Ryan Smith

CS 540: AI

HW #3

1.

FullJointProbTablePlayer_rasmith

My full joint probability table player uses joint probabilities among variables representing the state of the game board from my player's perspective.

We use variables representing:

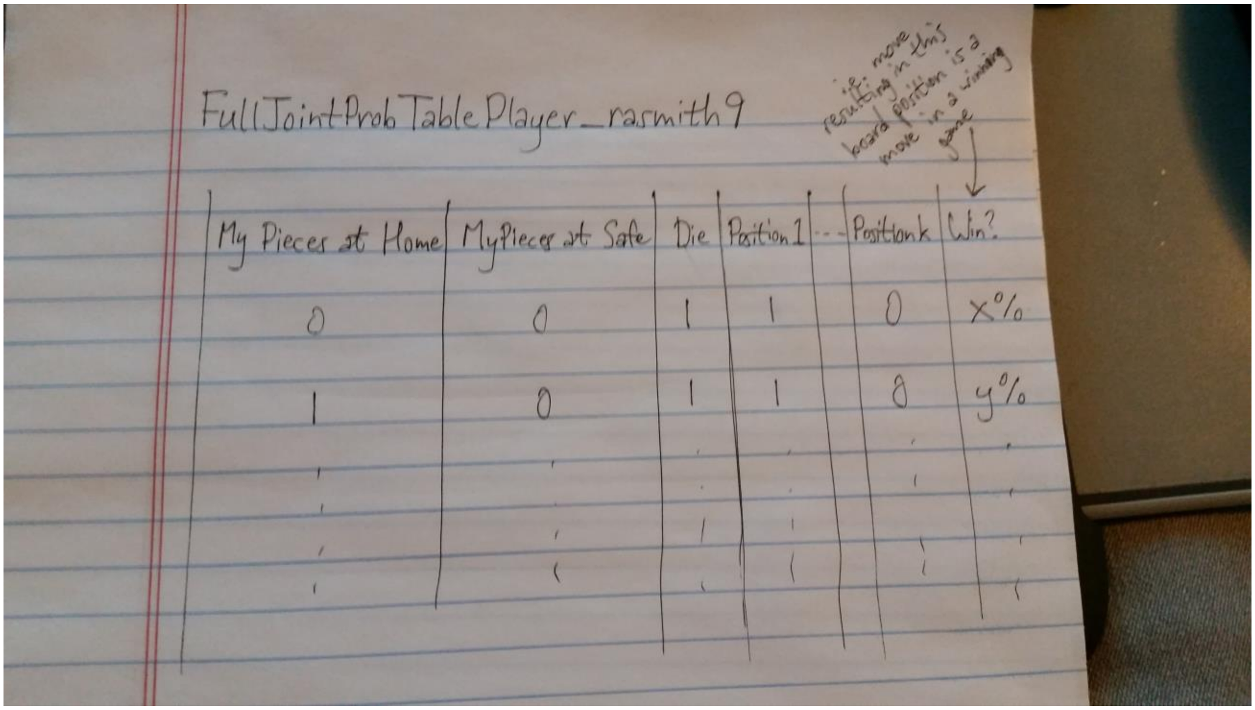
- The number of my pieces currently at home, from among {0, 1, 2, 3}
- The number of my pieces currently at safe, from among {0, 1, 2, 3}
- The die value of my roll, from among {0, 1, 2, 3, 4, 5, 6, 7} (0 if the die value is -1, as in the beginning of the game)
- The piece (or absence of one) in each of the 6 board cells in the board size this player supports, from among {0, 1, 2}

We store these variables as nodes in two 9-dimensional arrays:

- count_given_move_in_win
- count_given_move_in_loss

So we capture the complete board state with respect to my player at each move. When updating statistics, for each move in a game we win, we increment the count_given_move_in_win's value with nodes representing the state of each move. When updating statistics, for each move in a game we lose, we increment the count_given_move_in_loss's value with nodes representing the state of each move.

When choosing a move, we consider each legal move available, and for each possible move we consider the resulting board state. We take the resulting board state of a move under consideration and insert its constituent variables into our "count_given_move" arrays. We then compute the probability that the move under consideration is a move in a winning game by taking the count_given_move_in_win value over the count_given_move_in_win value plus the count_given_move_in_loss value for the move. We then store this probability for each legal move and select the move for which this probability is greatest.



FullJointProbTablePlayer_rasmith9

i.e. move resulting in this board position is a move in a winning game

My Pieces at Home	MyPieces at Safe	Die	Position 1	- - -	Position k	Win?
0	0	1	1		0	x%
1	0	1	1		8	y%
⋮	⋮	⋮	⋮		⋮	⋮

2.

NaiveBayesNetPlayer_rasmith9

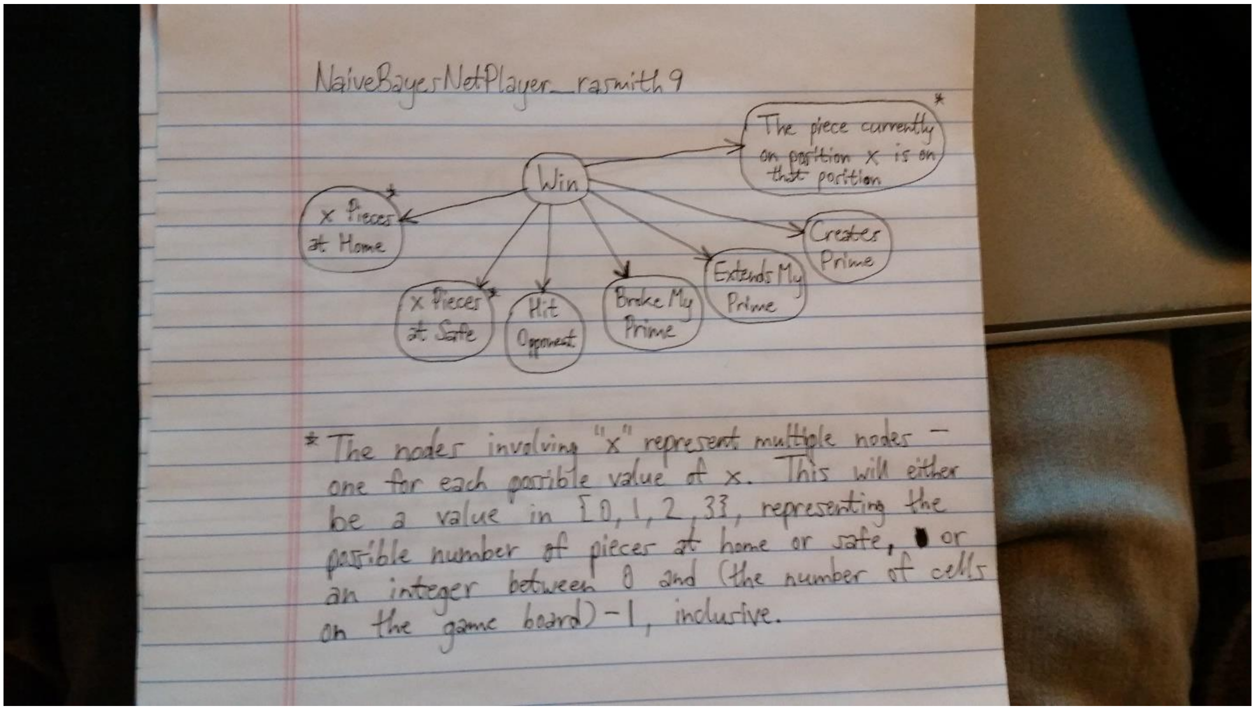
My Naïve Bayes net player uses 14 independent random variables (or arrays of variables), representing (twice – once each for wins and losses):

- The number of my pieces currently at home (whether it's equal to empty, me, or opponent)
- The number of my pieces currently at safe (whether it's equal to empty, me, or opponent)
- The state of each cell on the game board, tracked independently
- Whether the move hits the opponent
- Whether the move breaks a prime of mine
- Whether the move extends a prime of mine
- Whether the move creates a prime of mine

These are each stored as separate random variables. In the case where we win a game, we increment the "win versions" of these variables for the target board state of the move under consideration. Likewise, in the case where we lose a game, we increment the "lose versions" of these variables for the target board state of the move under consideration.

When choosing a move, we apply our Naïve Bayes assumption to calculate the probability that a given move will be a winning move, and then select the best move from among the legal moves in a manner similar to our full joint probability table player.

To calculate the probability for a given move, we multiply the probability that each random variable occurs in a win – for each variable we calculate the probability that it occurred in a winning game by taking the number of occurrences of the variable in winning games over the number of games won in the statistics-gathering phase. Then we multiply all of these together. Then we do the same thing for the variables in losing games. Finally, we take the overall probability of a win and multiply it by our "win product" and do the same for our "lose product". The probability we're after is given by the "win product" over the "lose product".



NaiveBayesNetPlayer_rasmith9

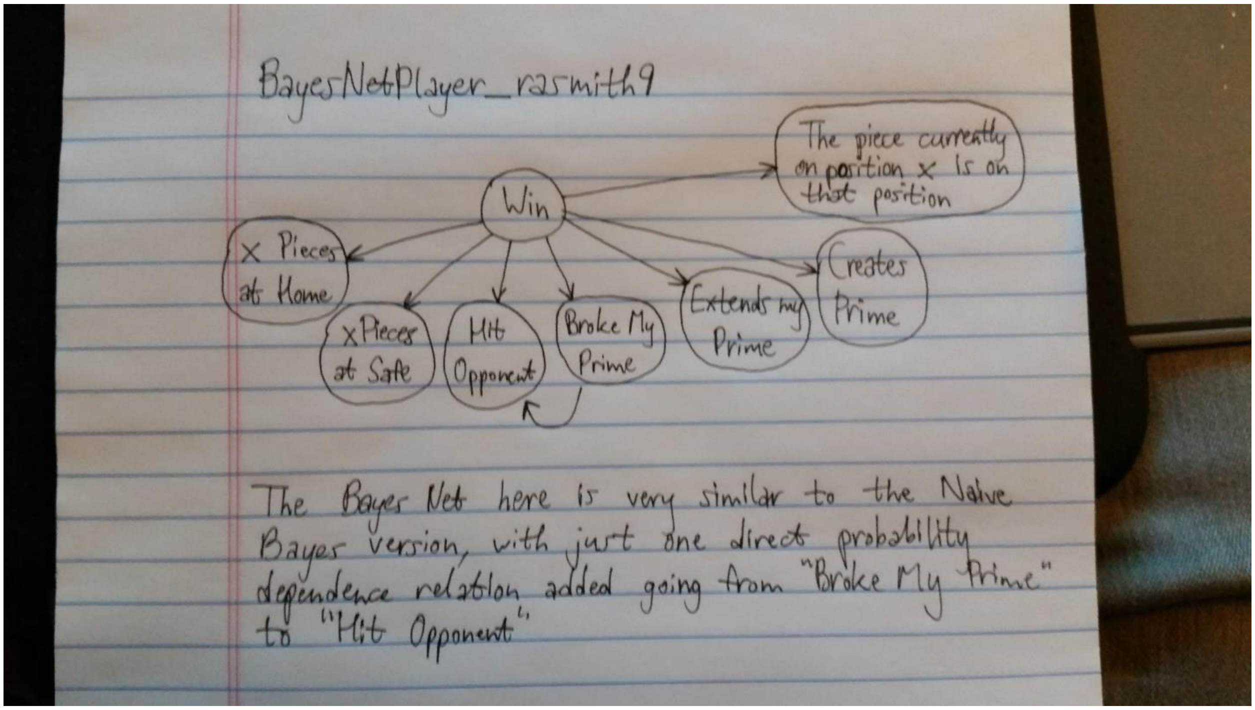
* The nodes involving "x" represent multiple nodes – one for each possible value of x. This will either be a value in {0, 1, 2, 3}, representing the possible number of pieces at home or safe, or an integer between 0 and (the number of cells on the game board) – 1, inclusive.

3.

BayesNetPlayer_rasmith9

The design and behavior of my Bayes Net player is almost identical to my Naïve Bayes net player. The only dependency between random variables I added was one between two move-effects. I theorized that it would be helpful to include a probabilistic dependency that a winning move hit an opponent is impacted by the probability that the move broke a prime of mine.

The resulting chose move calculation uses a product between independent variables similar to the one used by my Naïve Bayes player, the only difference being that we count instances of a move being a hit and a move breaking a prime of mine jointly, and we include the combined probability of a state with respect to these two variables in our product calculations rather than tracking these independently and multiplying them.



BayesNetPlayer_rasmith9

The Bayes Net here is very similar to the Naive Bayes version, with just one direct probability dependence relation added going from "Broke My Prime" to "Hit Opponent"

4.

	MyFullJoint	MyNB	MyBN	HandCoded	Random
MyFullJoint		44.62%	43.42%	49.19%	35.53%
MyNB			48.56%	54.31%	42.10%
MyBN				55.65%	42.98%

My full joint probability table player was my best player, winning more than 50% of games against other players. This seems reasonable since the full joint probability table player retains the most discrete information – given the high volume of burn-in games played and the small size of the game board, this player should perform reasonably well because we have robust statistics covering possible board states.

My Naïve Bayes and Bayes Net players performed about equally well. Considering my design, where my Bayes Net player is “almost naïve”, incorporating only one dependency between random variables (other than from the “win” variable), this is expected. The fact that my Bayes Net player performed worse suggests that my model, which ties two “effect of moves” variable together, wasn’t a very accurate model of the relationship between the probabilities of these random variables as they relate to winning.