Evaluating Model

CS: 4990:0005

Professor Denise Szecsei

Isaac Phillips

Models

With the amount of information, we were able to produce with the simulator it was a bit overwhelming to start. I wanted to use as much as possible from the simulator, but I had to consider what Cosmo will know in a game of blackjack. The true count is hard for a human to determine but for a robot its easier and can be much more accurate. While it is important to note that I have yet to include count into my model; I am topics one, so it’s not required but that’s not stopping me from trying to get a model with count. I think it would improve the accuracy in the end as well so only positives.

I first started creating my first model with Dylan and Isaac. I was having problems with my csv in the beginning but quickly solved that and got into pandas. I took a panda’s class previously, so I had some old files with the train, test, split. Train, test, split, is a function in pandas that allows me to split the data from the csv file we previously made with the simulator. So, with old files pulled up, I look at my csv and the information Cosmo would know and be able to use. In the first iteration of model making, it was just creating data frame to read the csv, pulling the right information from the csv, and then run train, test, split, which we can use the results to predict what action should have been taken in the simulator. In my first models’ iteration I passed ‘dealer\_card’, ‘player\_initial\_hand’, ‘hit\_stay’, ‘count’, ‘dealer\_bust’, ‘dealer\_value’, and ‘outcome’. Dealer card was passed because it is a key part of blackjack and the players decision. We are not doing any logic for decision making here but we need it for accuracy. Then the player initial value, this is key because we must see what our first two cards were and compare to outcomes to see if it was the right choice or not. With hit, stay parameter we can compare that to the players initial hand and the outcome to see if the ‘hit\_stay’ should be a ‘1’ meaning hit or ‘0’ meaning stay. I think count behaves in a similar behavior but haven’t had any luck getting it to work well with my graphs that are produced. So, I don’t have much to say for count/true count as of right now. However, I am trying to digest it like hit and outcome, so I can see how accurate count is on its own. Dealer bust and dealer value were two mistakes I had in my first iteration, there wasn’t much I really did with these two columns only because after I passed them, I realized that I don’t know the second card of the dealer yet, only the first. Which leads into the dealer bust; since we don’t know the second card of the dealer, we don’t know if the dealer bust or not since his turn has been done yet. We are focused on the accuracy of the decision the simulator made during the player’s hand. So, we have only the information when it’s the players turn.

That helped me with the second iteration, as well as the basic model the professor posted on ICON gave me more of an idea of what I was really supposed to do. The base model produced an ROC curve to display the predictions from the train, test, split functions. The base model adds five layers to the model one at 16, 128, 32, 8, and 1. I messed around with these numbers for a bit but found that not much changed, as well as my understanding for the layers. The model called for 10 epochs as well. However, I raised it to 20 after finding a recommendation to increase the epoch size while the batch size is lowered. Yet, I kept the batch size the same since when I ran the program it resulted in more attractive predictions. I’m still messing around with the number of epochs there are and the size of the batch to see if there is a combination better to use. Then finally the program plots the ROC curve which looks something like these:

A graph of a curve

Description automatically generatedA graph of a line

Description automatically generated

The one on the left is a better run than the curve on the right. The shows that, in the beginning of the model run, it was positively predicting what should have been played in the simulator. My logic for the simulator was a bit more aggressive than conservative and it somewhat shows in the csv as well as these graphs. The graph on the right was just as common as the graph on the left yet, we see a significant difference. The right graph shows that it gets the prediction right around 50% of the time, which is significantly worse, a whole 20%. With how these graphs turned out, I wasn’t a fan of how much variance there was between runs in these graphs so the addition I made was to get the true accuracy of the predictions using the “predict\_y\_test” created from the train, test, split function previously stated. Assuming that the data is binary, 1 or 0 for hit or stay respectively, actuals from the function as well, and a threshold range from 0 to 60, we can plot a line graph that will show us the actual accuracy of the predictions based on the threshold. With all of that and getting the maximum and minimum, of the actual accuracy using “metrics.accuracy\_score” after looking up the metrics with the suggested link and appending it to the new list. The line graphs at first were a bit alarming but it was because I had set the range wrong, a bit higher than it should have been. Those first graphs looked something like this:

A graph with blue dots

Description automatically generated

Right out of the gate we can see that the data is just a little high and needs to be adjusted lower to see the full graph. That looks like:

A green line graph with white text

Description automatically generated

As you can see the curve of this line graph is like the ROC curve however, it plateaus at the .2 threshold area and reaches its best of 89.11% accuracy near the end of the graph. The labels were easy to come up with, I wanted to get the accuracy of the predictions, thus the title. The x-axis is the thresholds, for understanding where in the model it starts to improve or struggle with the predictions. The y-axis is the prediction precent. What percent of the predictions are accurate at a given threshold. I did begin to work with the confusion matrix, but it would a little concerning so for that purpose I’ll only include the one to show you what it is I am dealing with. A red squares with numbers and a white background

Description automatically generated

Its saying that the model is only predicting 44 accurate hits, and 18 stays that should have been hits. In other words, it predicted we hit but instead the simulator stayed. I think I have found the issue; however, it only works every so often, so I am still working out the kinks to the program.

Overall, I found it confusing in the beginning. But the tips Isaac and Dylan gave me to work with the panda’s train, test, split function, and me previously having worked with that helped get a better understanding. I think I might be able to get a contender for casino night and hopefully go up against those in topics 2 with ease and get the count incorporated into the model as well.