Hugo Escobar

20 March 2022

Probability and Applied Stats

The first program that I created was the function plotter. The purpose of this plotter is to print X and Y values into a csv file. It takes the X value and puts it into a formula. Then it outputs the Y value. The formula I chose was very simple, I chose the slope intercept form. It then takes these two values, X and Y, and puts it into a csv file. You can choose how many X values you want by using the parameter in the method call. In this case, I chose 1000 as the amount of trials being ran.

The next program that was created was the function salter. What this program does is take in a csv file, read each line, change the Y value of each line, then print it into a new csv file. The first step it does is use a scanner to read the file. It reads the csv file that was created using the function plotter. After opening the initial file, I used a loop that continued as long as there was a new line after the current one. Within that loop I decided to locate the comma within each line. I figured that using that method was a good one since it is the one character in every line that does not change. By doing that I was able to set variables that were equal to whatever was before the comma, the X value, and anything after the comma is the Y value. It had initially saved them as strings so I used a method that converted them to doubles. After that, the Y value was randomly chosen to get the user inputted bound either added or subtracted to it. I then set the new Y value to that answer. After all of that I then created a new csv file containing the original X value and the new Y value.

The next program was the function smoother. What it does is find the average of a certain amount of Y values at the current one and sets a new Y value to that average. It uses the csv file from the function salter. It reads the Y values and places them in an arraylist. After that, depending on the user inputted bound, it finds the average of Y values based on what is placed in the bound parameter. If it is 10, then it finds the average of 10 values in front of the current placements and 10 behind it. After that, the new Y value is that average. However, I was not able to fully implement this. So what I actually did was just found the average of the total y values as it assign each one. It is then placed in a new csv file and printed there. You can choose how many X values you want based on the parameter you place.

The next thing that was created was the updated stats library. There were five distributions added to this library. The first being the binomial distribution. This distribution is used when there is a fixed amount of trials. It takes in 4 variables, the total which is N, the probability of success which is P, the probability of failure which is Q, and what is being asked is Y in this distribution. By finding the combination of N and Y then finding P raised to the Y and Q to the power of N minus Y then multiplying it all will give you the probability of whatever you are looking for.

The next distribution is the geometric distribution. You would use this distribution when you’re trying to find the amount of failures until a success. This distribution only take three variables. It takes the probability of success, which is P. It takes the probability of failure, which is Q. And what you are trying to find is considered as Y. I then did Q to the power of Y minus one multiplied by the probability of success, P. That would then give you the probability of a success after a certain amount of failure.

The next distribution that I implemented in the stats library is the hypergeometric distribution. You would use this when trying to find something specific within a set. This distribution uses four variables. The set total being N, the subset being n, the amount of certain items you are looking for being r, and y being what you are choosing. An example of this could be 20 total marbles, 5 marbles chosen at random, 8 red marbles, and y being the probability of getting all 5 marbles of being red. I first found the combination of r and y. Then the combination of N minus r and n minus y. Then the combination of N and n. Then the first two combinations get multiplied. After that I divide the two multiplied combinations by the final combination.

The next distribution is the Poisson distribution. You would use this when trying to find the amount of something happening within a certain rate. An example could be how many miles being driven in 2 hours when driving 60 miles per hour. It only uses two variables, one being Y and the other being lambda, the rate in this formula. I then found e to the negative lamda times lambda to the power of Y divided by the factorial of Y.

The next thing that was implemented was Chebyshev’s theorem. You use this when you are trying to find the probability of something occurring in between two bounds. The method takes in four variables. The standard deviation, the mean, the upper limit, and the lower limit. First it compares the upper limit minus the mean and the mean minus the lower limit. If those two answers are the same, then k is equal to the upper limit minus the mean divided by the standard deviation. Then I did one minus one divided by k squared. That then gives the probability.

The next and final program was the poker hand tester. This program creates 52 card, each with their own suit and number assigned to it. In a separate class, a hand of card is assigned. In this hand there are five random cards assigned to it. I then made a pair method that compares the different placements in the arraylist to see if there is a pair. I then made a three of a kind method that uses three loops to compare different placements within each hand. After that I made a four of a kind method that works similar to the three of a kind except it uses a fourth for loop for an extra card comparison. After that method I made a method that finds two pairs in a hand. It works the same way as the single pair method but if it returns 2 then it is two pairs but if it is 1 then it is one pair. When it came to the method that checked if there was a straight, all I did were two if statements that compared the numbers that were next to each other to see if they increment or decrement. After this method, I realized that maybe a method to check two cards numbers might be useful. I also made a similar method that compared suits of two cards. Then using these two methods, I made one that checked if a hand has a flush. By using the suit check method, I compare every card that was next to each other and if they are all true then a true value is returned. This next part is where it gets a bit tricky. Making the full house method made me realize that there were different ways a full house could be made. So instead of just making one large beefy full house method, I broke it into two different ones. Let’s first start with the front house method. What this method does is run three for loops to check if there is a three of a kind. If there was a three of a kind, I then ran two for loops within the if statement to find a pair. If these criteria are met, the pairs variable is set to true. Next is the backhouse method. This works similarly to the front house method. It first runs two for loops to check if there is a pair. Then an if statement is ran that continues more loops if it is true. So if it is true then three more loops are ran to see if there is a three of a kind following the pair. If there is a three of a kind then it sets the pairs variable to true and if not then it is false. Then it all goes back to the original full house method that I initially created. Within that method, I made an if statement that if the front house method or back house method are true then a full house is in the hand. I had originally tried to use arraylist in some way but could not figure it out so I did this method in the best way I knew how. I had also made sure to put many comments within my code as well.