

ARTIFICIAL INTELLIGENCE

---

# BLACKJACK STRATEGY: BEATING THE CASINO

---

December 15, 2018

Ryan Worley  
Stanford University  
CS Department  
[rcworley1592@gmail.com](mailto:rcworley1592@gmail.com)

# Contents

<b>1</b>	<b>Introduction</b>	<b>2</b>
<b>2</b>	<b>Project Goal</b>	<b>2</b>
<b>3</b>	<b>Basic Overview</b>	<b>2</b>
3.1	Baseline and Oracle . . . . .	3
<b>4</b>	<b>Models</b>	<b>3</b>
4.1	Simplified MDP . . . . .	3
4.2	Full Blackjack MDP . . . . .	4
<b>5</b>	<b>Value Iteration</b>	<b>5</b>
5.1	Algorithm . . . . .	5
5.2	Card Count Analysis . . . . .	5
5.3	Mid-Card Analysis . . . . .	6
<b>6</b>	<b>Betting Policies and Simulation</b>	<b>7</b>
6.1	Prediction of Expected Value of Different Betting Policies . . . . .	7
6.2	Simulation Test . . . . .	9
6.3	Results and Error Analysis . . . . .	9
<b>7</b>	<b>Conclusion</b>	<b>9</b>
<b>8</b>	<b>Moving Forward, Final Product</b>	<b>10</b>
<b>9</b>	<b>Appendix</b>	<b>11</b>

# 1 Introduction

Blackjack is marketed as a simple game, one that casino novices should can play with ease. The premise is simple: draw cards until your total card value is as close to 21 as possible and to beat the dealer's hand. Past analysis on the game has found that blackjack has the best odds of any game in the casino, with only a 1.0% - 0.5% house edge when played properly.

While casino novices decide to risk their money on the blackjack table, they often don't play with the optimum policy. Blackjack is a game with many states, and knowing how to maneuver through all of them to give yourself the best chance to win can be confusing.

Some companies try to capitalize on this by selling 'Basic Strategy Cards', which claim to give the player information on how to approach betting at every state. However, these strategies are very basic and don't incorporate more advanced strategies involving keeping track of the state of the deck. These additional factors can be leveraged to give the player an edge against the casino. I believe that I can make a program that can improve the basic strategy of the starting player, by providing information on how to bet, and what their odds are in every situation on the board.

## 2 Project Goal

The goal of this project is to evaluate card counting as a strategy and decide how useful it is in gain advantage over the casino. Doing this will first involve deriving an optimal policy by which to play card counting blackjack, and the testing this policy through simulation. The utility of different betting policies will also be analyzed, and a recommendations will be made to the player on how to succeed in the game of Blackjack. A basic tool will be developed to display the optimum strategy to the player through an easy to use interface.

## 3 Basic Overview

Blackjack is a card game, where the player (agent) plays against an adversarial dealer with a set policy. The goal of the player is simple: get a hand value as close as possible to 21 without going over while also beating the dealer. The player can keep drawing cards until they are satisfied with their hand. The dealer must keep drawing cards until the value of his hand is between 17 and 21.

Card counting is a strategy that can employed by the player that gives them an advantage over the dealer. In a normal game, the casino will have anywhere between a 0.5 - 1 percent edge over the player. However, while using a card counting strategy, you can gain information about the relative probabilities of cards left in the deck. This all comes from knowing the value of 'True Card Count', which is the number of low cards (2, 3, 4, 5, 6) that have been drawn from the deck minus the number of high cards (10, J, Q, K, A) all divided by the total number of decks (52 cards) left to be drawn.

$$TrueCount = (LowCardsDrawn - HighCardsDrawn) / (52 * DecksRemaining)$$

The number of mid-card (7, 8, 9) that are still in the deck are not accounted for in this metric. The relative density of these cards in the deck is needed in order to have full knowledge of the relative probability of drawing a low, mid, or high card.

### 3.1 Baseline and Oracle

#### *Baseline:*

The baseline for this project is to achieve a utility of about -0.01 as a result of a card count move and betting policy. This would represent the approximately 1% house edge built into a normal game of blackjack without card counting. As literature states, this should be about the expected value that would be achieved if one unit was bet for each hand and basic strategy was played.

#### *Oracle:*

The oracle for this challenge lies with an individual that claims to have won over \$100,000 dollars a year playing advantage blackjack. One team out of Seattle even claim that they won over 3.5 million in just a couple of years. While these total value numbers are astounding, it doesn't provide an estimate of the expected value of the game, since the total investment is unknown. However, there is documentation of card counting providing a 1.2% advantage to the player, with a standard deviation of 3.5 (very high). This 1.2% advantage will act as our baseline for the experiment.

## 4 Models

### 4.1 Simplified MDP

#### *Purpose:*

A simplified MDP model was created to produce an optimum policy and expected values for the game of Blackjack including card counting. This MDP operates under the assumption that the probability of drawing a low card, mid-card, or high-card will be dependent on the true card count. The transition probability for drawing a certain card doesn't change, regardless of what previous cards have been drawn. Despite the simplifications in this MDP and its differences from how Blackjack is actually played, its results should be of use.

#### *States: (PlayerState, DealerState, CardCount)*

PlayerState and DealerState are set up in the following format: **State** = [XX]\*[keys]. [XX] represents the maximum legal value of either the player or dealer's hand. If the hand has ace, the value of the ace is 11 unless this makes the total value greater than 21. Then the ace is given value 1. [keys] represents letter keys added to state string that provide information about possible next moves. If 'D' or 'S' are in the key portion of the state, then this means that a double or split move is legal on the next turn. If 'A' is in keys, this means the player has an ace in their hand with value 11. If the ace has a value 1, then the 'A' key is deleted.

#### *Actions: (Stay, Draw, Double, Begin)*

*Stay* signifies that the player will stop drawing cards and the dealer will draw from the deck until their value is greater than 17. *Draw* means that the player will take another card. If this card puts the player over a value of 21, the player loses. *Double* means that the player doubles their bet and draws one more card. On the next move, the player must stay. This can only happen after the *Begin* action. *Begin* is always the first move in a new hand. Here, the dealer draws one card and the player two cards.

#### *Transitions:*

The transition probabilities for the simplified MDP are directly related to the card-count and mid-card value. These transition probabilities do not change throughout the MDP. If more

than one card is drawn per action, then the transition probability would involve multiplying individual card probabilities:

$$P(Card1, \dots, CardN) = P(Card1) * P(Card2) \dots * P(CardN)$$

*Rewards:*

Rewards are only earned at the end of the MDP. Typically, the player either earns or loses one betting unit depending on whether they win or lose. All ties result in a push. Some edge cases includes ‘BLACKJACK’, where the player’s hand has value 21 after the *Begin* action and the dealer does not. In this case, the player receives 1.5 times their bet value. Additionally, if the player chooses to *Double* on a hand, the reward (whether win or lose) will be doubled.

*Initialization of the Class:*

There are two inputs needed from the user in order to initialize the simplified MDP class. Firstly, a true count value is inputted. This count value must be an integer, and is taken as a constant for the MDP, meaning it will not be altered in any successor state. The second value is the mid-card value. The mid-card value is defined as the number of mid-cards (7, 8, 9) per 52 cards remaining in the deck. In a game where no cards are drawn, the number of mid-cards would be 12. However, once a blackjack game begins, this number will change. The mid-card value affects the probability of drawing any card, and is a necessary in addition to true count to calculate exact card draw probabilities.

## 4.2 Full Blackjack MDP

*Purpose:*

Created a more representative MDP for an actual casino blackjack game in order to test the optimum policies learned in the previous value iteration MDP using simulation. In this model, exact card draw probabilities are calculated and probabilities of drawing cards without replacement is considered.

*State: (playerState, dealerState, CardsRemaining, CardCount)*

*PlayerState* and *dealerState* are represented the same as in the approximate MDP. The new variable *CardsRemaining*, is a tuple of integers that represents the number of each card remaining in the deck. The *CardCount* variable is no longer a constant, and is changed to reflect the current deck state. This value is rounded to the nearest integer value.

*Actions:* Same as simplified MDP.

*Transitions:*

Transition states were calculated based on the ‘CardsRemaining’. The probability of each card being drawn is:

$$P(Card_i) = CardsRemaining_i / \sum_{n=1}^i CardsRemaining$$

In the case where multiple cards were drawn as the result of an action, probabilities will be multiplied similar to the procedure shown for the simplified MDP.

*Reward:* Same as simplified MDP.

## 5 Value Iteration

### 5.1 Algorithm

Value iteration was used to solve the simplified MDP problem resulting in an expected utility and optimal policy. Two different analysis experiments were performed using the data collected from this algorithm. First, a true card count analysis was run, analyzing how different count values affect the result of the MDP. Second, a mid-card analysis was run to observe how sensitive utility and policy values are to a change in the number of mid cards in the deck. Both of these methods will be further described below.

### 5.2 Card Count Analysis

The MDP was run for 31 different values of true card count (-15 to 15 incremented by 1). The optimum policy and expected utility was saved for each MDP solution. The expected value for the starting state of each MDP trial was extracted and plotted. The starting state's expected value is most relevant because this is the state at which players place their bets. As the true card count increases, the expected value of a hand increases approximately linearly. An important feature is the point where the expected utility changes from negative to positive, which is the point where the player now has the edge over the casino. This is the point in time where the player should bet big. When the count approaches the large positive or negative numbers, the count seems to approach an asymptotic maximum or minimum value.

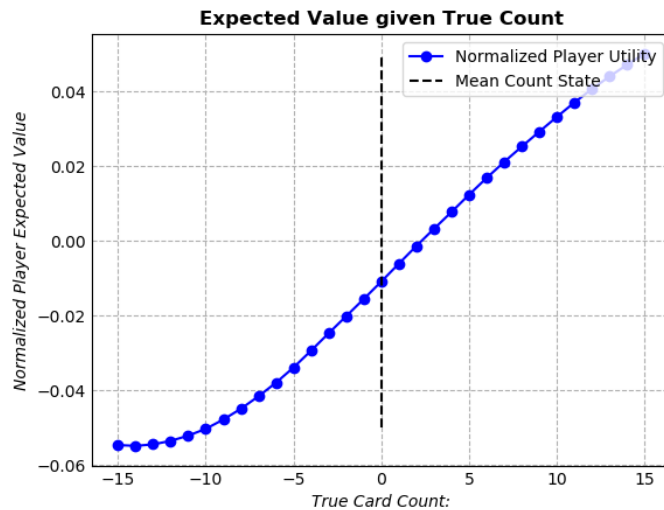


Figure 1:  $E[\text{Utility}|\text{Count}]$

Literature states that the casino typically has a house edge between 0.5% and 1%. From our expected value based on card count, we see that a typical game (card count of 0) has an expected house edge of just over 1%. However, we do not take in some additional rare edge cases in our program for complexity reasons (such as doubling or insurance), so it would make sense that the simplified MDP has a slightly lower expected value. If these extra moves were incorporated into the MDP, they would most likely only slightly affect the outcome.

In addition to variable expected utility for different card counts, the optimum policy for playing the game also changes. The plot below shows the percentage of actions in the optimum policy that are similar to the optimum policy to those at the mean state (count is 0). As the count

increases or decreases, the optimum policy continually changes, and the shared actions with the mean state decreases. If a player wanted to really make use of a card counting strategy in the casino, they're playing policy would vary significantly from basic strategy (up to 15%).

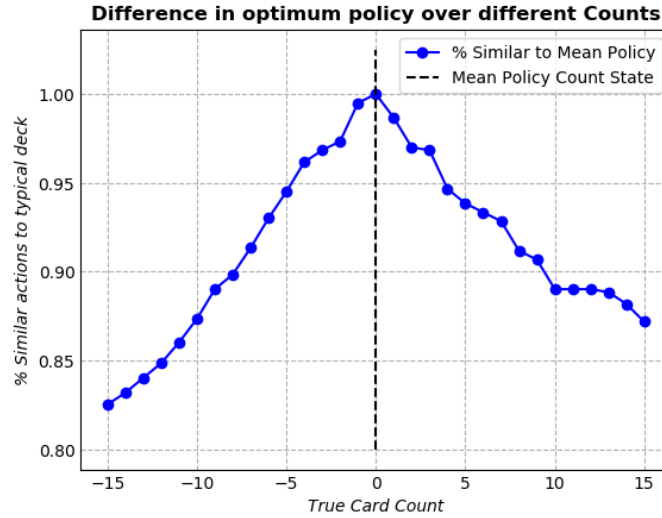


Figure 2: % Difference in Policy Actions over Different Count

### 5.3 Mid-Card Analysis

While card count is a useful metric to understand relative high-card and low-card probability, it does not give any information regarding the concentration of 7, 8, 9 (mid-cards) in the deck. For the previous value iteration count experiment, the mid-card value was assumed to be 12 (average for a deck of 52 cards). However, in an actual game, this will not always be the case. For this experiment, the number of mid-cards in the deck were varied and value iteration was run (the count was kept at an arbitrary value of 0).

The expected utility from the start state of a hand was found to change with respect to the number of mid-cards in the deck. However, this amount is fairly small (only about .2% of the bet). Additionally, the expected value seems to change favorably as values of mid-card move away from the mean (the mean is almost at a local minima).

The one possible flaw for this procedure is that this mid-card analysis was conducted over only one count value. To be complete, the analysis could be conducted over many different count values, to ensure that card count doesn't affect the mid-card v. expected utility relationship. If expected utility was not found to vary significantly over different mid-card values for all values of count, we could conclude that card count alone is a sufficient identifier to estimate the expected utility. However, due to lack of resources and time (and generally wanting to move into more interesting analysis) this aspect wasn't completed. In addition to slight expected value change, mid-card changes also induced slight variability in the policy as well. This was deemed to be insignificant, as there was no more than a 6% difference in policy actions between mid-card states, which shouldn't contribute to much change in our expected value.

Overall, we can tentatively say that our simplified MDP provided a solid optimum policy for which to base our card counting strategy. While differing mid-card values may slightly affect

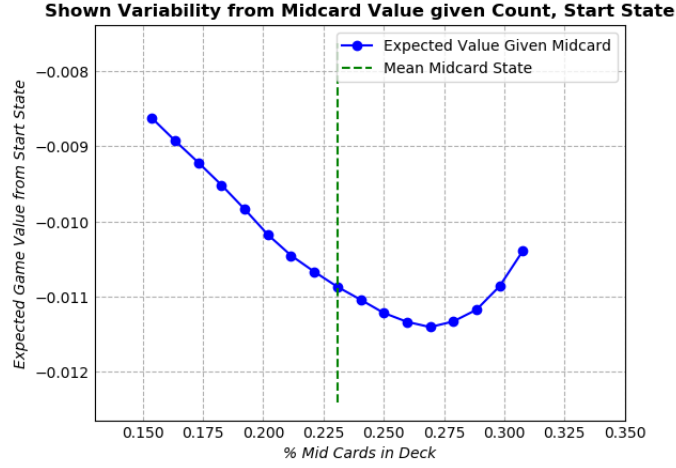


Figure 3:  $E[\text{Utility}|\text{Mid-Card}, \text{Count}=0]$

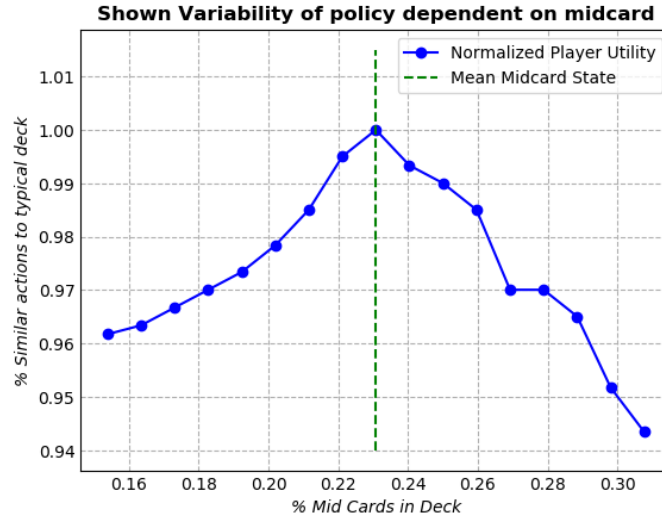


Figure 4: Policy Actions Comparison for Mid-Cards

the actions and expected value of this policy, these deviations will be small and won't end up hurting our expected value very much.

## 6 Betting Policies and Simulation

A Monte Carlo Model Fixed simulation algorithm was run using the `succProbReward` function of the accurate Blackjack MDP. The optimum policy created and saved from the simplified MDP was used as the model for the simulation. Using this method, we can test the optimum policy based on count and test different betting policies in order to maximize expected return.

### 6.1 Prediction of Expected Value of Different Betting Policies

Probability theory was used to get an estimate of expected value of different betting policies using the following equation:



$$E[\text{Reward}|\pi_{bet}] = \int \pi_{bet} * E[\text{Reward}|\text{Count}] * P(\text{Count})d\text{Count}$$

$P(\text{Count})$ :

A set of true card count values was found by simulating 45,000 games of blackjack. A normal distribution and a Cauchy distribution (adjusted normal distribution) were used along with a least squares regression was used to find an approximate distributin fit. It was determined that the Cauchy PDF distribution was the best fit due to it's much lower  $R^2$  value.

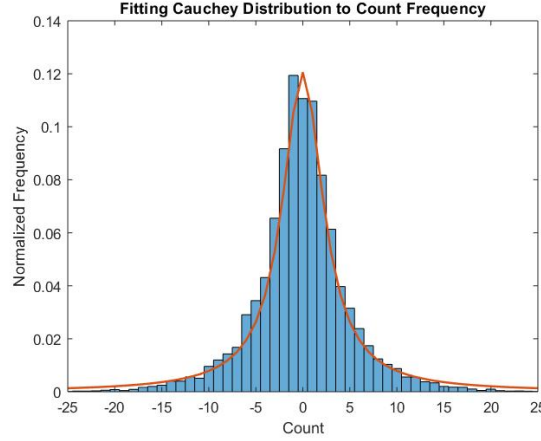


Figure 5: Cauchy PDF Fit

Unfortunately,  $P(\text{Count}) \neq \text{PDF}(\text{Count})$  for a continuous distribution function. However, we can discretize the Cauchy distribution at each unit count value. This will create a discrete PDF function that will sum to one, and can be used to integrate to find the expected utility.

$E[\text{Reward}|\text{Count}]$ : Derived earlier; see Figure 1.

$\pi_{bet}$ :

This is a policy function that plots initial bet amount against the true count of the game. Five different betting policies were created and tested.

Policy				
#	Policy vector	If count < 0	If 0<count<2	If count>2
1	[.5, 1, 5]	.5 unit	1 unit	5 units
2	[0, 1, 5]	0	1 unit	5 units
3	[0, 0, 5]	0	0	5 units
4	[1, 1, 1]	1 unit	1 unit	1 unit
5	[.5, 1, 2.5]	.5 unit	1 unit	2.5 units

In general, policies were created to improve player expected value. Thus, for higher values of count where the player has a larger chance of winning, the bet is raised. All betting policy charts can be seen in the appendix.

The expected reward, betting policy and count distribution multiplied and integrated to marginalize out the true count and find the expected value of the deck of a given bet.

## 6.2 Simulation Test

For each of the five betting policies, a simulation algorithm was run playing approximately 50,000 games of Blackjack. The average expected utility of each hand was calculated and compared to the expected utility found via the probabilistic analysis done in Section 6.1. The summary of all values can be seen below.

Policy	Estimated Utility from Simplified MDP	Simulation Utility using learned optimal policy	Residual
[.5, 1, 5]	.0251	.0315	.006
[0, 1, 5]	.0322	.00091	.0231
[0, 0, 5]	.0338	.0628	.0390
[1, 1, 1]	-.0089	-.0020	.0069
[.5, 1, 2.5]	.0082	.0102	.002

## 6.3 Results and Error Analysis

The results for trial 4 (the baseline case) indicate that playing with a uniform betting policy will allow use to achieve around a -.01 expected utility. This is very similar to what we hypothesized for our baseline, which has been achieved with this very simple policy.

When examining the other possible betting strategies, the expected value residuals were slightly higher than expected for trials two and three. However, the expected values from simulation are lower and higher for trials 2 and 3 respectively than the expected values from the simplified MDP, showing that one approach isn't biased in one way or another.

Thus, it seems likely that the source of error for trials 2 and 3 is due to a insufficient sample size. Only about 27% of hands are played with a count greater than two according to the Cauchy distribution. Statistically, betting a large number on a smaller sample will create higher variation in the expected value. In order to get better simulation data, more trials would have to be conducted. Unfortunately, due to lack of time and computing power, more simulation trials were not able to be run for the final project. Each 50,000 trial simulation takes around 1 hour. If I decide to keep working on this project in the future to refine and check the results, this is where I would start.

To address an earlier point, it doesn't seem like the variation in mid-card value (accounted for in the simulation) is significantly affecting the expected utility. In general, simulation values are slightly higher than the utility estimated from the simplified MDP (which does not account for mid-card variation). Thus, I believe we can conclude that mid-card values don't significantly affect the optimum policy derived by focusing on card count alone.

## 7 Conclusion

Though true card count does not paint a complete picture of the state of the deck, it does provide the user with enough valuable information to gain advantage in their play over the casino. The value iteration and simulation results indicate the playing with different optimum policies (learned via the simplified MDP) is a viable way to maximize your return at the table, despite not taking into account factors such as mid-card probability.

Both the simulation and the probability approach show that betting policies that involve betting big for large count values are optimal in blackjack. The larger the bets you place when you have the advantage (true card count 3 or above), the larger your expected returns will be. However, casino's have gotten wise to the idea of card counting, and are always on the lookout for players that alter their bet over a wide range based on the count. Though counting cards is in no way illegal, the casino is well within their right to ask you to leave. Due to this, I recommend that you use the [.5, 1, 2.5] betting strategy. This still will give you a positive expected utility, but will keep you generally under the radar.

## 8 Moving Forward, Final Product

File

State Initial Inputs: Card = (2, 3, 4, 5, 6, 7, 8, 9, T, J, Q, K, A)

Player Card 1:



Player Card 2:


Player Card 3:

Player Card 4:

Shown Dealer Card:

Card Count:

Player:  

Dealer: 

Optimum Policy: Double

Expected Reward: 0.39451

Figure 6: Application Training Tool Interface

A Blackjack card counting program application was made to display the optimum policy learned during the value iteration process. For any player training about how to play blackjack more efficiently, this is an excellent way to do so. Inputs to the GUI are player card values, dealer card values, and the current card count at the time. The player card and dealer card fields can be left blank if the user wants to know the expected reward associated with a certain true count at the start of the round. The application works by using a database of the optimum policy and expected values generated during the value iteration process. This application is basically an easier way to parse through the database and receive information based on your current state. This is a great way to train yourself for any blackjack situation, and a great learning tool for those looking to improve their game and win some money. The variable betting strategy has not been implemented into the GUI application yet, but could be at a later date.

## 9 Appendix

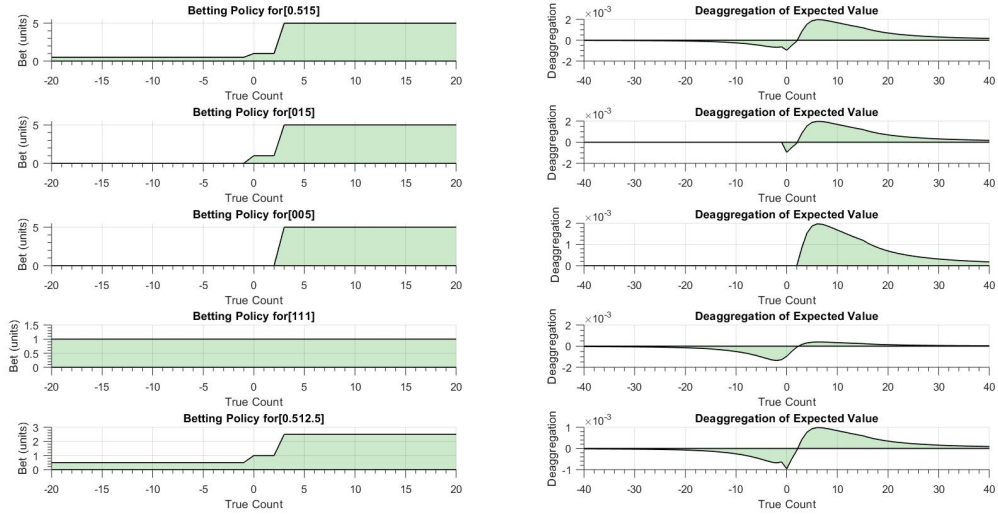


Figure 7: All Betting Policies and Deaggregation of Expected Value Charts

## References

- [1] Jensen, Kamron, *The Expected Value of an Advantage Blackjack player*. All Graduate Plan B and other Reports. 524
- [2] Wong, Stanford 1994. *Professional Blackjack*, page 31, 1994 ed.