Name: Shivam Pandey
Roll no:47
Experiment no.3
Explore Inferential Statistic on the given dataset
Date of Performance: 16-02-2024
Date of Submission: 16-02-2024

Aim: Explore Inferential Statistic on the given dataset

Objective: Able to perform various inferential statistics on the given dataset.

Theory:

Z-Test & T-Tests are Parametric Tests, where the Null Hypothesis is less than, greater than or equal to some value. • A z-test is used if the population variance is known, or if the sample size is larger than 30, for an unknown population variance. • If the sample size is less than 30 and the population variance is unknown, we must use a t-test. T test is a type of inferential statistic used to study if there is a statistical difference between two groups. Mathematically, it establishes the problem by assuming that the means of the two distributions are equal (H_0: $\mu_1=\mu_2$). If the t-test rejects the null hypothesis (H_0: $\mu_1=\mu_2$), it indicates that the groups are highly probably different. The statistical test can be one-tailed or two-tailed. The one-tailed test is appropriate when there is a difference between groups in a specific direction. It is less common than the two-tailed test. When choosing a t test, you will need to consider two things: whether the groups being compared come from a single population or two different populations, and whether you want to test the difference in a specific direction.

There are three main types of t-test :

- One Sample t-test : Compares mean of a single group against a known/hypothesized/ population mean.
- Two Sample: Paired Sample T Test: Compares means from the same group at different times.
- Two Sample: Independent Sample T Test: Compares means for two different groups.

One Sample t-test:

$$t=\frac{\text{(Sample Mean} - \text{Population Mean)}}{\text{Standard Error}}$$

$$t = \frac{\bar{x} - \mu}{s/\sqrt{n}}$$

$\bar{x}$ Sample mean
μ Population mean
s Sample standard deviation
n Sample size

Two-sample - Paired Sample t-test

$$t = \frac{\bar{d}}{s/\sqrt{n}}$$

$\bar{d}$ =Mean of the difference
s=Standard deviation of the difference
n =is the sample size (i.e., size of d)

If the calculated t value is less than critical t value or greater that the critical value (obtained from a critical value table called the T-distribution table) then reject the null hypothesis.

P-value <significance level (a) => Reject your null hypothesis in favor of your alternative hypothesis. Your result is statistically significant.

P-value >= significance level (a) => Fail to reject your null hypothesis. Your result is not statistically significant.

Code:

Exp: 03

February 16, 2024

[55]:
```
import numpy as np
import pandas as pd
from scipy import stats
from google.colab import drive
drive.mount("/content/drive")
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call
drive.mount("/content/drive", force_remount=True).
```

Reliance Data Mart Dataset

[56]:
```
RDM=pd.read_excel('/content/drive/MyDrive/ADS Lab/RelianceDataMart.xlsx')
RDM
```

[56]:

	Rice_Bag_Weight
0	24.50
1	24.70
2	25.60
3	25.00
4	24.70
5	23.30
6	23.30
7	24.00
8	25.10
9	24.30
10	23.30
11	24.10
12	24.10
13	24.20
14	25.20
15	24.90
16	24.70
17	24.10
18	25.00
19	24.70
20	24.90
21	25.00
22	24.00

```
23              23.98
24              24.30
25              24.20
26              24.56
27              24.50
28              24.70
```

[57]:
```python
print(RDM.mean())
```

```
Rice_Bag_Weight    24.446207
dtype: float64
```

[58]:
```python
RDM.describe()
```

[58]:

	Rice_Bag_Weight
count	29.000000
mean	24.446207
std	0.569463
min	23.300000
25%	24.100000
50%	24.500000
75%	24.900000
max	25.600000

[59]:
```python
one_sample_result=stats.ttest_1samp(RDM,24.446)
print(one_sample_result)
```

```
TtestResult(statistic=array([0.00195653]), pvalue=array([0.99845279]),
df=array([28]))
```

Crocin Data ST Dataset

[60]:
```python
CDS=pd.read_excel('/content/drive/MyDrive/ADS Lab/Crocin_Data_ST.xlsx')
CDS
```

[60]:

	Before_Crocin	After_Crocin	diff	Unnamed: 3	Unnamed: 4	Unnamed: 5
0	101.0	99	2.000000	NaN	NaN	NaN
1	99.0	98	1.000000	NaN	NaN	NaN
2	101.0	97	4.000000	NaN	NaN	NaN
3	99.9	99	0.900000	NaN	NaN	NaN
4	99.8	98	1.800000	NaN	NaN	NaN
5	98.0	97	1.000000	NaN	NaN	NaN
6	97.0	99	-2.000000	NaN	NaN	NaN
7	101.0	98	3.000000	NaN	NaN	NaN
8	102.0	96	6.000000	NaN	NaN	NaN
9	103.0	98	5.000000	NaN	NaN	NaN
10	99.0	94	5.000000	NaN	NaN	NaN
11	99.9	96	3.900000	NaN	NaN	NaN

12	99.8	97	2.800000	NaN	NaN	NaN
13	99.7	99	0.700000	NaN	NaN	NaN
14	101.1	98	3.100000	NaN	NaN	NaN
15	102.3	97	5.300000	NaN	NaN	NaN
16	101.0	99	2.000000	NaN	NaN	NaN
17	99.0	98	1.000000	NaN	NaN	NaN
18	101.0	97	4.000000	NaN	NaN	NaN
19	99.9	99	0.900000	NaN	NaN	NaN
20	99.8	98	1.800000	NaN	NaN	NaN
21	98.0	96	2.000000	NaN	NaN	NaN
22	97.0	97	0.000000	NaN	NaN	NaN
23	101.0	99	2.000000	NaN	NaN	NaN
24	102.0	97	5.000000	NaN	NaN	NaN
25	103.0	99	4.000000	NaN	NaN	NaN
26	99.0	98	1.000000	NaN	NaN	NaN
27	99.9	97	2.900000	NaN	NaN	NaN
28	99.8	99	0.800000	NaN	NaN	NaN
29	NaN	mean	2.444828	NaN	t val	7.071713
30	NaN	std dev	1.861755	NaN	NaN	NaN
31	NaN	sq root n	5.385165	NaN	NaN	NaN

[61]:
```
CDS = CDS.iloc[:, 0:3]
CDS
```

[61]:

	Before_Crocin	After_Crocin	diff
0	101.0	99	2.000000
1	99.0	98	1.000000
2	101.0	97	4.000000
3	99.9	99	0.900000
4	99.8	98	1.800000
5	98.0	97	1.000000
6	97.0	99	-2.000000
7	101.0	98	3.000000
8	102.0	96	6.000000
9	103.0	98	5.000000
10	99.0	94	5.000000
11	99.9	96	3.900000
12	99.8	97	2.800000
13	99.7	99	0.700000
14	101.1	98	3.100000
15	102.3	97	5.300000
16	101.0	99	2.000000
17	99.0	98	1.000000
18	101.0	97	4.000000
19	99.9	99	0.900000
20	99.8	98	1.800000
21	98.0	96	2.000000

```
22          97.0         97  0.000000
23         101.0         99  2.000000
24         102.0         97  5.000000
25         103.0         99  4.000000
26          99.0         98  1.000000
27          99.9         97  2.900000
28          99.8         99  0.800000
29           NaN       mean  2.444828
30           NaN    std dev  1.861755
31           NaN  sq root n  5.385165
```

[62]:
```
CDS = CDS.iloc[:29]
```

[63]:
```
print(CDS.mean())
```

```
Before_Crocin    100.134483
After_Crocin      97.689655
diff               2.444828
dtype: float64
```

[64]:
```
CDS.describe()
```

[64]:

	Before_Crocin	diff
count	29.000000	29.000000
mean	100.134483	2.444828
std	1.561427	1.861755
min	97.000000	-2.000000
25%	99.000000	1.000000
50%	99.900000	2.000000
75%	101.000000	4.000000
max	103.000000	6.000000

[65]:
```
two_sample_result = stats.ttest_rel(CDS ["Before_Crocin"], CDS ["After_Crocin"])
two_sample_result
```

[65]:
```
TtestResult(statistic=7.071712959273876, pvalue=1.0800112658101922e-07, df=28)
```

Pre_Post_Score Dataset

[66]:
```
pps=pd.read_excel('/content/drive/MyDrive/ADS Lab/Pre_Post_Score.xlsx')
pps
```

[66]:
```
   Pre_Score   Post_Score      Diff  Unnamed: 3  Unnamed: 4 Unnamed: 5  \
0       18.0           22 -4.000000         NaN         NaN        NaN
1       21.0           25 -4.000000         NaN         NaN        NaN
2       16.0           17 -1.000000         NaN         NaN        NaN
3       22.0           24 -2.000000         NaN         NaN        NaN
4       19.0           16  3.000000         NaN         NaN        NaN
```

```
5        24.0            29 -5.000000        NaN        NaN        NaN
6        17.0            20 -3.000000        NaN        NaN        NaN
7        21.0            23 -2.000000        NaN        NaN        NaN
8        23.0            19  4.000000        NaN        NaN        NaN
9        18.0            20 -2.000000        NaN        NaN        NaN
10       14.0            15 -1.000000        NaN        NaN        NaN
11       16.0            15  1.000000        NaN        NaN        NaN
12       16.0            18 -2.000000        NaN        NaN        NaN
13       19.0            26 -7.000000        NaN        NaN        NaN
14       18.0            18  0.000000        NaN        NaN        NaN
15       20.0            24 -4.000000        NaN        NaN        NaN
16       12.0            18 -6.000000        NaN        NaN        NaN
17       22.0            25 -3.000000        NaN        NaN     t val=
18       15.0            19 -4.000000        NaN        NaN        NaN
19       17.0            16  1.000000        NaN        NaN        NaN
20        NaN          mean -2.050000        NaN        NaN        NaN
21        NaN       std dev  2.837252        NaN        NaN        NaN
22        NaN  sq root  of n  4.472136        NaN        NaN        NaN

    Unnamed: 6
0          NaN
1          NaN
2          NaN
3          NaN
4          NaN
5          NaN
6          NaN
7          NaN
8          NaN
9          NaN
10         NaN
11         NaN
12         NaN
13         NaN
14         NaN
15         NaN
16         NaN
17   -3.231253
18         NaN
19         NaN
20         NaN
21         NaN
22         NaN
```

[67]:
```
pps = pps.iloc[:, 0:3]
pps
```

[67]:

	Pre_Score	Post_Score	Diff
0	18.0	22	-4.000000
1	21.0	25	-4.000000
2	16.0	17	-1.000000
3	22.0	24	-2.000000
4	19.0	16	3.000000
5	24.0	29	-5.000000
6	17.0	20	-3.000000
7	21.0	23	-2.000000
8	23.0	19	4.000000
9	18.0	20	-2.000000
10	14.0	15	-1.000000
11	16.0	15	1.000000
12	16.0	18	-2.000000
13	19.0	26	-7.000000
14	18.0	18	0.000000
15	20.0	24	-4.000000
16	12.0	18	-6.000000
17	22.0	25	-3.000000
18	15.0	19	-4.000000
19	17.0	16	1.000000
20	NaN	mean	-2.050000
21	NaN	std dev	2.837252
22	NaN	sq root of n	4.472136

[68]:

```
pps = pps.iloc[:20]
```

[69]:

```
two_sample_result = stats.ttest_rel (pps ["Pre_Score"], pps ["Post_Score"])
two_sample_result
```

[69]: TtestResult(statistic=-3.231252665580312, pvalue=0.004394965993185664, df=19)

Conclusion:

One sample t-test has been done on the reliance data mart dataset and it has been found that difference exists between the rice bag population mean and rice bag sample mean. Two sample paired t-test has been done on the prescore-post score dataset and Crocin dataset. In the prescore-post score dataset difference exists between the mean pre-score before studying the module and mean prescore after studying the module. In the crocin dataset it is found that temperature difference exists before and after having the crocin tablet.