International Baccalaureate Diploma Program

Extended Essay

RQ: What is the best strategy in a one versus one Blackjack game with the use of probability theory and simulation?

Subject: Mathematics

May 2018

Word Count: 3892

Table of Contents

1.1 Introduction 1

1.2 What is Blackjack? 1

1.3 Probability and Blackjack- A Literature Review 2

2.1 Main Body 3

2.2 Game Created to Understand the Concept: Easy 21 3

2.3 Possible outcomes of Easy 21 4

2.4 Optimum Strategies for Easy 21 4

2.5 The so-called best strategy for Blackjack 6

2.6 Assumed Blackjack Rules 7

2.7 Mathematical Checking for the Best Strategy that is Given 9

2.8 Data from program written with the strategy that is given as the best 11

2.9 Explanation of the written code 11

2.10 Data from program written with the strategy that is given as the best without split tactic 13

2.11 Data from program written with the average strategy 14

2.12 Data Analysis 16

2.13 Best value to stop picking cards from graph 17

4.1 Conculusion 18

5. References 20

6. Appendices 21

6.1 Appendix 1 - Rules and Terms of Blackjack 21

6.2 Appendix 2- The program written with split tactic 22

6.3 Appendix 2 - Program written without split tactic 31

6.4 Appendix 2 - Program written with the average value 32

1.1 Introduction

Blackjack is a card game, which is played in casinos. There are so many books and films about Blackjack and how to play it properly. In novels and films such as "21" and "Rainman", mathematics behind Blackjack is used to gain advantage. There are also many casino players that put out their strategy to win money from Blackjack.

This essay focuses on to find the best strategy to use in a Blackjack game under certain circumstances. There are some known tactics that are used by professional Blackjack players but even all of those tactics use probabilistic reasoning, there are still some differences within tactics because of the changing casino rules. In this essay I try to explain what would be the best tactic in Blackjack under certain circumstances. First, rules of the casino will be set, then the most known tactic that famous gamblers proposed in the internet which many people rely on while they play Blackjack will be tested under the rules that is created, and after testing the known strategy, a better strategy will be offered. This research question is worthy of investigation because size of online gambling by itself was worth of 37 billion dollars in 2015 (Size of the online gambling, 2017). On top of that this essay has potential to change the so-called best strategy which many gamblers rely on. The research question is to find what is the best strategy in a one versus one Blackjack game with the use of probability theory and simulation? In order to answer this question, firstly, Blackjack will be introduced and then the studies done on finding the optimal strategy in Blackjack will be shared.

1.2 What is Blackjack?

Blackjack, also known as twenty-one, is a popular game that is played in casinos. In a Blackjack table there could be up to 8 players, but Blackjack is a card game that is between the dealer and player/s, which means other players loses or wins doesn't affect the rest. It can be played with one or more deck of cards, in casinos they usually use more than 6 decks and even in some casinos they get rid of some random cards to eliminate people counting the cards. There are two ways to win at Blackjack; one is having higher points than dealer without passing 21 or letting the dealer draw cards until he passes 21. So a general objective would be get as much close to 21 without exceeding 21. Detailed explanations of rules and terms is given in Appendix 1.

1.3 Probability and Blackjack- A Literature Review

When one searches for games and mathematics, one of the methodologies that are used to create or test strategies is game theory. However, game theory is not only for games, it can be more generally defined as Myerson (1997) did with the words "the study of mathematical models conflict and cooperation between intelligent rational decision-makers." (p.1). Every game has different rules and different play styles so games are grouped in different ways, such as: Number of players, simultaneous game (all players play their turn at the same time and they don't know what other players did), sequential games (some players make their decisions after seeing other players' decisions), zero-sum game (in order one person to win one person needs to lose), nonzero-sum game (players don't benefit from other players' wins or loses). Blackjack is also a zero-sum game where either the dealer or the player wins.

Now the question is, what is the best strategy for Blackjack? One of the earlier studies belongs to Baldwin et al. (1956), in which the authors tried to find solutions to the basic problems of strategy like when to hit or stand, when to double down and when to split without the idea that one can count the cards opened in the game and change the strategy. However, Thorp (1966), in his famous book "Beat the Dealer" had moved the study of Baldwin et al. beyond and make and utilized computer programs to better understand the optimal strategy when the player was assumed to count the cards, which are opened up to the decision and form tactics accordingly. In order to do this, player's expectations were calculated with all possible subsets of a deck. Since Thorp's strategies make the players advantageous over the house, the casinos took precautions like using more than one deck in a play. Moreover, the casino staff started to detect card counters and ban these people entering the casino. Though simulation gained much more importance in terms of determining better strategies for Blackjack.

Simulations and statistical analysis related with Blackjack have increased following the Thorp's studies. Markov Chain Analysis and Monte Carlo simulations were a few of the methodologies to approach this popular game. Markov Chain analysis is used for non deterministic events when the probability distribution of each phase of the event depends on only the value of one step prior to that phase and the nature of the game Blackjack could be modelled as a Markov Chain (Coltin, 2012). Monte Carlo Simulations utilizes random sampling and conducts a large amount of experiments and it could be used in determining Blackjack strategies (Vaidyanathan, 2014). In this essay a simulation that I wrote will be used based on probability theory, and this methodology could be regarded as it is close to Monte Carlo Simulations.

Besides these academic works done on the strategies, many people who play Blackjack gave decisions according to famous tactic websites. In this essay, one of the most famous strategy shared in a betting website (Blackjack Strategy, 2017) will be questioned. Before trying to find a better strategy than the one given on this website, a better understanding of the game is needed.

2.1 Main Body

To have a better understanding of the game, a mini game will be created, which will use the same fundamental idea as Blackjack before calculating the probability of Blackjack and generating millions of games with different strategies. That game will be called "Easy 21".

2.2 Game Created to Understand the Concept: Easy 21

Easy 21 will be played with six cards, three of them being fives and the other three of them being twos. There will be two players just like black jack one dealer and one player, and everyone begins with two cards. Dealer and player can see each of their hands so the game is played with cards opened. The player that is closest to ten without going over ten wins. There are 2 possible ways to win as a player. If player has ten, player wins; and if dealer is over ten player wins. The game is played with turns, and there are only two possible choices that players have, they can hit or stand. Player is the first one that starts playing, and dealer starts playing after player stands.

2.3 Possible outcomes of Easy 21

There are 2 main possible outcomes in the game; player winning or dealer winning. Player can only win if dealer gets over 10, because player starts first and dealer waits him to stand, meaning that dealer won't stop hitting unless he is over player or 10. For example if player stopped taking cards when he was at 9, dealer will hit until he hits 10 because if he doesn't he loses anyways. Dealer can win if he gets over player without getting over ten, or dealer can win if player gets over ten. Draw is not a possible outcome because in total there are only three fives and three twos meaning that both player and dealer can only have the same value when they both get 7 and dealer will hit anyways.

2.4 Optimum Strategies for Easy 21

This table shows all the possible hands and best strategies for them found with probability for player. Player's Hand: the first 2 cards that player picks. Dealer's Hand: the first 2 cards that dealer picks. Remaining Cards: the last two cards that are left for picking. Probability of Hit: player's wining chance by hitting. Probability of Stand: player's winning chance by standing. Optimum Strategy: the best strategy for player to perform.

Player’s Hand	Dealer’s Hand	Remaining Cards	Probability of Hit	Probability of Stand	Optimum Strategy
2,2	2,5	5,5	100%	0%	Hit
2,2	5,5	2,5	0%	0%	Lost
2,5	2,2	5,5	0%	0%	Lost
2,5	2,5	2,5	50%	50%	Hit, Stand
2,5	5,5	2,2	0%	0%	Lost
5,5	2,2	2,5	0%	100%	Stand
5,5	2,5	2,2	0%	100%	Stand

Table 1: Easy 21 outcomes

Player shouldn’t bet when (4,10), (7,4), (7,10) because when player has those hands he will lose no matter what. When the hands are (7,7) player has 50% of wining no matter what he does. Player must bet when he has (4,7), (10,4), (10,7). When player has (4,7) if player hits, player will always win because, player will pick a five, as the only cards left are five, making the total value (9,7) and player will win because if dealer hits dealer will take a five, making the value 12. Player will win if he stands with (10,4) and (10,7) starting hands as player already has the best value. Same type of reasoning and calculations will be used in 21.

Now, a very popular table of tactics of Blackjack among gamblers will be shared in order to discuss what will be done in order to achieve a better strategy.

2.5 The so-called best strategy for Blackjack

Dealer's Card

Player's Hand

	2	3	4	5	6	7	8	9	10	A
8	H	H	H	H	H	H	H	H	H	H
9	H	DD	DD	DD	DD	H	H	H	H	H
10	DD	DD	DD	DD	DD	DD	DD	DD	H	H
11	H	H	DD	DD	DD	DD	DD	DD	DD	H
12	H	H	S	S	S	H	H	H	H	H
13	S	S	S	S	S	H	H	H	H	H
14	S	S	S	S	S	H	H	H	H	H
15	S	S	S	S	S	H	H	H	H/R	H
16	S	S	S	S	S	H	H	H/R	H/R	H/R
17	S	S	S	S	S	S	S	S	S	S
A,2	H	H	H	DD	DD	H	H	H	H	H
A,3	H	H	H	DD	DD	H	H	H	H	H
A,4	H	H	DD	DD	DD	H	H	H	H	H
A,5	H	H	DD	DD	DD	H	H	H	H	H
A,6	H	DD	DD	DD	DD	H	H	H	H	H
A,7	S	DD	DD	DD	DD	H	H	H	H	H
A,8	H/P	H/P	S	S	S	S	S	S	S	S
A,9	H/P	H/P	S	S	S	S	S	S	S	S
2,2	H	H	P	P	P	P	H	H	H	H
3,3	H	H	P	P	P	P	H	H	H	H
4,4	H	H	H	H/P	H/P	H	H	H	H	H
5,5	DD	DD	DD	DD	DD	DD	DD	DD	H	H
6,6	H/P	P	P	P	P	H	H	H	H	H
7,7	P	P	P	P	P	P	H	H	H	H
8,8	P	P	P	P	P	P	P	P	P	P
9,9	P	P	P	P	P	S	P	P	S	S
10,10	S	S	S	S	S	S	S	S	S	S
A,A	P	P	P	P	P	P	P	P	P	P

H: Hit DD: Double Down S: Stand P: Split H/R: Split if allowed to Double afterwards, or Hit H/P:Surrender if allowed, or Hit

Table 2: Popular known strategy found from Internet from a famous gambler. (Blackjack Strategy, 2017)

2.6 Assumed Blackjack Rules

To make the calculations easier and more understandable, some casino rules will be changed from the known strategy's rules. In created rules the game will be played with infinitely many decks, because in real casinos they use 6 deck of cards and they get rid of some random cards to prevent counting cards, another reason for it to be calculated with infinity decks is that every casino uses different amount of decks and as more decks are added to the game outcome probabilities come closer of a game with infinity decks. There will be only one player and a dealer which is actually not a good idea to do in a real casino, because in a normal game a dealer plays against more than one people so he/she has a huge advantage, which is to start picking cards after everybody stops picking cards. In this essay money won't be used so when a person wins, winner gains one point, and loser doesn't lose any point, the reason that loser doesn't lose any point is to make the data of our program easier because our program will be counting the points and giving out win percentage so if loser were to lose point it wouldn't be able to calculate the win percentage. The reason that points are used because of the fact that money is betted before players see cards, so money is just a luck factor as it won't change any expected values, to be able to calculate the probability, or simulate games points is a good way to measure. Because money is not getting used, in rules created here there is no option to double down or take insurance, as they are options related to money. Surrender option won't be viable, because with surrender player takes a part of his money back but because money won't be used it is hard to determine what happens when the player surrenders. Under the rules created draw means nothing so if both the player and the dealer have the same value at the end they both gain a point, which cancels each other. So to use the best strategy table from the strategy given above under created rules, there won't be DD, H/R, H/S and H replaces them. The strategy table of the given strategy will be like Table 3 under generated rules.

Dealer's Hand

Player's Hand

	2	3	4	5	6	7	8	9	10	A
8	H	H	H	H	H	H	H	H	H	H
9	H	H	H	H	H	H	H	H	H	H
10	H	H	H	H	H	H	H	H	H	H
11	H	H	H	H	H	H	H	H	H	H
12	H	H	S	S	S	H	H	H	H	H
13	S	S	S	S	S	H	H	H	H	H
14	S	S	S	S	S	H	H	H	H	H
15	S	S	S	S	S	H	H	H	H	H
16	S	S	S	S	S	H	H	H	H	H
17	S	S	S	S	S	S	S	S	S	S
A,2	H	H	H	H	H	H	H	H	H	H
A,3	H	H	H	H	H	H	H	H	H	H
A,4	H	H	H	H	H	H	H	H	H	H
A,5	H	H	H	H	H	H	H	H	H	H
A,6	H	H	H	H	H	H	H	H	H	H
A,7	S	H	H	H	H	H	H	H	H	H
A,8	S	S	S	S	S	S	S	S	S	S
A,9	S	S	S	S	S	S	S	S	S	S
2,2	H	H	P	P	P	P	H	H	H	H
3,3	H	H	P	P	P	P	H	H	H	H
4,4	H	H	H	H	H	H	H	H	H	H
5,5	H	H	H	H	H	H	H	H	H	H
6,6	H	P	P	P	P	H	H	H	H	H
7,7	P	P	P	P	P	P	H	H	H	H
8,8	P	P	P	P	P	P	P	P	P	P
9,9	P	P	P	P	P	S	P	P	S	S
10,10	S	S	S	S	S	S	S	S	S	S
A,A	P	P	P	P	P	P	P	P	P	P

H: Hit S: Stand P: Split

Red: The numbers that are written in red are the sum of the values of your hand

Green: The numbers that are written in green are for some special cases

Table 3: Best strategy (Table 2) changed under the generated rules

2.7 Mathematical Checking for the Best Strategy that is Given

The dealer will hit when he has a total value of 8, 9, 10 and 11 in any case because of the fact that he wants to get closer to 21 and hitting for one card doesn't make him go over 21 and still makes the total value close to 21.

When the player has a total value of 12 if he gets a card with a value of 10 he will loose and he still needs to be higher than the dealer without exceeding 21. To do so table suggests that player should stand when the dealer has 4,5 and 6 because when he has a 4 there are 3 possible ways that he can win without hitting and they are (4,9), (4,10) or (4, A). When player has the total of 12 if dealer' visible card is 4, dealers wining probability is

$$\frac{1}{13}+\frac{4}{13}+\frac{1}{13}=\frac{6}{13}\approx\ 0.4615 \ldots\ldots. (1)$$

but dealer starts taking cards when player stops taking cards so the probabilities of dealer loosing when he hits should be calculated. Dealer doesn't need to hit if his sum is equal or higher than 12 so he will only hit when he has lower than 12. It is impossible to lose when you hit at 11. So it is impossible that dealer loses when he hits when he has a sum of 11. When player has a total of 12, the only possible way to win is that when dealer has a total value of 12 at the start and picks a card with the value of 10, there are hands that the sum of the is 12: (2,10), (3,9), (4,8), (5,7), (6,6), (6,6), (7,5), (8,4), (9,3), (10,2). The probability of having those hands is;

$$\frac{1}{13}.\frac{4}{13}.2+4.\left(\frac{1}{13}.\frac{1}{13}.2\right)\approx 0.0947 \ldots\ldots\ldots (2)$$

and the probability of picking a card that has the value of 10 is;

$$\left[\frac{1}{13}.\frac{4}{13}.2+4\left(\frac{1}{13}.\frac{1}{13}.2\right)\right].\frac{4}{13}\approx\ 0.291\ldots\ldots\ldots (3)$$

So when player has the total value of 12 he should hit in any case, as without hitting player's winning chance is 2.91%, which shows that according to math this strategy that called the best strategy isn't the best.

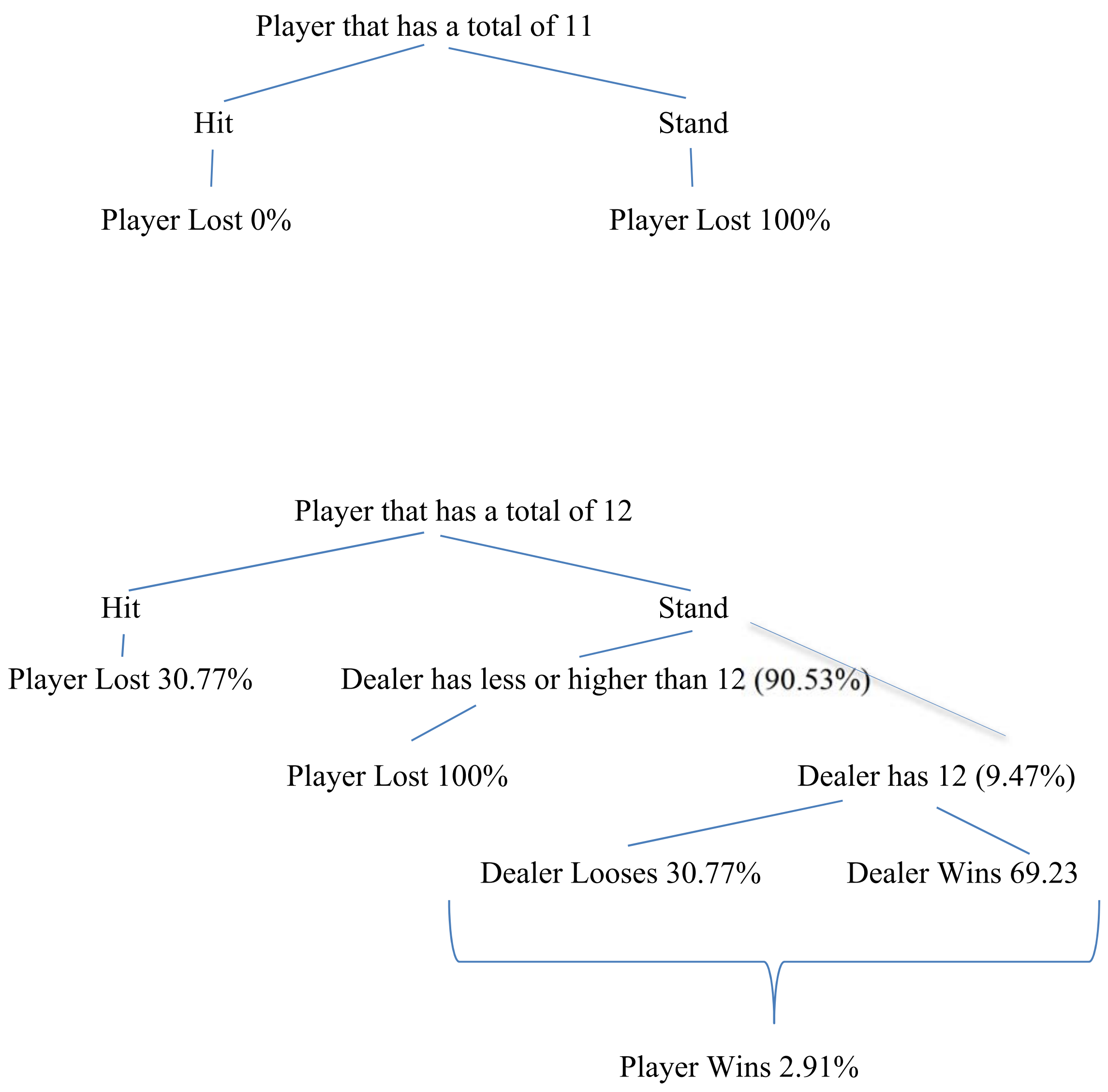


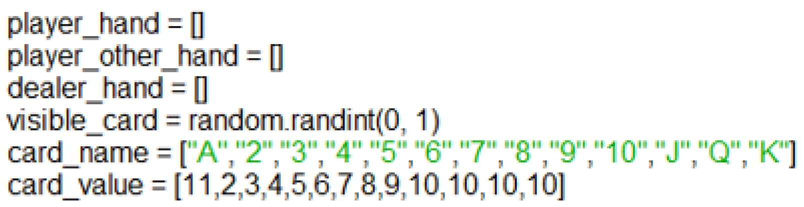
Diagram 1: A diagram of calculated in (1,2,3)

2.8 Data from program written with the strategy that is given as the best

As the probabilities are getting calculated with hand the number of the branches in the diagram increases, which is hard to calculate because there are so many possibilities such as player performing so many small hits will make the calculations more complex because when player picks a card new outcomes rise and there are so many outcomes that points to the same value. So a program is written in Python, which simulates Blackjack matches with the best strategy that is given and the created rules. The program created using logic, algorithmic reasoning, and functions. First the functions were defined and then the main game was created with tactics. The program is written to generate 1000000 games of Blackjack with rules that are created and with the tactics that Blackjack professionals gave, and at the end of those 1000000 it showed that if the player uses those tactics, player will win 32.45 % of his games.

2.9 Explanation of the written code

Programing uses mathematical thinking and mathematical reasoning so some of the process of writing the program will be explained. First some main concepts needed to be defined to computer such as the player, the dealer, and a deck. To do that lists were created, which can be thought as domain of the functions that will be defined for the program.



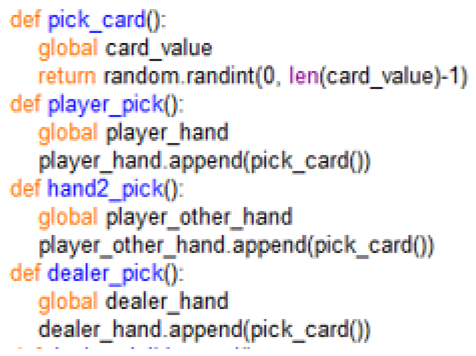
```
player_hand = []
player_other_hand = []
dealer_hand = []
visible_card = random.randint(0, 1)
card_name = ["A","2","3","4","5","6","7","8","9","10","J","Q","K"]
card_value = [11,2,3,4,5,6,7,8,9,10,10,10,10]
```

Figure 1. Initial definitions in the coding.

Dealer's, and player's cards will be defined as empty sets, and cards will be given at the beginning of the game. Card names and values defined differently because Aces need to be counted, and there are some special hands that needed different tactics, so by defining them separately without disturbing the value, which cards they have will be known. There is also a group in there called “player_other_hand” which is created for split tactic, so it will used as the new hand created in split.

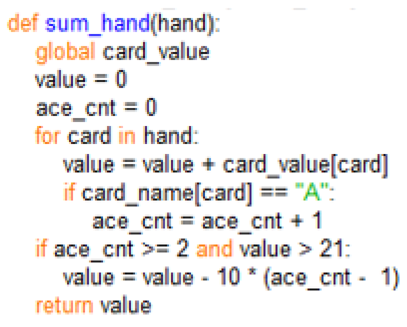
Then functions were created and these were used in tactics in order to make the program work faster, and to make the code look cleaner.

To define picking a card a function “pick_card” (Figure 2) was created to give a random number between 0 and 12 including 0, and 12, and then for player picking a card “player_pick” was created to append the random number generated in to the created “player_hand” (in Figure 1.). So this number [0,12] is stored and this is not the value of the card but an address for the card. And when wanted to get the value of the card this address will be used to get it (in Figure 3.)



```
def pick_card():
    global card_value
    return random.randint(0, len(card_value)-1)
def player_pick():
    global player_hand
    player_hand.append(pick_card())
def hand2_pick():
    global player_other_hand
    player_other_hand.append(pick_card())
def dealer_pick():
    global dealer_hand
    dealer_hand.append(pick_card())
```

Figure 2.

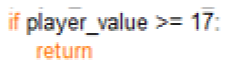


```
def sum_hand(hand):
    global card_value
    value = 0
    ace_cnt = 0
    for card in hand:
        value = value + card_value[card]
        if card_name[card] == "A":
            ace_cnt = ace_cnt + 1
    if ace_cnt >= 2 and value > 21:
        value = value - 10 * (ace_cnt -  1)
    return value
```

Figure 3

(Figure 3) shows how the created random number will be used the adres created for adding the values of the cards. To understand the function completely for and if functions needed to be learned but this essays focus is not python.

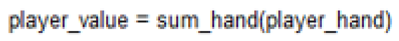
The tactics given will be executed with the help of created sets, and functions, and easy example would be standing if player's hand is bigger or equal to 17 which can be seen from Table 3.



```
if player_value >= 17:
    return
```

Figure 4

This function basicly tells us to stand if the total of player's hand is bigger or equal to 17. "player_value" is defined by using "sum_hand" for "player hand" (Figure 5). The reason there is no stand funciton is because of the way that Blackjack was explained to the computer. To make the programing easier Blackjack explained as a turn based game in the program without cahanging anything.



```
player_value = sum_hand(player_hand)
```

Figure 5

There are so many other functions, groups and tactics in the code but this will give us a simple understanding of the code, and this also shows that mathematical reasoning was used.

<u>2.10 Data from program written with the strategy that is given as the best without split tactic</u>

This win rate isn't good and as calculated in mathematical calculations this strategy isn't the best strategy with infinite amount of decks. To improve this strategy, split tactic should be removed, because no matter which tactic the player uses dealer always has higher chance to win as dealer has the advantage to start picking cards when player stops picking cards which is a major advantage as it makes the probability of being under the player 0%. So basically splitting helps dealer by playing more games. When the program is changed so that it won't use the splitting tactic the win rate of player increased to 33.31% as expected.

Dealer's visible card

Player's Hand

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

H: Hit S: Stand

Table 4: The Tactic Table without Split

2.11 Data from program written with the average strategy

After realizing that the "Best strategy for Blackjack" can be improved for the created rules, The strategy is improved more by getting the avarage value of the card that the player can pick. The calculation of the avarage value of the card that can be picked by simply adding all the values that player can pick and dividing it by 13:

$$\frac{2+3+4+5+6+7+8+9+10+10+10+10+11}{13} = 7.3.$$

Sum is divided by 13 because from the rules created there are infinity amount of cards in the deck so the card that the player is holding doesn't effect the probability. It means that the card that player will pick is estimetied as 7.3, so to win player should stop taking cards when his hand's value is 13.7 but as player can only pick natural numbers by rouding 13.7, best value to stand can be found as 14. When the program is programed to use this tactic player's winrate increased to 33.73%. Which is better than the tactic without split.

Dealer's Hand

Player's Hand

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

H: HiT S: Stand

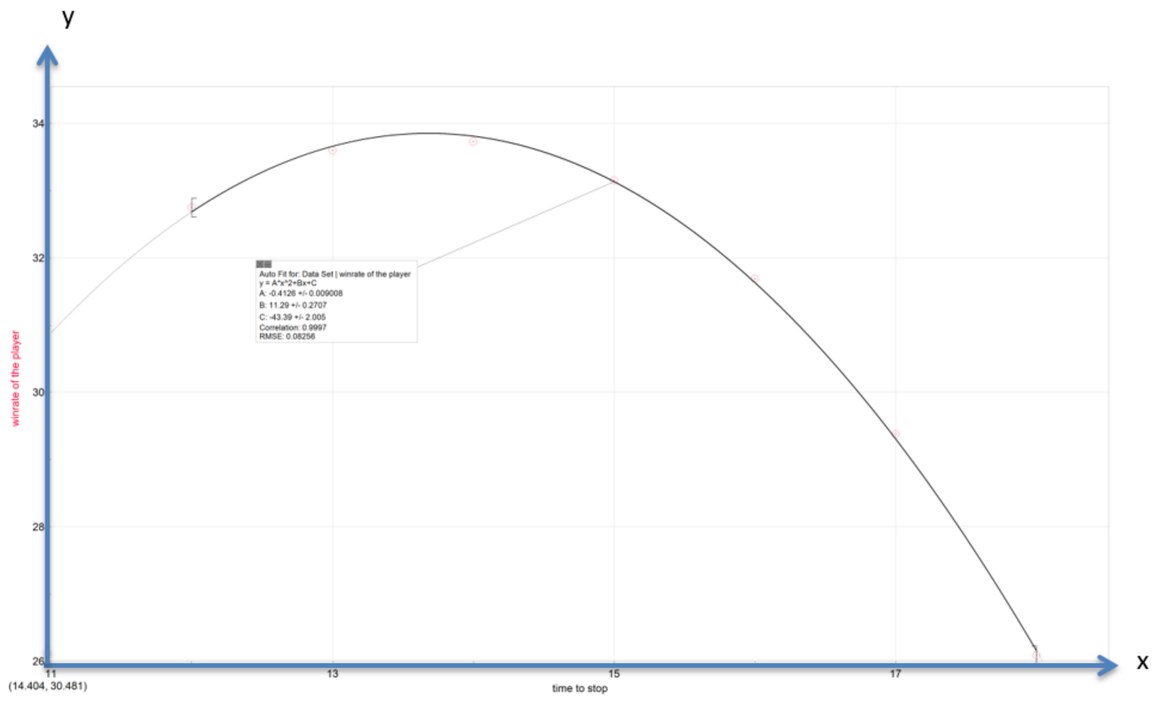
Table 5: The average tactic

2.12 Data Analysis

Tabele of the data produced from our program proves that the best strategy to win is to stop taking cards when player has a total of 14

The value to stop taking cards	Avarage winrate
12	32.75
13	33.60
14	33.73
15	33.15
16	31.69
17	29.38
18	26.09

Table 6: Calculated winrates



```
Auto Fit for: Data Set | winrate of the player
y = A*x^2+Bx+C
A: -0.4126 +/- 0.009008
B: 11.29 +/- 0.2707
C: -43.39 +/- 2.005
Correlation: 0.9997
RMSE: 0.08256
```

Graph 1: x representing : The value to stop taking cards; y representing : Avarage winrate

This graph will be used to determine the best value to stop. As this is a quadratic function first the x intercept point will be found to determine if this graph is resonable by using our function of the graph. Then if function of the graph is reasonable, derivative of the function will be solved for 0, which will give us the point with the highest winrate.The function of the graphed found from the graphing program "logger pro".

The function fitted to the data is :

$$y = (-0.4126 \pm 0.009008)x^2 + (11.29 \pm 0.2707)x - (43.39 \pm 2.005)$$

So by leting $y = 0$ x intercept will be found. $0 = (-0.4126\,)x^2 + (11.29)x - (43.39)$ to solve this quadratic formula will be used $x = \frac{-b \pm \sqrt{b^2-4ac}}{2a}$. From that formula x intercepts will be found as 4.62 and 22.74. That shows that there are some uncertaintys but the function of the graph is pretty good because when player stops picking cards when he hits 22 he will always lose by going over 21. And our graph shows that if player stops picking cards when his hand is 22.74 he will always lose which is pretty close to 22. So this graph is pretty reasonable for the winrates.

2.13 Best value to stop picking cards from graph

To find the best value to stop the highest value of y (winrate) must be found, and to do that the derivitive of the function will be taken and set it equal to 0.

$$f'(x) = \frac{d}{d_x}[(-0.4126\,)x^2 + (11.29)x - (43.39)]$$

The reason that the derivitive of the function is set to 0 is because of the fact slope of the tangent to the function from it's highest point will be 0; because this quadratic function first rises then drops so the highest value will be the point that where the slope of a tangent will be 0.

$f'(x) = -0.8252x + 11.29$, when $f'(x) = 0, x = 13.682$ so the best value to stop picking cards is 13.682 and the winrate of that is 33.842, which is calculated from inserting the best value to the main function.

$(-0.4126)13.682^2 + (11.29)13.682 - (43.39) = 33.842.$

So from all these it can be said that the best value to stop picking cards is 13.68, but a whole number need to be chosen so the graph suggest that best time to stop picking cards in a real game is 14 by just rounding 13.68, as the program and estimating the card value showed.

4.1 Conculusion

In conclusion, at the beginning of journey, the research question which was needed to be answered was: "What is the best strategy in a one versus one Blackjack game with probability theory and simulation?". To answer this research question, I first created a new set of casino rules, one of the most important rule was to play with infinite amount of decks so it was always the same probability to pick a card. After setting the rules most known best strategy was found and integrated with the new casino rules, to test the best strategy with probability theory first I created an easy version of 21, and analysed the probabilities over that to get a complete understanding. After understanding the probability theory, I calculated some outcomes from the known strategy, and tested the best known strategy with a program written – which simulated a million games to calculate the probabilities. After the calculations and data from the program I tried to improve the strategy with probability theory, and calculus. So this essay put out another strategy to use in Blackjack under given circumstances. In the essay probability theory was used to find the best strategy, to prove that it was a better strategy, data analysis and the program written to generate games were used. Although this essay used different methods to be sure about the core idea behind those methods, the biggest limitation of the study was limited analytical approach since mathematical calculations of these types of complex games is very difficult without computers. This essay was written with the purpose to show that one the popular strategy among gamblers is not appropriate under the given circumstances in this essay and a better strategy, which excludes the split tactic from the game, is offered. In conclusion, one versus one Blackjack game against a dealer isn’t fair and but if one decides to play this unfair game, one should use the average tactic as it has the highest win rate.

5. References

Baldwin, R., Cantey, W., Maisel, H. and MacDermott, J. (1956) The optimal strategy in Blackjack. *American Statistical Association Journal. 275*(51), 429-439.

Blackjack Strategy Learn the best way to maximise your winnings. (2017, July 17) Retrieved from www.bettingexpert.com/casino/blackjack/blackjack-strategy.

Blackjack – Card Game Rules (2017, July 16) Retrieved from www.bicyclecards.com/how-to-play/blackjack/.

Coltin, K. (2013) Optimal Strategy for Casino Blackjack: A Markov Chain Approach Retrieved from www.semanticscholars.org

Myerson, R.B (1997) *Game Theory: Analysis of Conflict.* Boston: Harvard University Press

Size of the online gambling market from 2009-2020. (2017, August 10) Retrieved from https://www.statista.com/statistics/270728/market-volume-of-online-gaming-worldwide/

Thorp, E.O. (1966) *Beat the dealer: a winning strategy for the game of twenty one*. New York: Vintage Books

Vaidyanathan, A. (2014) *Monte Carlo comparison of strategies of Blackjack.* An Undergraduate Theses, Department of Mathematics of Darmouth College.

6. Appendices

6.1 Appendix 1 - Rules and Terms of Blackjack

In blackjack each player starts with 2 cards and one of the dealer's card is shown to the players. Value of each card is its own number except aces, and face cards. Aces can be counted as 11 or 1, so ace is a card that changes from situation to situation. Face cards $\{King, Queen, Jack\}$ counts as 10 points. In a one on one game, game starts with dealer dealing two cards face up to a player and 1 card face up, and 1 downward to himself. Player adds up his/her cards values and stands, hits, double downs, surrenders, or splits depending to player's hand and dealer's hand. After player stops taking cards which means the player stands, dealer starts taking cards until dealer hits a higher value than player. The one that who has the highest value under 21 or a 21 wins the game.

Blackjack: Having the value of 21 with the starting hand. For this to happen player needs to have an ace and a card with the value of 10.

Bust: The total value of the cards is being over 21.

Hand: The cards that are in the game, such as player's hand meaning player's cards.

Hit: Taking an additional card to your hand.

Stand: Stop taking additional cards, and letting dealer take cards. By thinking blackjack as a turn based game, stand would mean finishing player's turn and letting dealer play.

Double down: Increasing the amount of money you bet in the table by 100%. This rule changes casino to casino, in some casinos you can increase the amount of money you put as you please.

Soft Hand: A hand, which contains an Ace and it, can have the value of 11 without causing a bust.

Hard Hand: All the other hands rather than soft hands.

Split: If the two cards that the player gets are the same, player can split the hand, which means the two cards are separated and instead of playing with 1 hand player starts playing with 2 hands. After the player splits, dealer deals one card per hand to the person who split and, player starts playing with two independent hands in the same game; but to split player must pay the entrance money for the other hand as well.

Surrender: When the player surrenders dealer takes half of the player's bet. Players can't surrender after standing or going over 21.

Insurance: The dealer offers insurance when the open card that dealer has is an ace. Insurance is a side bet, meaning that the dealer has Blackjack (21).

6.2 Appendix 2- The program written with split tactic

```
# Main
import random
import time
print("calculating")
player_hand = []
player_other_hand = []
dealer_hand = []
visible_card = random.randint(0, 1)
card_name = ["A","2","3","4","5","6","7","8","9","10","J","Q","K"]
card_value = [11,2,3,4,5,6,7,8,9,10,10,10,10]

# GAME RELATED FUNCTIONS
def pick_card():
    global card_value
    return random.randint(0, len(card_value)-1)
def player_pick():
    global player_hand
    player_hand.append(pick_card())
def hand2_pick():
    global player_other_hand
    player_other_hand.append(pick_card())
def dealer_pick():
    global dealer_hand
    dealer_hand.append(pick_card())
def dealer_visible_card():
    global dealer_hand
    global visible_card
    assert (len(dealer_hand) > visible_card), "dealer hand longer ??"
    return dealer_hand[visible_card]
def sum_hand(hand):
    global card_value
    value = 0
    ace_cnt = 0
    for card in hand:
        value = value + card_value[card]
        if card_name[card] == "A":
            ace_cnt = ace_cnt + 1
    if ace_cnt >= 2 and value > 21:
        value = value - 10 * (ace_cnt -  1)
    return value
def card_name_to_id(name):
    global card_name
    z = -1
    i = 0
    while i < len(card_name):
        if card_name[i] == name:
            z = i
            break
        i = i + 1
    return z
def count_hand(hand, name):
    global card_name
    id = card_name_to_id(name)
    assert (id != -1), "Card not found"
    count = 0
    for card in hand:
        if card == id:
            count = count + 1
    return count


def dealer_won(hand, dealer):
    hand_d = (sum_hand(hand) - 21)
    dealer_d = (sum_hand(dealer) - 21)
    if hand_d > 0:
        return True
    if dealer_d > 0:
        return False
    hand_d = abs(hand_d)
    dealer_d = abs(dealer_d)
    return dealer_d < hand_d
# SPLIT
```

```
def split():
    global player_hand
    global player_other_hand

    assert (len(player_hand) == 2), "card != 2 "
    assert (len(player_other_hand) == 0), "you already have 2 hands "

    hand2_pick()
    player_other_hand.append(player_hand[1])
    player_hand[1] = pick_card()

    while sum_hand(player_hand) <= 11:
        player_pick()
    while sum_hand(player_other_hand) <= 11:
        hand2_pick()

    dealer_visible_val = card_value[dealer_visible_card()]
    player_value = sum_hand(player_hand)

    if player_value == 12:
        if dealer_visible_val >= 4 and dealer_visible_val <= 6:
            return
        else:
            player_pick()
            player_value = sum_hand(player_hand)

    while player_value >= 13 and player_value <= 16:
        if dealer_visible_val <= 6:
            return
        else:
            player_pick()
            player_value = sum_hand(player_hand)

    if player_value >= 17:
        return

    if player_value == 12:
        if dealer_visible_val >= 4 and dealer_visible_val <= 6:
            return
        else:
            hand2_pick()
            player_value = sum_hand(player_other_hand)

    while player_value >= 13 and player_value <= 16:
        if dealer_visible_val <= 6:
            return
        else:
            hand2_pick()
            player_value = sum_hand(player_other_hand)

    if player_value >= 17:
        return
    return
def dealer_play():
    global dealer_hand
    global player_hand
    player_value = min(sum_hand(player_hand), sum_hand(player_other_hand))
    dealer_value = sum_hand(dealer_hand)
    if dealer_value < player_value:
        dealer_pick()



def play_round():
    global player_hand
    global card_value

    ace_cnt = count_hand(player_hand, "A")

    other_card = -1
    dealer_visible_val = card_value[dealer_visible_card()]

    for card in player_hand:
        if card_name[card] != "A":
            other_card = card
            break
# TACTICS
    if ace_cnt == 2:
        split()
        return
    elif ace_cnt == 1:
        if other_card <= 6:
            player_pick()
        elif other_card == 7:
            if dealer_visible_val == 2:
                return
            else:
                player_pick()
```

```
elif other_card >= 8 and other_card <= 10:
    return
elif count_hand(player_hand, "2") == 2 or count_hand(player_hand, "3") == 2:
    if dealer_visible_val >= 4 and dealer_visible_val <= 7:
        split()
        return
    else:
        player_pick()
elif count_hand(player_hand, "4") == 2 or count_hand(player_hand, "5") == 2:
    player_pick()
elif count_hand(player_hand, "6") == 2:
    if dealer_visible_val >= 3 and dealer_visible_val <= 6:
        split()
        return
    else:
        player_pick()
elif count_hand(player_hand, "7") == 2:
    if dealer_visible_val <= 7:
        split()
        return
    else:
        player_pick()
elif count_hand(player_hand, "8") == 2:
    split()
    return
elif count_hand(player_hand, "9") == 2:
    if dealer_visible_val == 7 or dealer_visible_val == 10 or dealer_visible_val == 11:
        return
    else:
            split()
            return
    elif count_hand(player_hand, "10") == 2:
        return

    player_value = sum_hand(player_hand)
    while player_value <= 11:
        player_pick()
        player_value = sum_hand(player_hand)

    if player_value == 12:
        if dealer_visible_val >= 4 and dealer_visible_val <= 6:
            return
        else:
            player_pick()
            player_value = sum_hand(player_hand)

    while player_value >= 13 and player_value <= 16:
        if dealer_visible_val <= 6:
            return
        else:
            player_pick()
            player_value = sum_hand(player_hand)
    player_value = sum_hand(player_hand)
    if player_value >= 17:
        return

# Output
dealer_win = 0
```

```
player_win = 0
for round in range(0, 1000000):
    for k in range(0, 2):
        player_pick()
        dealer_pick()

    play_round()
    dealer_play()



    if len(player_other_hand) != 0:
        if dealer_won(player_hand, dealer_hand):
            dealer_win = dealer_win + 1
        else:
            player_win = player_win + 1
    else:
        if dealer_won(player_hand, dealer_hand):
            dealer_win = dealer_win + 1
        else:
            player_win = player_win + 1

        if dealer_won(player_other_hand, dealer_hand):
            dealer_win = dealer_win + 1
        else:
            player_win = player_win + 1



    player_hand = []
    player_other_hand = []
    dealer_hand = []
    visible_card = random.randint(0, 1)


print(player_win, dealer_win)
print("player winrate is :  ",player_win/(player_win + dealer_win)*100,"%")

time.sleep(30)
```

6.3 Appendix 2 - Program written without split tactic

Only copy this and paste on top of the # TACTICS in 6.2

```
    player_value = sum_hand(player_hand)
    while player_value <= 11:
        player_pick()
        player_value = sum_hand(player_hand)


    if player_value == 12:
        if dealer_visible_val >= 4 and dealer_visible_val <= 6:
            return
        else:
            player_pick()
            player_value = sum_hand(player_hand)


    while player_value >= 13 and player_value <= 16:
        if dealer_visible_val <= 6:
            return
        else:
            player_pick()
            player_value = sum_hand(player_hand)
```

```
   player_value = sum_hand(player_hand)
   if player_value >= 17:
      return
```

6.4 Appendix 2 - Program written with the average value

Only copy this and paste on top of the # TACTICS in 6.2

```
   player_value = sum_hand(player_hand)
   while player_value <= 13:
         player_pick()
         player_value = sum_hand(player_hand)

   if player_value >= 14 :
      return
```