Question 6

Data Set	Classification Rate	Tree Size
Dummy 1	1.0	3
Dummy 2	.65	11
Car	.921	408
Connect4	. 765350	41605

Dummy Set #1 and Dummy Set #2 were the same data sets with regard to the number of attributes (10), type of attributes (binary), labels (binary), and number of training and test examples (20 each). Dummy Set #1 performed much better in both the classification rate and tree size than Dummy Set #2. Dummy Set #1 performed extremely well because one attribute, the fifth one, determined the output. None of the other attributes played a role in the decision process. This allowed for the tree size to be as small as possible. Because the test set was a randomly chosen subset of the test set, the classification was easy.

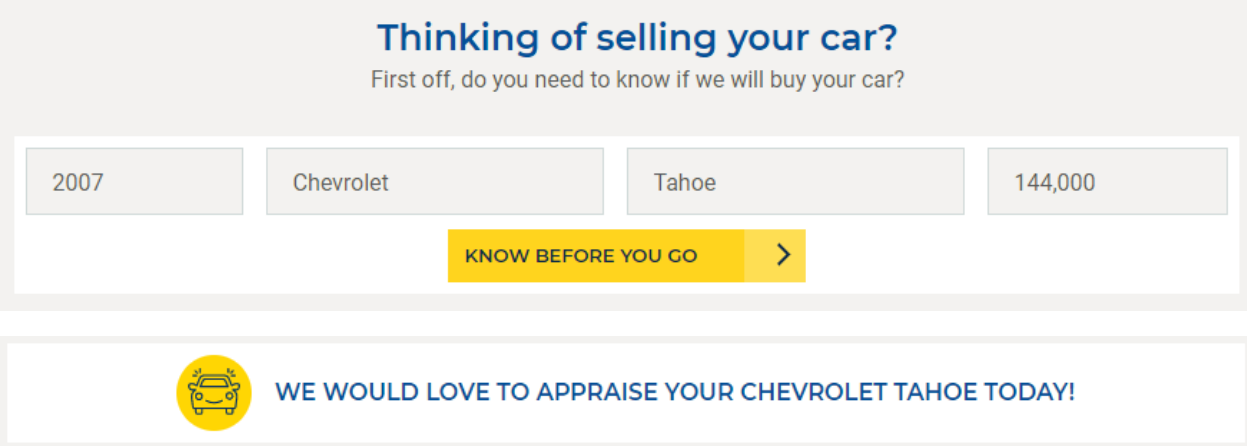
Dummy Set #2 had many more attributes that were important in the decision. Dummy Set #2 was inaccurate because there was not enough training data for the decision tree to generalize well. The small number of examples in the training set did not allow the decision tree to be accurately constructed because there are more meaningful attributes. The higher quantity of attributes that provide meaningful information for the decision increased the size of the decision tree.

The Car data set has a very high classification rate. This is because the data set is not very big, meaning there isn't much noise near the leaves so overfitting is less likely to happen. The decision tree has a relatively small size because there are few attributes, with few values each. Though each attribute is meaningful in the decision, some attributes were much more important than others (much higher info gain) so the classification was easier to make.

The Connect4 data set showed a mediocre classification rate for the decision tree. The training set had an extremely large number of examples so there is a lot of noise near the leaves of the tree. Because of the nature of the dataset, each individual attribute provides extremely little information gain. The combination of the noise and the low info gain is what makes this decision tree have a poor classification rate. The massive size of the tree is due to number of attributes and the fact that the data set represents the 8-ply positions. Each attribute contributes very little to the decision so many attributes are needed to classify an example.

Question 7

The car classifying decision tree is perfect for a used-car website for when they are making a decision to buy someone's car to put it on their lot. Used-car dealers must make the decision of whether the car is worth the time to take a deeper look into. See the below pictures for an example. Once the information is filled in, the program would query its database about the number of persons, its safety rating, the number of doors, etc. to make a decision if they should schedule an appointment to see my car for further analysis. The car in this example (2007 Chevy Tahoe with 144k miles) classifies as an acceptable car.



A decision tree would also be useful for the used-car website to present relevant choices for the buyer. The dataset for this decision tree would contain certain filters that a user would want such as safety, mpg, price, style, etc. The user's preferences would be collected, then applied to a decision tree which would return a recommended model (the classification).

The Connect4 Decision Tree can be extremely useful with the minimax algorithm. The minimax algorithm would determine which move to make next by keeping a score for a possible next move and that move's implications further down the game. The computer tries to maximize the score when it is the computer's turn to make a move, and minimize the score for when it is its opponent's turn. Because the Connect4 search space is massive, the decision tree would be extremely helpful for determining a good move without exploring the entire space. At a certain state (the n^{th} move), the algorithm would create a list of all next possible legal moves (the $(n+1)^{th}$ move) that could be made and then query the $n+1$ ply decision tree to see if that move would result in a win, lose, or draw. Using only the win-labeled moves (or lose-labeled if minimizing opponent's score), it could then find next possible moves from there (the $(n+2)^{th}$ move), query the $n+2$ ply decision tree. This would repeat to a limited depth (as to avoid massive search times) and the minimax algorithm would identify the move that maximizes the probability of winning.

Question 8

Standard Decision Tree

Data Set	Classification Rate	Tree Size
Poker Hand	.614	33813

Pruned Decision Tree

Data Set	Classification Rate	Tree Size
Poker Hand	.6038	3841

Question 6

The dataset I used is intended to classify a poker hand based on the cards in hand. There are 10 attributes, suit and rank of each of the 5 cards in hand. The label was the poker hand: nothing, one pair, two pair, all the way to royal flush.

This dataset performed poorly with both the decision tree and the pruned decision tree. This is likely due to the nature of the dataset. Because a two-pair hand could be {p, p, x, x, x}, {p, x, p, x, x}, … {x, x, x, p, p} with p being the rank of the pair. 13 different ranks and each pair could have a variety of combinations of suits, creates an extremely large variety of combinations for a pair (2860). No single attribute consistently gives meaningful data for the decision tree to make an appropriate classification.

The number of attributes and the large number of values per attribute made the tree size extremely large. After pruning the tree, the tree size is reduced to a tenth of the size.

The pruned decision tree and normal decision tree have about the same classification rate. This makes the pruned decision tree a better option because of the speed at which it can classify hands.

Question 7

This decision tree would be extremely useful for an online poker game. For each “computer” player in the game, the algorithm could be run on their cards to analyze what they have. This helpful because the poker hand is what determines betting strategy and, ultimately, who wins. Because of the poor performance of this classifier, a hard-coded option for the poker hand classifier would likely work better.