedy	11.05	4.00	11.03	11.07
5s	Rules-Based: Glenn 2007	14.39	4.76	14.36	14.42
6s	Greedy	12.57	4.70	12.54	12.59
6s	Rules-Based: Glenn 2007	17.58	5.46	17.55	17.61
3 of a kind	Greedy	22.85	4.13	22.82	22.88
3 of a kind	Rules-Based: Glenn 2007	17.40	6.00	17.36	17.44
4 of a kind	Greedy	17.46	9.41	17.40	17.52
4 of a kind	Rules-Based: Glenn 2007	14.51	7.56	14.46	14.56
Full house	Greedy	23.69	5.56	23.66	23.73
Full house	Rules-Based: Glenn 2007	13.80	12.43	13.72	13.87
Small straight	Greedy	29.98	0.76	29.98	29.99
Small straight	Rules-Based: Glenn 2007	27.57	8.18	27.52	27.62
Large straight	Greedy	36.23	11.69	36.16	36.30
Large straight	Rules-Based: Glenn 2007	24.50	19.49	24.38	24.62
Yahtzee	Greedy	16.30	29.66	16.11	16.48
Yahtzee	Rules-Based: Glenn 2007	30.23	45.51	29.94	30.51
Chance	Greedy	21.74	3.25	21.72	21.76
Chance	Rules-Based: Glenn 2007	19.85	4.82	19.82	19.88
Upper section bonus	Greedy	1.14	6.21	1.10	1.18
Upper section bonus	Rules-Based: Glenn 2007	8.25	14.86	8.16	8.34
Yahtzee bonus	Greedy	2.20	14.68	2.11	2.29
Yahtzee bonus	Rules-Based: Glenn 2007	9.82	29.76	9.64	10.01

Table 9. Sample means and standard deviations for % scored as zero

		% Scored as Zero			

Category	Strategy	Sample Mean	Sample Standard Deviation	95% Confidence Interval, Lower Bound	95% Confidence Interval, Upper Bound
1s	Greedy	21.7%	41.2%	21.4%	21.9%
1s	Rules-Based: Glenn 2007	6.7%	25.0%	6.5%	6.8%
2s	Greedy	3.1%	17.4%	3.0%	3.2%
2s	Rules-Based: Glenn 2007	6.2%	24.2%	6.1%	6.4%
3s	Greedy	0.7%	8.3%	0.6%	0.7%
3s	Rules-Based: Glenn 2007	6.0%	23.7%	5.8%	6.1%
4s	Greedy	0.2%	4.2%	0.2%	0.2%
4s	Rules-Based: Glenn 2007	6.1%	23.9%	5.9%	6.2%
5s	Greedy	0.0%	2.2%	0.0%	0.1%
5s	Rules-Based: Glenn 2007	2.8%	16.5%	2.7%	2.9%
6s	Greedy	0.0%	1.0%	0.0%	0.0%
6s	Rules-Based: Glenn 2007	2.3%	15.0%	2.2%	2.4%
3 of a kind	Greedy	0.1%	2.6%	0.1%	0.1%
3 of a kind	Rules-Based: Glenn 2007	1.1%	10.4%	1.0%	1.2%
4 of a kind	Greedy	15.8%	36.4%	15.5%	16.0%
4 of a kind	Rules-Based: Glenn 2007	9.5%	29.3%	9.3%	9.7%
Full house	Greedy	5.2%	22.2%	5.1%	5.4%
Full house	Rules-Based: Glenn 2007	44.8%	49.7%	44.5%	45.1%
Small straight	Greedy	0.1%	2.5%	0.0%	0.1%
Small straight	Rules-Based: Glenn 2007	8.1%	27.3%	7.9%	8.3%
Large straight	Greedy	0.1%	2.5%	0.0%	0.1%
Large straight	Rules-Based: Glenn 2007	8.1%	27.3%	7.9%	8.3%
Yahtzee	Greedy	71.8%	45.0%	71.5%	72.1%
Yahtzee	Rules-Based: Glenn 2007	59.2%	49.1%	58.9%	59.5%

Chance	Greedy	0.0%	0.0%	0.0%	0.0%
Chance	Rules-Based: Glenn 2007	0.0%	0.0%	0.0%	0.0%

Table 10. Sample means and standard deviations for order scored

		Round Scored			
Category	**Strategy**	**Sample Mean**	**Sample Standard Deviation**	**95% Confidence Interval, Lower Bound**	**95% Confidence Interval, Upper Bound**
1s	Greedy	11.41	1.20	11.40	11.41
1s	Rules-Based: Glenn 2007	6.99	3.28	6.97	7.01
2s	Greedy	10.28	1.51	10.27	10.29
2s	Rules-Based: Glenn 2007	6.95	3.35	6.93	6.97
3s	Greedy	8.97	1.90	8.96	8.98
3s	Rules-Based: Glenn 2007	7.00	3.37	6.98	7.02
4s	Greedy	7.75	2.05	7.74	7.76
4s	Rules-Based: Glenn 2007	7.12	3.39	7.10	7.15
5s	Greedy	6.84	2.00	6.82	6.85
5s	Rules-Based: Glenn 2007	5.52	3.40	5.49	5.54
6s	Greedy	5.88	1.91	5.87	5.89
6s	Rules-Based: Glenn 2007	5.06	3.37	5.04	5.08
3 of a kind	Greedy	3.83	2.09	3.81	3.84
3 of a kind	Rules-Based: Glenn 2007	6.21	3.07	6.19	6.23
4 of a kind	Greedy	7.42	3.56	7.39	7.44
4 of a kind	Rules-Based: Glenn 2007	6.24	3.33	6.22	6.26
Full house	Greedy	6.12	2.96	6.10	6.14
Full house	Rules-Based: Glenn 2007	8.73	3.72	8.71	8.76
Small straight	Greedy	2.89	2.01	2.88	2.90

Small straight	Rules-Based: Glenn 2007	5.68	3.58	5.66	5.70
Large straight	Greedy	5.64	3.90	5.62	5.67
Large straight	Rules-Based: Glenn 2007	7.92	4.36	7.90	7.95
Yahtzee	Greedy	11.56	2.96	11.54	11.58
Yahtzee	Rules-Based: Glenn 2007	10.18	4.08	10.16	10.21
Chance	Greedy	2.42	1.24	2.41	2.43
Chance	Rules-Based: Glenn 2007	7.40	2.90	7.38	7.41

Table 11. Category score sample means vs. optimal strategy

	Sample Mean Category Score		
Category	**Greedy**	**Rules-Based: Glenn 2007**	**Optimal Strategy**
1s	1.66	2.38	1.88
2s	4.24	4.78	5.28
3s	6.66	7.25	8.57
4s	8.81	9.70	12.16
5s	11.05	14.39	15.69
6s	12.57	17.58	19.19
3 of a kind	22.85	17.40	21.66
4 of a kind	17.46	14.51	13.1
Full house	23.69	13.80	22.59
Small straight	29.98	27.57	29.46
Large straight	36.23	24.50	32.71
Yahtzee	16.30	30.23	16.87
Chance	21.74	19.85	22.01
Upper section bonus	1.14	8.25	23.84
Yahtzee bonus	2.20	9.82	9.58

Between the two sub-optimal strategies, the greedy strategy generally tends to overindex on lower section scores (with the exception of Yahtzee) compared to the rules-based strategy. Category-level scores for all lower section categories except Yahtzee are statistically significantly higher for the greedy strategy than for the rules-based strategy. The converse is also true – the greedy strategy's scores for upper section categories are statistically significantly lower than the rules-based strategy's scores.

Additionally, the greedy strategy tends to leave scoring 1s and 2s to the very end of the game (1s are scored at round 11.41, on average, while 2s are scored at round 10.28) and is also statistically significantly more likely to score the 1s category as zero (21.7% vs. 6.7%). Interestingly, the optimal strategy tends to behave more like the greedy strategy than the rules-based strategy, at least regarding the 1s category: the average score for 1s with the optimal strategy is 1.88 vs. 2.38 for the rules-based strategy. The rules-based strategy generally reserves the full house category as its "throwaway" category to mark as zero (zero 45% of the time) but these comparisons suggest that the 1s category is a better option, presumably because its expected score of 0.83 is the lowest of any category.

These results also suggest that Yahtzees are not the primary determinant of overall score. I already found in the previous section that the rules-based strategy that optimizes toward achieving a Yahtzee does successfully generate the most Yahtzees on average but underperforms on the final score. Now, comparing the more successful rules-based strategy category scores against the optimal strategy's category scores, I see that the optimal strategy is more closely in line with the greedy strategy, with an average Yahtzee score of 16.87 vs. 30.23 for the rules-based strategy.

The largest discrepancies between category-level scores for the rules-based strategy and the optimal strategy that contribute to the latter's overall higher final score are in the large straight category and the upper section bonus. On average, the rules-based strategy scores 24.50 in the large straight category and earns an average upper section bonus of 8.25, compared against 32.71 and 23.84, respectively, for the optimal strategy. As I found in the previous section, scores in the 5s and 6s categories are major determinants of whether a strategy achieves the upper section bonus. Indeed, while the rules-based strategy scores higher in these categories than the greedy strategy, the optimal strategy produces even higher scores that imply that the average roll applied is at least three-of-a-kind (15.69 and 19.19 for 5s and 6s, respectively).

If I were to iterate further on the rules-based strategy, these are a few modifications I would consider simulating:

1. Keep not only [2, 3, 4] and [3, 4, 5] but also other sequences of consecutive numbers or near-consecutive numbers (e.g. [1, 3, 4]) in order to increase probability of achieving a large straight.
2. Keep tripletons and doubles only if the upper section has not been earned for a particular category.
3. Delay scoring 4s, 5s, and 6s until at least three of a kind are obtained and/or at least six turns have elapsed in order to increase probability of earning the upper section bonus.
4. If all potential scores are zero, always choose the 1s category to apply the zero.

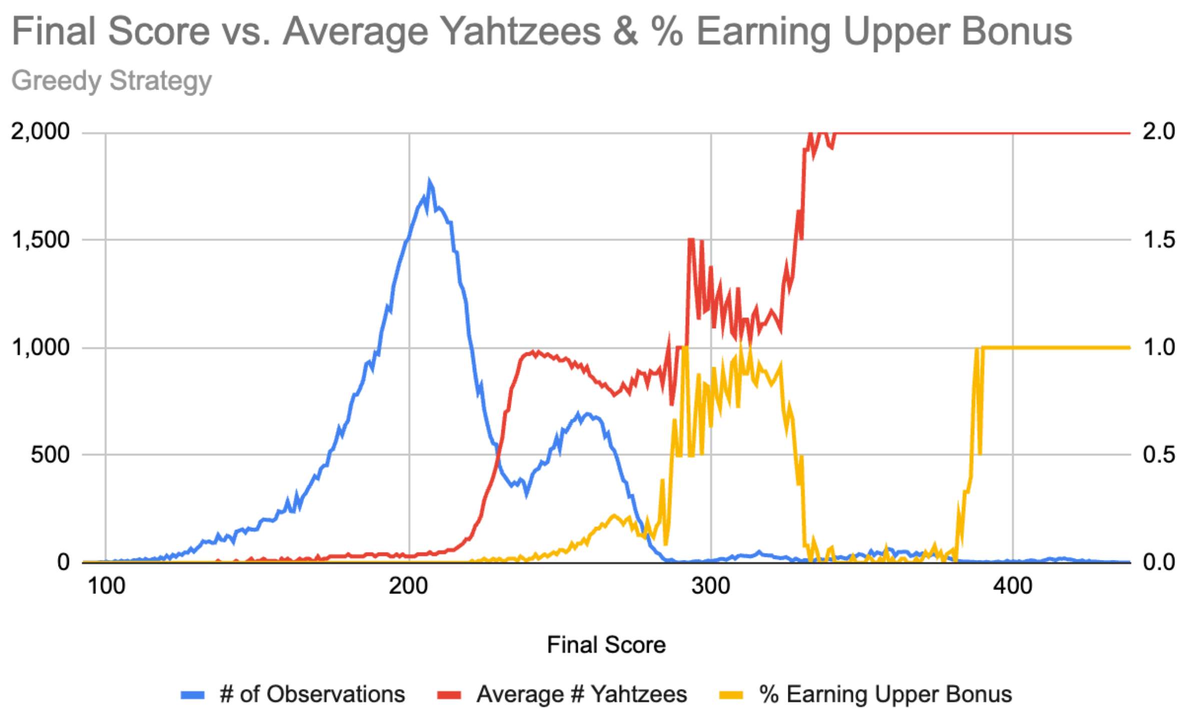
As with before, I could apply the Kim-Nelson procedure to evaluate how these further modifications compare against the existing rules-based and greedy strategies.

VI. Conclusion

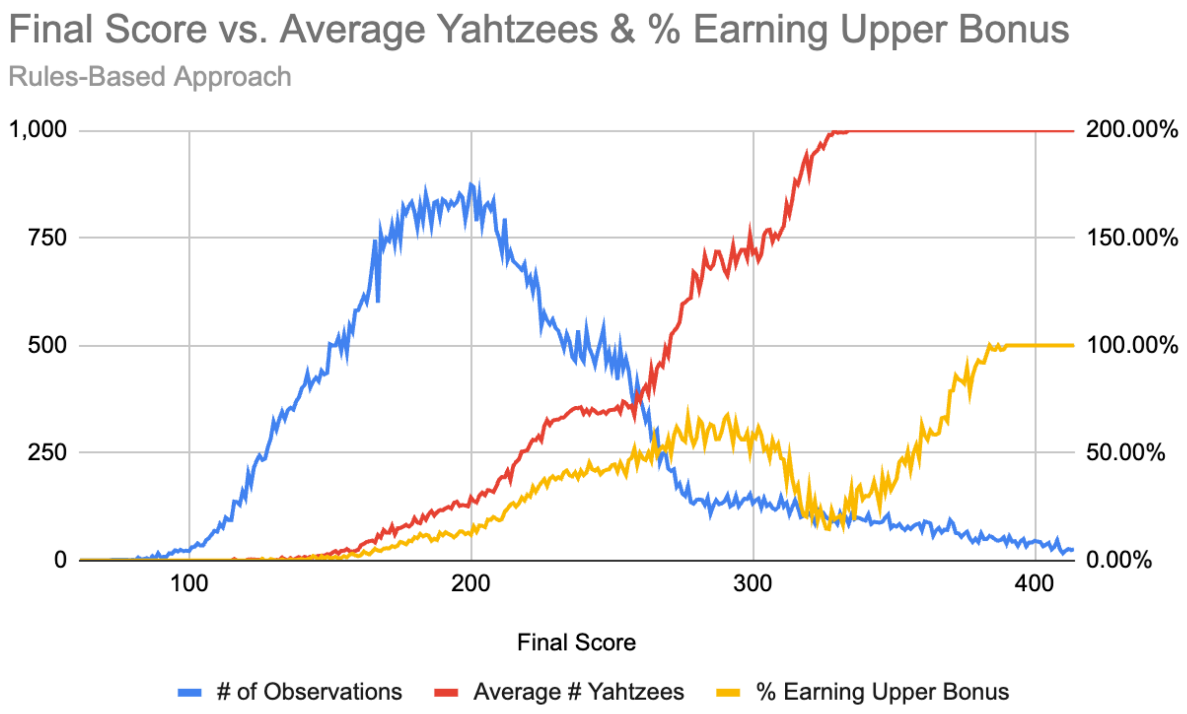
Simulation is a powerful tool to evaluate performant strategies when there are too many potential paths to enumerate a strategy deterministically, as is the case with Yahtzee. Among the strategies I implemented, I find that a greedy strategy is most performant, followed closely by a rules-based strategy, though there are opportunities for improvement to the latter to achieve outcomes closer to the true optimal strategy. Given a rules-based strategy is more intelligible to a real player than a greedy strategy, this is the approach I would suggest to someone who is looking to improve their Yahtzee score.

Appendix

Appendix Figure 1. Final score vs. average Yahtzees & % earning upper bonus (greedy strategy)

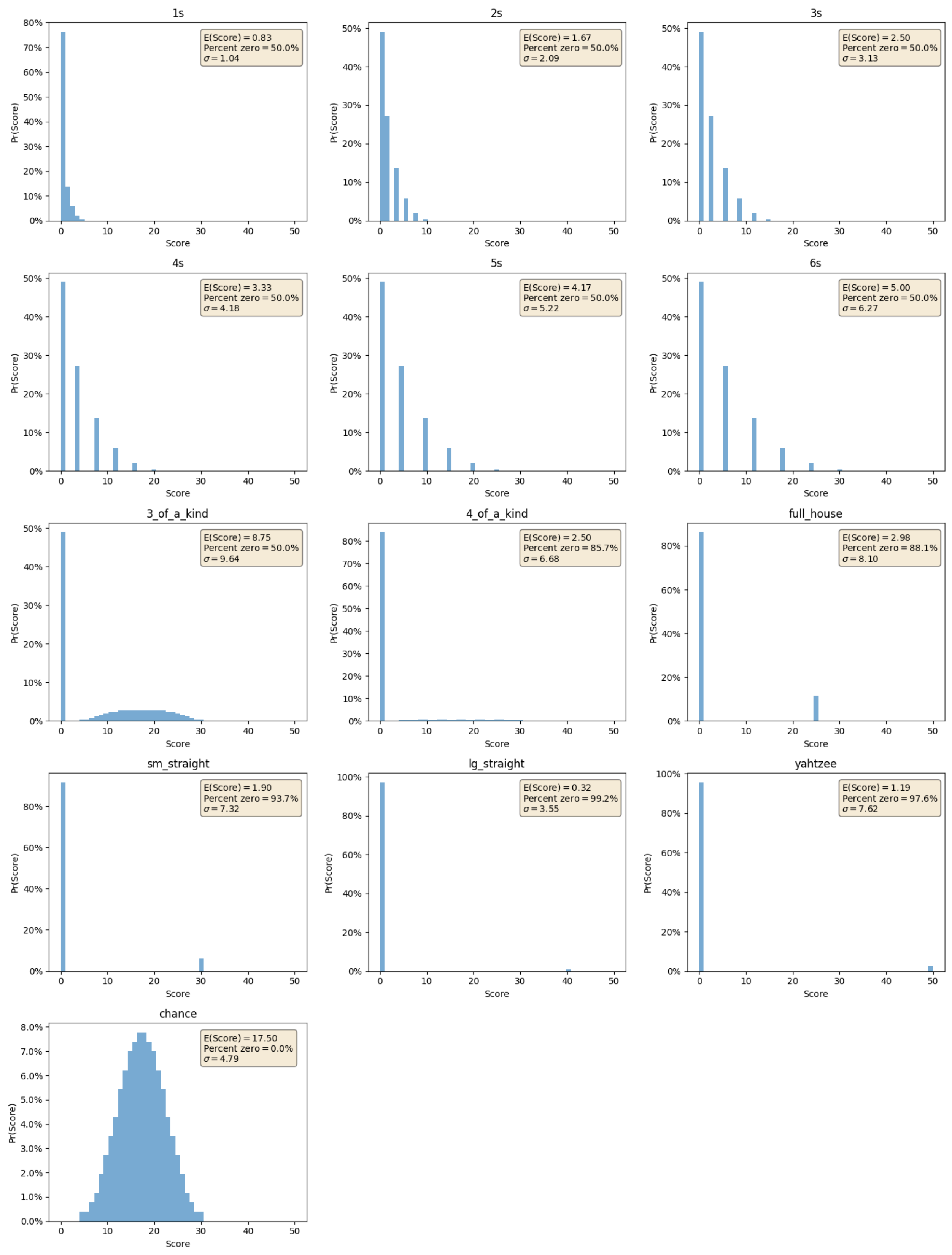


Appendix Figure 2. Final score vs. average Yahtzees & % earning upper bonus (Rules-based approach)



Appendix Figure 3. Expected score by category for a single roll

Distribution of Scores by Category for a Single Roll



References

[1] J. R. Glenn, *Computer strategies for solitaire Yahtzee*, IEEE Symposium on Computational Intelligence and Games (CIG), 2007.

[2] Hasbro, *Yahtzee classic instructions*, available at https://instructions.hasbro.com/api/download/00950_en-ca_yahtzee-classic.pdf.

[3] Hasbro, *Yahtzee History*, available at https://web.archive.org/web/20070808043322/http://www.hasbro.com/default.cfm?page=ci_history_yahtzee.

[4] K. Kelly and J. Liese, *Let's get rolling! Exact optimal solitaire Yahtzee*, Math. Mag. 95 (2022), no. 3, 205--219.

[5] S.-H. Kim and B. L. Nelson, *A fully sequential procedure for indifference-zone selection in simulation*, ACM Trans. Model. Comput. Simul. 11 (2001), no. 3, 251--273.

[6] NumPy Developers, *Permuted congruential generator (64-bit, PCG64)*, NumPy Documentation, available at https://numpy.org/doc/stable/reference/random/bit_generators/pcg64.html#numpy.random.PCG64.

[7] T. Verhoeff, *How to maximize your score in solitaire Yahtzee*, Technical Report, Eindhoven University of Technology, Faculty of Mathematics and Computing Science, June 1999.

[8] P. Woodward, *Yahtzee®: The solution*, CHANCE 16 (2003), no. 1, 18--22.

Additional Sources

Coding assistance:

Matplotlib development team, *Placing text boxes*, Matplotlib 3.3.4 Documentation, available at https://matplotlib.org/3.3.4/gallery/recipes/placing_text_boxes.html.

NumPy Developers, *numpy.std*, NumPy Documentation, available at https://numpy.org/doc/stable/reference/generated/numpy.std.html.

Stack Overflow, *Get count of tuples in list regardless of element orders*, available at https://stackoverflow.com/questions/49724651/get-count-of-tuples-in-list-regardless-of-elements-orders.

Stack Overflow, *How to see if the list contains consecutive numbers*, available at https://stackoverflow.com/questions/33575235/how-to-see-if-the-list-contains-consecutive-numbers.

Stack Overflow, *Numpy array of objects*, available at https://stackoverflow.com/questions/4877624/numpy-array-of-objects.

README poetry instructions were largely taken from the standard setup used at my company.

Artificial intelligence:

The text of this paper and the coding are my original work. I have access to GitHub Copilot through my job but have autocomplete and inline suggestions disabled on my personal computer and only employ the chat sparingly. I sought out assistance from artificial intelligence for the following specific uses:

1. GitHub Copilot chat: Asked for advice on how to optimize the runtime of the greedy strategy. It suggested caching rolls outside of the run_strategy() function.
2. GitHub Copilot chat: Asked to find a bug that was resulting in lower overall score for greedy strategy than expected. It found that I had forgotten to convert an array to a list.
3. GitHub Copilot chat: Asked to double check the logic of my implementation of the Kim and Nelson (2001) sequential procedure. No issues found.
4. GitHub Copilot chat: Asked to refactor *run_strategy*() function for random and greedy strategies to add print out statements (already manually added to rules-based strategy).

5. GitHub Copilot chat: Asked how to display integers as percentages on y-axis in matplotlib graphs.
6. Google Gemini: Uploaded links to sources and asked it to generate citations in AMS style.