Timothy Adams
[redacted]

[redacted]
June 1st, 2020

The Optimization of the Blackjack Strategy with Genetic Algorithms

Introduction

Blackjack is one of the most popular betting games in the world and is seen in each and every casino there is. Like most casino games, blackjack has been heavily studied in order for the player to extract some sort of an advantage over the casino and shift the odds of winning their bet in their favor. Specifically, the aspect of blackjack that is most often studied to accomplish this goal is the player strategy. The player strategy is essentially a set of moves that the player should make in every possible situation in blackjack that would maximize the chance of winning their bet. Put more simply, the development of a player strategy is an optimization problem in which we wish to maximize the probability of the player winning their bet. An interesting technique for optimization for the given problem is the genetic algorithm. Therefore, an interesting question arises: what would the blackjack strategy developed by a genetic algorithm look like? Additionally, how would the performance of this strategy compare to other player strategies such as the well-established basic strategy or even a completely random strategy.

Model/Theory

In order to to accomplish the optimization of the blackjack strategy, we will be using what is known as a genetic algorithm. This reinforcement learning algorithm takes principles from Darwin's theory of natural selection and applies them to a computational setting. The overview of this optimization process is that the best performing (fittest) individuals, in our case blackjack strategies, of a population reproduce in order to create offspring which are more fit than the previous generation. This process repeats itself over many generations until an optimal solution is found. To develop such an algorithm for blackjack strategies, we must consider the important aspects of the genetic algorithm: the chromosome, the population, the fitness function, selection, crossover and mutation. Addtionally, we must establish how these various components can be applied to learning blackjack.

The chromosome is an ordered set of parameters (known as genes) which represent a possible solution to the problem. In the context of the optimization of the blackjack strategy, the chromosome is simply the blackjack strategy itself. The blackjack model used in this paper limits the player moves to two possibilities: hitting and standing. Therefore, the strategy model which corresponds to this particular model of blackjack can be represented by a pair of 2-dimensional arrays detailing which move to make based on the player's count and the dealer's face-up card. The representation requires two tables as the strategy differs depending on whether the player has a hard or soft count. Given that the strategy is limited to two moves, the entries of the arrays can have binary values (1 indicating *Hit*; 0 indicating *Stand*). This strategy model can be illustrated in the following figure representing the well-established basic strategy (Shackleford, “Blackjack Basic Strategy for Infinite Decks”, 17:17).

Hard Count	Dealer’s Face-Up Card									
Player Count	2	3	4	5	6	7	8	9	10	11
4	1	1	1	1	1	1	1	1	1	1
5	1	1	1	1	1	1	1	1	1	1
6	1	1	1	1	1	1	1	1	1	1
7	1	1	1	1	1	1	1	1	1	1
8	1	1	1	1	1	1	1	1	1	1
9	1	1	1	1	1	1	1	1	1	1
10	1	1	1	1	1	1	1	1	1	1
11	1	1	1	1	1	1	1	1	1	1
12	1	1	0	0	0	1	1	1	1	1
13	0	0	0	0	0	1	1	1	1	1
14	0	0	0	0	0	1	1	1	1	1
15	0	0	0	0	0	1	1	1	1	1
16	0	0	0	0	0	1	1	1	1	1
17	0	0	0	0	0	0	0	0	0	0
18	0	0	0	0	0	0	0	0	0	0
19	0	0	0	0	0	0	0	0	0	0
20	0	0	0	0	0	0	0	0	0	0
21	0	0	0	0	0	0	0	0	0	0

Soft Count	Dealer’s Face-Up Card									
Player Count	2	3	4	5	6	7	8	9	10	11
12	1	1	1	1	1	1	1	1	1	1
13	1	1	1	1	1	1	1	1	1	1
14	1	1	1	1	1	1	1	1	1	1
15	1	1	1	1	1	1	1	1	1	1
16	1	1	1	1	1	1	1	1	1	1
17	1	1	1	1	1	1	1	1	1	1
18	0	0	0	0	0	0	0	1	1	1
19	0	0	0	0	0	0	0	0	0	0
20	0	0	0	0	0	0	0	0	0	0
21	0	0	0	0	0	0	0	0	0	0

1: Hit

0: Stand

Figure 1. Basic Strategy Chromosome Model

Next, the population consists of set of individuals with unique chromosomes which compete against one another in order to contribute their genetic material to the following generations. In our case, these individuals have a chromosome representing a blackjack strategy. In the initial population, individuals are given randomized chromosomes as there is not a previous generation from which they can inherit genes. Additionally, the chromosomes are randomized to produce the genetic diversity required to converge to the fittest possible solution.

The fitness function assigns a fitness value to each individual in a population. This fitness value indicates the effectiveness of an individual’s solution to the optimization problem (chromosome). Additionally, the fitness score determines the probability that an individual will be selected to pass their genes to the next generation. For the given problem, the fitness function would consist of a blackjack simulation using an individual’s strategy for a specified number of hands. The fitness value extracted from this simulation would represent the net change in the player’s cash (final cash amount – initial cash amount).

Following the assessment of each individual’s fitness score, the fittest individuals are selected in order to pass on their genetic material to the next generation’s population. While there are many selection methods available (roulette wheel selection, ranked selection, …), the selection technique used in this evolutionary blackjack strategy algorithm was 4 candidate tournament selection. This selection method groups 4 individuals from the population at random and selects the individual having the highest fitness within this sub-group. This process is repeated throughout the entire population until a quarter of the initial population has been selected for reproduction (Sommerville).

Crossover is the process by which selected individuals having a high fitness score relative to their population produce offspring in order to populate the next generation. Ideally, the offspring would inherit the optimal portions of their parents’ genetic material resulting in an increase in the population’s fitness score when compared to the previous generation. Crossover requires two parent individuals (which we will call parent A and parent B) in order to produce two offspring individuals (which we will call offspring 1 and offspring 2). The crossover technique used in this paper is based on single point crossover. However, the crossover method had to be expanded as single point crossover applies to one-dimensional chromosomes while the chromosomes present in this problem are two-dimensional. The process begins by randomly selecting an x-axis and a y-axis which will define which portions of each parent individual's genetic material will be inherited by each of their offspring. We can view the portions of genetic material sectioned off by the set of axes as quadrants similar to the quadrants defined by a two-dimensional Cartesian coordinate system (top right section: Q1, top left section: Q2, …). After the axes have been defined, offspring 1 will receive genetic material from parent A's third and fourth quadrant and from parent B's first and third quadrants. On the other hand, offspring 2 will receive the genes present in parent A's first and third quadrant as well as from parent B's second and fourth quadrants. While this definition is quite abstract, this crossover method can be visualized quite nicely through the following figure:

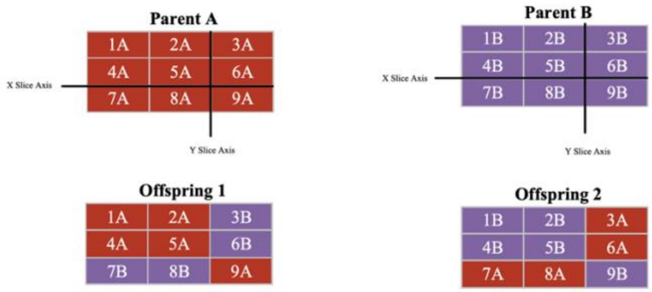

Figure 2. Two-Dimensional Crossover Method

The final important aspect of genetic algorithm is mutation. Mutation is a low probability event in which one or more genes are flipped meaning that the value is changed to its binary counterpart in order to maintain the genetic diversity of the population. However, mutation has been excluded in the model developed for this paper as it has been shown that higher rates of mutation make for worse performing solutions in the context of the optimization of blackjack strategies (Sommerville).

Numerical Methods

Now that we have a description of the model used in the algorithm, let us address how these features were implemented into the computational environment. The individual is represented by the Individual class. This class has the sole purpose of holding the Individual's chromosome as the chromosome is too large to be displayed in the representation of the population. While on the subject of the chromosome, this entity was integrated as a list containing two entries which represent the hard and soft count strategies. Each of these strategies is represented through Pandas DataFrame objects which have an index representing the player's count and columns representing the dealer's face-up card. The entries of these DataFrames are integers of value 1 (indicating *Hit*) or 0 (indicating *Stand*).

Next, the population is also represented by a Pandas DataFrame having two columns and a number of rows corresponding to the population size. The DataFrame has a column for Individual objects and a column for their respective fitness value. The initial population was created by assigning the individuals column to a list of Individual objects having randomized strategies created by a dedicated random strategy creation function.

The fitness function is modeled as a function which takes a population DataFrame as a parameter and enters the fitness value of each Individual object in the fitness column of the inputted DataFrame. To accomplish this task, the function iterates over every individual in the dedicated column of the population. For each individual, the function simulates a specified number of hands of blackjack with the given individual's strategy using various classes (Player, Dealer, Shoe) as well as game logic defined in the function itself. Once the function has completed the simulation for an individual's strategy, the fitness value is recorded as the net change in cash (final cash – initial cash). Finally, the function enters the fitness value in the dedicated column of the population DataFrame in the row corresponding to the Individual object being evaluated.

The selection process is represented by a function which takes a population DataFrame as a parameter and returns a list of the selected individuals from the population entered. First, the function iterates over the population and isolates sub-groups of 4 individuals. The individual with the highest fitness score within their respective group is appended to the list of selected individuals which is returned as the value of the function call once the function has iterated over the entire population.

Finally, the reproduction process is modeled using two functions: the crossover function and the reproduction function. The crossover function requires two parent individuals as parameters and returns a list of two entries containing the offspring Individual objects. First, the function selects an x-axis and a y-axis at random. Next, the quadrants of the parent chromosomes defined by the set of axes are isolated using Pandas DataFrame indexing and rearranged according to the crossover technique described in the *Model/Theory* section of the paper using Pandas' join and concatenate functions in order to form the chromosomes of the offspring. Finally, a list of two individuals which are equipped with the newly created strategies is returned. Moving on, the reproduction function takes a list of selected individuals, representing a quarter of the initial population, created by the selection function as an input and returns a list of Individual objects representing the population of following generation. To accomplish this task, the function iterates over the inputted list and selects pairs of individuals to reproduce. This pair of individuals is passed into the crossover function four times in order to balance the fact that a quarter of the population has been selected to reproduce. The list of individuals created by the crossover function is appended to the list of individuals representing the next generation's population. This list is shuffled and returned as the value of the function call.

Results

Now that we have established the model being evaluated as well as the computational techniques used to apply it, we may now examine the results of the evolutionary blackjack strategy algorithm. In order to obtain the following solution, the population size was set to 400 individuals; the number of blackjack hands played per simulation in the fitness function was limited to 10,000; and the number of generations over which the genetic algorithm was allowed to evolve was 100 generations. The blackjack strategy obtained from the evolutionary algorithm using these parameters can be represented as follows:

Hard Count	Dealer's Face-Up Card: 2	3	4	5	6	7	8	9	10	11
Player Count 4	1	1	1	1	1	1	1	1	1	1
5	1	1	1	1	1	1	1	1	1	1
6	1	1	1	1	1	1	1	1	1	1
7	1	1	1	1	1	1	1	1	1	1
8	1	1	1	1	1	1	1	1	1	1
9	1	1	1	1	1	1	1	1	1	1
10	1	1	1	1	1	1	1	1	1	1
11	1	1	1	1	1	1	1	1	1	1
12	1	1	1	1	1	1	1	1	1	1
13	1	1	0	1	0	1	1	1	1	1
14	0	0	0	0	0	1	1	1	1	1
15	0	0	0	0	0	1	1	1	1	1
16	0	0	0	0	0	1	1	1	1	1
17	0	0	0	0	0	0	0	0	1	0
18	0	0	0	0	0	0	0	0	0	0
19	0	0	0	0	0	0	0	0	0	0
20	0	0	0	0	0	0	0	0	0	0
21	0	0	0	0	0	0	0	0	0	0

Soft Count	Dealer's Face-Up Card: 2	3	4	5	6	7	8	9	10	11
Player Count 12	1	1	1	1	1	1	1	1	1	1
13	1	1	1	1	1	1	1	1	1	1
14	1	1	1	1	1	1	1	1	1	1
15	1	1	1	1	1	1	1	1	1	1
16	1	1	1	1	1	1	1	1	1	1
17	1	1	1	1	1	1	1	1	1	1
18	0	1	0	0	1	1	0	1	1	1
19	0	0	0	0	0	0	0	0	1	1
20	0	0	0	0	0	0	0	0	0	0
21	0	0	0	0	0	0	0	0	0	0

1: Hit

0: Stand

Figure 3. Genetic Algorithm Blackjack Strategy Solution

Let us test this result against two solutions which lie on opposite ends of the spectrum of effectiveness in order to see where the effectiveness of the strategy obtained using this algorithm lies. For the upper bound of efficiency, we will use the established optimal solution known as basic strategy (refer to *Figure 1* for strategy representation). For the lower bound of proficiency, we will use a randomly generated strategy similar to the strategies generated for the initial population. For this test, we will simulate the use of the various strategies over 100,000 hands of blackjack at an initial cash amount of 100,000$ while betting 1$ per hand. The results can be illustrated through the following graph of the cash amount versus the turn number:

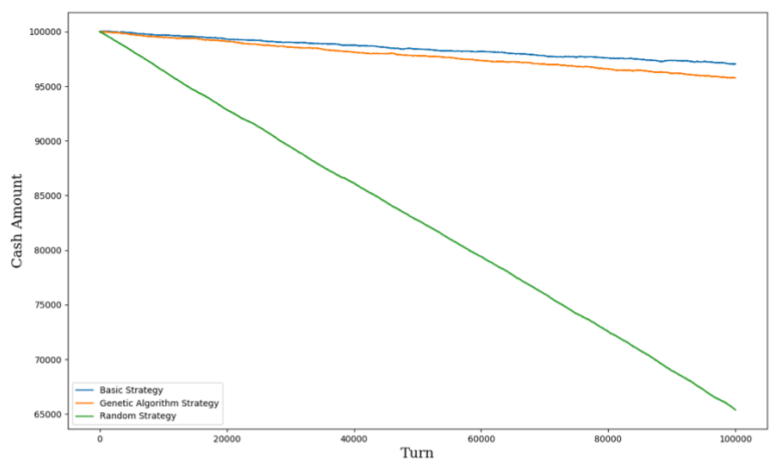

Figure 4. Cash Amount versus Turn for Various Strategies

As you can see, the strategies' performance ranked in descending order is basic strategy (final cash amount: 97044.5$) followed closely by the genetic algorithm strategy (final cash amount: 95759.5$) and finally the randomized strategy (final cash amount: 65379.5$). However, the final cash amount used to assess the effectiveness of each of these strategies is unique to the betting conditions (number of wagers and amount per wager). Therefore, we should evaluate these strategies using a metric which is independent of these parameters. A metric independent of betting conditions is the house edge which represents the percentage of the player's wager that they lose on average (Carter). This value can be expressed by the equation:

$$House\ Edge = \frac{Cash_{final} - Cash_{initial}}{nbr\ of\ hands\ played \times bet\ ammount} \times 100\%$$

Applying this formula to the performance of each of the strategies, we can determine that using basic strategy yields a house edge of 2.96%, employing the genetic algorithm strategy yields 4.24% and applying the randomized strategy yields 34.6%.

Discussion

From these results, we can observe that the blackjack strategy developed by the genetic algorithm is quite successful and is very similar to the unanimous optimal blackjack strategy, basic strategy, in terms of the composition of the strategy as well as its performance. However, the algorithm did not completely converge to the ultimate solution. This can be due to a number of reasons, but the principal cause for error lies in the number of blackjack hands played in each of the fitness function simulations. The fewer hands of blackjack played in a simulation make for greater variability in the fitness score as blackjack is an innately random game meaning that a large number of hands must be played in order to lessen the effect of this unpredictability (Sommerville). This inaccuracy in the fitness score effects the algorithm when it gets to later generations as the strategies become more similar and are only differentiated by a couple of genes. In fact, the fitness function will not be able to accurately assess the small differences between each of the strategies meaning that it cannot accurately identify the optimal solutions in a given population. In our case, the use of 10,000 hands of blackjack per simulation was therefore insufficient. Greg Sommerville in his article on machine learning blackjack even states that "a minimum of 100,000 hands is probably reasonable, because that is the point at which the variability starts to flatten out" (Sommerville). However, this minimum number of hands was not used due to the amount of time these simulations would take.

Furthermore, the results of this test illustrate the inaccuracy of the blackjack model used in this paper through the discrepancy between the experimental and theoretical value the house edge of basic strategy. In fact, the experimental value was found to be 2.96% while the theoretical value is 0.447% (Shackleford, "Blackjack House Edge"). This discrepancy is likely due to the blackjack model which limits the player moves to hitting and standing. If the model were to include other moves, such as doubling and splitting, the experimental value of the house edge would become closer to the theoretical value as these moves increase the player's chances at winning cash thereby decreasing the house edge.

References

Carter, James. "Blackjack House Edge." *Gaming the Odds*, 2015, Web. Date of Access: 26/05/2020.

Mallawaarachchi, Vijini. "Introduction to Genetic Algorithms — Including Example Code." *Towards Data Science*, 2017, Web. Date of Access: 28/04/2020.

Sommerville, Greg. "Winning Blackjack using Machine Learning." *Towards Data Science*, 2019, Web. Date of Access: 24/04/2020.

Shackleford, Michael. "Blackjack House Edge." *The Wizard of Odds*, 2015, Web. Date of Access: 26/05/2020.

Shackleford, Michael. "Blackjack Basic Strategy for Infinite Decks." *Youtube*, 2014, Web. Date of Access: 12/04/2020