John Santoro (U87402138) and Jim Bui

CIS4930.002F16

AI Project Report - Blackjack!

Overview:

This project is designing an AI agent to play against the house in the game of Blackjack. The AI initially knows nothing about when to draw or stand, and will learn this behavior from playing multiple games against the dealer.

System Description:

In this version of the Blackjack game, the dealer and an agent square off in a 1-on-1 battle. The dealer is given two cards, one of which is face-up and the other of which is face-down, and the agent is given two cards face-up. The agent is then given the opportunity to accept as many cards as it wants until either (1) the total value of all the cards in its hand is 21, (2) it goes bust, or (3) it chooses to stop accepting cards. Afterwards, the dealer draws until he has a hand valued at 17 or more, and then he stops. The dealer's face-down card is then revealed, and whoever's hand is closest in value to 21 wins.

The computer agent is initialized such that it knows only to draw cards until it goes bust. Naturally, it loses most of its first few games as a result. In order to 'learn' to play Blackjack, the agent does as follows:

1. Saves past experiences to a file.

   The file that the agent saves contains three columns and an unlimited number of rows. The formatting is described here

VAL_CARDS	PREV_DECISION	RESULT
<...>	<...>	<...>

   Where VAL_CARDS is the value of the hand at the time a decision is made, PREV_DECISION is whether or not the agent chose to hit (1 if hit, 0 if stand), and RESULT is the result of the agent's decision (1 if win, 0 if loss, -1 if not a loss or draw).

   Experiences are recorded after each move: in hits that do not result in a win or a loss, -1 is recorded in the result column.

2. Uses past experiences to create a knowledge table.

In order to make intelligent moves, the program converts the past experiences to a knowledge table.

If there are no past experiences, the knowledge table is initialized as follows:

------------------------------------ Knowledge ------------------------------------

VAL_CARDS	NOLOSS_IF_HIT	NOLOSS_IF_STAND	ERR_HIT	ERR_STAND
2	1	0	1	1
3	1	0	1	1
4	1	0	1	1
5	1	0	1	1
6	1	0	1	1
7	1	0	1	1
8	1	0	1	1
9	1	0	1	1
10	1	0	1	1
11	1	0	1	1
12	1	0	1	1
13	1	0	1	1
14	1	0	1	1
15	1	0	1	1
16	1	0	1	1
17	1	0	1	1
18	1	0	1	1
19	1	0	1	1
20	1	0	1	1

VAL_CARDS is as in the past experiences table; NOLOSS_IF_HIT is the chance that the agent will not lose if he chooses to take a hit; NOLOSS_IF_STAND is the chance that the agent will not lose if he chooses to stand; ERR_HIT is the margin of error in the NOLOSS_IF_HIT column; and ERR_STAND is the error in the NOLOSS_IF_STAND column.

3. Uses the knowledge table to make decisions.

The agent will use the knowledge table to determine whether to hit or stand when its cards are of a specific value. In particular, if the chance of not losing when accepting a hit is greater than the chance of not losing when standing, the agent will *normally* try to accept a hit. If the opposite is true, the agent will *normally* stand. There is, however, gray-area in the decision-making process: Suppose that the agent has a hand valued at 15, that NOLOSS_IF_HIT is 0.75 and NOLOSS_IF_STAND is .667. If the margin of error columns contain large numbers, agent will not always choose to accept a hit, though it "knows" that it is a better decision. This is done to ensure that the data does not become skewed easily, showing higher or lower probabilities than it ought.

Please see PlayerAI.h for more information.

Summary of Experimental Results:

With the initial games of an AI with no prior games, the AI has no knowledge to base its decisions off of. Instead, the AI randomly picks whether to stand or to draw another card. Whether the new AI win or loses is purely based off of chance.

With the untrained AI, we played five sessions of 500 games each, and recorded the wins, draws, and losses. We then trained an AI with a sufficient number of games, and repeated the same process. For the purposes of the win rate, draw is considered a win.

Inexperienced AI

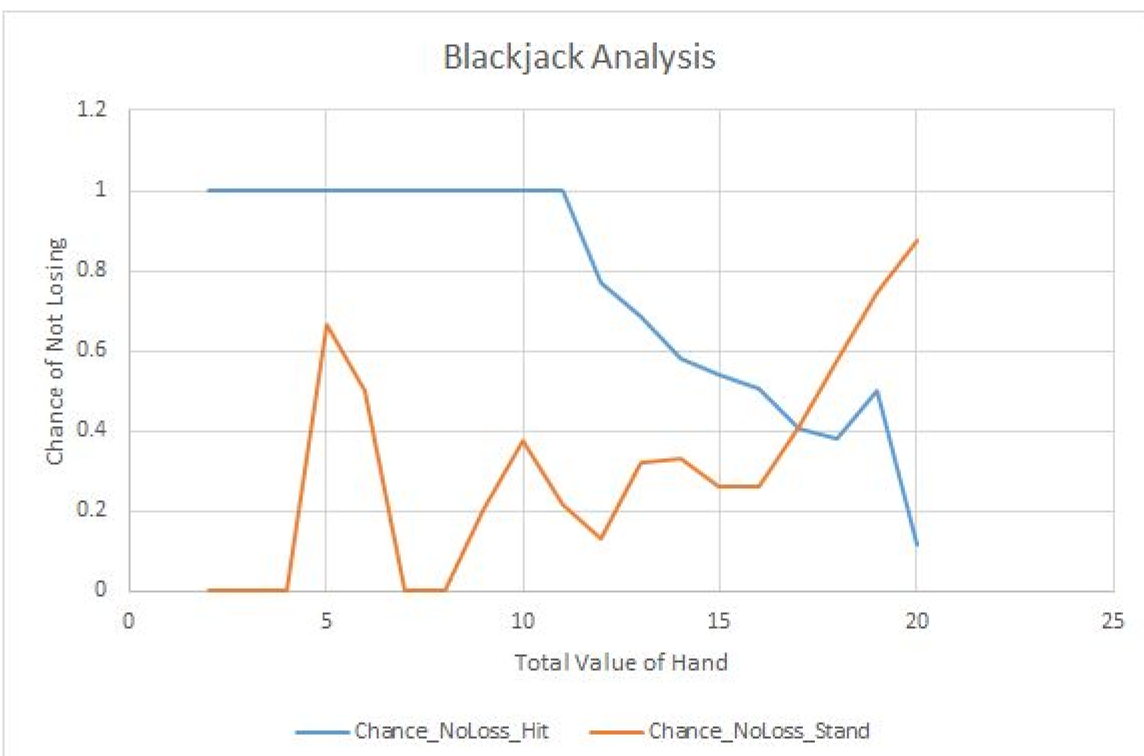
Wins	Draws	Loses	Win Rate
141	11	348	0.30
125	7	368	0.26
133	7	360	0.28
141	4	355	0.29
129	6	365	0.27

Experienced AI

Wins	Draws	Loses	Win Rate
217	37	246	0.51
215	41	244	0.51
221	44	235	0.51
222	40	238	0.52
214	45	241	0.52

We can see that the AI improves by more than 20% after it has been trained, and indeed, the AI does learn from its past games. The AI does not improve any more after this point.

This next chart shows the chances of losing if you either decide to stay or draw another card based on the current value of your hand.



We can see that your chances of not losing if you draw another card are non-existent if you don't have at least a value of 12 in your hand. From there, the chances of not losing steadily goes down. The chances of not losing from not drawing are less consistent in data, but generally it is more likely to lose if the hand is low valued.

Interestingly enough, we can see that the point where the chances of standing and drawing intersect is a hand value of 17, which is the value off which the dealer plays for.

Conclusions:

The AI was able to improve its skill such that it stopped drawing cards before it went bust in most cases, and was able to improve its win-rate by over 20% from learning. It seems like Blackjack is technically beatable (without counting cards), but only over many games, since the win-rate is only about 51-52%. Additionally, the player must be able to play perfectly in order to have a chance statistically.