Probability & Statistics Project 3

Arvind Pawar

November 21, 2018

Probability Distribution is a function which is used to either generate or describe all likelihoods and possible values, which is taken by random variables in a given range. By understanding the dataset, we can plot the data and then plot various distribution and see which one most closely fits.

Central Limit Theorem is an important concept in Inferential Statistics which shows us how the means of the samples of the same size taken from the same population vary about the population mean/

We have a given a scenario of ancient Chinese game. In one electronic version of this game, a player selects 20 numbers from the set of numbers 1 through 100. The computer then randomly draws another set of 20 numbers from the set 1 through 100, and the player is rewarded according to how many of his selected numbers have been matched by the 20 numbers drawn by the computer. Let X be the number of matches between a player's 20 selected numbers and the 20 numbers drawn by the computer. Then X may range from 0 (no match) to 20 (all match) and follows a hyper-geometric probability distribution. I have done this project in R programming.

Part 1

a. Construct Tabular Distribution for X. Introduction: To start the analysis of this data set, I have to calculate the probability for the number randomly picked by a user to be matched with a number randomly picked by computer and is calculated by the concept Hypergeometry Probability distribution function. Hypergeometry distribution is a discrete probability distribution that describes the probability of k successes which are random for which the object has drawn and from a finite population of size N that contains exactly K objects with that feature, and has a specified feature in n draws without replacement, wherein each draw is either a success or a failure.
dhyper(c(0:20), m=20, n=80, k=20, log = FALSE)
In the above code I have used dhype(x, m, n, k, log = FALSE) function which will generate hypergeometry distribution.

x = Range of number of matches of the numbers picked by player and computer

m = Numbers picked by player from the numbers 1 to 100

n = Remaining numbers

k = Numbers picked by computer

Analysis:

The player picks 20 numbers randomly, and the computer picks the 20 numbers randomly, and this sampling has done without replacement. So, we have used Hypergeometry Distribution. In Hypergeometry distribution the probability of success changes on each draw, as each draw decreases the population because the sampling was without replacement. In the given scenario we can see that there is more possibility that player will score 4.

b.
Construct cumulative distribution for x

By using **phyper(c(0:20), m=20, n=80, k=20, log = FALSE)** I generated cumulative probability distribution.

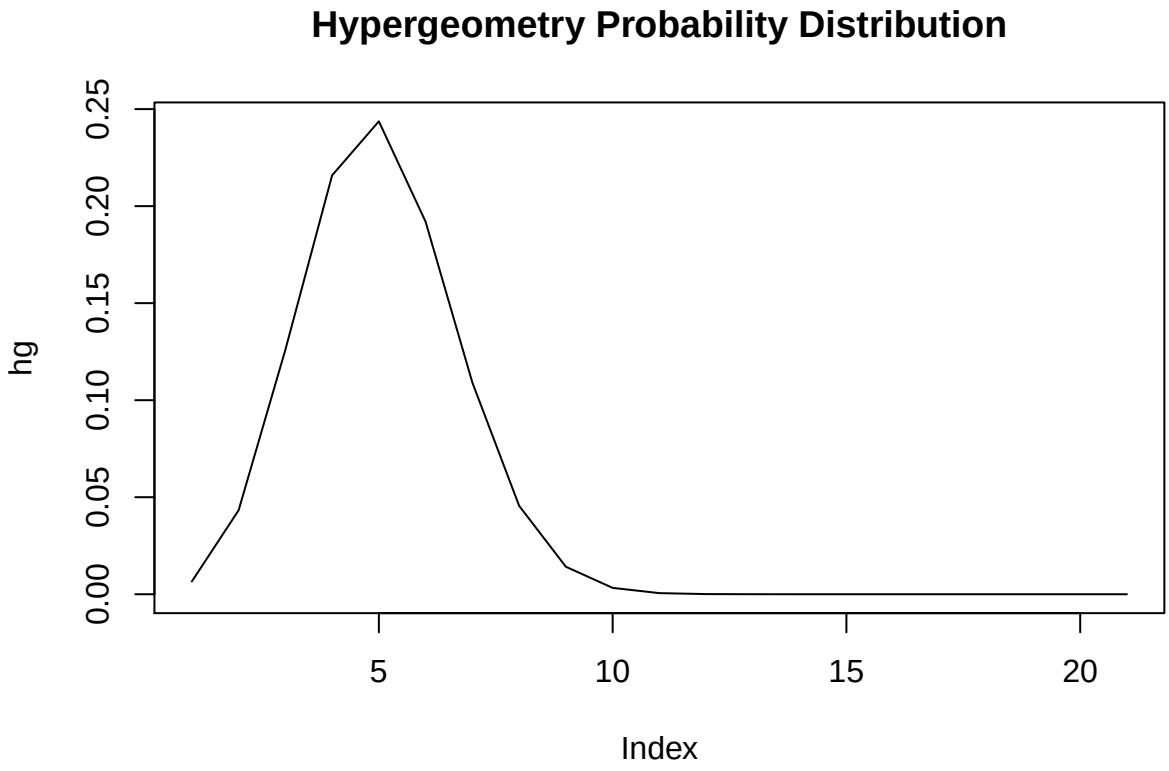
x- Number of Matches	P(X = x)	P(X <= x)
0	0.0065959	0.0065959
1	0.0432521	0.0498480
2	0.1259194	0.1757674
3	0.2158618	0.3916292
4	0.2436877	0.6353170
5	0.1919510	0.8272680
6	0.1090630	0.9363310
7	0.0455786	0.9819096
8	0.0141595	0.9960691
9	0.0032834	0.9993525
10	0.0005676	0.9999200
11	0.0000727	0.9999927
12	0.0000068	0.9999995
13	0.0000005	1.0000000
14	0.0000000	1.0000000
15	0.0000000	1.0000000
16	0.0000000	1.0000000
17	0.0000000	1.0000000
18	0.0000000	1.0000000
19	0.0000000	1.0000000
20	0.0000000	1.0000000

c.

Introduction:

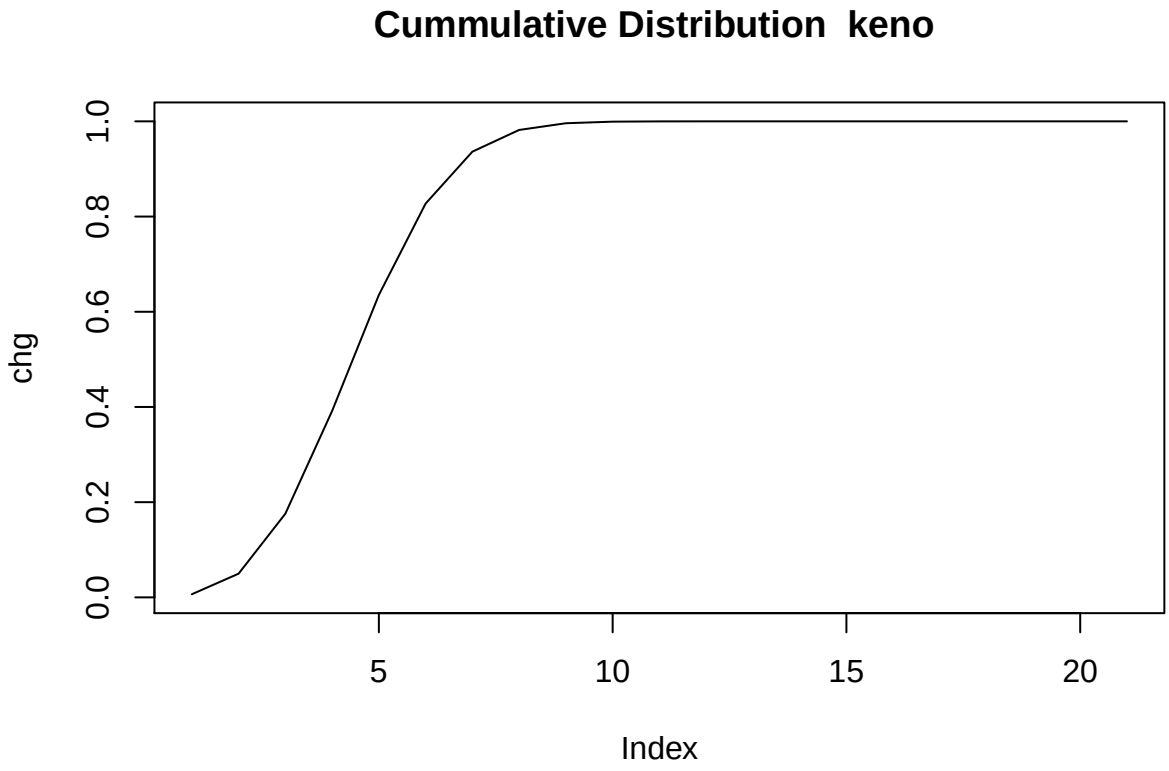
I have graphed Probability Distribution based on the values generated by Hypergeometry distribution. by using plot(dhyper(c(0:20), m=20, n=80, k=20, log = FALSE)), graphed the Hypergeometry Probability Distribution. **Analysis:**

From the Hypergeometry Probability Distribution we can see that probability of getting score 8 or more is almost 0. There is more probability that 4 numbers will get matched with the numbers drawn by computer. The distribution right skewed.



d.
Introduction:

I have graphed Cumulative Frequency graph based on the cumulative frequency distribution. Used plot(phyper(c(0:20), m=20, n=80, k=20, log = FALSE)) function to create graph.



e.

Introduction:

In this step, I have asked to calculate the theoretical mean, variance and standard deviation. * So, to calculate the theoretical mean I have to calculate the sum of the product of the number of matches and its corresponding probability distribution, so I used **sum(hg*c(0:20))___ function. 'hg' is variable where I stored Hypergeometry Distribution and c(0:20) is a count of matches from 0 to 20. The Theoretical mean is 4.***

To calculate the variance the formula is

$variance = \sum(x-\mu)^2 * P(X = x)$___ where x is the number of matches,

$$\mu$$

is theoretical mean and P(X=x) is probability distribution.

* To calculate variance, I executed below snippet,

___hg.var<- sum((c(0:20)-hg.mean)^2*hg)___

Assigned a variable hg.var and stored the value of variance in that variable.

Then, I calculated standard deviation which is square root of variance.

hg.sd <- sqrt(hg.var)

Theoretical Mean

```
## [1] 4
```

Theoretical Variance

```
## [1] 2.585859
```

Theoretical Standard Deviation

```
## [1] 1.608061
```

Analysis:
The calculated values are theoretical values which are being calculated using formulas. From mean we can interpret that there are 4 numbers which can get matched with the numbers picked by player and the same by computer. After getting the mean I calculated the variance which tells us the spread between numbers in a data set.

f.
Introduction: In this step I am going to Generate 1000 random values according to standard uniform probability distribution. I used runif function to generate 1000 uniformly distributed random values.
rv <- runif(1000)
Analysis:
Many elements of statistical practice depend on randomness via random numbers. By generating this randomness, we can simulate the number of experiments which helps to predict what can be the results in some situations based on that we can take decisions. Which is creating a scenario of assuming the possibilities of getting the scores in Keno Game.
Conclusion:
Here I have generated 1000 random values based on standard uniform probability distribution which are similar to the probability distribution.
rv <- runif(1000)

g.
Introduction:
I am going to simulate the probability distribution with the 1000 uniformly distributed random numbers. I wrote the following code,
a <- vector(mode='double', length=1000L)

k <- 1

for (x in rv){

a[k]<-min(which(chg>=x))-1

v<-v+1

}

First, I created a vector of type double because the values are in fractions. v is a counting variable. rv is variable where I stored uniformly distributed random values. And then I used for loop.
Analysis:
It has generated the 1000 simulations with respect to Cumulative Distribution and 1000 uniformly distributed random values. I observed that number that are occurring more number of times has high probability.
Conclusion:
This step has generated the values according to their probability distribution.
h.
Introduction:
As we have generated a pattern based on the calculated probabilities is an experiment. Now, let's calculate the mean, variance and Standard Deviation of these experiments. 'a' is the vector where I have stored the experimental values and I used **mean(a), var(a), sd(a)** to find the experimental mean, variance and Standard Deviation.
Analysis:
The values of theoretical and Experimental mean, variance and standard deviation are almost similar.

	x- Number of Matches	Experimental (Simulated)
Expected Value of X	4.000000	3.982000
Variance of X	2.585859	2.616292
SD of X	1.608060	1.617496

i.

Now let's calculate the mean successively from 20, 40, 60, 80, 100, 200, 300, 400, 500, 600, 700, 800, 900, and 1000 interval.

I wrote below code to calculate mean.

```
sim.mean<-vector(mode='double', length=14)
k<-1
sim<- c(20, 40, 60, 80, 100, 200, 300, 400, 500, 600, 700, 800, 900,1000)
for (x in sim){
sim.mean[k]<-mean(a[1:x])
k<- k+1
}
```

It has calculated the mean of first 20 experimental values of x and then first 40 values and so on. I stored the values of means in vector 'sim.mean'.

The simulated means are:

```
##  [1] 4.300000 4.400000 4.200000 3.987500 3.930000 3.975000 3.913333
##  [8] 3.930000 3.954000 3.956667 3.975714 3.982500 4.003333 3.982000
```

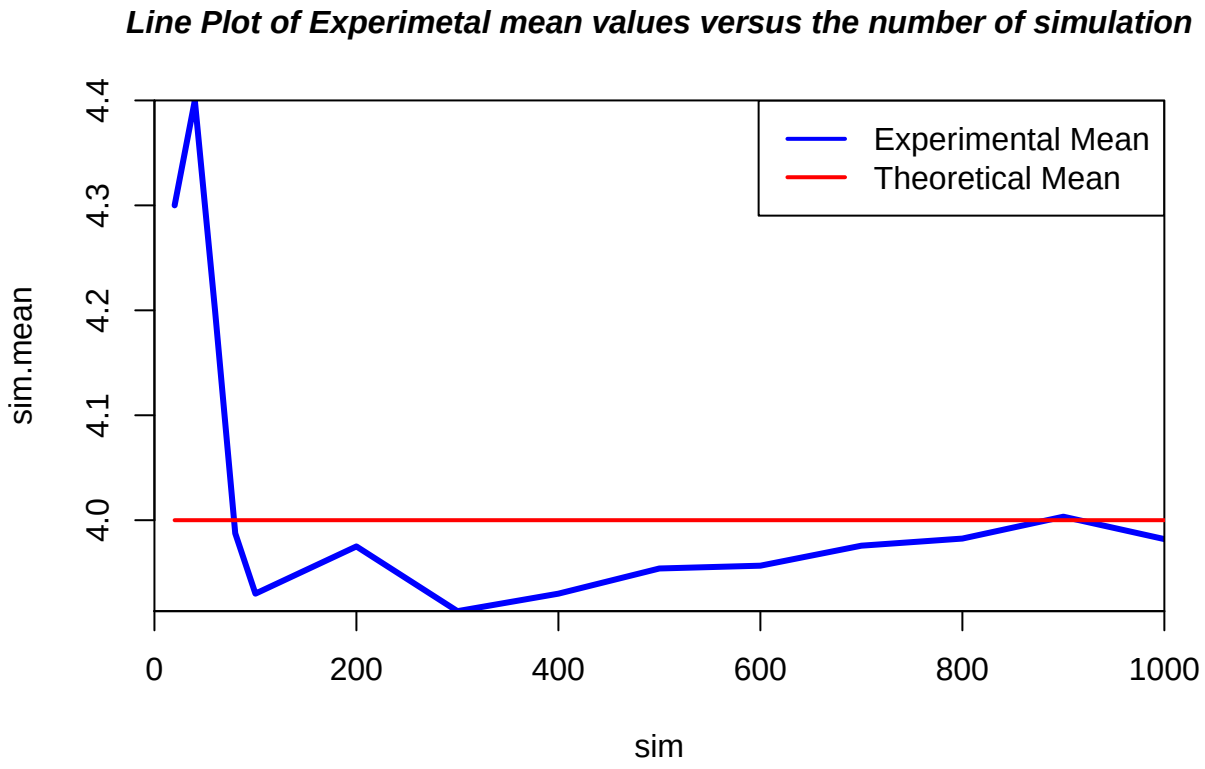
j.

Introduction: After getting the mean values of experiment intervals, I am going to represent those values with the theoretical mean I have calculated earlier. To plot the graph, I used following code.

```
plot(sim, sim.mean, type="l", lwd=3, col="blue", xlim=c(0,1000), xaxs="i", yaxs="i")
lines(sim,th1.mean, lwd=2, col="red")
legend("topright", legend=c("y","z"), lwd=c(2,2), col=c("blue","red"))
```

Analysis:

The red line represents the theoretical mean, and the blue line represents experimental interval means. We can see the blue line starts from the top and then drops and comes near to that of theoretical mean. As the number of experiments increases the line of the experimental means overlaps the line of theoretical mean. Which is proving the 'The Law of Large Numbers' which states that if we do the experiments in large number, the experimental values become closer to theoretical values.



Part 2

a:
Introduction:
I have given a normal population in worksheet. I have created the csv file of that data and read file in R to calculated Mean, Variance and Standard Deviation. **Edata<-read.csv("C:/Users/Arvind/Desktop/Probability Theory & Statistics/Discussion Week 3/Project 3/Edata.csv",header=FALSE)**
Edata<-unlist(Edata)
typeof(Edata)
mean(Edata)
var(Edata)
sd(Edata)
To calculate mean, variance and standard deviation, I did above steps.

The values are as follows.
Mean

```
## [1] 49.16
```

Variance

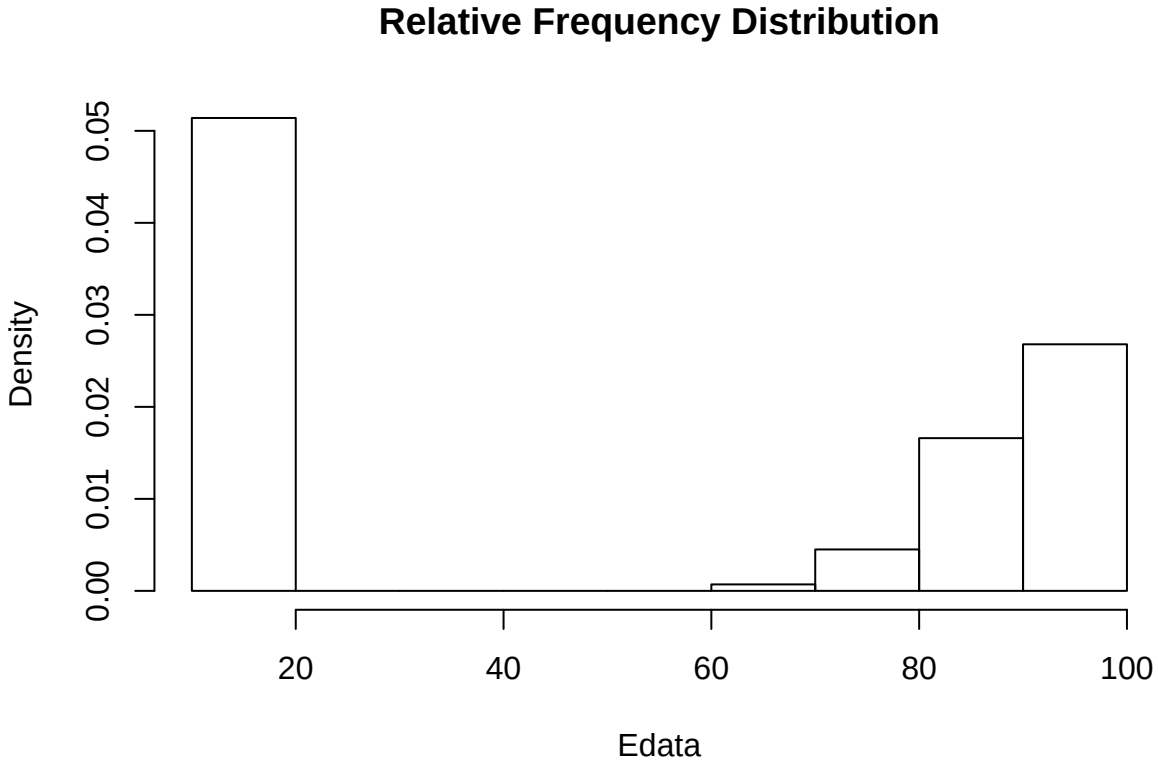
```
## [1] 1610.111
```

Standard Deviation

```
## [1] 40.12618
```

Analysis:
From Standard deviation 40.12618 we can conclude that we can cover almost 80% population if we go one standard deviation away from mean to left or right.
b.

Introduction:
Now I have to construct a relative frequency for the given population. To construct the relative frequency Histogram I used hist(Edata, freq=FALSE), This function in R, automatically calculates relative frequency and plot the histogram.
Analysis:
From the shape of the histogram we can say that it is not random, most of our population set is between range 0-20 and 5% of the population is between 90-100.



c:
Introduction:
I have to generate 30 samples of 30 random numbers each from the Normal Population. For this I wrote below code.

```
list30<- vector(mode='list', length=30L)
count<-1

for(val in c(1:30))
{
list30[[count]]<- sample(Edata,30,replace=FALSE)
count<-count+1
}
```

I have created vector list30 which will store the 30 samples. Sample function will take 30 random numbers from normal population without repetition and have put it into for loop to generate 30 samples.

d:
Introduction:
Now I have to calculate the mean, variance and standard deviation for each sample. I have calculated it using **list30_mean <- unlist(lapply(list30, mean))**, have created variable list30_mean, where I have store the mean of the sample. I used lapply function which calculates the mean of each sample at a time.

After executing this function, I got mean values of all 30 samples. I applied the unlist function to convert the values of samples into integers because the samples are in the list and we cannot make any calculations on a list. Similarly, I followed these steps to calculate variance and Standard deviation of all the 30 samples.

list30_var <- unlist(lapply(list30, var))

list30_sd <- unlist(lapply(list30, sd))

Analysis:

As per my observation I see all 30 samples have almost nearby values of means, variances and standard deviation are almost similar. The mean of the samples represents the whole population from where we have taken samples.

e:

I am going to calculate average of means, variances and standard deviations.

final_mean <- mean(list30_mean)

Var_mean <- mean(list30_var)

sd_mean<- mean(list30_sd)

f:

Introduction:

I am going to comparing the mean, variance and standard deviation of samples and the same of the population from where we have taken samples.

Analysis:

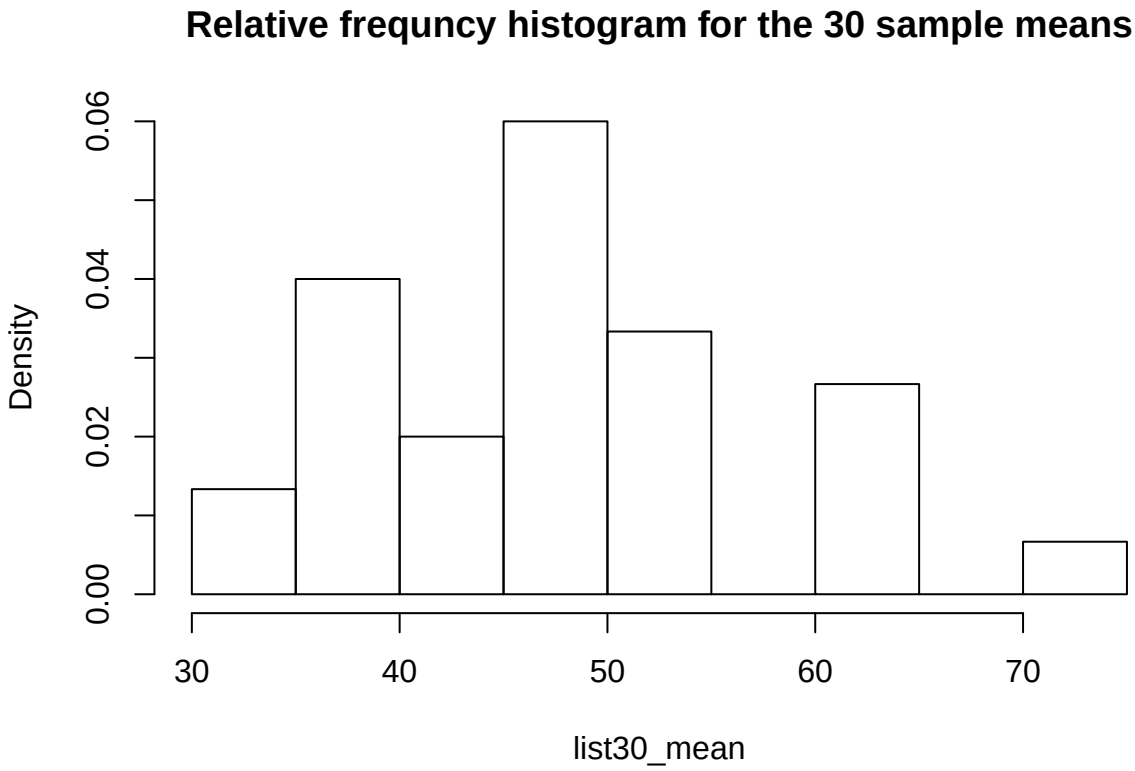
All these values are almost like the population mean, variance and standard deviation., this is a proof of one of the points of Central Limit Theorem, that sample taken from the population will have a similar mean, variance and standard deviation to that of the whole population.

g:

Now I am going to create relative frequency distribution of means of 30 samples.

Analysis:

The shape of the relative frequency looks like a normal distribution, not exactly but alike, which is proving the other point of Central Limit Theorem which states that If samples taken from population are of a large enough size, then the distribution of their averages will approximately be a normal distribution.



h:
From all the above steps and analysis, I can conclude that if we select a large number of samples from the population, the mean, variance and standard deviation will be similar, and the distribution tends to be normal. When we select a large number of samples, it decreases the risk of outcomes.

I have done the module 3 project in R, and I got comfortable with a portion of the capacity in R programming. This undertaking gave me a chance to build up a logic while doing a few stages like producing 1000 uniformly distributed random numbers with cumulative distribution, creating 30 tests from a normal population. I additionally came to realize that how random numbers help to extend tests likewise, the tasks performed all in all samples and population comprehends the ideas of as far as a possible hypothesis. The qualities we ascertained hypothetical and the qualities we got from trials were relatively comparative which demonstrates 'The Law of Large Numbers', as we have done the experiments number of times utilizing uniformly distributed random numbers and cumulative distribution.

References:
1. (n.d.). Retrieved from https://www.cliffsnotes.com/study-guides/statistics/sampling/central-limit-theorem
2. R Language what is difference between rnorm and runif. (n.d.). Retrieved from https://stats.stackexchange.com/questions/49370/r-language-what-is-difference-between-rnorm-and-runif
3. (n.d.). Retrieved from https://web.ma.utexas.edu/users/mks/statmistakes/skeweddistributions.html
4.Skewness. (2018, November 25). Retrieved from https://en.wikipedia.org/wiki/Skewness
5. Stat Trek. (n.d.). Retrieved from https://stattrek.com/probability-distributions/hypergeometric.aspx
6. Hypergeometric distribution. (2018, October 25). Retrieved from https://en.wikipedia.org/wiki/Hypergeometric_distribution#Multivariate_hypergeometric_distribution
7. https://northeastern.acrobatiq.com/courseware/NEU_MPSA_PROB_THEORY_INTRO_STATS_ALY6010_V2_2/week_3_-_probability_distributions_and_the_central_limit_theorem___sampling/module_overview___probability_distributions_and_the_central_limit_theorem___sampling/wbp_module_overview___probability_distributions_and_the_central_limit_theorem___sampling