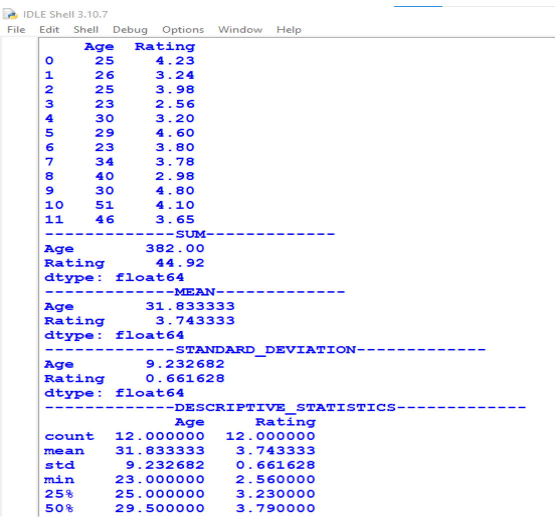
Practical 1

A. Write a program for obtaining descriptive statistics of data.

```
import pandas as pd

d={'Age':pd.Series([25,26,25,23,30,29,23,34,40,30,51,46]),

  'Rating':pd.Series([4.23,3.24,3.98,2.56,3.20,4.6,3.8,3.78,2.98,4.80,4.10,3.65])

  }

df=pd.DataFrame(d)

print(df)

print('-------------SUM-------------')

print(df.sum())

print('-------------MEAN-------------')

print(df.mean())

print('-------------STANDARD_DEVIATION-------------')

print(df.std())

print('-------------DESCRIPTIVE_STATISTICS-------------')

print(df.describe())
```



```
IDLE Shell 3.10.7
File  Edit  Shell  Debug  Options  Window  Help
    Age  Rating
0    25    4.23
1    26    3.24
2    25    3.98
3    23    2.56
4    30    3.20
5    29    4.60
6    23    3.80
7    34    3.78
8    40    2.98
9    30    4.80
10   51    4.10
11   46    3.65
-------------SUM-------------
Age       382.00
Rating     44.92
dtype: float64
-------------MEAN-------------
Age       31.833333
Rating     3.743333
dtype: float64
-------------STANDARD_DEVIATION-------------
Age       9.232682
Rating    0.661628
dtype: float64
-------------DESCRIPTIVE_STATISTICS-------------
             Age     Rating
count  12.000000  12.000000
mean   31.833333   3.743333
std     9.232682   0.661628
min    23.000000   2.560000
25%    25.000000   3.230000
50%    29.500000   3.790000
```

Using Excel 2013

STEP1 - Go to File Menu

STEP2 - Choose Options

STEP3 - Choose Add-Ins

STEP4 - Select Analysis ToolPak

STEP5 - Press OK

STEP6 - GO to Data Menu

STEP7 - Click on DataAnalysis

STEP8 - Descriptive Statistics

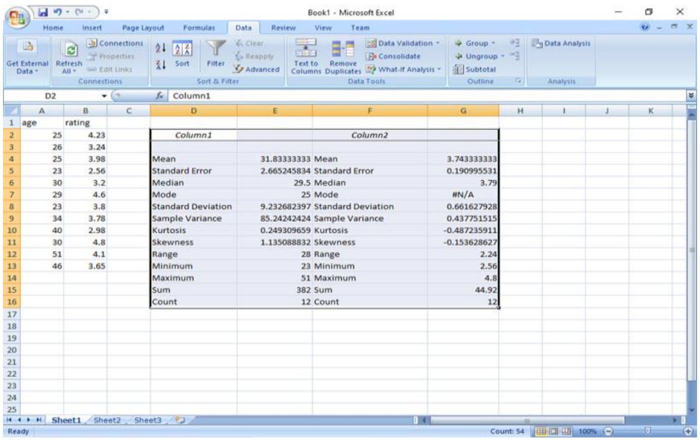
STEP9 - Press OK

STEP10 - Descriptive Statistics

Input range ------------select age range from value one to last

Output range ----------- select different columns for output

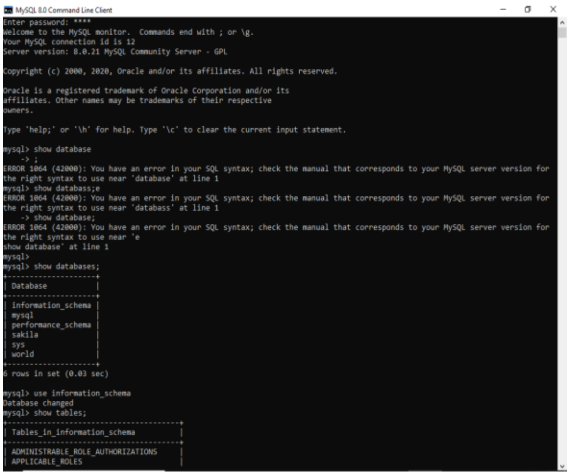
OUTPUT:

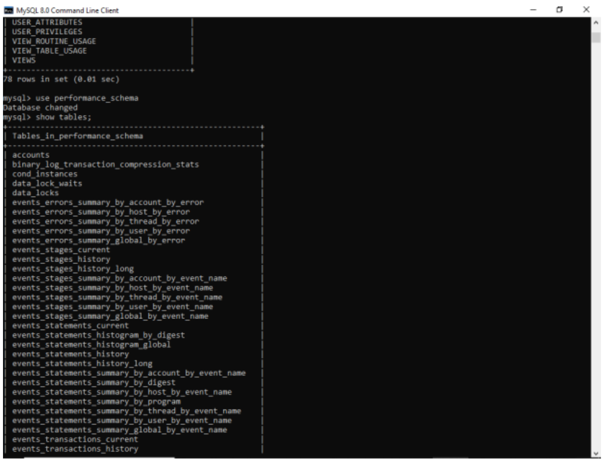


B. Import data from different data sources (from Excel,csv, mysql, sql server, oracle to R/Python/Excel)

```
import mysql.connector

conn = mysql.connector.connect(host='localhost',database='information_schema',
user='root',password='root')

conn.connect

if(conn.is_connected):

   print('###### Connection With MySql Established Successfullly ##### ')

else:

   print('Not Connected -- Check Connection Properites')

import mysql.connector

mycursor = conn.cursor()

mycursor.execute("show tables;")

myresult = mycursor.fetchall()

for x in myresult:

   print(x)
```

```
Python 3.10.7 (tags/v3.10.7:6cc6b13, Sep  5 2022, 14:08:36) [MSC v.1933 64 bit (
AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.

== RESTART: C:/Users/User-25/AppData/Local/Programs/Python/Python310/RIC 3.py ==
###### Connection With MySql Established Successfullly #####
('student',)

== RESTART: C:/Users/User-25/AppData/Local/Programs/Python/Python310/RIC 3.py ==
###### Connection With MySql Established Successfullly #####
(101, 'abc', '1st year')
(102, 'pqr', '1st year')
(103, 'xyz', '1st year')
```





Practical 2:

A. Design a survey form for a given case study, collect the primary data and analyse it

Case 1:
A researcher wants to conduct a Survey in colleges on Use of ICT in higher education from Mumbai, Thane and Navi Mumbai. The survey focuses on access to and use of ICT in teaching and learning, as well as on attitudes towards the use of ICT in teaching and learning.
Design questionnaire addressed to teachers seeks information about the target class, his experience using ICT for teaching, access to ICT infrastructure, support available, ICT based activities and material used, obstacles to the use of ICT in teaching, learning activities with the target class, your skills and attitudes to ICT, and some personal background information.

Arrange question in following groups:

1. Information about the target class you teach
2. Experience with ICT for teaching
3. ICT access for teaching
4. Support to teachers for ICT use
5. ICT based activities and material used for teaching
6. Obstacles to using ICT in teaching and learning
7. Learning activities with the target class
8. Teacher skills
9. Teacher opinions and attitudes
10. Personal background information

Case 2:
A research agency wants to study the perception about App based taxi service in Mumbai, Thane and Navi Mumbai. The survey focuses on customers attitude towards app base taxi service as well as on attitudes towards regular taxi cab.
Design questionnaire seeks information about the target taxi service, his experience using taxi services, access, support available, obstacles and some personal background information, with the following objectives:

1. To find out the customer satisfaction towards the App based-taxi services.
2. To find the level of convenience and comfort with App based -taxi services.
3. To know their opinion about the tariff system and promptness of service.
4. To ascertain the customer view towards the driver behaviour and courtesy.
5. To provide inputs to enhance the services to delight the customers.

6. To examine relationship between service quality factors and taxi passenger satisfaction.
7. To suggest better regulations for transportation authorities regarding
8. customer protection and effective monitoring of taxi services.

Case 3:

A popular electronic store want to conduct a survey to develop awareness of branded laptop baseline estimates and determine popularity of different company's laptop. It suggests steps to be initiated or strengthened in the field of demand in a region. The key indicators are among the general population, demand branded laptop and the problem users.

The objectives of this particular study are:-

1. To know the preferences of different types of branded laptops by students and professionals.
2. To study which factor influence for choosing different types of branded laptops.
3. To know about the level of satisfaction towards different types of branded laptops.
4. To identify the perception of consumers towards the laptop positioning strategy.
5. To know the consumer preference towards laptop in the present era.

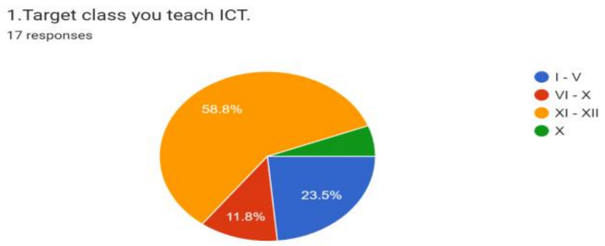
Use the collected data for analysis.

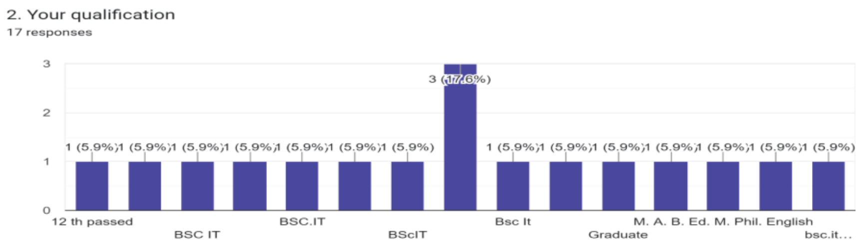
B. Perform analysis of given secondary data.

Steps in Secondary Data Analysis

1. **Determine your research question** – Knowing exactly what you are looking for.
2. **Locating data**– Knowing what is out there and whether you can gain access to it. A quick Internet search, possibly with the help of a librarian, will reveal a wealth of options.
3. **Evaluating relevance of the data** – Considering things like the data's original purpose, when it was collected, population, sampling strategy/sample, data collection protocols, operationalization of concepts, questions asked, and form/shape of the data.
4. **Assessing credibility of the data** – Establishing the credentials of the original researchers, searching for full explication of methods including any problems encountered, determining how consistent the data is with data from other sources, and discovering whether the data has been used in any credible published research.
5. **Analysis** – This will generally involve a range of statistical processes.Example: Analyze the given Population Census Data for Planning and Decision Making by using the size and composition of populations.

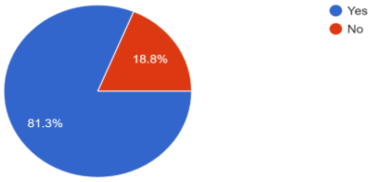
Analysis for Case I –





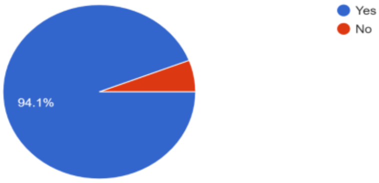
Do you have IT Lab

16 responses



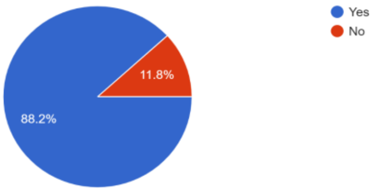
3. Do you have proper Internet Connection in IT Lab

17 responses



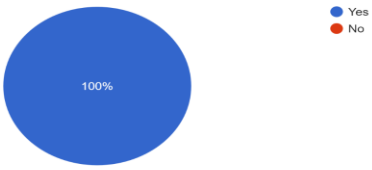
4. Is MS Office latest version installed

17 responses



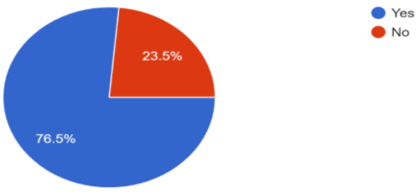
5. Does each student gets hands on practice

17 responses



6. Are there enough competitive exams or IT Fest around you taking place that enhances students skils.

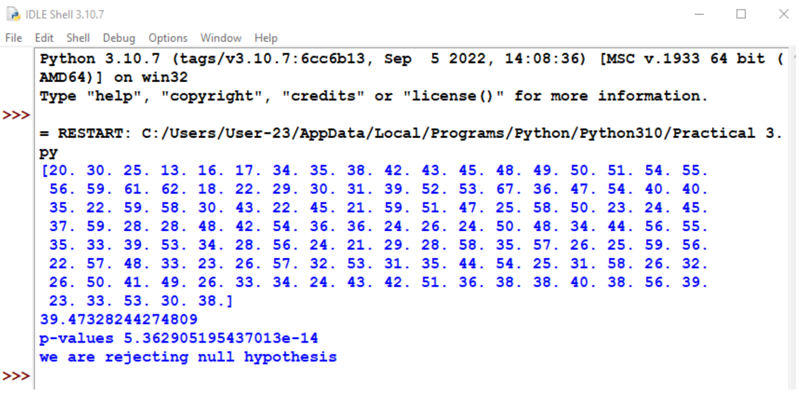
17 responses



Practical 3

A. Perform testing of hypothesis using one sample t-test.

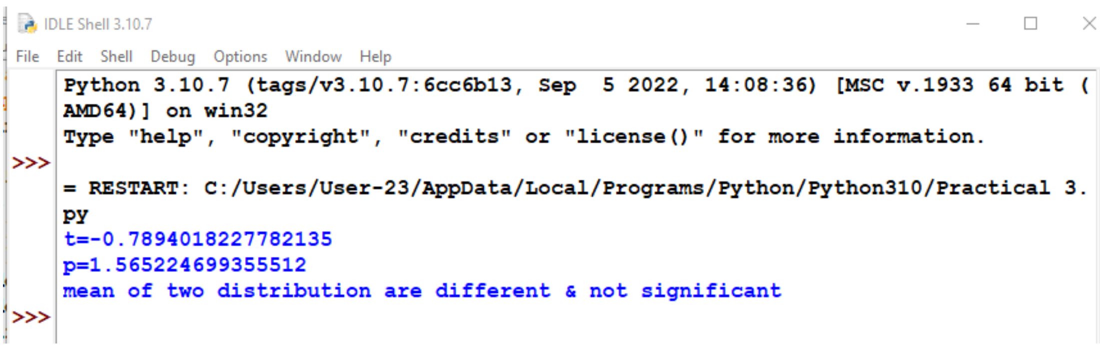
```
from scipy.stats import ttest_1samp

import numpy as np

ages=np.genfromtxt('ages.csv')

print(ages)

ages_mean=np.mean(ages)

print(ages_mean)

tset,pval=ttest_1samp(ages,30)

print('p-values',pval)

if pval<0.05:

   print("we are rejecting null hypothesis")

else:

   print("we are accepting null hypothesis")
```



```
IDLE Shell 3.10.7
File Edit Shell Debug Options Window Help
Python 3.10.7 (tags/v3.10.7:6cc6b13, Sep  5 2022, 14:08:36) [MSC v.1933 64 bit (
AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:/Users/User-23/AppData/Local/Programs/Python/Python310/Practical 3.
py
[20. 30. 25. 13. 16. 17. 34. 35. 38. 42. 43. 45. 48. 49. 50. 51. 54. 55.
 56. 59. 61. 62. 18. 22. 29. 30. 31. 39. 52. 53. 67. 36. 47. 54. 40. 40.
 35. 22. 59. 58. 30. 43. 22. 45. 21. 59. 51. 47. 25. 58. 50. 23. 24. 45.
 37. 59. 28. 28. 48. 42. 54. 36. 36. 24. 26. 24. 50. 48. 34. 44. 56. 55.
 35. 33. 39. 53. 34. 28. 56. 24. 21. 29. 28. 58. 35. 57. 26. 25. 59. 56.
 22. 57. 48. 33. 23. 26. 57. 32. 53. 31. 35. 44. 54. 25. 31. 58. 26. 32.
 26. 50. 41. 49. 26. 33. 34. 24. 43. 42. 51. 36. 38. 38. 40. 38. 56. 39.
 23. 33. 53. 30. 38.]
39.47328244274809
p-values 5.362905195437013e-14
we are rejecting null hypothesis
>>>
```

B. Perform testing of hypothesis using two sample t-test.

```
import numpy as np

from scipy import stats

from numpy.random import randn

N=20

a=[35,40,12,15,21,14,46,10,28,48,16,30,32,48,31,22,12,39,19,25]

b=[2,27,31,35,1,5,19,1,34,3,1,2,1,3,1,2,1,3,29,37,2]

a=5*randn(100)+50

b=5*randn(100)+51

var_a=a.var(ddof=1)

var_b=b.var(ddof=1)

s=np.sqrt((var_a+var_b)/2)

t=(a.mean()-b.mean())/(s*np.sqrt(2/N))

df=2*N-2

p=1-stats.t.cdf(t,df=df)

print("t="+str(t))

print("p="+str(2*p))

if t>p:

  print('mean of two distribution are different & significant')

else:

    print('mean of two distribution are different & not significant')
```



B. Using Excel 2013

STEP1 - Go to File Menu

STEP2 - Choose Options

STEP3 - Choose Add-Ins

STEP4 - Select Analysis ToolPak

STEP5 - Press OK

STEP6 - GO to Data Menu

STEP7 – Click on t-Test: Paired Two Sample for Means

STEP8 - Press OK

STEP9 - F- t-Test: Paired Two Sample for Means

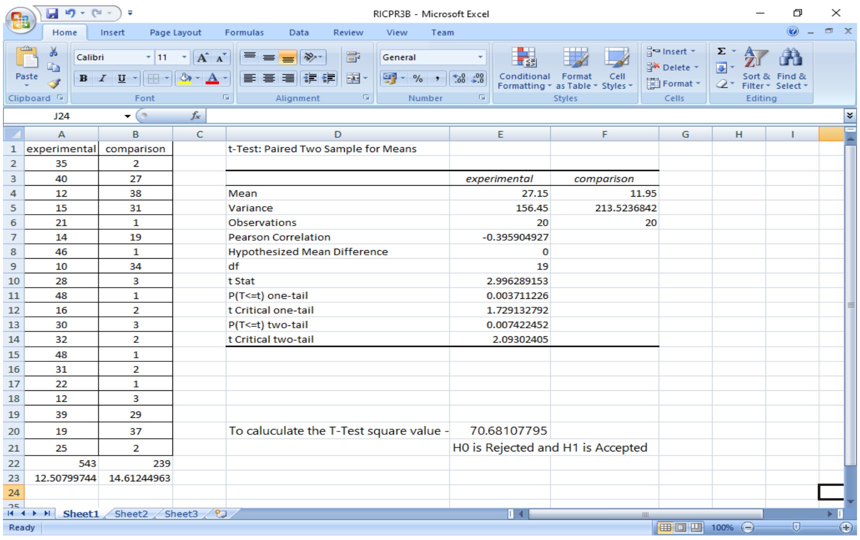
Input range ------------Variable 1 range -----select experimrntal range

Variable 2 range------select comparison range

STEP 10 – Select Labels

STEP11 - Output range ----------- select any blank cells in same sheet or new worksheet or new workbook

OUTPUT:

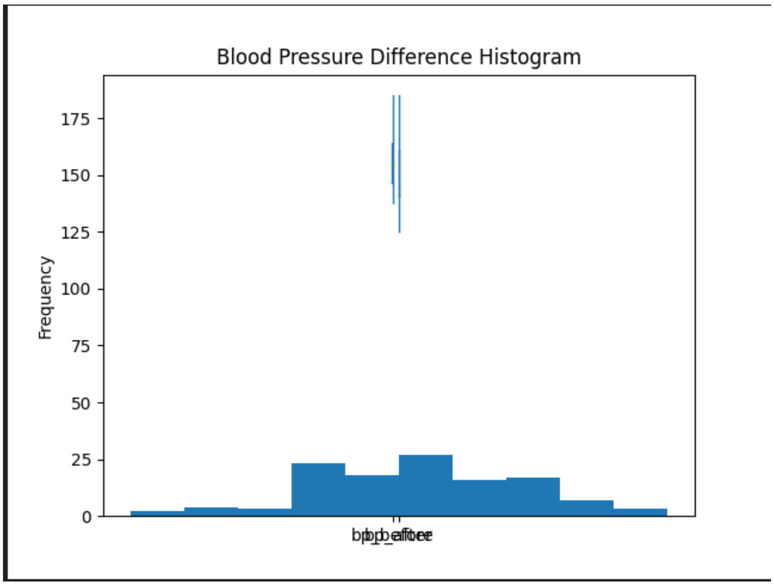


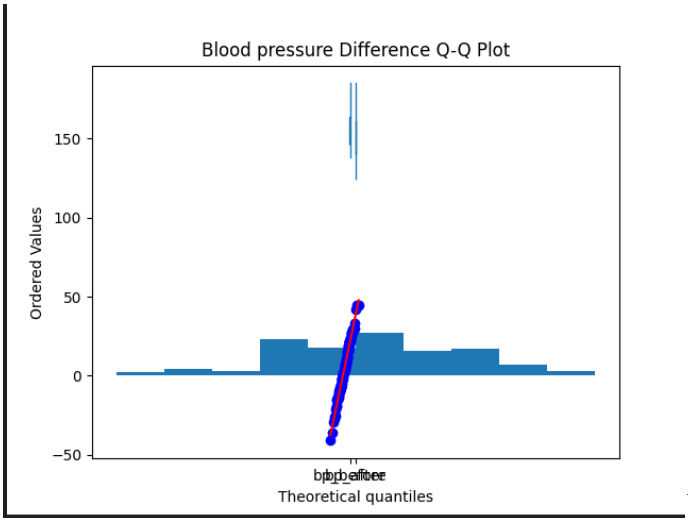
	A	B	C	D	E	F
1	experimental	comparison		t-Test: Paired Two Sample for Means		
2	35	2				
3	40	27			*experimental*	*comparison*
4	12	38		Mean	27.15	11.95
5	15	31		Variance	156.45	213.5236842
6	21	1		Observations	20	20
7	14	19		Pearson Correlation	-0.395904927	
8	46	1		Hypothesized Mean Difference	0	
9	10	34		df	19	
10	28	3		t Stat	2.996289153	
11	48	1		P(T<=t) one-tail	0.003711226	
12	16	2		t Critical one-tail	1.729132792	
13	30	3		P(T<=t) two-tail	0.007422452	
14	32	2		t Critical two-tail	2.09302405	
15	48	1				
16	31	2				
17	22	1				
18	12	3				
19	39	29				
20	19	37		To caluculate the T-Test square value -	70.68107795	
21	25	2			H0 is Rejected and H1 is Accepted	
22	543	239				
23	12.50799744	14.61244963				

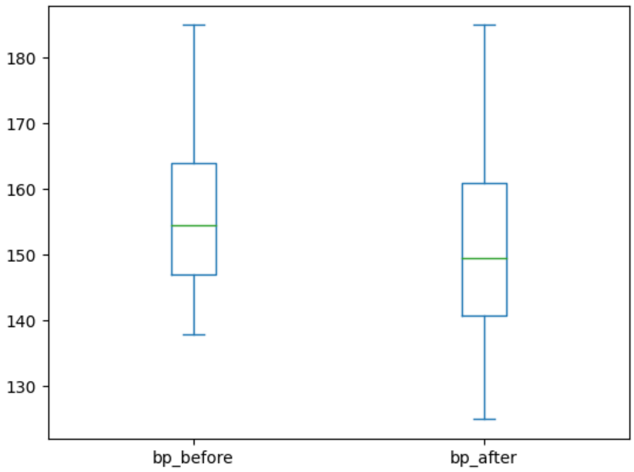
C. Perform testing of hypothesis using paired t-test.

```
from scipy import stats
import matplotlib.pyplot as plt
import pandas as pd
df = pd.read_csv("blood_pressure.csv")
print(df[['bp_before','bp_after']].describe())
#First let’s check for any significant outliers in
#each of the variables.
df[['bp_before', 'bp_after']].plot(kind='box')
# This saves the plot as a png file
plt.savefig('boxplot_outliers.png')
# make a histogram to differences between the two scores.
df['bp_difference'] = df['bp_before'] - df['bp_after']
df['bp_difference'].plot(kind='hist', title= 'Blood Pressure Difference Histogram')
#Again, this saves the plot as a png file
plt.savefig('blood pressure difference histogram.png')
stats.probplot(df['bp_difference'], plot= plt)
plt.title('Blood pressure Difference Q-Q Plot')
plt.savefig('blood pressure difference qq plot.png')
stats.shapiro(df['bp_difference'])
stats.ttest_rel(df['bp_before'], df['bp_after'])
```

```
IDLE Shell 3.10.7
File Edit Shell Debug Options Window Help
Python 3.10.7 (tags/v3.10.7:6cc6b13, Sep  5 2022, 14:08:36) [MSC v.1933 64 bit (
AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:/Users/User-23/AppData/Local/Programs/Python/Python310/Practical 3C
.py
        bp_before   bp_after
count  120.000000  120.000000
mean   156.450000  151.358333
std     11.389845   14.177622
min    138.000000  125.000000
25%    147.000000  140.750000
50%    154.500000  149.500000
75%    164.000000  161.000000
max    185.000000  185.000000
>>>
```


Blood Pressure Difference Histogram
Frequency
175
150
125
100
75
50
25
0
bp_before
bp_after


Blood pressure Difference Q-Q Plot
Ordered Values
150
100
50
0
−50
bp_before
bp_after
Theoretical quantiles


180
170
160
150
140
130
bp_before
bp_after

PRACTICAL 4

A. Perform testing of hypothesis using chi squared goodness of fit test

Problem:

A system administrator needs to upgrade the computers for his division. He wants to know what sort of computer system his workers prefer. He gives three choices: Windows, Mac, or Linux. Test the hypothesis or theory that an equal percentage of the population prefers each type of computer system.

H0 : The population distribution of the variable is the same as the proposed distribution

HA : The distributions are different

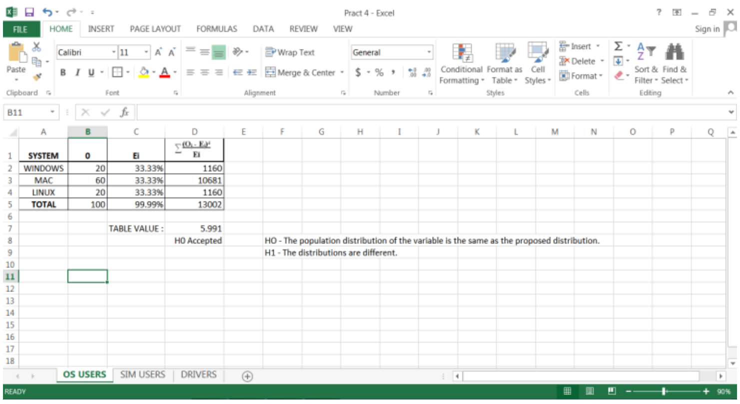
To calculate the Chi –Squared value for Windows go to cell D2 and type =((B2-C2)*(B2-C2))/C2

To calculate the Chi –Squared value for Mac go to cell D3 and type =((B3-C3)*(B3-C3))/C3

To calculate the Chi –Squared value for Mac go to cell D3 and type =((B4-C4)*(B4-C4))/C4 Go to Cell D5 for and type=SUM(D2:D4)

To get the table value for Chi-Square for α = 0.05 and dof = 2, go to cell D7 and type =CHIINV(0.05,2) At cell D8 type =IF(D5>D7, "H0 Accepted","H0 Rejected")

OUTPUT:



B. Perform testing of hypothesis using chi-squared test of independence

In a study to understand the performance of M. Sc. IT Part -1 class, a college selects a random sample of 100 students. Each student was asked his grade obtained in B. Sc. IT. The sample is as given below

Sr. No	Roll No	Student's Name	Gen	Grade
1	1	Gaborone	m	O
2	2	Francistown	m	O
3	5	Niamey	m	O
4	13	Maxixe	m	O
5	16	Tema	m	O
6	17	Kumasi	m	O
7	34	Blida	m	O
8	35	Oran	m	O
9	38	Saefda	m	O
10	42	Constantine	m	O
11	43	Annaba	m	O
12	45	Bejaefa	m	O
13	48	Medea	m	O
14	49	Djelfa	m	O
15	50	Tipaza	m	O
16	51	Bechar	m	O
17	54	Mostaganem	m	O
18	55	Tiaret	m	O
19	56	Bouira	m	O
20	59	Tebessa	m	O
21	61	El Harrach	m	O
22	62	Mila	m	O
23	65	Fouka	m	O
24	66	El Eulma	m	O
25	68	SidiBel Abbes	m	O
26	69	Jijel	m	O
27	70	Guelma	m	O
28	85	Khemis El Khechna	m	O
29	87	Bordj El Kiffan	m	O
30	88	Lakhdaria	m	O
31	6	Maputo	m	D
32	12	Lichinga	m	D
33	15	Ressano Garcia	m	D
34	19	Accra	m	D
35	27	Wa	m	D
36	28	Navrongo	m	D
37	37	Mascara	m	D
38	44	Batna	m	D
39	57	El Biar	m	D
40	60	Boufarik	m	D
41	63	OuedRhiou	m	D
42	64	Souk Ahras	m	D
43	71	Dar El Befda	m	D
44	86	Birtouta	m	D
45	18	Takoradi	m	C
46	22	Cape Coast	m	C
47	29	Kwabeng	m	C
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]

Sr. No	Roll No	Student's Name	Gen	Grade
62	3	Maun	f	O
63	7	Tete	f	O
64	9	Chimoio	f	O
65	11	Pemba	f	O
66	14	Chibuto	f	O
67	25	Mampong	f	O
68	36	Tlemcen	f	O
69	40	Adrar	f	O
70	41	Tindouf	f	O
71	46	Skikda	f	O
72	47	Ouargla	f	O
73	10	Matola	f	D
74	20	Legon	f	D
75	21	Sunyani	f	D
76	72	Teenas	f	D
77	73	Kouba	f	D
78	75	HussenDey	f	D
79	77	Khenchela	f	D
80	82	HassiBahbah	f	D
81	84	Baraki	f	D
82	91	Boudouaou	f	D
83	95	Tadjenanet	f	D
84	4	Molepolole	f	C
85	8	Quelimane	f	C
86	23	Bolgatanga	f	C
87	58	Mohammadia	f	C
88	83	Merouana	f	C
89	24	Ashaiman	f	B
90	76	N'gaous	f	B
91	90	Bab El Oued	f	B
92	92	BordjMenael	f	B
93	93	Ksar El Boukhari	f	B
94	74	Reghaa	f	A
95	78	Cheria	f	A
96	79	Mouzaa	f	A
97	80	Meskiana	f	A
98	81	Miliana	f	A
99	94	Sig	f	A
100	99	Kadiria	f	A

Null Hypothesis - H0 : The performance of girls students is same as boys students. Alternate **Hypothesis - H1 :** The performance of boys and girls students are different.

Open Excel Workbook

	O	A	B	C	D	Total	$\sum\frac{(O_i - E_i)^2}{Ei}$
Girls	11	7	5	5	11	**39**	6.075
Boys	30	4	3	10	14	**61**	6.075
Total	41	11	8	15	25	**100**	**12.150**
Ei	**20.5**	**5.5**	**4**	**7.5**	**12.5**	**50**	

Prepare a contingency table as shown above.

To calculate Girls Students with 'O' Grade Go to Cell N6 and type =COUNTIF($J$2:$K$40,"O") To calculate Girls Students with 'A' Grade Go to Cell O6 and type =COUNTIF($J$2:$K$40,"A") To calculate Girls Students with 'B' Grade Go to Cell P6 and type =COUNTIF($J$2:$K$40,"B") To calculate Girls Students with 'C' Grade Go to Cell Q6 and type =COUNTIF($J$2:$K$40,"C") To calculate Girls Students with ‘D’ Grade Go to Cell R6 and type =COUNTIF($J$2:$K$40,"D") To calculate Boys Students with ‘O’ Grade Go to Cell N7 and type=COUNTIF($D$2:$E$62,"O") To calculate Boys Students with ‘A’ Grade Go to Cell O7 and type=COUNTIF($D$2:$E$62,"A") To calculate Boys Students with ‘B’ Grade Go to Cell P7 and type =COUNTIF($D$2:$E$62,"B") To calculate Boys Students with ‘C’ Grade Go to Cell Q7 and type =COUNTIF($D$2:$E$62,"C") To calculate Boys Students with ‘D’ Grade Go to Cell R7 and type =COUNTIF($D$2:$E$62,"D")

To calculate the expected value Ei

Go to Cell N9 and type =N8/2

Go to Cell O9 and type =O8/2

Go to Cell P9 and type =P8/2

Go to Cell Q9 and type =Q8/2

Go to Cell R9 and type =R8/2

Go to Cell S6 and calculate total girl students = SUM(N6:R6)

Go to Cell S7 and calculate total girl students = SUM(N7:R7)

Now Calculate

Go to cell T6 and type

=SUM((N6-$N$9)^2/$N$9,(O6-$O$9)^2/$O$9,(P6-$P$9)^2/$P$9,(Q6-Q$9)^2/$Q$9,(R6-$R$9)^2/$R$9)

Go to cell T7 and type

=SUM((N7-$N$9)^2/$N$9,(O7-$O$9)^2/$O$9,(P7-$P$9)^2/$P$9,(Q7-Q$9)^2/$Q$9,(R7-$R$9)^2/$R$9)

To get the table value go to cell T11 and type

=CHIINV(0.05,4) Go to cell O13 and type =IF(T8>=T11," H0 is Accepted", "H0 is Rejected")

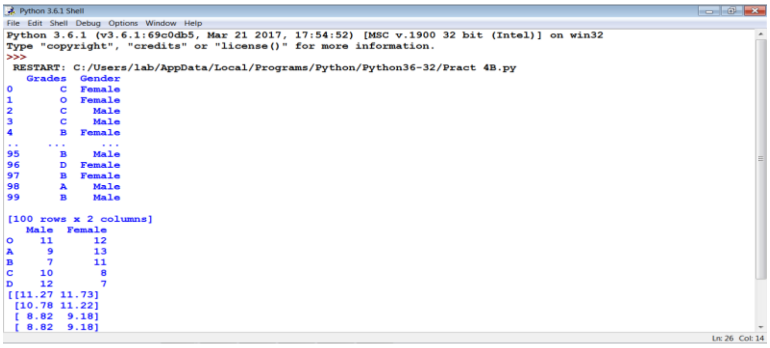
OUTPUT:

USING EXCEL:

M	N	O	P	Q	R	S	T
H0 : Performance of boys and girls are equal							
Frequency Table							$(O_i - E_i)^2$
	O	**A**	**B**	**C**	**D**	**Total**	**Ei**
Girls	11	7	5	5	11	**39**	6.075
Boys	30	4	3	10	14	**61**	6.075
Total	41	11	8	15	25	**100**	**12.150**
Ei	**20.5**	**5.5**	**4**	**7.5**	**12.5**	**50**	
Critcal Value of α =0.05 for df = (2-1) * (5-1)							9.487729
Decesion		**H0 is Accepted**					

USING PYTHON:

```
import numpy as np
import pandas as pd
import scipy.stats as stats
np.random.seed(10)
stud_grade=np.random.choice(a=["O","A","B","C","D"],
                     p=[0.20,0.20,0.20,0.20,0.20],size=100)
stud_gen=np.random.choice(a=["Male","Female"],p=[0.5,0.5],size=100)
mscpart1=pd.DataFrame({"Grades":stud_grade,"Gender":stud_gen})
print(mscpart1)
stud_tab=pd.crosstab(mscpart1.Grades,mscpart1.Gender, margins=True)
stud_tab.columns=["Male","Female","row_totals"]
stud_tab.index=["O","A","B","C","D","col_totals"]
observed=stud_tab.iloc[0:5,0:2]
print(observed)
expected=np.outer(stud_tab["row_totals"][0:5],stud_tab.loc["col_totals"][0:2])/100
print(expected)
chi_squared_stat=(((observed-expected)**2)/expected).sum().sum()
print('calculated:',chi_squared_stat)
crit=stats.chi2ppf(q=0.95,df=4)
print('table value ',crit)
if chi_squared_stats>=crit:
   print("HO IS accpeted")
else:
   print("Ho is rejected")
```



```
Python 3.6.1 Shell
File Edit Shell Debug Options Window Help
Python 3.6.1 (v3.6.1:69c0db5, Mar 21 2017, 17:54:52) [MSC v.1900 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
 RESTART: C:/Users/lab/AppData/Local/Programs/Python/Python36-32/Pract 4B.py
   Grades  Gender
0       C  Female
1       O  Female
2       C    Male
3       C    Male
4       B  Female
..    ...     ...
95      B    Male
96      D  Female
97      B  Female
98      A    Male
99      B    Male

[100 rows x 2 columns]
   Male  Female
O    11      12
A     9      13
B     7      11
C    10       8
D    12       7
[[11.27 11.73]
 [10.78 11.22]
 [ 8.82  9.18]
 [ 8.82  9.18]
Ln: 26  Col: 14
```

```
Python 3.6.1 Shell
File Edit Shell Debug Options Window Help
0       C  Female
1       O  Female
2       C    Male
3       C    Male
4       B  Female
..    ...     ...
95      B    Male
96      D  Female
97      B  Female
98      A    Male
99      B    Male

[100 rows x 2 columns]
   Male  Female
O    11      12
A     9      13
B     7      11
C    10       8
D    12       7
[[11.27 11.73]
 [10.78 11.22]
 [ 8.82  9.18]
 [ 8.82  9.18]
 [ 9.31  9.69]]
Calculated :  3.158915138993211
Table Value :  9.487729036781154
H0 is Accepted
>>>
Ln: 26  Col: 14
```

PRACTICAL 5

A. Perform testing of hypothesis using Z Test using one sample test

Perform testing of hypothesis using Z-test.

Use a Z test if:

1. Your sample size is greater than 30. Otherwise, use a t test.
2. Data points should be independent from each other. In other words, one data point isn't related or doesn't affect another data point.
3. Your data should be normally distributed. However, for large sample sizes (over 30) this doesn't always matter.
4. Your data should be randomly selected from a population, where each item has an equal chance of being selected.
5. Sample sizes should be equal if at all possible. Ho - Blood pressure has a mean of 156 units

Program Code for one-sample Z test.

```
from statsmodels.stats import weightstats as stests
import pandas as pd
from scipy import stats
df=pd.read_csv("blood_pressure.csv")
df[['bp_before','bp_after']].describe()
print(df)
ztest,pval=stests.ztest(df['bp_before'],x2=none,value=156)
print(float(pval))
if pval<0.05:
   print("reject null hypothesis")
else:
   print("accept null hypothesis")
```

```
     patient  gender agegrp  bp_before  bp_after
0          1    Male  30-45        143       153
1          2    Male  30-45        163       170
2          3    Male  30-45        153       168
3          4    Male  30-45        153       142
4          5    Male  30-45        146       141
..       ...     ...    ...        ...       ...
115      116  Female    60+        152       152
116      117  Female    60+        161       152
117      118  Female    60+        165       174
118      119  Female    60+        149       151
119      120  Female    60+        185       163

[120 rows x 5 columns]
```

Practical 6:

A. Perform testing of hypothesis using One-way ANOVA.

ANOVA Assumptions

- The dependent variable (SAT scores in our example) should be continuous.
- The independent variables (districts in our example) should be two or more categorical groups.
- There must be different participants in each group with no participant being in more than one group. In our case, each school cannot be in more than one district.
- The dependent variable should be approximately normally distributed for each category.
- Variances of each group are approximately equal.

From our data exploration, we can see that the average SAT scores are quite different for each district. Since we have five different groups, we cannot use the t-test, use the 1-way **ANOVA** test anyway just to understand the concepts.

H0 - There are no significant differences between the groups' mean SAT scores.

$$\mu 1 = \mu 2 = \mu 3 = \mu 4 = \mu 5$$

H1 - There is a significant difference between the groups' mean SAT scores.

If there is at least one group with a significant difference with another group, the null hypothesis will be rejected.

```
import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

fromscipy import stats

data = pd.read_csv("scores.csv")

data.head()

data['Borough'].value_counts()

############### There is no total score column, have to create it.

########In addition, find the mean score of the each district across all schools.

data['total_score'] = data['Average Score (SAT Reading)'] + \

data['Average Score (SAT Math)'] + \
```

```
data['Average Score (SAT Writing)']

data = data[['Borough', 'total_score']].dropna()

x = ['Brooklyn', 'Bronx', 'Manhattan', 'Queens', 'Staten Island']

district_dict = {}

#Assigns each test score series to a dictionary key

for district in x:

district_dict[district] = data[data['Borough'] == district]['total_score']

y = []

yerror = []

#Assigns the mean score and 95% confidence limit to each district

for district in x:

y.append(district_dict[district].mean())

yerror.append(1.96*district_dict[district].std()/np.sqrt(district_dict[district].shape[0]
))

print(district + '_std : {}'.format(district_dict[district].std()))

sns.set(font_scale=1.8)

fig = plt.figure(figsize=(10,5))

ax = sns.barplot(x, y, yerr=yerror)

ax.set_ylabel('Average Total SAT Score')

plt.show()

###################### Perform 1-way ANOVA

print(stats.f_oneway(

district_dict['Brooklyn'], district_dict['Bronx'], \

district_dict['Manhattan'], district_dict['Queens'], \

district_dict['Staten Island']

))

districts = ['Brooklyn', 'Bronx', 'Manhattan', 'Queens', 'Staten Island']

ss_b = 0
```

for d in districts:

ss_b += district_dict[d].shape[0] * \

np.sum((district_dict[d].mean() - data['total_score'].mean())**2)

ss_w = 0

for d in districts:

ss_w += np.sum((district_dict[d] - district_dict[d].mean())**2)

msb = ss_b/4

msw = ss_w/(len(data)-5)

f=msb/msw

print('F_statistic: {}'.format(f))

ss_t = np.sum((data['total_score']-data['total_score'].mean())**2)

eta_squared = ss_b/ss_t

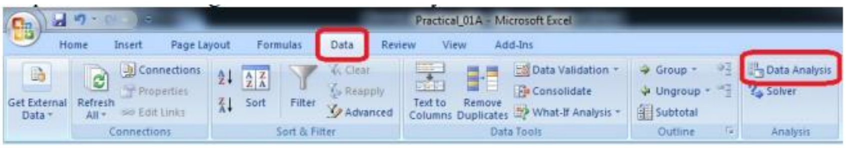
print('eta_squared: {}'.format(eta_squared))

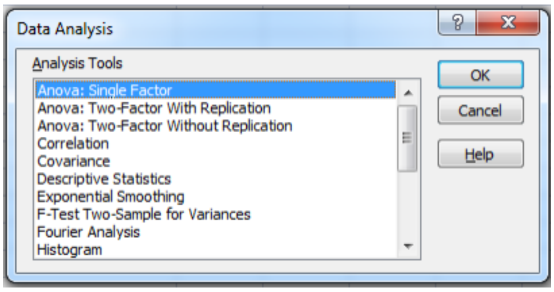
Since the resulting pvalueis less than 0.05. The null hypothesis is rejected and conclude that there is a significant difference between the SAT scores for each district.

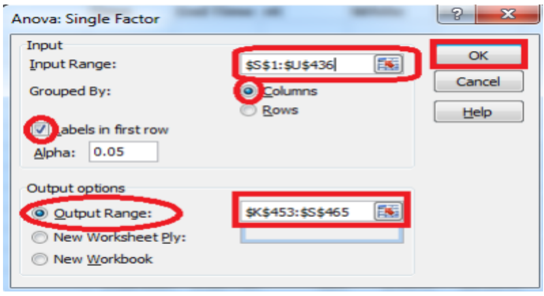
Using Excel

- ☆ H0 - There are no significant differences between the Subject's mean SAT scores.µ1 = µ2 = µ3 = µ4 = µ5
- ☆ H1 - There is a significant difference between the Subject's mean SAT scores.

To perform ANOVA go to data →Data Analysis







Input Range : $S$1:$U$436(Select columns to be analyzed in group)

Anova: Single Factor						
SUMMARY						
Groups	*Count*	*Sum*	*Average*	*Variance*		
Average Score (SAT Math)	375	162354	432.944	5177.144		
Average Score (SAT Reading)	375	159189	424.504	3829.267		
Average Score (SAT Writing)	375	156922	418.4587	4166.522		
ANOVA						
Source of Variation	*SS*	*df*	*MS*	*F*	*P-value*	*F crit*
Between Groups	39700.57	2	19850.28	4.520698	0.01108	3.003745
Within Groups	4926677	1122	4390.977			
Total	4966377	1124				

Since theresulting pvalueis less than 0.05. The null hypothesis (H0) is rejected and conclude that there is a significant difference between the SAT scores for each subject.

B. Perform testing of hypothesis using Two-way ANOVA.

Program Code:

```
import pandas as pd
import statsmodels.api as sm
from statsmodels.formula.api import ols
from statsmodels.stats.anova import anova_lm
from statsmodels.graphics.factorplots import interaction_plot
import matplotlib.pyplot as plt
from scipy import stats
def eta_squared(aov):
 aov['eta_sq'] = 'NaN'
 aov['eta_sq'] = aov[:-1]['sum_sq']/sum(aov['sum_sq'])
 return aov
def omega_squared(aov):
,  mse = aov['sum_sq'][-1]/aov['df'][-1]
 aov['omega_sq'] = 'NaN'
 aov['omega_sq'] = (aov[:-1]['sum_sq']-(aov[:-
1]['df']*mse))/(sum(aov['sum_sq'])+mse)
 return aov
datafile = "ToothGrowth.csv"
data = pd.read_csv(datafile)
fig = interaction_plot(data.dose, data.supp, data.len,
 colors=['red','blue'], markers=['D','^'], ms=10)
N = len(data.len)
df_a = len(data.supp.unique()) - 1
df_b = len(data.dose.unique()) - 1
df_axb = df_a*df_b
df_w = N - (len(data.supp.unique())*len(data.dose.unique()))
```

```
grand_mean = data['len'].mean()
#Sum of Squares A – supp
ssq_a = sum([(data[data.supp ==l].len.mean()-grand_mean)**2 for l in data.supp])
#Sum of Squares B – supp
ssq_b = sum([(data[data.dose ==l].len.mean()-grand_mean)**2 for l in data.dose])
#Sum of Squares Total
ssq_t = sum((data.len - grand_mean)**2)
vc = data[data.supp == 'VC']
oj = data[data.supp == 'OJ']
vc_dose_means = [vc[vc.dose == d].len.mean() for d in vc.dose]
oj_dose_means = [oj[oj.dose == d].len.mean() for d in oj.dose]
ssq_w = sum((oj.len - oj_dose_means)**2) +sum((vc.len - vc_dose_means)**2)
ssq_axb = ssq_t-ssq_a-ssq_b-ssq_w
ms_a = ssq_a/df_a #Mean Square A
ms_b = ssq_b/df_b #Mean Square B
ms_axb = ssq_axb/df_axb #Mean Square AXB
ms_w = ssq_w/df_w
f_a = ms_a/ms_w
f_b = ms_b/ms_w
f_axb = ms_axb/ms_w
p_a = stats.f.sf(f_a, df_a, df_w)
p_b = stats.f.sf(f_b, df_b, df_w)
p_axb = stats.f.sf(f_axb, df_axb, df_w)
results = {'sum_sq':[ssq_a, ssq_b, ssq_axb, ssq_w],
 'df':[df_a, df_b, df_axb, df_w],
 'F':[f_a, f_b, f_axb, 'NaN'],
 'PR(>F)':[p_a, p_b, p_axb, 'NaN']}
columns=['sum_sq', 'df', 'F', 'PR(>F)']
```

aov_table1 = pd.DataFrame(results, columns=columns,

index=['supp', 'dose',

'supp:dose', 'Residual'])

formula = 'len ~ C(supp) + C(dose) + C(supp):C(dose)'

model = ols(formula, data).fit()

aov_table = anova_lm(model, typ=2)

eta_squared(aov_table)

omega_squared(aov_table)

print(aov_table.round(4))

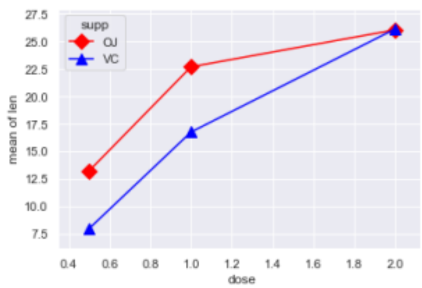
res = model.resid

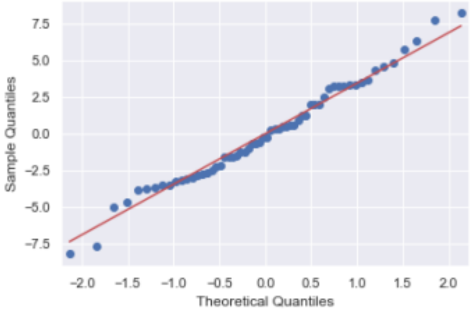
fig = sm.qqplot(res, line='s')

plt.show()

Output:

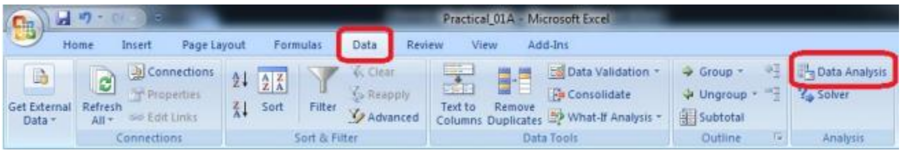
```
In [40]: runfile('K:/Research In Computing/Practical Material/Programs/
Practical_06/Annova_2_Way.py', wdir='K:/Research In Computing/Practical
Material/Programs/Practical_06')
                    sum_sq    df       F  PR(>F)  eta_sq  omega_sq
C(supp)           205.3500   1.0  15.572  0.0002  0.0595    0.0555
C(dose)          2426.4343   2.0  92.000  0.0000  0.7029    0.6926
C(supp):C(dose)   108.3190   2.0   4.107  0.0219  0.0314    0.0236
Residual          712.1060  54.0     NaN     NaN     NaN       NaN
```

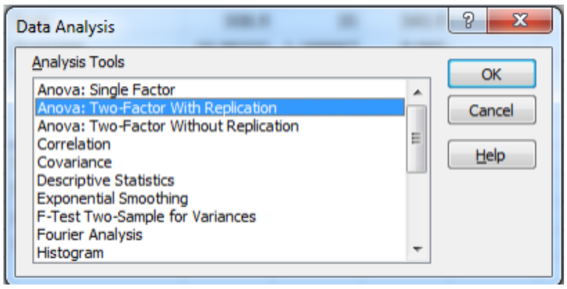


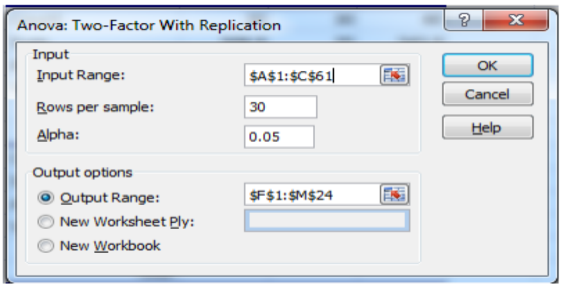


Using Excel:

Go to Data tab → Data Analysis







Input Range - $A$1:$C$61

Rows Per Sample – 30 (Beacause 30 Patients are given each dose)

Alpha – 0.05

Output Range - $F$1:$M$24

Anova: Two-Factor With Replication						
SUMMARY	len	dose	Total			
1						
Count	30	30	60			
Sum	508.9	35	543.9			
Average	16.96333	1.166667	9.065			
Variance	68.32723	0.402299	97.22333			
31						
Count	30	30	60			
Sum	619.9	35	654.9			
Average	20.66333	1.166667	10.915			
Variance	43.63344	0.402299	118.2854			
Total						
Count	60	60				
Sum	1128.8	70				
Average	18.81333	1.166667				
Variance	58.51202	0.39548				
ANOVA						
Source of Variation	*SS*	*df*	*MS*	*F*	*P-value*	*F crit*
Sample	102.675	1	102.675	3.642079	0.058808	3.922879
Columns	9342.145	1	9342.145	331.3838	8.55E-36	3.922879
Interaction	102.675	1	102.675	3.642079	0.058808	3.922879
Within	3270.193	116	28.19132			
Total	12817.69	119				

P-value = 0.0588079 column in the ANOVA Source of Variation table at the bottom of the output. Because the p-values for both medicin dose and interaction are less than our significance level, these factors are statistically significant. On the other hand, the interaction effect is not significant because its p-value (0.0588) is greater than our significance level. Because the interaction effect is not significant, we can focus on only the main effects and not consider the interaction effect of the dose.

Practical 7:

A. Perform the Random sampling for the given data and analyse it.

Example 1: From a population of 10 women and 10 men as given in the table in Figure 1 on the left below, create a random sample of 6 people for Group 1 and a periodic sample consisting of every 3rd woman for Group 2. You need to run the sampling data analysis tool twice, once to create Group 1 and again to create Group 2. For Group 1 you select all 20 population cells as the Input Range and Random as the Sampling Method with 6 for the Random Number of Samples. For Group 2 you select the 10 cells in the Women column as Input Range and Periodic with Period 3.Open existing excel sheet with population data Sample Sheet looks as given below:

	A	B	C	D	E	F	G	H	I	J	K
1	Sr. No	Roll No	Student's Name	Gender	Grade		Sr. No	Roll No	Student's Name	Gender	Grade
2	1	1	Gaborone	m	O		62	3	Maun	f	O
3	2	2	Francistown	m	O		63	7	Tete	f	O
4	3	5	Niamey	m	O		64	9	Chimoio	f	O
5	4	13	Maxixe	m	O		65	11	Pemba	f	O
6	5	16	Tema	m	O		66	14	Chibuto	f	O
7	6	17	Kumasi	m	O		67	25	Mampong	f	O
8	7	34	Blida	m	O		68	36	Tlemcen	f	O
9	8	35	Oran	m	O		69	40	Adrar	f	O
10	9	38	Saefda	m	O		70	41	Tindouf	f	O
11	10	42	Constantine	m	O		71	46	Skikda	f	O
12	11	43	Annaba	m	O		72	47	Ouargla	f	O
13	12	45	Bejaefa	m	O		73	10	Matola	f	D
14	13	48	Medea	m	O		74	20	Legon	f	D
15	14	49	Djelfa	m	O		75	21	Sunyani	f	D
16	15	50	Tipaza	m	O		76	72	Teenas	f	D
17	16	51	Bechar	m	O		77	73	Kouba	f	D
18	17	54	Mostaganem	m	O		78	75	Hussen Dey	f	D
19	18	55	Tiaret	m	O		79	77	Khenchela	f	D
20	19	56	Bouira	m	O		80	82	Hassi Bahbah	f	D
21	20	59	Tebessa	m	O		81	84	Baraki	f	D
22	21	61	El Harrach	m	O		82	91	Boudouaou	f	D
23	22	62	Mila	m	O		83	95	Tadjenanet	f	D
24	23	65	Fouka	m	O		84	4	Molepolole	f	C

Set Cell O1 = Male and Cell O2 = Female

To generate a random sample for male students from given population go to Cell O1 and type=INDEX(E$2:E$62,RANK(B2,B$2:B$62))

Drag teh formula to the desired no of cell to select random sample.

Now, to generate a random sample for female students go to cell P1 and type =INDEX(K$2:K$40,RANK(H2,H$2:H$40))

Drag teh formula to the desired no of cell to select random sample.

Output:

O	P
Male	Female
A	A
A	A
A	A
B	A
C	B
C	C
D	C
D	C
D	C
D	C
D	D
D	A
D	B
D	B
O	D
O	D
O	D
O	D
O	O
O	O
O	O
O	O
O	A

B. Perform the Stratified sampling for the given data and analyse it.

We are to carry out a hypothetical housing quality survey across Lagos state, Nigeria. And we looking at a total of 5000 houses (hypothetically). We don’t just go to one local government and select 5000 houses, rather we ensure that the 5000 houses are a representative of the whole 20 local government areas Lagos state is comprised of. This is called stratified sampling. The population is divided into homogenous strata and the right number of instances is sampled from each stratum to guarantee that the test-set (which in this case is the 5000 houses) is a representative of the overall population. If we used random sampling, there would be a significant chance of having bias in the survey results.

Code:-

```
import pandas as pd

import numpy as np

import matplotlib

import matplotlib.pyplot as plt

plt.rcParams['axes.labelsize'] = 14

plt.rcParams['xtick.labelsize'] = 12

plt.rcParams['ytick.labelsize'] = 12

import seaborn as sns

color = sns.color_palette()

sns.set_style('darkgrid')

importsklearn

fromsklearn.model_selection import train_test_split

housing =pd.read_csv('housing.csv')

print(housing.head())

print(housing.info())

#creating a heatmap of the attributes in the dataset

correlation_matrix = housing.corr()

plt.subplots(figsize=(8,6))

sns.heatmap(correlation_matrix, center=0, annot=True, linewidths=.3)
```

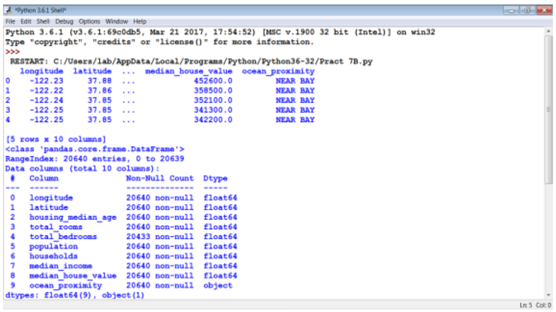
corr =housing.corr()

print(corr['median_house_value'].sort_values(ascending=False))

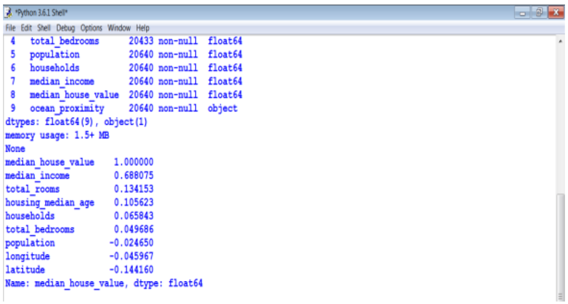
sns.distplot(housing.median_income)

plt.show()

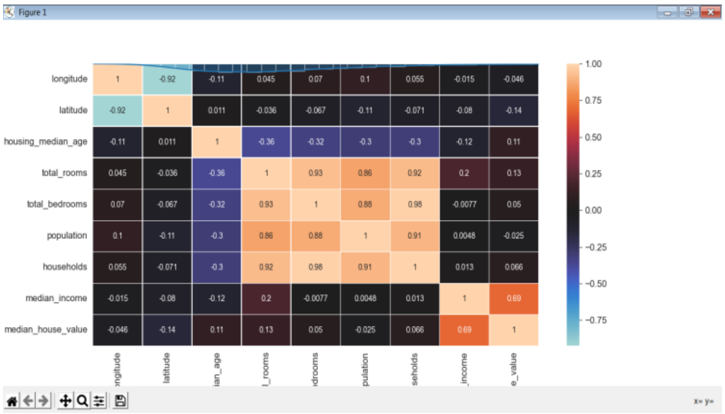
output:



```
Python 3.6.1 (v3.6.1:69c0db5, Mar 21 2017, 17:54:52) [MSC v.1900 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
 RESTART: C:/Users/lab/AppData/Local/Programs/Python/Python36-32/Pract 7B.py
   longitude  latitude  ...  median_house_value  ocean_proximity
0    -122.23     37.88  ...            452600.0         NEAR BAY
1    -122.22     37.86  ...            358500.0         NEAR BAY
2    -122.24     37.85  ...            352100.0         NEAR BAY
3    -122.25     37.85  ...            341300.0         NEAR BAY
4    -122.25     37.85  ...            342200.0         NEAR BAY

[5 rows x 10 columns]
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20640 entries, 0 to 20639
Data columns (total 10 columns):
 #   Column              Non-Null Count  Dtype
---  ------              --------------  -----
 0   longitude           20640 non-null  float64
 1   latitude            20640 non-null  float64
 2   housing_median_age  20640 non-null  float64
 3   total_rooms         20640 non-null  float64
 4   total_bedrooms      20433 non-null  float64
 5   population          20640 non-null  float64
 6   households          20640 non-null  float64
 7   median_income       20640 non-null  float64
 8   median_house_value  20640 non-null  float64
 9   ocean_proximity     20640 non-null  object
dtypes: float64(9), object(1)
```



```
 4   total_bedrooms      20433 non-null  float64
 5   population          20640 non-null  float64
 6   households          20640 non-null  float64
 7   median_income       20640 non-null  float64
 8   median_house_value  20640 non-null  float64
 9   ocean_proximity     20640 non-null  object
dtypes: float64(9), object(1)
memory usage: 1.5+ MB
None
median_house_value    1.000000
median_income         0.688075
total_rooms           0.134153
housing_median_age    0.105623
households            0.065843
total_bedrooms        0.049686
population           -0.024650
longitude            -0.045967
latitude             -0.144160
Name: median_house_value, dtype: float64
```



Practical 8:

Write a program for computing different correlation.

A. Positive Correlation

Let’s take a look at a positive correlation. Numpy implements a corrcoef() function that returns a matrix of correlations of x with x, x with y, y with x and y with y. We’re interested in the values of correlation of x with y (so position (1, 0) or (0, 1)).

Import numpy as np

importmatplotlib.pyplot as plt

np.random.seed(1)

1000 random integers between 0 and 50

x = np.random.randint(0, 50, 1000)

Positive Correlation with some noise

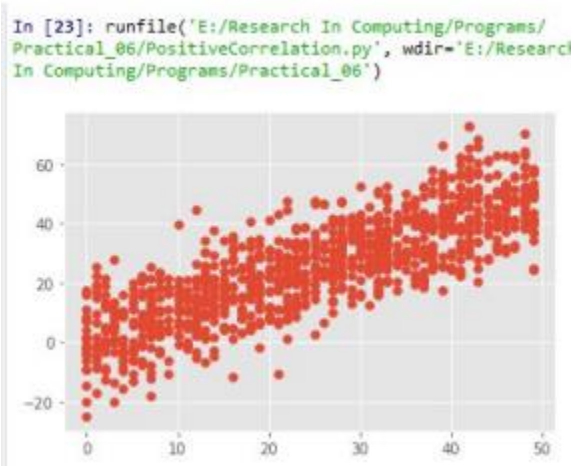
y = x + np.random.normal(0, 10, 1000)

np.corrcoef(x, y)

matplotlib.style.use('ggplot')

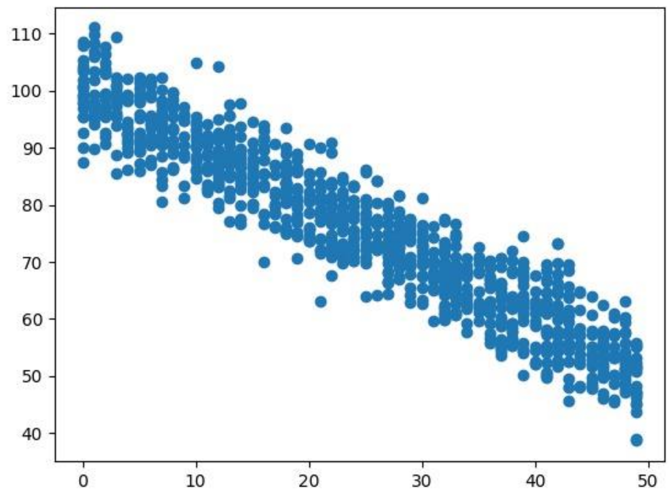
plt.scatter(x, y)

plt.show()



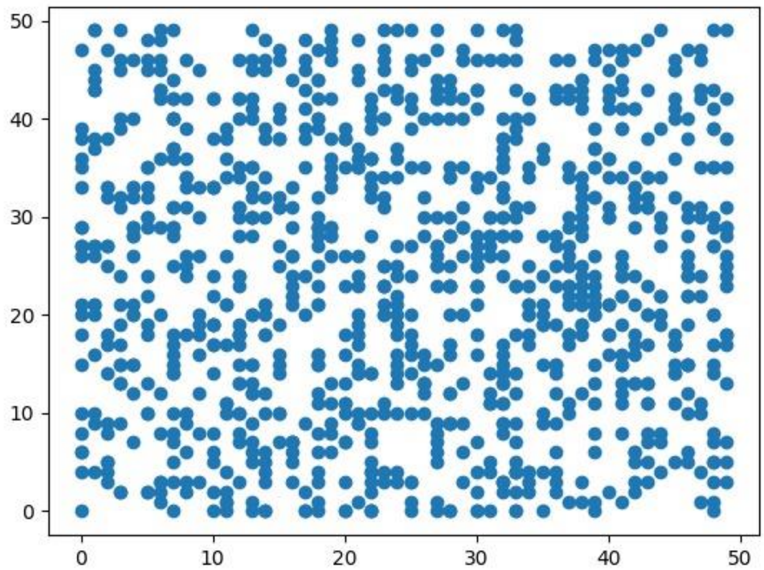
B. Negative Correlation:

```
Import numpy as np
Import matplotlib.pyplot as plt
np.random.seed(1)
# 1000 random integers between 0 and 50
x = np.random.randint(0, 50, 1000)
# Negative Correlation with some noise
y = 100 - x + np.random.normal(0, 5, 1000)
np.corrcoef(x, y)
plt.scatter(x, y)
plt.show()
```



C. No/Weak Correlation:

```
importnumpy as np
importmatplotlib.pyplot as plt
np.random.seed(1)
x = np.random.randint(0, 50, 1000)
y = np.random.randint(0, 50, 1000)
np.corrcoef(x, y)
plt.scatter(x, y)
plt.show()
```

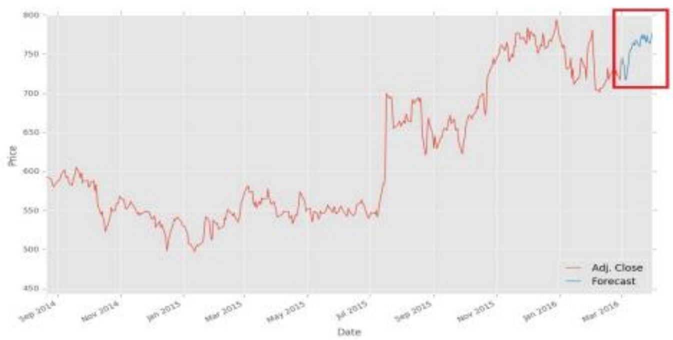


Practical 9:

A. Write a program to Perform linear regression for prediction.

```
import Quandl, math
import numpy as np
import pandas as pd
from sklearn import preprocessing, cross_validation, svm
from sklearn.linear_model import LinearRegression
import matplotlib.pyplot as plt
from matplotlib import style
import datetime
style.use('ggplot')
df = Quandl.get("WIKI/GOOGL")
df = df[['Adj. Open', 'Adj. High', 'Adj. Low', 'Adj. Close', 'Adj. Volume']]
df['HL_PCT'] = (df['Adj. High'] - df['Adj. Low']) / df['Adj. Close'] * 100.0
df['PCT_change'] = (df['Adj. Close'] - df['Adj. Open']) / df['Adj. Open'] * 100.0
df = df[['Adj. Close', 'HL_PCT', 'PCT_change', 'Adj. Volume']]
forecast_col = 'Adj. Close'
df.fillna(value=-99999, inplace=True)
forecast_out = int(math.ceil(0.01 * len(df)))
df['label'] = df[forecast_col].shift(-forecast_out)
X = np.array(df.drop(['label'], 1))
X = preprocessing.scale(X)
X_lately = X[-forecast_out:]
X = X[:-forecast_out]
df.dropna(inplace=True)
y = np.array(df['label'])
X_train, X_test, y_train, y_test = cross_validation.train_test_split(X, y, test_size=0.2)
```

```
clf = LinearRegression(n_jobs=-1)
clf.fit(X_train, y_train)
confidence = clf.score(X_test, y_test)
forecast_set = clf.predict(X_lately)
df['Forecast'] = np.nan
last_date = df.iloc[-1].name
last_unix = last_date.timestamp()
one_day = 86400
next_unix = last_unix + one_day
for i in forecast_set:
 next_date = datetime.datetime.fromtimestamp(next_unix)
 next_unix += 86400
 df.loc[next_date] = [np.nan for _ in range(len(df.columns)-1)]+[i]
df['Adj. Close'].plot()
df['Forecast'].plot()
plt.legend(loc=4)
plt.xlabel('Date')
plt.ylabel('Price')
plt.show()
```



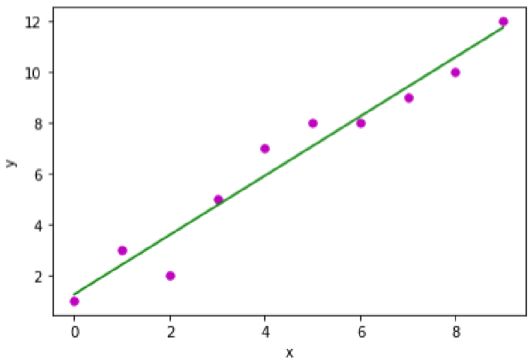
B. Perform polynomial regression for prediction.

```
importnumpy as np
importmatplotlib.pyplot as plt
defestimate_coef(x, y):
# number of observations/points
n = np.size(x)
# mean of x and y vector
m_x, m_y = np.mean(x), np.mean(y)
# calculating cross-deviation and deviation about x
SS_xy = np.sum(y*x) - n*m_y*m_x
SS_xx = np.sum(x*x) - n*m_x*m_x
# calculating regression coefficients
b_1 = SS_xy / SS_xx
b_0 = m_y - b_1*m_x
return(b_0, b_1)
defplot_regression_line(x, y, b):
# plotting the actual points as scatter plot
plt.scatter(x, y, color = "m",
marker = "o", s = 30)
# predicted response vector
y_pred = b[0] + b[1]*x
# plotting the regression line
plt.plot(x, y_pred, color = "g")
# putting labels
plt.xlabel('x')
plt.ylabel('y')
# function to show plot
plt.show()
```

```
def main():
# observations
x = np.array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])
y = np.array([1, 3, 2, 5, 7, 8, 8, 9, 10, 12])
# estimating coefficients
b = estimate_coef(x, y)
print("Estimated coefficients:\nb_0 = {} b_1 = {}".format(b[0], b[1]))
# plotting regression line
plot_regression_line(x, y, b)
if __name__ == "__main__":
main()
```

Output:

```
In [22]: runfile('E:/Research In Computing/Programs/
Practical_07/Practical_7B.py', wdir='E:/Research In
Computing/Programs/Practical_07')
Estimated coefficients:
b_0 = 1.2363636363636363  b_1 = 1.1696969696969697
```



Practical 10:

A. Write a program for multiple linear regression analysis.

Step #1: Data Pre Processing

- ☆ Importing The Libraries.
- ☆ Importing the Data Set.
- ☆ Encoding the Categorical Data.
- ☆ Avoiding the Dummy Variable Trap.
- ☆ Splitting the Data set into Training Set and Test Set.

Step #2: Fitting Multiple Linear Regression to the Training set

Step #3: Predicting the Test set results.

```
importnumpy as np

importmatplotlib as mpl

from mpl_toolkits.mplot3d import Axes3D

importmatplotlib.pyplot as plt

defgenerate_dataset(n):

x = []

y = []

random_x1 = np.random.rand()

random_x2 = np.random.rand()

fori in range(n):

x1 = i

x2 = i/2 + np.random.rand()*n

x.append([1, x1, x2])

y.append(random_x1 * x1 + random_x2 * x2 + 1)

returnnp.array(x), np.array(y)

x, y = generate_dataset(200)

mpl.rcParams['legend.fontsize'] = 12

fig = plt.figure()

ax = fig.gca(projection ='3d')
```

```
ax.scatter(x[:, 1], x[:, 2], y, label ='y', s = 5)
ax.legend()
ax.view_init(45, 0)
plt.show()
defmse(coef, x, y):
returnnp.mean((np.dot(x, coef) - y)**2)/2
def gradients(coef, x, y):
returnnp.mean(x.transpose()*(np.dot(x, coef) - y), axis = 1)
defmultilinear_regression(coef, x, y, lr, b1 = 0.9, b2 = 0.999, epsilon = 1e-8):
prev_error = 0
m_coef = np.zeros(coef.shape)
v_coef = np.zeros(coef.shape)
moment_m_coef = np.zeros(coef.shape)
moment_v_coef = np.zeros(coef.shape)
t = 0
while True:
error = mse(coef, x, y)
if abs(error - prev_error) <= epsilon:
break
prev_error = error
grad = gradients(coef, x, y)
t += 1
m_coef = b1 * m_coef + (1-b1)*grad
v_coef = b2 * v_coef + (1-b2)*grad**2
moment_m_coef = m_coef / (1-b1**t)
moment_v_coef = v_coef / (1-b2**t)
delta = ((lr / moment_v_coef**0.5 + 1e-8) *
(b1 * moment_m_coef + (1-b1)*grad/(1-b1**t)))
```

```
coef = np.subtract(coef, delta)
returncoef
coef = np.array([0, 0, 0])
c = multilinear_regression(coef, x, y, 1e-1)
fig = plt.figure()
ax = fig.gca(projection ='3d')
ax.scatter(x[:, 1], x[:, 2], y, label ='y',
s = 5, color ="dodgerblue")
ax.scatter(x[:, 1], x[:, 2], c[0] + c[1]*x[:, 1] + c[2]*x[:, 2],
label ='regression', s = 5, color ="orange")
ax.view_init(45, 0)
ax.legend()
plt.show()
```

Output: