Alfie Skues

Registration number 100397928

2025

Implementation and Evaluation of Strategies in Blackjack

Supervised by Dr Ben Milner

University of East Anglia
Faculty of Science
School of Computing Sciences

Abstract

This project investigates the effectiveness of various strategies within simulated games of Blackjack. The project began with the implementation of Blackjack mechanics and included a baseline strategy that mimics the dealer's behaviour. Additional strategies such as Never Bust and all variations of Basic Strategy were then implemented. Each strategy was simulated over 10,000 games to evaluate performance and generate reliable long-term results. Each game play method produced negative results but gradually reduced the house's edge, with Full Basic Strategy being the most advantageous. Card counting methods, such as Hi-Lo, Omega II and Wong Halves, were then implemented with the most effective game play strategy. Each of these methods produced long-term profits for the player. This project demonstrated that incorporating successful card counting, on top of optimal play, can shift the edge in favour of the player under the right conditions.

Acknowledgements

I would like to thank my advisor, Dr Ben Milner, for giving his time, knowledge and assistance throughout all stages of this project.
Also, I would like to give acknowledgements to my family for giving me motivation and support throughout university.

Contents

List of Figures

List of Tables

1. Introduction

1.1. Problem statement

Blackjack has been the most popular casino game for decades, known for its simple rules and quick gameplay. Unlike many other casino games, Blackjack is a game of odds that can be influenced by the decisions and skill of the player. Through the use of optimal strategies, such as basic strategy and card counting, players can significantly adjust their expected outcomes.

However, accurately measuring the effectiveness of these strategies in real-world play can be difficult due to human error, countermeasures by the casinos and time. Simulating Blackjack offers a controlled and repeatable environment where the impact of different strategies can be objectively evaluated over hundreds of thousands of games, which would take years in real-life.

1.2. Main Aim

The main aim of this project was to develop a simulation system for Blackjack that can evaluate different playing strategies, such as game play and card counting methods - while allowing for a statistical comparison of their performance over a large number of games.

1.3. Objectives

1. Develop a Blackjack game that supports standard rules, including hitting, standing, splitting, doubling and dealer behaviour.
2. Create a simulation environment to conduct tests and produce results.
3. Implement different methods of game play through Mimic the Dealer, Never Bust and Basic Strategy.
4. Simulate and evaluate the game play methods to find the most effective.
5. Using the most effective game play method, implement various card counting strategies such as Hi-Lo, Omega II and Wong Halves.

6. Compare outcomes of the card counting methods.
7. Produce findings and results in a visual manner to support the outcomes identified.

1.4. Structure of the Report

This report is organised as follows:

- **Section 2: Background** - Provides an overview of game simulation concepts and introduces context of Blackjack
- **Section 3: Blackjack Implementation** - Describes the technical implementation of the Blackjack game, including Blackjack basics, key design and simulation mechanics.
- **Section 4: Gameplay Strategies and Analysis** - Examines different Blackjack strategies, presents results and discusses observations to lead to the next section.
- **Section 5: Card Counting Techniques** - Explores card counting methods, evaluates their impact on gameplay and compares to find the most effective card counting technique.
- **Section 6: Real-World Practicality** - Assesses the viability of deploying these strategies in casinos, discussing the challenges the player could face.

2. Background

Simulating games allows researchers to explore complex scenarios under controlled and repeatable conditions. In the case of Blackjack, a widely played casino game, simulation offers an efficient and safe way to evaluate strategies over tens of thousands of games. This project aims to simulate and assess the effectiveness of various strategies in Blackjack, particularly involving Basic Strategy and card counting.

2.1. Simulation

Simulation involves modelling real-world processes to study behaviour, test hypotheses and predict outcomes. In this project, simulation enables large-scale testing of Blackjack without the cost financially or the variability found in live play. This allows for the strengths and weaknesses of particular strategies to emerge across a range of conditions, as the simulation can alter multiple independent variables and test their outcomes. (Morris et al., 2019)

2.2. Blackjack Simulation

Apart from its appeal as a casino game, Blackjack has gained the attention from mathematicians and professional gamblers. The combination of chance and player decisions makes it an important subject for studying probability, risk and strategies. Blackjack offers the opportunity for skilled play to influence the outcome of each game, thus making the studies and optimising of the game a natural focus.

One of the most influential contributions to Blackjack analysis is Edward O Thorp's Beat the Dealer (Thorp, 1962), which applied probability theory and early computer simulations to demonstrate that the game could be 'beaten' using optimal strategies. Thorp's work introduced the first computationally tested version of Basic Strategy and card counting systems.

This built on earlier work by Baldwin et al. (1956), who manually calculated a statistically optimal strategy, proving that the house edge could be reduced to approximately 0.10% (Thorp, 1969). Thorp expanded this research using simulations, showing that combining optimal strategies with card counting could give the player a long-term advantage.

2.3. Basic Strategy and Card Counting

Basic Strategy charts show the best action a player should take for every combination of their hand and the dealer's visible card. This process was initially done using combinatorial analysis and refined later through computer simulations. The charts aim to minimise the house edge when followed perfectly.

HARD TOTALS										
	DEALER UPCARD									
	2	3	4	5	6	7	8	9	10	A
17	S	S	S	S	S	S	S	S	S	S
16	S	S	S	S	S	H	H	H	H	H
15	S	S	S	S	S	H	H	H	H	H
14	S	S	S	S	S	H	H	H	H	H
13	S	S	S	S	S	H	H	H	H	H
12	H	H	S	S	S	H	H	H	H	H
11	D	D	D	D	D	D	D	D	D	D
10	D	D	D	D	D	D	D	D	H	H
9	H	D	D	D	D	H	H	H	H	H
8	H	H	H	H	H	H	H	H	H	H

KEY		
	H	Hit
	S	Stand
	D	Double if allowed, otherwise hit

Figure 1: Example of a Basic Strategy chart for Blackjack showing when to hit or stand on hard totals (see full chart Appendix 9).

For example, if the dealer show a 7, the player should hit until they reach 17 or above, then stand. (Apprentice, 2000)

Following Basic Strategy, card counting techniques were developed to overcome the house edge overtime. Card counting is a technique which tracks the amount of high and low cards that have been seen and informs betting amounts and playing decisions based on this (Thorp and Walden, 1973). There is an advantage when the remaining deck is rich in high-value cards. Card counting's mathematical foundations are well-established (Thorp, 1962) , however there is limited comparative research on how different strategies perform in practice, particularly under varied circumstances.

This project uses the foundational work of Thorp and others and simulates large volumes of Blackjack using various gameplay and card counting methods. The goal is to effectively compare the performance of these strategies and explore whether these systems maintain their theoretical advantages under real-world conditions.

3. Implementing Blackjack

This section outlines the technologies, algorithms and simulation plan of this project. A baseline strategy was tested to confirm the validity of the implementation. The aim of the simulation is to provide significant statistics into the effectiveness of each strategy across a large number of games.

3.1. Technologies

The simulation was developed entirely in Python. Although Python is not the fastest programming language in terms of execution time, it excels in readability, development speed, and ease of use with advantages for a project that prioritises clear logic and iteration over performance. Languages such as C or C++ offer increased performance but come with a steeper learning curve and increased development complexity. Python's clean syntax and large community made it the most practical choice for building a simulation framework within a reasonable timeframe.

This project also benefited from Python's vast library support, which aided with the development of statistical analysis and visual components. The following libraries were used:

- Matplotlib - Comprehensive library used for creating visualisations within Python.
- numPy - Provides mathematical functions in Python
- Pandas - Library that provides access to DataFrames within Python to reshape data.
- Seaborn - Another statistical visualisation library

3.2. Game Logic

This section outlines the core mechanics and foundation of the Blackjack simulation used throughout this project. The game logic follows standard casino rules for Blackjack and provides a controlled environment for testing various strategies.

3.2.1. Overview of Blackjack Mechanics

The Blackjack simulation was built to accurately reflect the standard rules and structure of the casino game. These rules define the mechanics for card values, turns within the game, dealer behaviour and win conditions. The implementation focused on replicating this flow of gameplay in code - ensuring that at every decision point, the player's choice is realistic. The simulation logic was designed to be modular, allowing any strategy to be applied dynamically to the decision-making process of the player. Each hand proceeds through a sequence: selecting a strategy, dealing cards, applying the selected strategy, executing dealer actions, and resolving the outcome. This core structure forms the foundation for all strategy comparisons and experimental analysis carried out in this project.

3.2.2. Simulation Structure

Cards within the code were represented as three element lists containing the rank, the connecting word 'of' and the suit. For example, the ace of spades was represented as ['A', 'of', 'Spades']. This allowed for clear identification of the decks properties while maintaining a readable format.

Decks were created using a function that generated all the combinations of card's ranks and suits. The function defined arrays containing the suits ('Heart', 'Diamond', 'Spade', Club') and ranks ('A', 2-10, 'J', 'Q', 'K'). A standard 52-card deck was then constructed and used within the program. Casinos deck count can range from one to eight decks, a higher deck count is more common as it grants them an advantage. For games requiring multiple decks, this process was repeated within another function to return the requested amount of decks. After the decks were created, they were shuffled using Python's built-in random.shuffle() to ensure a properly randomised card order before the game began.

In real-life card games, cards are dealt by taking the top card from the deck and removing it from play until the next shuffle. This behaviour is replicated in the program by using a stack to pop cards from the top of the deck as they are dealt and the deck is reconstructed whenever a shuffle occurs.

Each card in Blackjack has a specific value: number cards are worth their face value, face cards are worth 10 and aces are worth either 1 or 11 depending on the total value of the hand. The total hand value is calculated as the sum of all individual card values.

3.2.3. Game Flow

The game flow was implemented to strictly follow standard Blackjack rules. Before each round, every player must place a bet. In the simulation, a flat bet was used across all simulations (except card counting) meaning if it were real life, the player would be betting the minimum bet each time. This allows for efficient and isolated comparison between strategies and minimises the variance between results, making comparisons statistically fair. The player is also assigned a pot, each loss is taken out of the pot and each win is added to it. For example, the pot starts at 0 units, the player will bet 1 unit and lose the round - leaving the pot at -1 after that round. Each round began with two cards being dealt to all players and the dealer, with only the dealer's first card (the up card) visible.

After receiving the initial cards, players could choose to hit (receive another card), stand (end their turn), double down (double their bet and receive one more card then stand), or split (separate two matching cards into two hands with the first bet on each, also doubling the bet). Splitting is limited to once per hand.

Once all players have completed their turns, the dealer will reveal their hidden card and have to follow one rule: hit until reaching a hand value of 17 or higher, then stand.

3.2.4. Hand evaluation

As stated in the simulation structure section, aces can be valued as either 1 or 11 depending on the total value of the hand. The program evaluates if treating an ace as 11 would cause the hand to exceed a total of 21. If it would, the ace is valued as 1, otherwise the ace is valued as 11 and the hand is classed as 'soft'. In Blackjack a soft hand is one that contains an ace that could be counted as 1 or 11 without busting, meaning the hand has two potential values. This implementation can be seen in Appendix 11.

During the dealing process, the program checked for two important states:

1. **Blackjack** - This is achieved when the player's first two cards consist of an ace and a 10-value card, creating a total of 21. This condition is evaluated after the first two cards are dealt.

2. **Bust** - This occurs when the total value of a hand exceeds 21. The program checked for this state after each card is dealt to a hand, ensuring that busts are identified as they happen.

3.2.5. Payout rules and Gameplay Simplifications

To ensure the simulation is focused on the task and efficient, certain aspects of standard Blackjack have been simplified. This section outlines the payout rules used in the program and the gameplay assumptions made.

Players receive a payout of 3:2 for natural Blackjack, meaning the player receives 1.5 times their original bet in winnings. For a standard win the player will receive a payout of 1:1 and a payout of 0:1 for a loss. Finally, a push (tie with the dealer) results in the player receiving their total bet.

To reduce complexity, the simulation excluded certain player actions, such as surrendering, side bets and surrendering. The dealer was programmed to stand on all 17s including soft 17.

3.2.6. Strategy System

The simulation was designed to evaluate different strategies in Blackjack. Each strategy was implemented as a separate module that determined the player's action. Strategies are modular and interchangeable, allowing for direct comparisons over a large number of simulated games.

3.3. Baseline Strategy

To validate the implementation and provide a baseline for comparison, a simple strategy was first tested. The chosen strategy was Mimic the Dealer. As the name suggests, the player will mimic the dealer actions: hitting on 16 or less and standing on 17 or more.

This approach served two purposes. First, the result of this simulation helped confirm that the game mechanics function as expected over a large simulation. Secondly, as it is poor strategy, it provides a comparison against more advanced strategies.

Pseudocode for Mimic the Dealer:

```
if hand total <= 16:
    hit
else:
    stand
```

3.3.1. Simulation Setup:

Each strategy was evaluated under identical conditions to ensure a fair comparison. Specifically:

- Each strategy was tested over 10,000 simulated games, each consisting of 100 hands. 100 hands was chosen as a player would average this many in a single session.

- The same set of rules (e.g. 1 deck, dealer stands on soft 17, no surrender) was applied across all simulations, unless stated.
- After each hand, the player's pot was updated based on the outcome. The pot resets at the start of each game.
- Final results include average profit per hand and outcomes across all games.

Results:

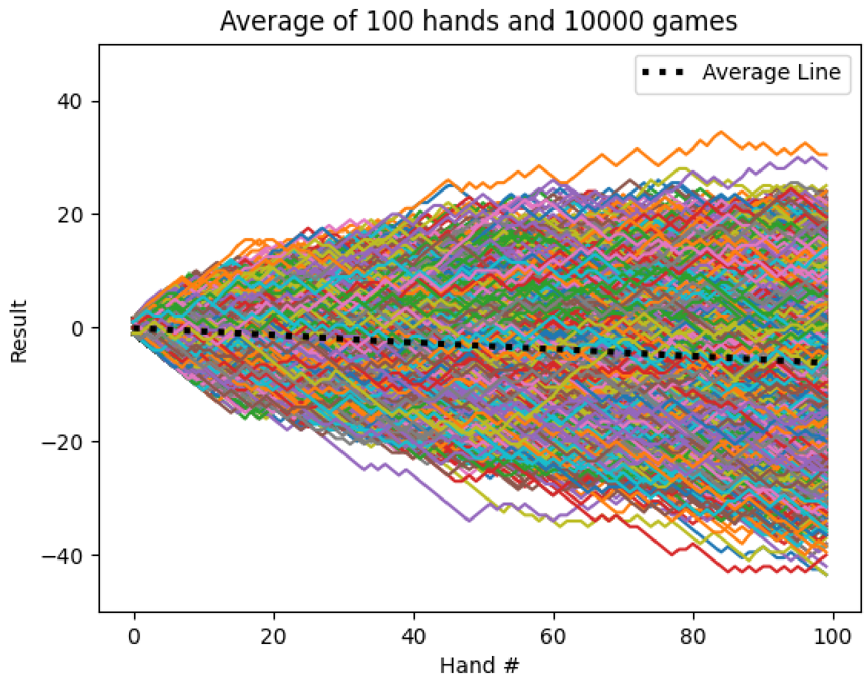

Figure 2: Graph showing the pot from each of the 10,000 games using the Mimic the Dealer strategy and highlighting the average profit.

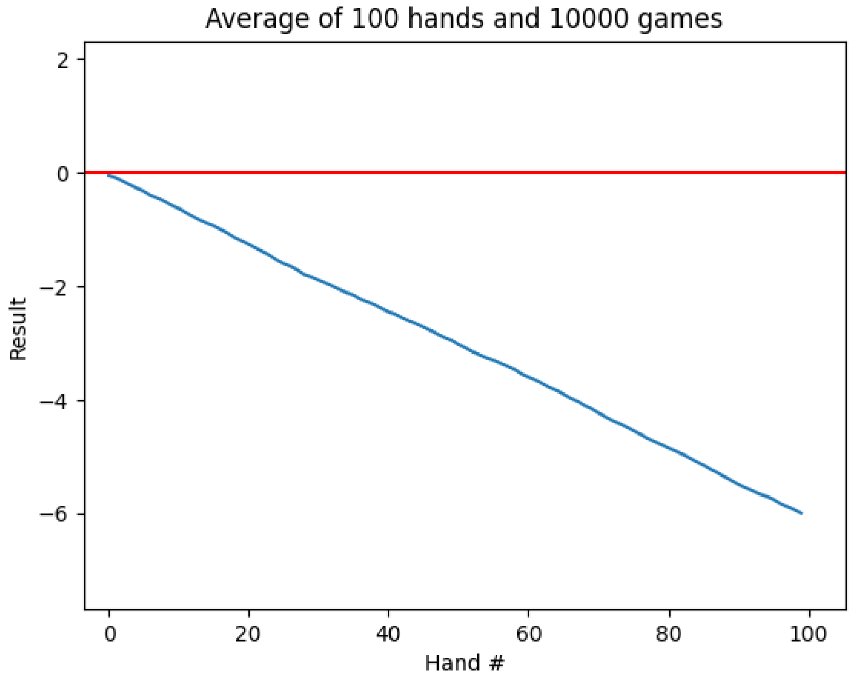

Figure 3: Graph showing the average return per hand across all 10,000 games.

The results reflect the house edge and highlights the need for more advanced strategies to shift the edge to the player. Specifically, the average profit per hand was approximately -0.062, meaning the player lost around 6.2% of their bet per hand on average per hand. These results are consistent with expectations for Mimic the Dealer strategy, which offers no strategic advantage to the player.

4. Game play Strategies

Following the baseline assessment, additional gameplay strategies were implemented to evaluate their effectiveness within the same simulation environment. These strategies incorporate various levels of complexity and aim to improve the player outcomes. Each strategy was compared using relevant metrics such as win rate, loss rate and average profit per hand. The goal of this comparison is to identify the most effective strategy, to be used in combination with card counting techniques in later simulations.

4.1. Overview of Strategies

The following strategies were implemented:

- **Never Bust** - The player stands on any hand value of 12 or more. Designed to avoid busting at all costs but often results in weak hand values that lose to the dealer.
- **Basic Strategy** - A statistically optimal approach derived from probabilities of winning in different scenarios as described in the Background section. The strategy considers both the player's total and the dealer's up card then provides an action accordingly (see Appendix 9). Within the code, a dictionary was used to store all the combinations of Basic Strategy and to efficiently look up actions, see Appendix 10. Basic Strategy uses more advanced actions compared to the other strategies as it considers splitting and doubling down. To consider this while testing, other versions of Basic Strategy were created, these include:
  - **Basic Basic Strategy** - This strategy uses Basic Strategy logic where it compares the player's and dealer's cards but only uses basic actions through hitting and standing. This allows for a better comparison between the weaker game play strategies and against more advanced Basic Strategy.
  - **Basic Strategy (No Splitting)** - This version allows for doubling down to be chosen as an action, but not splitting. Having versions that show the impact of implementing actions let the simulation display the benefits and difference between them.
  - **Basic Strategy (No Doubling)** - This version is identical for reasoning as the one above but does not allow for doubling down and lets splitting be a game play option.
  - **Full Basic Strategy** - This is the final version of Basic Strategy and contains all possible actions. This version is expected to have the highest potential.

Each strategy was simulated under the exact same conditions as previously stated.

4.2. Simulation Comparison

Figure 4 provides a visual comparison of the average profit per hand across 10,000 games for each strategy tested, offering a comparison between their relative performance. This graph highlights which strategies performed well in terms of profitability. In addition to the graph, Table 4.2 presents the detailed metrics of each strategy, including win rate, push rate and loss rate, sorted in ascending order by **average profit per hand**.

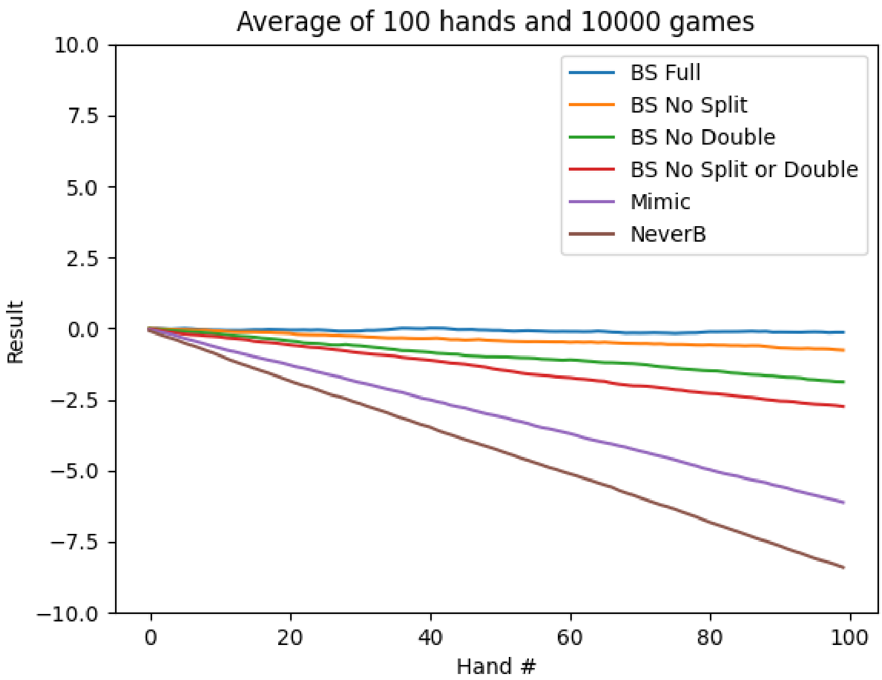

Figure 4: Combined graph of average profit per hand across 10,000 games for all tested strategies.

Strategy	**Avg. pft p hand**	**Win rate (%)**	**Push rate (%)**	**Loss rate (%)**
Never Bust	-8.18%	41.91	6.20	**51.91**
Mimic the Dealer	-6.21%	41.15	**9.62**	49.22
Basic Strategy (No Double/Split)	-2.66%	43.44	8.60	47.96
Basic Strategy (No Double)	-1.99%	**43.63**	8.73	47.64
Basic Strategy (No Split)	-1.05%	43.43	8.42	48.15
Basic Strategy (Full)	**-0.22%**	43.45	8.41	48.00

4.2.1. Discussion of Results

As seen in Figure 4 and Table 4.2, the strategies yield different outcomes over time:

- **Mimic the Dealer** serves as the baseline, showing a consistent downward slope due to offering no strategic advantage. Average loss per hand is **-6.21%**. As this strategy mimics the dealer, there is more chance they finish the hand with the same hands, which is shown with the highest push rate of **9.62%**.

- **Never Bust** performs the worst overall, highlighted by the lowest average profit of **-8.18%**. While it avoids busting, it will frequently lose to the dealer's stronger hands, shown by its high loss rate of **51.91%**.

- **Basic Strategy (No Split or Double)** performed significantly better than the previous two, with an average profit per hand of **-2.66%**. The lack of any advanced actions limited its effectiveness and profitability.

- **Basic Strategy (No Double)** shows a more gentle decline, with an average profit per hand of **-1.99%**. The ability to split helped to gain some advantage but the lack of doubling still limited the strategy.

- **Basic Strategy (No Split)** performs better than the previous Basic Strategy, with an average profit per hand of **1.05%**. Doubling allows the player to take advantage of certain hands, but is still underperforming.

- **Full Basic Strategy** is the best performing strategy, with an average profit per hand of **-0.22%**. It maintains a near flat line near break even, showcasing how effective the full strategy is at reducing the house edge and maximising the player potential over the long run.

4.3. Summary

The goal of this analysis was to assess how different Blackjack strategies perform over a large number of hands and to identify what strategy is the most effective in reducing the house edge.

Full Basic Strategy produced the best overall results, achieving the highest average profit per hand and one of the most balanced distributions of win, loss and push rates. This performance closely aligns with established research, which demonstrates that when both doubling and splitting are permitted, the player can reduce the house edge significantly. Compared to the simpler strategies and restricted versions, the full implementation consistently outperformed across all key metrics. However, whilst Full Basic Strategy performs the best, the player still loses on average 0.22% per hand - emphasising the need for further strategies, which is why this strategy was used in combination with card counting.

4.4. Comparison with Existing Results

The Full Basic Strategy simulation in this project yielded an average player loss of 0.22% per hand. This compares favourably with Thorp's published house edge ranging from -1% to +1% under similar rule conditions Thorp (1962). The difference in value may be due to variations in simulation structure or the way metrics were calculated.

Furthermore, the higher losses observed with simpler strategies, such as Mimic the Dealer (-6.21%) and Never Bust (-8.18%), align with the established literature and validate the implementation and accuracy of the simulation system used in this project. These results complete and validate the the first four objectives in this project.

5. Card Counting Strategies

This section explains the principles of card counting, outlines the specific strategies implemented within the simulation and presents the expected outcomes based on statistical analysis.

5.1. Basics of Card Counting

Card counting is a technique used in Blackjack to assess whether the player or dealer has an advantage in the next hand. It involves tracking the ratio of high to low cards remaining in the deck. Theoretically, the player could gain a higher edge if they were to remember every card that has been dealt, but is impractical. Thus, simplified strategies such as card counting tables are used. To do this, the player maintains a **running count**, which is a total based on the values of cards that have already been dealt. For each card seen, a specific value is added or subtracted from the running count. A common card counting strategy that was used in the simulation is the **Hi-Lo** counting strategy. In this method, the player:

- Adds 1 for each low card (2 through 6) seen.
- Subtracts 1 for each high card (10, Jack Queen, King and Ace).
- Assigns 0 to neutral cards (7, 8, 9).

As more low-valued cards are dealt, the running count will increase, indicating to the counter that the deck is abundant in high-value cards. This benefits the player, as it increases the chances of achieving a Blackjack, receiving strong hands and having the dealer bust more often. A positive count suggests that the player has an edge, while a negative count will favour the house.

In games that use multiple decks, using the running count will be misleading as it does not account of how many cards are remaining. In a single deck game, a +5 count indicates a strong advantage to the player as it suggests that the deck is rich in high valued cards. However, in a six deck game, this +5 count is extended across all the remaining decks and has much less significance to the player.

To adjust for this, the **true count** is used. The true count is calculated by dividing the running count by the estimated number of decks remaining in the shoe.

$\text{True Count} = \frac{\text{Running Count}}{\text{Estimated Decks Remaining}}$

This adjustment provides an accurate measure of the player's advantage. The simulation takes this into account by tracking the number of cards dealt and calculating the number of decks left.

To take advantage of the count, the program implemented several **betting strategies** that adjust the player's bet:

- **Flat betting** - The player bet the same amount each hand, regardless of the count. This served as a baseline.
- **Dynamic betting** - The bet increases with the count in different brackets. Multiple strategies were created, each varying in aggressiveness, and will be discussed later.

These implementations enabled structured experiments to compare flat against dynamic betting strategies across different card counting methods.

5.2. Implemented Card Counting Strategies

In the simulation, three card counting strategies were implemented with each being more complex than the last. This aligns with the fifth objective of the project, which is to identify the most effective card counting strategy through simulation and statistical analysis. Each strategy was implemented and evaluated under consistent conditions to ensure accuracy in determining performance. These card counting strategies include:

- **Hi-Lo** - The standard strategy described above, widely used due to its simplicity and accuracy.
  - *Pros:* Easy to learn and apply; strong performance with minimal effort.
  - *Cons:* Lower precision compared to multi-level systems

Table 1: Blackjack Card Counting Values (Hi-Lo System)

2	3	4	5	6	7	8	9	10	J	Q	K	A
+1	+1	+1	+1	+1	0	0	0	−1	−1	−1	−1	−1

  For example, if the player sees a 3, 4 and 6, the count will be +3. This high/positive count indicates that more low cards have been played, meaning there are more high cards remaining in the deck. This indicates an advantage to the player, as they are more likely to draw strong hands and the dealer is more likely to bust.

- **Omega II** - A multi-level balanced card counting strategy that is designed for higher accuracy compared to simpler systems. The Omega II system uses a wider range of values from -2 to +2 to assign to cards.
  - *Pros:* Improved accuracy over simpler systems; higher edge when used correctly.
  - *Cons:* Requires more effect to learn and maintain count; less accessible to to beginners.

Table 2: Blackjack Card Counting Values (Omega II System)

2	3	4	5	6	7	8	9	10	J	Q	K	A
+1	+1	+2	+2	+2	+1	0	−1	−2	−2	−2	−2	−2

- **Wong Halves** - An advanced strategy which uses fractions within the count and is known as one of the strongest card counting methods.
  - *Pros:* Maximises precision and profit.
  - *Cons:* Very difficult to learn and put into practise.

Table 3: Blackjack Card Counting Values (Wong Halves System)

2	3	4	5	6	7	8	9	10	J	Q	K	A
+0.5	+1	+1	+1.5	+1	+0.5	0	-0.5	-1	-1	-1	-1	-1

The performance of each of these strategies were assessed using a controlled experimental simulation, as described in a following section.

5.3. Dynamic Betting

Dynamic betting is applied using the true count, as a higher true count indicates a greater player advantage. The betting amount is then increase proportionally to capitalise on this edge.

Each dynamic betting strategy was defined by a range, such as 1–8 units, where the minimum bet is placed when the true count is neutral or negative and higher bets are placed as the true count increases. More aggressive spreads like 1-16 increase both variance and maximum bet.

The following betting spreads were implemented:

- **1-8**: Most conservative approach, aimed at consistency and a baseline compared to the other spreads.

Table 4: Dynamic Betting (1-8)

True Count	-2	-1	0	1	2	3	4	5
Bet Amount	1	1	1	2	4	6	8	8

- **1-12**: Balanced spread as it is not conservative and not aggressive.

Table 5: Dynamic Betting (1-12)

True Count	-2	-1	0	1	2	3	4	5
Bet Amount	1	1	1	3	6	9	12	12

- **1-16**: Bets increase faster with the count, maximising profits but creating more risk and variance.

Table 6: Dynamic Betting (1-16)

True Count	-2	-1	0	1	2	3	4	5
Bet Amount	1	1	1	4	8	12	16	16

5.4. Results and Comparison

The effectiveness of each card counting technique was simulated over 10,000 games, played through Full Basic Strategy and bets were scaled dynamically based on the true count, the most conservative betting threshold was used. Additionally, this was simulated with single deck Blackjack. Some metrics like win rate will not be shown as they will be the same as Basic Strategy metrics.

5.4.1. Performance Overview

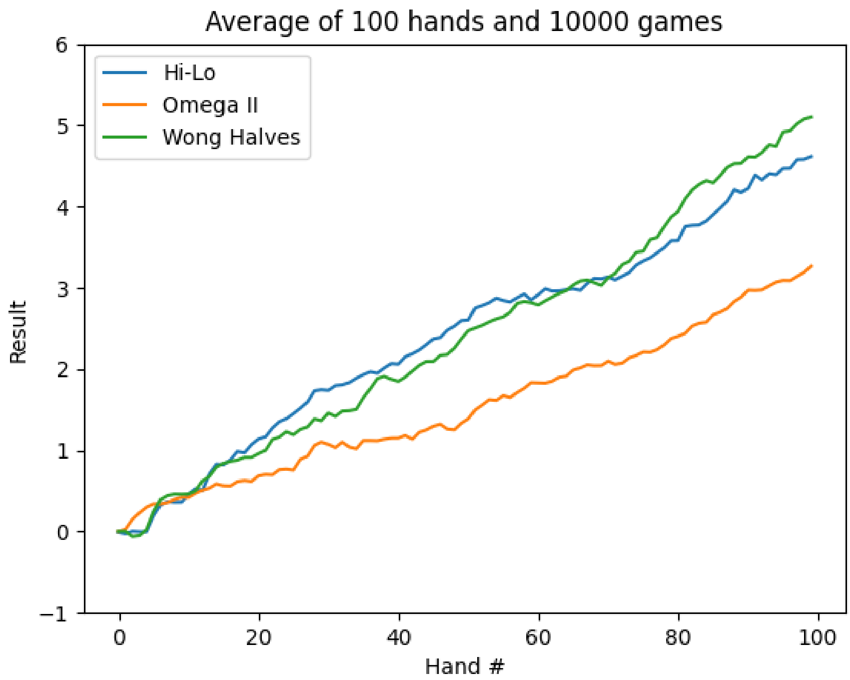

Figure 5: Graph presenting the average return per hand across all tested card counting methods.

- **Hi-Lo**: Provided the most consistent upward trend, highlighting its balanced nature. However, this method lagged behind the other two in average profit.
- **Omega II**: Performed the best in the early hands (0-50) but was overtaken around hand 70.
- **Wong Halves**: Surpassed the other two simpler methods and ended with the highest average profit due to its accuracy and advanced nature.

All of the card counting strategies show fairly smooth upward trends, in profit. This confirms the expected results that card counting allows the player to gain an edge, over a long time, against the casino under certain rules.

5.4.2. Table of Results

Table 7 shows the average profit per 100 hands of 10,000 games for each combination, using a single deck and the most conservative betting strategy:

Strategy	Avg. Profit per Hand
Hi-Lo	4.62%
Omega II	3.27%
Wong Halves	5.10%

Table 7: Average profit per hand and standard deviation across card counting strategies.

These results show that all card counting methods improved profitability over Basic Strategy alone and all became positive. More specifically, Hi-Lo performed consistently well whilst being the most simple strategy. Omega II, a more complex system, underperformed with its average profit and produced the worst results. Wong Halves, while also an advanced method, outperformed the other strategies as expected.

6. Real-life Practicality

While card counting has been shown to yield a statistical edge in simplified simulations, real-world conditions are significantly different. Specifically, the number of decks, betting systems, casino rules and human error all affect the effectiveness of strategies within Blackjack.

6.1. Increased Number of Decks

Most casinos use 6 to 8 decks rather than a single deck. This has two major effects:

- **Count Disruption:** With more decks in play, fluctuations in the true count will only appear near the shuffle point of the shoe (all the decks shuffled together), creating less advantageous positions for the player.
- **Reduced Edge:** Within a game, a high running count will be common. However, the running count is misleading in multiple decks and the true count will be used, as stated in 5.1.

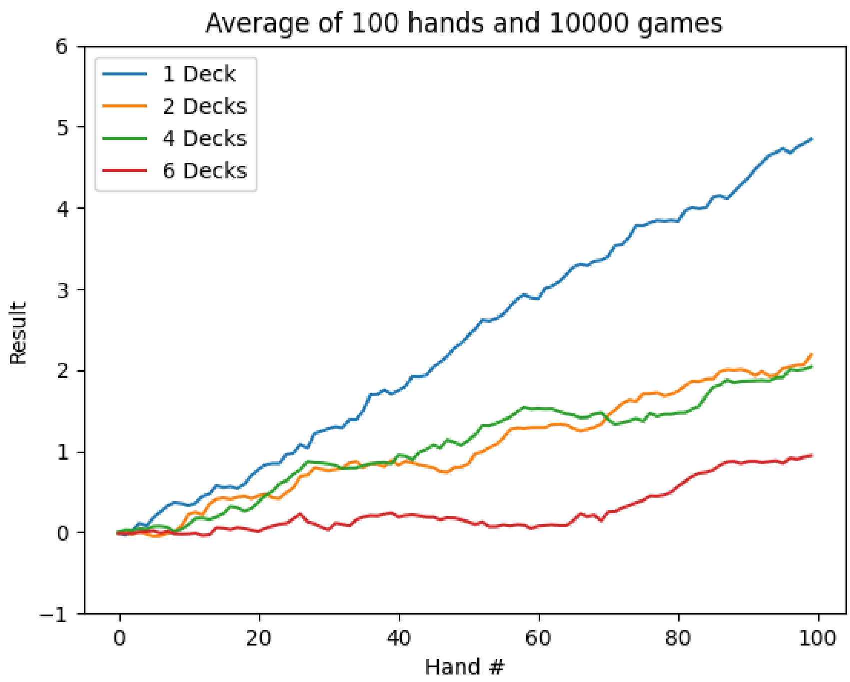

Figure 6: Graph that shows the average profit of the Hi-Lo card counting strategy with multiple decks.

Deck amount	**Avg. Profit per Hand**
1 Deck	4.85%
2 Decks	2.18%
4 Decks	2.04%
6 Decks	0.94%

Table 8: Average profit per hand and standard deviation across card counting strategies.

Figure 6 and Table 8 illustrates how the average profit per hand using the Hi-Lo card counting system decreases as the number of decks increases. With a single deck, the average profit is the highest, but the advantage declines progressively with two, four and six decks. This trend reflects a known limitation of card counting in real-life settings, where multiple decks are used specifically to reduce the effectiveness. As more decks are added, the accuracy of the true count decreases and the impact of each removed card is minimum, due to there being more cards in the deck, and resulting in less advantageous positions. Because of this, there are less high bets when the decks increase and decreases the variance, as shown by 6 decks in Figure 6

6.2. Betting Systems in Practice

Betting systems through flat and dynamic betting will behave differently under multiple decks.

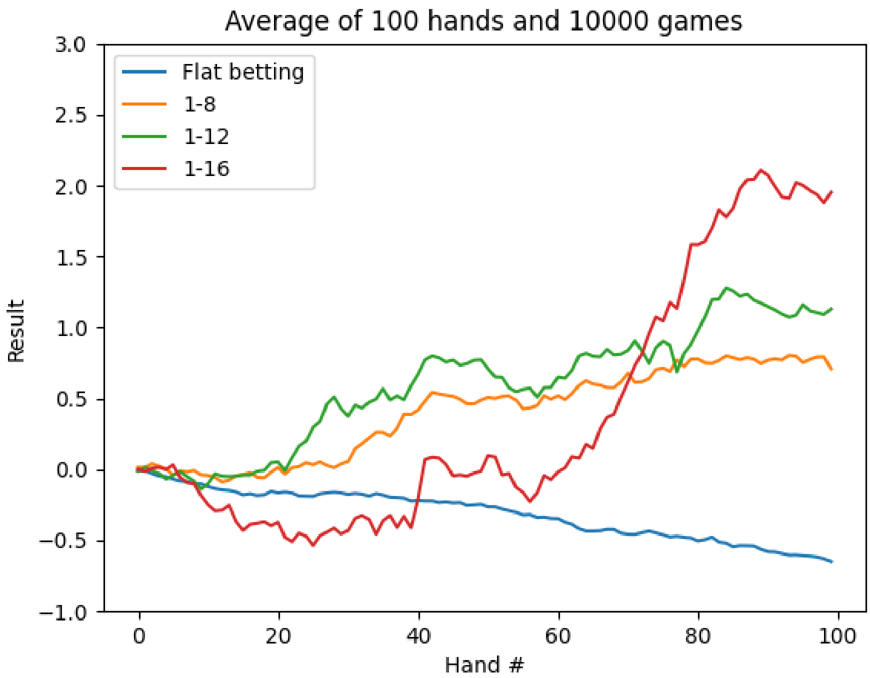

Figure 7: Graph illustrating the average profit of Hi-Lo using different betting systems.

Figure 7 displays how various betting spreads affect the average profit per hand when using the Hi-Lo card counting system across 10,000 games of 100 hands with six decks. Flat betting (betting 1 unit each hand regardless of the count) produced a consistent decline in profit with low variance, further reflecting the house's edge when the player is not using card counting systems.

As the betting spread increased, both the average profit and variance also increased. The 1-8 and 1-12 spreads showed positive average profit with moderate fluctuations.

However, the 1-16 spread, while initially returning negative results, achieved the highest average profit of approximately 2.1% per hand. This shows that Blackjack can be profitable for the player through the casino's countermeasures such as an increase in decks. However, effective game play, card counting, betting and mental strategy must be in place for this to occur, as the simulation shows how the perfect player will perform under simplified circumstances.

Assuming that around 7-8 cards are being played per hand and with a 75% penetration, the shoe will be shuffled around every 35 hands. This information directly correlates with the graph as sharp changes in profit tend to occur around this interval. This is where the true count becomes more accurate due to a greater proportion of cards having been played. This increased accuracy can cause large bets just before the deck is shuffled, resulting in more volatile and profitable outcomes.

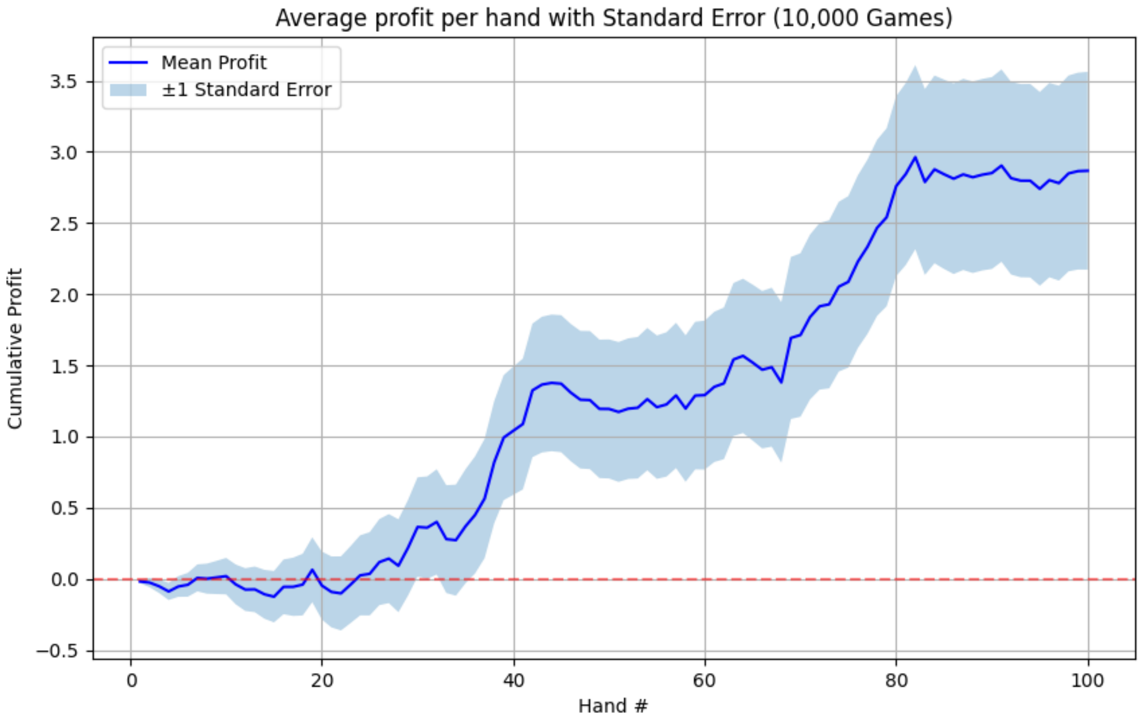

Figure 8: Graph showing the average profit of Hi-Lo using the 1-16 dynamic betting method and highlighting the standard error around the mean.

Standard error is a statistic that measures how much the mean varies across different samples. Figure 8 focuses on the 1-16 betting method and includes standard error to illustrate the variance in outcomes. In this graph, the blue line represents the mean profit, while the shaded region represents ±1 standard deviation. The graph illustrates a clear upward trend in profit after around the 35th hand, further supporting the shuffling point correlating with the count, whilst outlining the potential changes in profit due to higher bet size.

6.3. Casino Rules

While card counting systems can provide a statistical edge over the house, their real-world effectiveness is also influenced by casino rules. These rules were designed to protect the house edge and prevent potential gains from advantage play.

One of the significant factors that has not been discussed, is the penetration - the percentage of the shoe dealt before reshuffling. Higher penetration (75-90%) allows for more accurate counts and advantage based on that count, while low penetration (50-60%) disrupts this and reduces the counter's overall edge. Higher penetration is less common as card counting is well-known and the casino will try to keep their edge.

Additional rules also alter the house edge, such as:

- Dealer hitting or standing on soft 17 - In this simulation the dealer will stand on soft 17, but if they hit instead their edge can increase by around 0.2%.
- Re-splitting rules - Some casinos allow for multiple splitting in one hand, which alters the player edge.
- Payout ratios - Achieving a hand of Blackjack typically pays at a 3:2 ratio but some casinos will pay at 6:5 or 1:1 depending on their rules.

Majority of casinos object card counting and have measures against if caught. One measure is a forced shuffle in the middle of a game, regardless of the penetration. This disrupts the current count and removes any advantage for the player. Another method is removing/banning the card counter. This could be through physically banning them or not letting them request any of their winnings.

6.4. The Gambler

The most efficient strategies can fail when human intuition comes into play. While systems like basic strategy and card counting are mathematically optimal, gamblers deviate from these methods due to emotional, psychological or situational factors. While the simulation showed consistent profitable results, the skill of Basic Strategy and card counting can take years to master. This is due to the complex nature of these strategies, as remembering every possible Basic Strategy combination, card value and true count whilst in a casino environment is a challenge for the player. Additionally, Bond (1974) found that gamblers followed the recommended decisions 75% of the time. This information suggests that gamblers are not be motivated by the long-term probabilities as they are difficult to grasp whilst losing in the short-term.

Short-term outcomes, such as winning or losing streaks, have a stronger psychological affect, potentially causing a deviation from their strategies. Additionally, gamblers rarely make decisions under ideal conditions. Factors such as playing late at night, the casino environment and fluctuations in personal bankroll can all weaken judgment. Phillips and Amrhein (1989) observed that in Blackjack, the closer a player's total was to 21, the more time they needed to make their next decision. Under time pressure, players were also more conservative in their choices, indicating a shift when approaching higher risk situations.

7. Conclusion and Future Work

7.1. Summary

This project investigated and evaluated the casino game Blackjack, specifically common game play methods, card counting systems and betting spreads, to understand their validity and impact in a simulated environment. The Blackjack simulation was developed to reflect realistic casino conditions.

Different card counting techniques were implemented and tested across thousands of games to assess their long-term profitability. Various betting spreads were implemented and measured to understand their effect on return and level of risk. Numerous figures and tables were used to visualise the affect of all strategies and evaluate trends and consistency of outcomes.

The results produced from this project supported expectations while also highlighting other important factors such as, casino countermeasures, complexity and information on gamblers in general.

7.2. Conclusion

To conclude, the evaluation on the various game play strategies further supported Thorp's Basic Strategy as the foundation for optimal play by consistently outperforming other strategies in all metrics and providing the highest player edge.

Building on this foundation, the integration of card counting was shown to further reduce the house edge and shift the advantage toward the player. The evaluated card counting methods included Hi-Lo, Omega II and Wong Halves, with Wong Halves slightly performing better than the other methods, in a single deck game. However, Hi-Lo was selected deeper analysis due to its simplicity thus, making it more practical for real-world implementation.

This further investigation demonstrated the affect on return by increasing the decks within a shoe, reflecting typical casino rules. Additionally, the analysis onto different betting spreads illustrated that a player incorporating perfect Basic Strategy, Hi-Lo card counting with true count tracking and an aggressive betting spread has a positive expected profit over the long-term. This completes the objectives of the project by implementing, simulating and evaluating the effectiveness of strategies-specifically card counting-within Blackjack.

7.3. Future Work

If this project was to be further developed, a functional casino environment simulation would be the next step. This would allow for an analysis into factors that this project did not cover such as, multiple players at a table, human error and casino's removing players for card counting. The implementation of these features would produce more realistic results that could be expected in real-world play.

References

Apprentice, B. (2000). Card counting systems.

Baldwin, R. R., Cantey, W. E., Maisel, H., and McDermott, J. P. (1956). The optimum strategy in blackjack. *Journal of the American Statistical Association*, 51:429–439.

Bond, N. A. (1974). Basic strategy and expectation in casino blackjack. *Organizational Behavior and Human Performance*, 12:413–428.

Morris, T. P., White, I. R., and Crowther, M. J. (2019). Using simulation studies to evaluate statistical methods. *Statistics in Medicine*, 38:2074–2102.

Phillips, J. G. and Amrhein, P. C. (1989). Factors influencing wagers in simulated blackjack. *Journal of Gambling Behavior*, 5:99–111.

Thorp and Walden (1973). The fundamental theorem of card counting.

Thorp, E. O. (1962). *Beat the dealer : a winning strategy for the game of twenty one*. Vintage.

Thorp, E. O. (1969). Optimal gambling systems for favorable games. *Revue de l'Institut International de Statistique / Review of the International Statistical Institute*, 37:273.

A. Basic Strategy Chart

HARD TOTALS

	DEALER UPCARD									
	2	3	4	5	6	7	8	9	10	A
17	S	S	S	S	S	S	S	S	S	S
16	S	S	S	S	S	H	H	H	H	H
15	S	S	S	S	S	H	H	H	H	H
14	S	S	S	S	S	H	H	H	H	H
13	S	S	S	S	S	H	H	H	H	H
12	H	H	S	S	S	H	H	H	H	H
11	D	D	D	D	D	D	D	D	D	D
10	D	D	D	D	D	D	D	D	H	H
9	H	D	D	D	D	H	H	H	H	H
8	H	H	H	H	H	H	H	H	H	H

SOFT TOTALS

	DEALER UPCARD									
	2	3	4	5	6	7	8	9	10	A
A,9	S	S	S	S	S	S	S	S	S	S
A,8	S	S	S	S	Ds	S	S	S	S	S
A,7	Ds	Ds	Ds	Ds	Ds	S	S	H	H	H
A,6	H	D	D	D	D	H	H	H	H	H
A,5	H	H	D	D	D	H	H	H	H	H
A,4	H	H	D	D	D	H	H	H	H	H
A,3	H	H	H	D	D	H	H	H	H	H
A,2	H	H	H	D	D	H	H	H	H	H

PAIR SPLITTING

	DEALER UPCARD									
	2	3	4	5	6	7	8	9	10	A
A,A	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
T,T	N	N	N	N	N	N	N	N	N	N
9,9	Y	Y	Y	Y	Y	N	Y	Y	N	N
8,8	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
7,7	Y	Y	Y	Y	Y	Y	N	N	N	N
6,6	Y/N	Y	Y	Y	Y	N	N	N	N	N
5,5	N	N	N	N	N	N	N	N	N	N
4,4	N	N	N	Y/N	Y/N	N	N	N	N	N
3,3	Y/N	Y/N	Y	Y	Y	Y	N	N	N	N
2,2	Y/N	Y/N	Y	Y	Y	Y	N	N	N	N

SURRENDER

	DEALER UPCARD									
	2	3	4	5	6	7	8	9	10	A
16								SUR	SUR	SUR
15									SUR	
14										

INSURANCE OR EVEN MONEY: DON'T TAKE

KEY		
	H	Hit
	S	Stand
	D	Double if allowed, otherwise hit
	Ds	Double if allowed, otherwise stand
	N	Don't split the pair
	Y	Split the Pair
	Y/N	Split only if `DAS` is offered
	SUR	Surrender

Figure 9: Basic Strategy table used in the simulation, source: *https://www.Blackjackapprenticeship.com/Blackjack-strategy-charts/*.

B. Basic Strategy Dictionary in Code

```
hardHands = {
    21: {2: "Stand", 3: "Stand", 4: "Stand", 5: "Stand", 6: "Stand", 7: "Stand", 8: "Stand", 9: "Stand", 10: "Stand", 11: "Stand"},
    20: {2: "Stand", 3: "Stand", 4: "Stand", 5: "Stand", 6: "Stand", 7: "Stand", 8: "Stand", 9: "Stand", 10: "Stand", 11: "Stand"},
    19: {2: "Stand", 3: "Stand", 4: "Stand", 5: "Stand", 6: "Stand", 7: "Stand", 8: "Stand", 9: "Stand", 10: "Stand", 11: "Stand"},
    18: {2: "Stand", 3: "Stand", 4: "Stand", 5: "Stand", 6: "Stand", 7: "Stand", 8: "Stand", 9: "Stand", 10: "Stand", 11: "Stand"},
    17: {2: "Stand", 3: "Stand", 4: "Stand", 5: "Stand", 6: "Stand", 7: "Stand", 8: "Stand", 9: "Stand", 10: "Stand", 11: "Stand"},
    16: {2: "Stand", 3: "Stand", 4: "Stand", 5: "Stand", 6: "Stand", 7: "Hit", 8: "Hit", 9: "Hit", 10: "Hit", 11: "Hit"},
    15: {2: "Stand", 3: "Stand", 4: "Stand", 5: "Stand", 6: "Stand", 7: "Hit", 8: "Hit", 9: "Hit", 10: "Hit", 11: "Hit"},
    14: {2: "Stand", 3: "Stand", 4: "Stand", 5: "Stand", 6: "Stand", 7: "Hit", 8: "Hit", 9: "Hit", 10: "Hit", 11: "Hit"},
    13: {2: "Stand", 3: "Stand", 4: "Stand", 5: "Stand", 6: "Stand", 7: "Hit", 8: "Hit", 9: "Hit", 10: "Hit", 11: "Hit"},
    12: {2: "Hit", 3: "Hit", 4: "Stand", 5: "Stand", 6: "Stand", 7: "Hit", 8: "Hit", 9: "Hit", 10: "Hit", 11: "Hit"},
    11: {2: "Double", 3: "Double", 4: "Double", 5: "Double", 6: "Double", 7: "Double", 8: "Double", 9: "Double", 10: "Double", 11: "Double"},
    10: {2: "Double", 3: "Double", 4: "Double", 5: "Double", 6: "Double", 7: "Double", 8: "Double", 9: "Double", 10: "Hit", 11: "Hit"},
    9: {2: "Hit", 3: "Double", 4: "Double", 5: "Double", 6: "Double", 7: "Hit", 8: "Hit", 9: "Hit", 10: "Hit", 11: "Hit"},
    8: {2: "Hit", 3: "Hit", 4: "Hit", 5: "Hit", 6: "Hit", 7: "Hit", 8: "Hit", 9: "Hit", 10: "Hit", 11: "Hit"},
    7: {2: "Hit", 3: "Hit", 4: "Hit", 5: "Hit", 6: "Hit", 7: "Hit", 8: "Hit", 9: "Hit", 10: "Hit", 11: "Hit"},
    6: {2: "Hit", 3: "Hit", 4: "Hit", 5: "Hit", 6: "Hit", 7: "Hit", 8: "Hit", 9: "Hit", 10: "Hit", 11: "Hit"},
    5: {2: "Hit", 3: "Hit", 4: "Hit", 5: "Hit", 6: "Hit", 7: "Hit", 8: "Hit", 9: "Hit", 10: "Hit", 11: "Hit"},
    4: {2: "Hit", 3: "Hit", 4: "Hit", 5: "Hit", 6: "Hit", 7: "Hit", 8: "Hit", 9: "Hit", 10: "Hit", 11: "Hit"},
    3: {2: "Hit", 3: "Hit", 4: "Hit", 5: "Hit", 6: "Hit", 7: "Hit", 8: "Hit", 9: "Hit", 10: "Hit", 11: "Hit"},
    2: {2: "Hit", 3: "Hit", 4: "Hit", 5: "Hit", 6: "Hit", 7: "Hit", 8: "Hit", 9: "Hit", 10: "Hit", 11: "Hit"}
  }
```

Figure 10: Picture showcasing the Basic Strategy dictionary in code.

C. Hand Evaluation Code

```
def handType(hand):
    total = 0
    ace = 0
    for i in range(len(hand)):
        if hand[i][0] == "J" or hand[i][0] == "Q" or hand[i][0] == "K":
            total += 10
        elif hand[i][0] == "A":
            total += 11
            ace += 1
        else:
            total += int(hand[i][0])

    if total > 21 and ace > 0:
        return "hard"
    elif ace > 0:
        return "soft"
    else:
        return "hard"
```

Figure 11: Picture showing the hand evaluation function in code.