Ajay Kc
013213328
Project 5 Report

1. Effect of sample size on confidence intervals

Introduction

The experiment revolves around the concept of taking a sample and comparing it's mean to the mean to the main population. The experiment seeks to check the effect of sample size on the two confidence intervals i.e. 95% and 99%

Methodology

A total of 1,000,000 random numbers are generated with the mean of 75 and standard deviation of 7.5. From the generated list, sample of various sizes ranging from 1 to 200 is taken. For each sample, the mean of the sample is calculated and plotted on the graph. Two graphs with the same plot is created. The first graph checks the 95% confidence interval while the second one checks the 99% confidence interval.

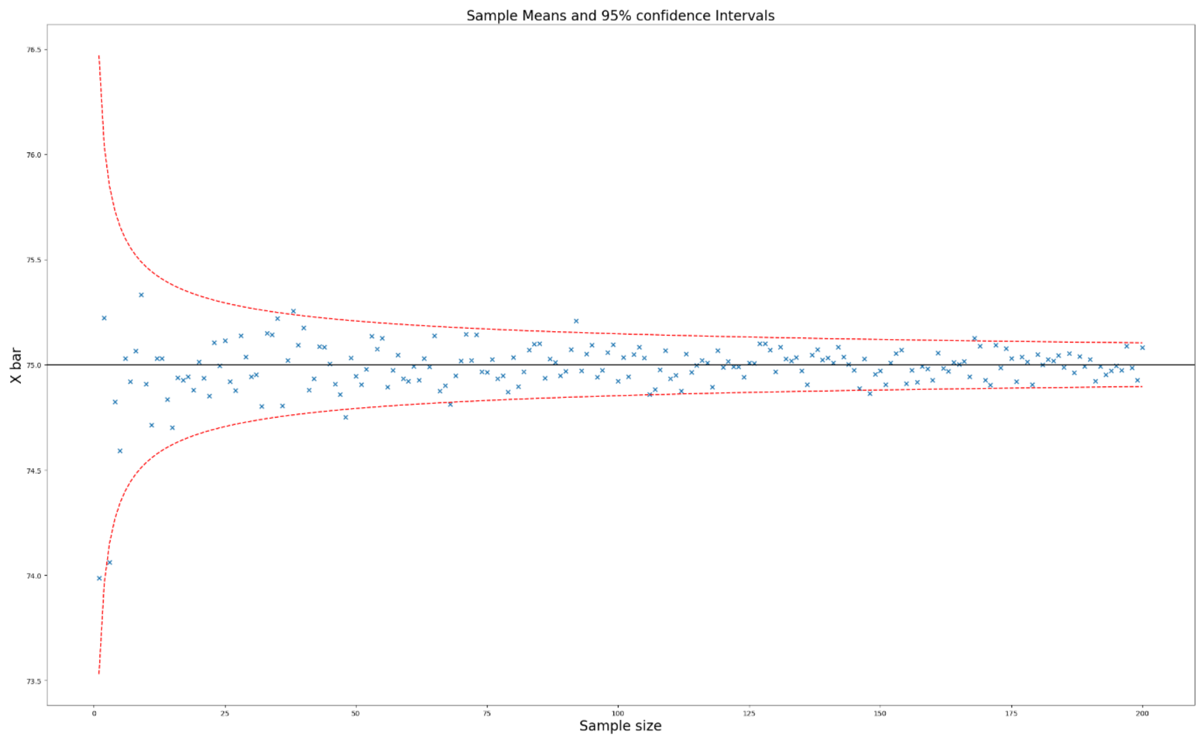
Source code

```
"""
Ajay Kc
013213328
EE381
Project 5 Part 1

The problem plots two plots showing the effect of
confidence intervals on sample size.
"""
import numpy as np
import matplotlib.pyplot as plt
import math as math
mean = 75
sd = 0.75

#method to generate 1000000 numbers with mean 75 and
standard deviation as 0.75
def generateNumbers():

    Nlist = np.random.normal(75,0.75,1000000)
    return Nlist

#method to generate a sample of size n from the list
def generateSample(list,n):

    sample = []
    sample = np.random.choice(list,n)
    return sample

def plotGraph(meanList,confidenceInterval):

    plt.xlabel('Sample size',fontsize=20)
    plt.ylabel('X bar',fontsize=20)
    plt.title('Sample Means and
'+str(confidenceInterval)+"% confidence
Intervals",fontsize=20)

    #plot a straight line through the mean
    plt.axhline(y=75, color='black')

    #plot the sample
    plt.scatter(range(1,201),meanList,marker='x')

    #plot the confidence interval
    e=0
    if(confidenceInterval==95):
        e = 1.96
    else:
        e = 2.58
    confidenceIntervalList1 = []
    confidenceIntervalList2 = []
    for i in range(1,201):

confidenceIntervalList1.append(mean+e*(sd/math.sqrt(i)))
        confidenceIntervalList2.append(mean - e * (sd /
math.sqrt(i)))

    plt.plot(range(1,201),confidenceIntervalList1,'--
',color='red')
    plt.plot(range(1, 201), confidenceIntervalList2, '--',
color='red')

    plt.show()

bearingList = generateNumbers()
meanList = []
standardDeviationList = []
for i in range(1,201):

    sampleList = generateSample(bearingList,i)

    meanList.append(np.mean(sampleList))
    standardDeviationList.append(np.std(sampleList))
```

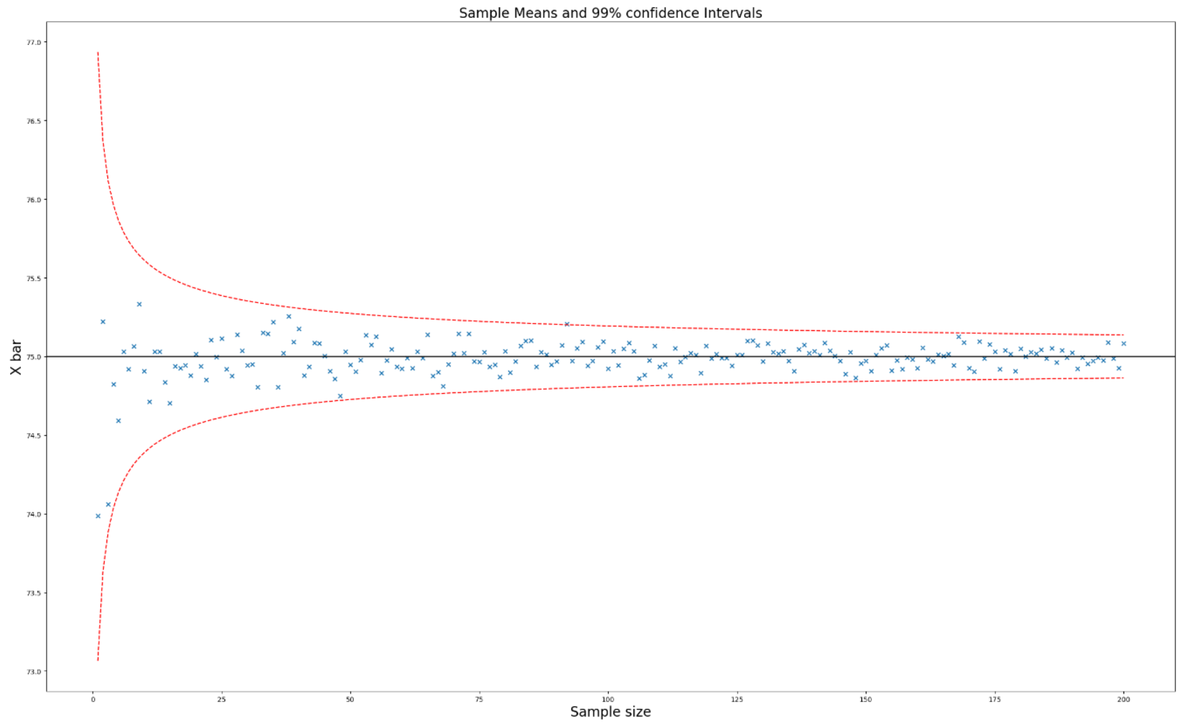
```
plotGraph(meanList,95)
plotGraph(meanList,99)
```

Plot of sample mean as a function of sample size with confidence interval

95% confidence intervals



99% confidence intervals



Ajay Kc
013213328
Project 5 Report

2. Using the sample mean to estimate the population mean

Introduction

The experiment also deals with the effect of sample size on the confidence interval. The experiment aims to check the accuracy of mean of a sample compared to the mean of the entire population. The accuracy of the mean is calculated based on the 95% and 99% confidence interval using the normal distribution as well as the student's t distribution.

Methodology

Similar to the first experiment, 1,000,000 random numbers with mean 75 and standard deviation 7.5 is generated. Then, a sample is taken from the list. The sample sizes for the experiment are 5,40,120. For each sample size, 10,000 samples are taken and the mean is calculated. Then, the mean of each sample is compared across difference confidence interval. If the mean falls inside the interval, it is considered a "success" and the number of "success" out of 10,000 is calculated for each sample size.

Source Code

```
"""
Ajay Kc
013213328
EE381
Project 5 Part 2

The problem calculates the percentage of the sample
mean fitting under the area between
various confidence intervals.
"""
import numpy as np
import matplotlib.pyplot as plt
import math as math

bearingList = np.random.normal(75,0.75,1000000)

def checkConfidenceInterval(mean,sd,n,num):
    upper = mean+ num*(sd/math.sqrt(n))
    lower = mean -num*(sd/math.sqrt(n))
    if((75>=lower)and(75<=upper)):
        return "Success"
    else:
        return "Failure"

def estimatePopulationMean(n, confidenceInterval):

    normalList = []
    tDistList = []
    num1=0
    num2={}
    if(confidenceInterval == 95):
        num1 = 1.96

        num2 = {5:2.78,40:2.02,120:1.98}
    else:
        num1 = 2.58
        num2 = {5:4.60,40:2.70,120:2.62}

    for i in range(0, 10000):
        sampleSet = np.random.choice(bearingList,n)
        mean = np.mean(sampleSet)
        sd = np.std(sampleSet)

normalList.append(checkConfidenceInterval(mean,sd,n
,num1))

tDistList.append(checkConfidenceInterval(mean,sd,n,
num2[n]))

    successCountNormal =
normalList.count("Success")
    successCounttDist = tDistList.count("Success")
    print("For n = %s" %n)
    print("For %s confidence interval, %s using
normal distribution"
%(confidenceInterval,successCountNormal/100))
    print("For %s confidence interval, %s using
student's t distribution" %(confidenceInterval,
successCounttDist / 100))
    print(" ")
```

Ajay Kc
013213328
Project 5 Report

```
estimatePopulationMean(5,95)
estimatePopulationMean(5,99)

estimatePopulationMean(40,95)
estimatePopulationMean(40,99)

estimatePopulationMean(120,95)
estimatePopulationMean(120,99)
```

Result

Sample size (n)	**95% Confidence** (Using Normal distribution)	**99% Confidence** (Using Normal distribution)	**95% Confidence** (Using Student’s t distribution	**99% Confidence** (Using Student’s t distribution)
5	83.75%	92.22%	92.94%	98.47%
40	94.45%	98.58%	95.23%	99%
120	94.41%	98.68%	94.68%	98.86%