Download (right-click, save target as ...) this page as a jupyterlab notebook from: ES-23

Exercise Set 23: "Confidence Intervals | AB Testing"

Medrano, Giovanni

R11521018

ENGR 1330 ES-23 - Homework

Exercise 1 (Interval Estimate):

From a normally distributed population, we randolmy took a sample of 200 dogs with a mean weight of 70 pounds. Suppose the standard deviation of the population is 20:

- What is the estimated true population mean for the 95% confidence interva?
- How about 90% confidence interval?
- How about 99% confidence interval?

In [3]:
```
# 95% confidence interval
size = 200
m = 70
p = size ** 0.5
std = 20

posInterval = m + (1.96 * (std / p))

smalInterval = m - (1.96 * (std / p))
print('The estimated true population mean for the 95% CI is between', smalInterval, 'to
```

```
The estimated true population mean for the 95% CI is between 67.22814141774873 to 72.771
85858225127
```

In [5]:
```
# 90% confidence interval

posInterval1 = m + (1.645 * (std / p))
smalInterval1 = m - (1.645 * (std / p))
print('The estimated true population mean for the 90% CI is between', smalInterval1, 't
```

```
The estimated true population mean for the 90% CI is between 67.67361868989626 to 72.326
38131010374
```

In [6]:
```
# 99% confidence interval

posInterval11 = m + (2.576 * (std / p))
smalInterval11 = m - (2.576 * (std / p))
print('The estimated true population mean for the 90% CI is between', smalInterval11, '
```

```
The estimated true population mean for the 90% CI is between 66.35698586332691 to 73.643
01413667309
```

Exercise 2 (Interval Estimate):

Download the data frame DogWeights. Describe the dataframe; how many rows?; what is the mean dog weight?; what is the standard deviation?; make a histogram of the dataframe

Assuming the dataframe is the entire population evaluate the value of your confidence interval estimates:

- For the 95% confidence interval simulate 20 random samples of size 200 from the population, from those samples estimate the mean (20 estimates). Then determine how many of your 20 estimates produce a mean value within the confidence interval you determine in Exercise 1 above.
- Repeat for the 99% confidence interval, but simulate 100 random samples of size 200. Again how many of the 100 estimates fall within the confidence interval you determined in Exercise 1 above.

In [9]:

```
# download the file
######### CODE TO AUTOMATICALLY DOWNLOAD THE DATABASE ################
#! pip install requests #install packages into local environment
import requests # import needed modules to interact with the internet
# make the connection to the remote file (actually its implementing "bash curl -O http:/
remote_url = 'http://54.243.252.9/engr-1330-webroot/8-Labs/Lab23/DogWeights.csv' # a cs
response = requests.get(remote_url) # Gets the file contents puts into an object
output = open('DogWeights.csv', 'wb') # Prepare a destination, local
output.write(response.content) # write contents of object to named local file
output.close() # close the connection

# how many rows?
# what is the mean dog weight?
# what is the standard deviation?
# make a histogram of the dataframe
```

In [19]:

```
import pandas as pd
import numpy as np
# describe the dataframe
dfd=pd.read_csv('DogWeights.csv')
print(dfd.head())
dfd=dfd.iloc[:,1]
print('=======================')
print(dfd.describe())
```

```
   Unnamed: 0  Weight_lbs
0           0   60.281985
1           1   60.986304
2           2   64.919400
3           3   82.327049
4           4   79.604253
=======================
count    2.500000e+06
mean     6.999733e+01
std      2.000243e+01
min     -3.017528e+01
25%      5.651350e+01
50%      6.998797e+01
75%      8.347878e+01
```

```
max      1.670066e+02
Name: Weight_lbs, dtype: float64
```

In [20]:
```
print('TOTAL AMOUNT OF ROWS' ,len(dfd))
```

```
TOTAL AMOUNT OF ROWS 2500000
```

In [21]:
```
print('Mean weight of the dogs',dfd.mean())
```

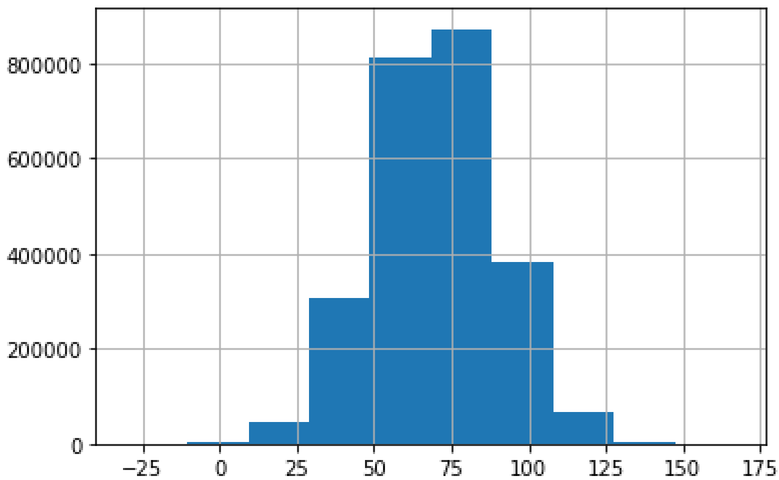
```
Mean weight of the dogs 69.99732967661204
```

In [22]:
```
print('Standard Deviation', dfd.std())
```

```
Standard Deviation 20.002431471929558
```

In [23]:
```
print(dfd.hist())
```

```
AxesSubplot(0.125,0.125;0.775x0.755)
```



In [35]:
```
# 95% confidence interval

import scipy.stats as stats
size = 20
sampleSize = 200
samps=[]
c=[]
avg =[]

for i in range(size):

    hold = df.sample(sampleSize)
    samps.append(hold.values)
    avg.append(hold.mean())

    c.append(stats.t.interval(alpha=0.95, df=len(samps[i])-1, loc=np.mean(samps[i]), sc

print(c)
print('\nTHERE ARE SEVERAL VALUES THAT CORRESPOND TO EXCERCISE ONE')
```

```
[(68.87365883708787, 74.44325904389319), (65.59672430160745, 71.66560351147706), (68.099
91373714615, 73.70037680623867), (68.14776297568513, 73.41662165818373), (66.08727697537
674, 72.1128366784832), (63.565749743560445, 69.21416907998045), (66.76794359536676, 72.
04877935934111), (65.27898839454416, 71.01560949587262), (63.5183848633407, 69.662011409
7265), (67.51072228036467, 72.94102134419701), (65.91346512477823, 71.39844367793613),
(68.04462072029942, 73.35098583308086), (66.42003368295896, 72.07338588232658), (67.6777
5283764131, 73.1990237226962), (65.53247734125016, 70.96786139463994), (67.8223136029893
1, 73.50802998551498), (68.28395249386642, 73.68165390769289), (68.20774770129687, 73.79
```

```
164414291644), (67.97034328415218, 73.91490862201628), (67.56658746096667, 73.1282203199
3733)]

THERE ARE SEVERAL VALUES THAT CORRESPOND TO EXCERCISE ONE
```

In [36]:

```python
# 99% confidence interval

import scipy.stats as stats
size = 100
sampleSize = 200
samps=[]
c=[]
avg =[]

for i in range(size):

    hold = df.sample(sampleSize)
    samps.append(hold.values)
    avg.append(hold.mean())

    c.append(stats.t.interval(alpha=0.99, df=len(samps[i])-1, loc=np.mean(samps[i]), sc

print(c)
print('\nTHERE ARE SEVERAL VALUES THAT CORRESPOND TO EXCERCISE ONE')
```

```
[(66.31078828999662, 73.97831534980921), (66.241474918036, 73.18581527656988), (66.40321
797414526, 73.7904590486036), (65.9334801051173, 73.59839151755733), (66.54998162308225,
73.62679358884215), (68.17299771731481, 75.43915289504389), (66.51626968107651, 74.13317
98973683), (65.5068802215818, 73.02209896940961), (66.8311915138965, 74.2569737064047),
(67.93136540481939, 75.47107746435509), (64.72699063411774, 71.48491290388766), (64.6044
8954525667, 71.63107352574632), (67.09415214514085, 74.35118039110871), (65.690166276896
72, 72.8917705766847), (65.74382197626014, 73.00428783059797), (65.64027437826216, 73.28
98039769065), (65.7268976227642, 73.21814668993778), (68.10902453494064, 75.440460816950
59), (65.8750397464296, 72.88874746996188), (66.98181168107956, 74.00239241758564), (66.
59609176010682, 73.47252609513151), (66.60505192292949, 73.35314403441639), (68.22245572
254926, 75.86836971702488), (67.11544772458724, 75.0658541570529), (64.72343688861775, 7
2.42812204409302), (68.46454041192432, 75.61539649520246), (65.9954946039233, 73.0684204
0439119), (66.24753594045409, 73.1876087846886), (64.2431898277704, 71.25503032584122),
(66.6862320300826, 73.91737943875981), (68.19297249852364, 75.67819495643751), (65.55964
247802227, 73.0816293165811), (66.87020691407687, 74.72706121256951), (68.7660599186651,
76.13169389824724), (66.7427690650204, 74.11392949352084), (65.8978532114237, 74.1928060
31349), (67.12001935445258, 75.00489750982052), (69.37475255539795, 76.31374827748633),
(67.4117232691754, 74.61149431874621), (66.76008535184758, 74.19669280937353), (65.71145
30298717, 72.04241635894653), (65.02709973880121, 72.14869131357446), (66.5133651458175
9, 73.98624560862784), (66.12652476318841, 73.32157067020044), (68.3283207256838, 75.116
27748488874), (67.1973694883492, 74.6054755497665), (66.20667939126021, 73.3561691431823
8), (66.61504993318441, 73.52846833496538), (65.26939188667708, 72.99236903585208), (64.
42798444143153, 72.47963485594411), (66.09254573772736, 73.41545502185994), (64.92041616
562018, 72.06626824100009), (66.24479058815693, 73.74120776760383), (66.45018602174466,
74.02648999354079), (66.08965911367397, 73.30455690229488), (64.68790763183972, 71.39262
416694734), (66.49971788168861, 73.58186656276541), (67.2540219180374, 74.7710655241241
8), (68.36120235744966, 75.95105104201704), (66.96484803034117, 73.75604119710563), (65.
92779448515195, 73.3046337602252), (65.2152564163, 72.42190604153537), (65.539558795403
4, 72.79880103257179), (65.31518065169091, 72.54537553122385), (69.2955089652179, 75.666
38914268829), (64.65079908539833, 71.5319365034335), (65.97042940719012, 73.566557753347
44), (68.6197015123152, 75.38036904477104), (67.55693746145417, 74.77781250549756), (63.
97516012610533, 71.36945154372553), (66.97795401431458, 74.49527123410783), (64.52613793
496414, 71.4019389150987), (66.56866052576154, 74.24826523369097), (63.90554679226721, 7
0.89007804869499), (65.33674336832318, 72.30074982981354), (67.18643521619413, 74.789111
79689862), (67.93812179138375, 74.81668197436592), (66.17937997248876, 73.338093992637),
(65.77799179694826, 73.17692889398093), (66.28229376013158, 73.29442901487499), (65.1933
4700969108, 72.39449604821655), (67.38305755333185, 74.44659767119869), (66.066546451583
3, 72.87711657779052), (66.69509396267048, 73.87776259328695), (68.36800983003192, 75.70
```

```
619396037239), (67.60270676759149, 74.6792263838538), (64.67490892575633, 72.39542499435
05), (66.96882868801202, 74.16889140711302), (68.0819266545625, 75.73793460804063), (67.
84629753835962, 75.76235813912938), (66.02341989068539, 73.39300488256744), (68.45165295
661064, 76.04786398766697), (67.77683808544795, 75.37209690192482), (64.03293858869937,
71.61053777225557), (66.42552874766734, 73.18531832482917), (64.90152433702711, 71.79254
487292233), (65.85011089226303, 73.3789884876859), (66.3310031219294, 72.9126675928823
7), (68.02710821707353, 75.9150805429988), (67.21467739766678, 74.06999602258222)]
```

```
THERE ARE SEVERAL VALUES THAT CORRESPOND TO EXCERCISE ONE
```

Exercise 3 (A/B Test):

Amazon is considering changing the color of their logo. The smile will be green instead of orange!

Let us assume out of 5000 users, they have directed 2500 to site A with the previous logo, and the rest to site B with the new logo. In the first group, 1863 users made a purchase. In the second group, 1904 users made a purchase. Is this a statistically significant result? Should Amazon change their logo in order to make more sells?

In [37]:

```

```

In [41]:

```
import pandas as pd
import numpy as np
from scipy.stats import norm


def get_confidence_ab_test(click_a, num_a, click_b, num_b):
    rate_a = click_a / num_a
```

```
    rate_b = click_b / num_b
    std_a = np.sqrt(rate_a * (1 - rate_a) / num_a)
    std_b = np.sqrt(rate_b * (1 - rate_b) / num_b)
    z_score = (rate_b - rate_a) / np.sqrt(std_a**2 + std_b**2)
    return norm.sf(z_score)

num_a= 2500
num_b = 2500
click_a= 1863
click_b = 1904
rate_a= click_a / num_a
rate_b = click_b / num_b

print(get_confidence_ab_test(click_a, num_a, click_b, num_b))
```

```
0.0892397034239209
```

In [ ]:

```
No they should not change their logo it doesnt meet the treshold of confidence
```