Name: Allan Rodrigues
Class: TE IT A
Roll no: 59
Pid:191104

St. Francis Institute of Technology, Mumbai-400 103
Department of Information Technology

A.Y. 2021-2022
Class: TE-ITA/B, Semester: VI

Subject: **Data Science Lab**

Experiment – 4: To implement statistical hypothesis tests.

1. **Aim: To implement statistical hypothesis tests Objectives:**
2. **After study of this experiment, the student will be able to**
    - Understand what are the various statistical tests available for evaluating probability of distribution of data
    - Steps involved in hypothesis testing
    - Various functions available in scipy package for hypothesis testing
    - Understand and practice analytical methods for solving real life problems based on Statistical analysis
3. **Outcomes:** After study of this experiment, the student will be able to
    - Understand the techniques of performing statistical tests available for evaluating probability of distribution of data
    - Various tests available in scipy package for the tests
    - Understand and practice analytical methods for solving real life problems based on Statistical analysis
    - Analyze the data using different statistical techniques

4. **Prerequisite:** Fundamentals of Python Programming and Database Management System.
5. **Requirements:** Python Installation, Personal Computer, Windows operating system, Internet Connection, Microsoft Word.

6. **Pre-Experiment Exercise:**

Brief Theory:

1) Normality tests

An important decision point when working with a sample of data is whether to use parametric or nonparametric statistical methods.

Parametric statistical methods assume Gaussian distribution. If a data sample is not Gaussian, then nonparametric statistical methods must be used.

Range of techniques that you can use to check if your data sample deviates from a Gaussian distribution, called normality tests.

--How whether a sample is normal dictates the types of statistical methods to use with a data sample.

--Graphical methods for qualifying deviations from normal, such as histograms and the Q-Q plot.

--Statistical normality tests for quantifying deviations from normal.

Normality Assumption There is also some middle ground where we can assume that the data is Gaussian-enough to use parametric methods or that we can use data preparation techniques to transform the data to be sufficiently Gaussian to use the parametric methods.

There are three main areas where you may need to make this evaluation of a data sample in a machine learning project; they are:

--Input data to the model in the case of fitting models.

--Model evaluation results in the case of model selection.

--Residual errors from model predictions in the case of regression.

Two classes of techniques for checking whether a sample of data is Gaussian:

1. Graphical Methods. These are methods for plotting the data and qualitatively evaluating whether the data looks Gaussian.
2. Statistical Tests. These are methods that calculate statistics on the data and quantify how likely it is that the data was drawn from a Gaussian distribution.

a) Shapiro-Wilk Test

Tests whether a data sample has a Gaussian distribution.

Assumptions

- Observations in each sample are independent and identically distributed (iid).

Interpretation

- H0: the sample has a Gaussian distribution.
- H1: the sample does not have a Gaussian distribution.

This test is named for Samuel Shapiro and Martin Wilk. The shapiro() SciPy function will calculate the Shapiro-Wilk on a given dataset. The function returns both the W-statistic calculated by the test and the p-value.

b) D'Agostino's K^2 Test

- The D'Agostino's K^2 test calculates summary statistics from the data, namely kurtosis and skewness, to determine if the data distribution departs from the normal distribution, named for Ralph D'Agostino.
- Skew is a quantification of how much a distribution is pushed left or right, a measure of asymmetry in the distribution.
- Kurtosis quantifies how much of the distribution is in the tail.
- It is a simple and commonly used statistical test for normality.(pointed means positive and flat means negative)

- The D'Agostino's K^2 test is available via the normaltest() SciPy function and returns the test statistic and the p-value.

2) Correlation tests

Variables within a dataset can be related for lots of reasons.

For example:

One variable could cause or depend on the values of another variable. One variable could be lightly associated with another variable. Two variables could depend on a third unknown variable.

A correlation could be positive, meaning both variables move in the same direction, or negative, meaning that when one variable's value increases, the other variables' values decrease. Correlation can also be neutral or zero, meaning that the variables are unrelated.

Positive Correlation: both variables change in the same direction. Neutral Correlation: No relationship in the change of the variables. Negative Correlation: variables change in opposite directions.

a) **Pearson's Correlation** The Pearson correlation coefficient (named for Karl Pearson) can be used to summarize the strength of the linear relationship between two data samples.

   The Pearson's correlation coefficient is calculated as the covariance of the two variables divided by the product of the standard deviation of each data sample. It is the normalization of the covariance between the two variables to give an interpretable score

b) **Spearman's Correlation** (non paramtric test)

   Two variables may be related by a nonlinear relationship, such that the relationship is stronger or weaker across the distribution of the variables.

   Further, the two variables being considered may have a non-Gaussian distribution.

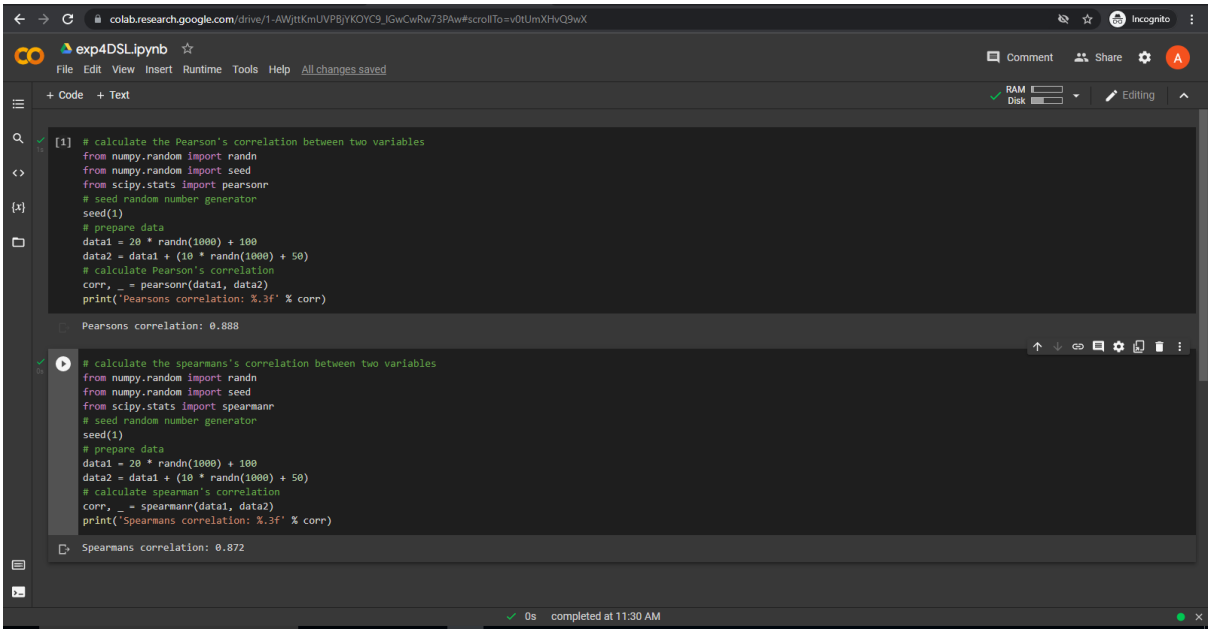
   If you are unsure of the distribution and possible relationships between two variables, Spearman correlation coefficient is a good tool to use.

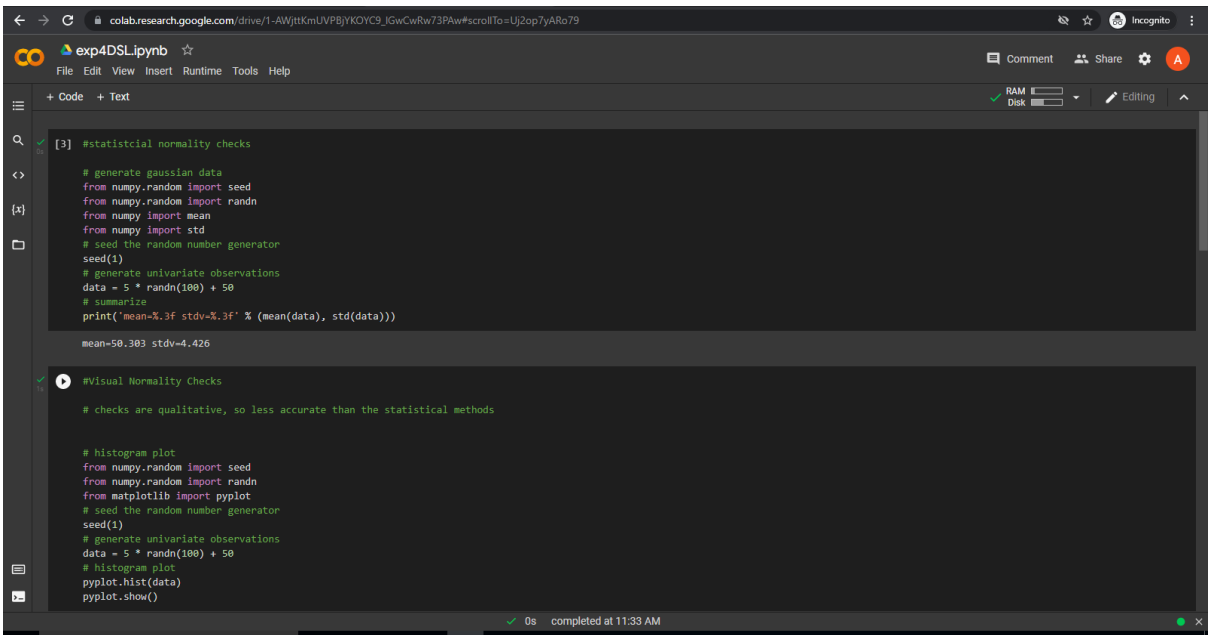
   The spearmanr() SciPy function can be used to calculate the Spearman's correlation coefficient between two data samples with the same length.

7. Laboratory Exercise

A. **Procedure:** (separate sheet of codes is attached with the experiment)

Paste Screenshots of performance of the above commands.


colab.research.google.com/drive/1-AWjttKmUVPBjYKOYC9_IGwCwRw73PAw#scrollTo=v0tUmXHvQ9wX
Incognito
exp4DSL.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
A
+ Code + Text
RAM
Disk
Editing
[1] # calculate the Pearson's correlation between two variables
from numpy.random import randn
from numpy.random import seed
from scipy.stats import pearsonr
# seed random number generator
seed(1)
# prepare data
data1 = 20 * randn(1000) + 100
data2 = data1 + (10 * randn(1000) + 50)
# calculate Pearson's correlation
corr, _ = pearsonr(data1, data2)
print('Pearsons correlation: %.3f' % corr)
Pearsons correlation: 0.888
# calculate the spearmans's correlation between two variables
from numpy.random import randn
from numpy.random import seed
from scipy.stats import spearmanr
# seed random number generator
seed(1)
# prepare data
data1 = 20 * randn(1000) + 100
data2 = data1 + (10 * randn(1000) + 50)
# calculate spearman's correlation
corr, _ = spearmanr(data1, data2)
print('Spearmans correlation: %.3f' % corr)
Spearmans correlation: 0.872
0s completed at 11:30 AM


colab.research.google.com/drive/1-AWjttKmUVPBjYKOYC9_IGwCwRw73PAw#scrollTo=Uj2op7yARo79
Incognito
exp4DSL.ipynb
File Edit View Insert Runtime Tools Help
Comment
Share
A
+ Code + Text
RAM
Disk
Editing
[3] #statistcial normality checks
# generate gaussian data
from numpy.random import seed
from numpy.random import randn
from numpy import mean
from numpy import std
# seed the random number generator
seed(1)
# generate univariate observations
data = 5 * randn(100) + 50
# summarize
print('mean=%.3f stdv=%.3f' % (mean(data), std(data)))
mean=50.303 stdv=4.426
#Visual Normality Checks
# checks are qualitative, so less accurate than the statistical methods
# histogram plot
from numpy.random import seed
from numpy.random import randn
from matplotlib import pyplot
# seed the random number generator
seed(1)
# generate univariate observations
data = 5 * randn(100) + 50
# histogram plot
pyplot.hist(data)
pyplot.show()
0s completed at 11:33 AM

colab.research.google.com/drive/1-AWjttKmUVPBjYKOYC9_IGwCwRw73PAw#scrollTo=Uj2op7yARo79

exp4DSL.ipynb

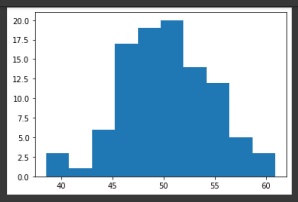
File Edit View Insert Runtime Tools Help All changes saved

Comment Share

+ Code + Text

RAM Disk Editing

[4]




```
[ ] #Statistical Normality Tests
```

Double-click (or enter) to edit

```
[5]
# Shapiro-Wilk Test
from numpy.random import seed
from numpy.random import randn
from scipy.stats import shapiro
# seed the random number generator
seed(1)
# generate univariate observations
data = 5 * randn(100) + 50
# normality test
stat, p = shapiro(data)
print('Statistics=%.3f, p=%.3f' % (stat, p))
# interpret
```

0s completed at 11:33 AM

colab.research.google.com/drive/1-AWjttKmUVPBjYKOYC9_IGwCwRw73PAw#scrollTo=Uj2op7yARo79

exp4DSL.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

+ Code + Text

RAM Disk Editing

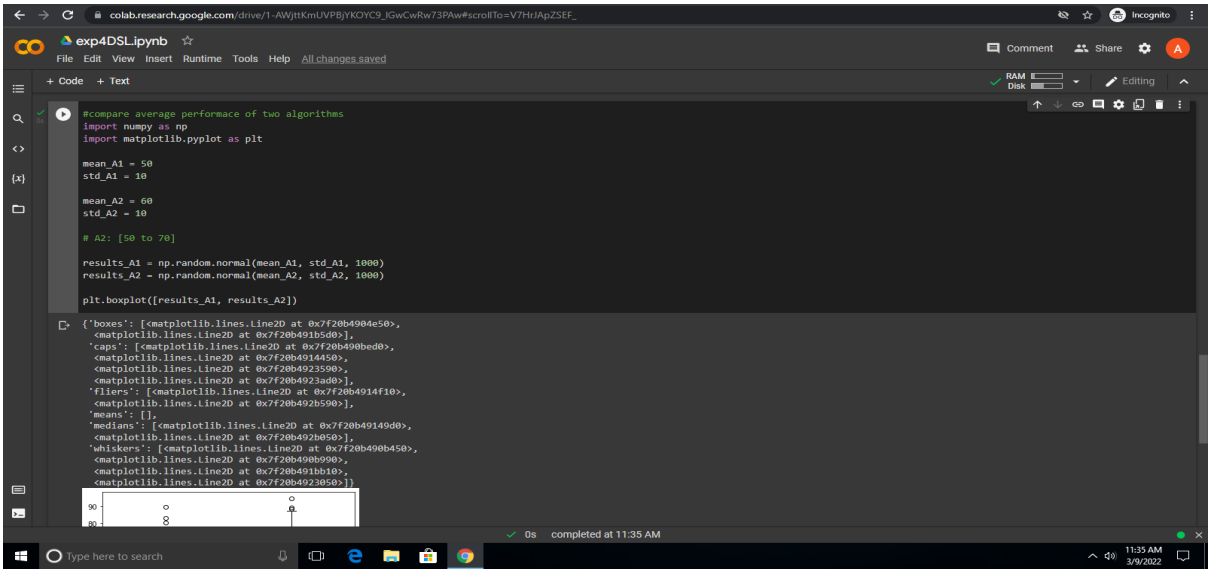
```
[5]
# normality test
stat, p = shapiro(data)
print('Statistics=%.3f, p=%.3f' % (stat, p))
# interpret
alpha = 0.05
if p > alpha:
  print('Sample looks Gaussian (fail to reject H0)')
else:
  print('Sample does not look Gaussian (reject H0)')
```

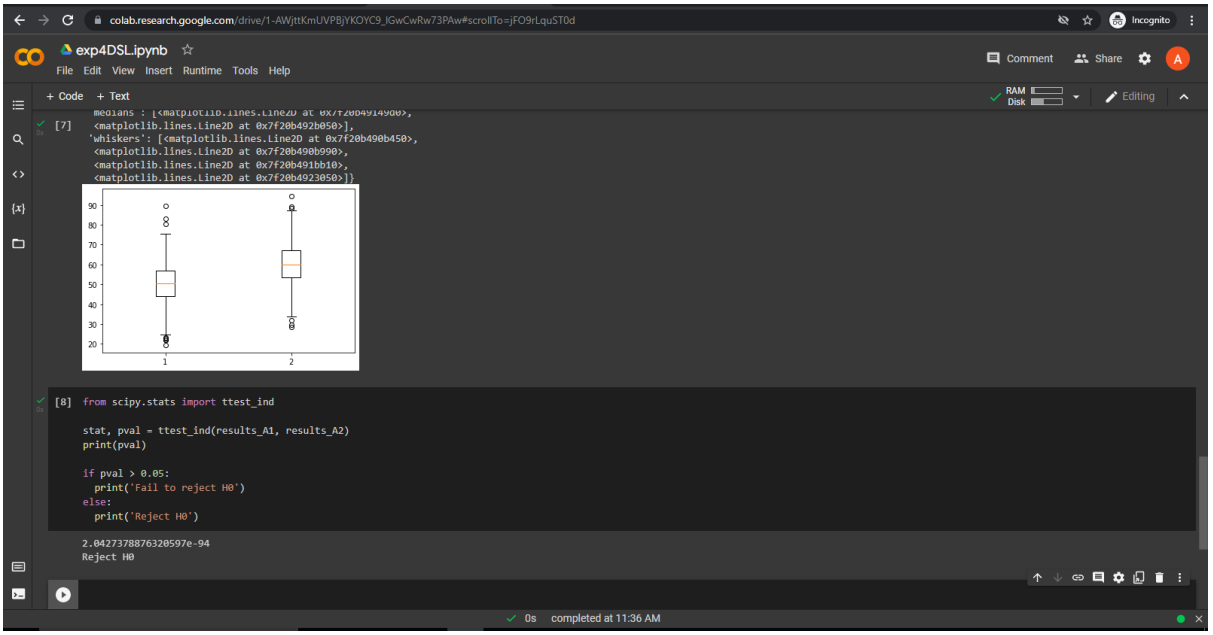
```
Statistics=0.992, p=0.822
Sample looks Gaussian (fail to reject H0)
```

```
[6]
# D'Agostino and Pearson's Test
from numpy.random import seed
from numpy.random import randn
from scipy.stats import normaltest
# seed the random number generator
seed(1)
# generate univariate observations
data = 5 * randn(100) + 50
# normality test
stat, p = normaltest(data)
print('Statistics=%.3f, p=%.3f' % (stat, p))
# interpret
alpha = 0.05
if p > alpha:
  print('Sample looks Gaussian (fail to reject H0)')
else:
  print('Sample does not look Gaussian (reject H0)')
```

```
Statistics=0.102, p=0.950
Sample looks Gaussian (fail to reject H0)
```

0s completed at 11:33 AM



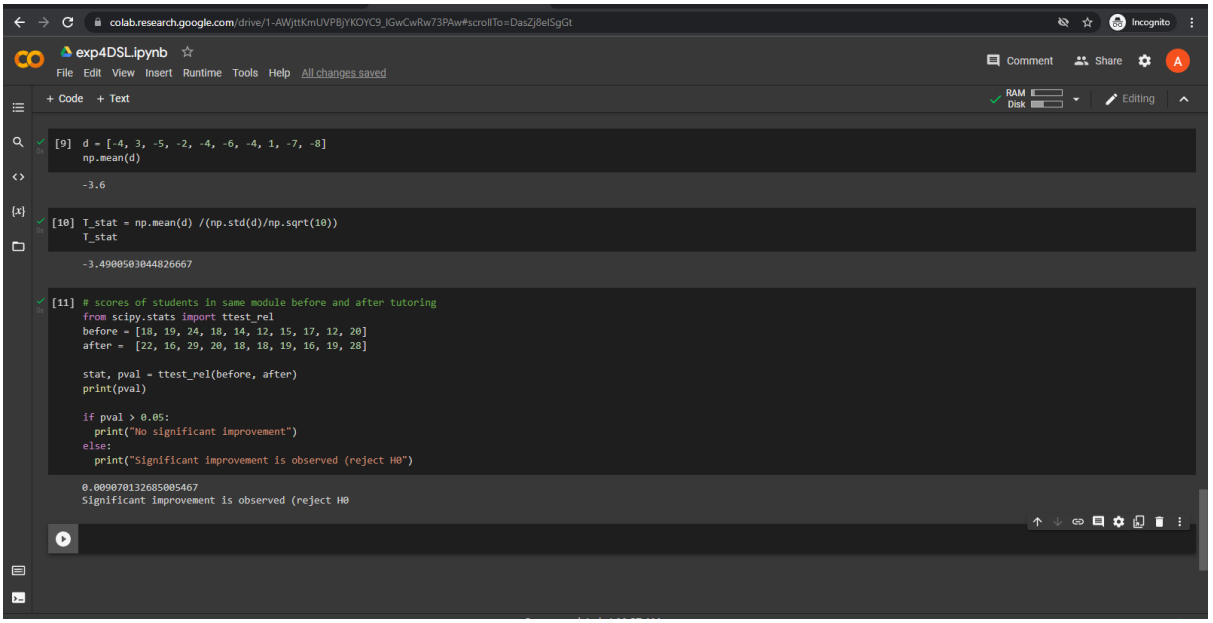


8. Post-Experiments Exercise

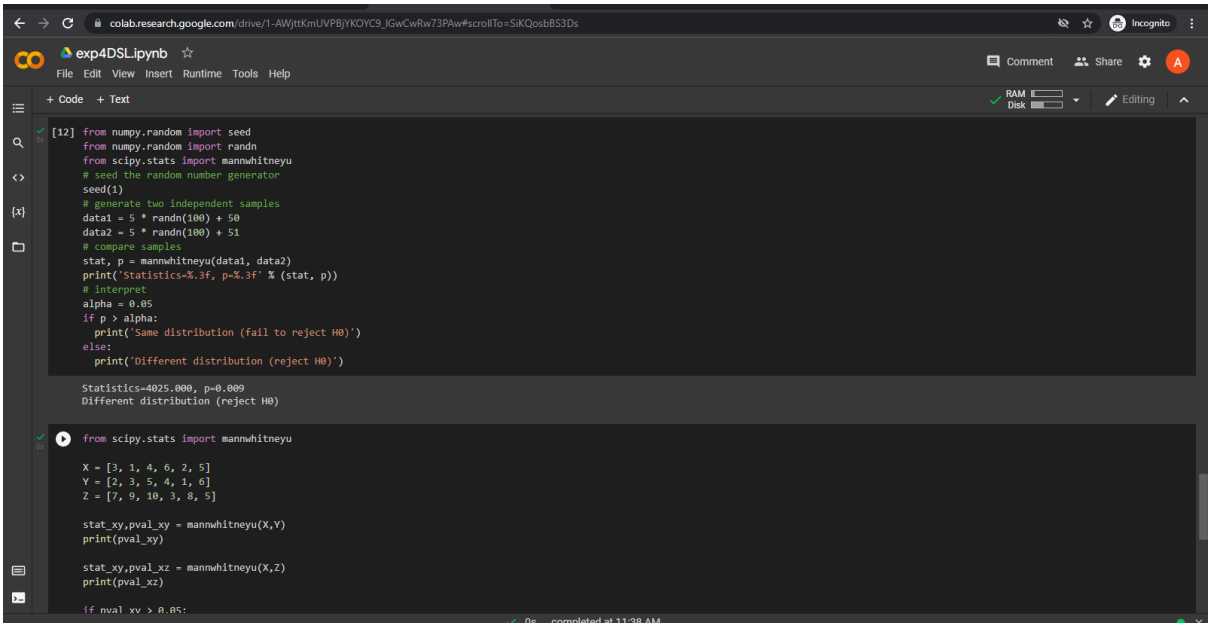
A. Extended Theory: (Soft Copy)

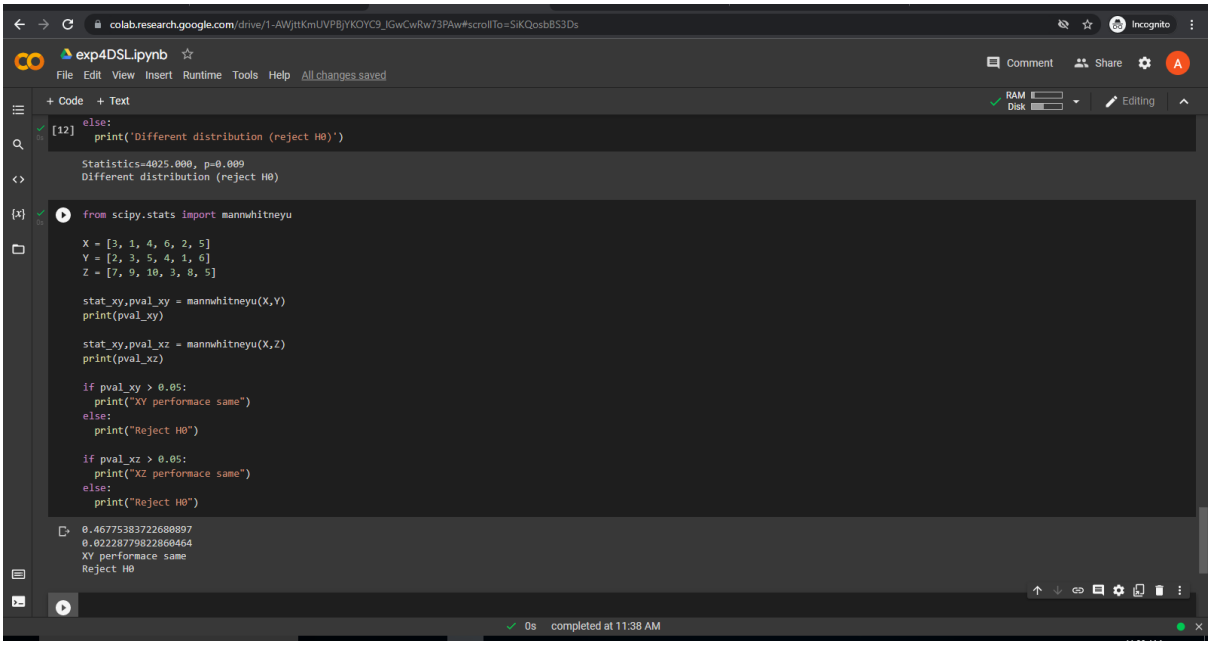
Mention Parametric Statistical Hypothesis Tests

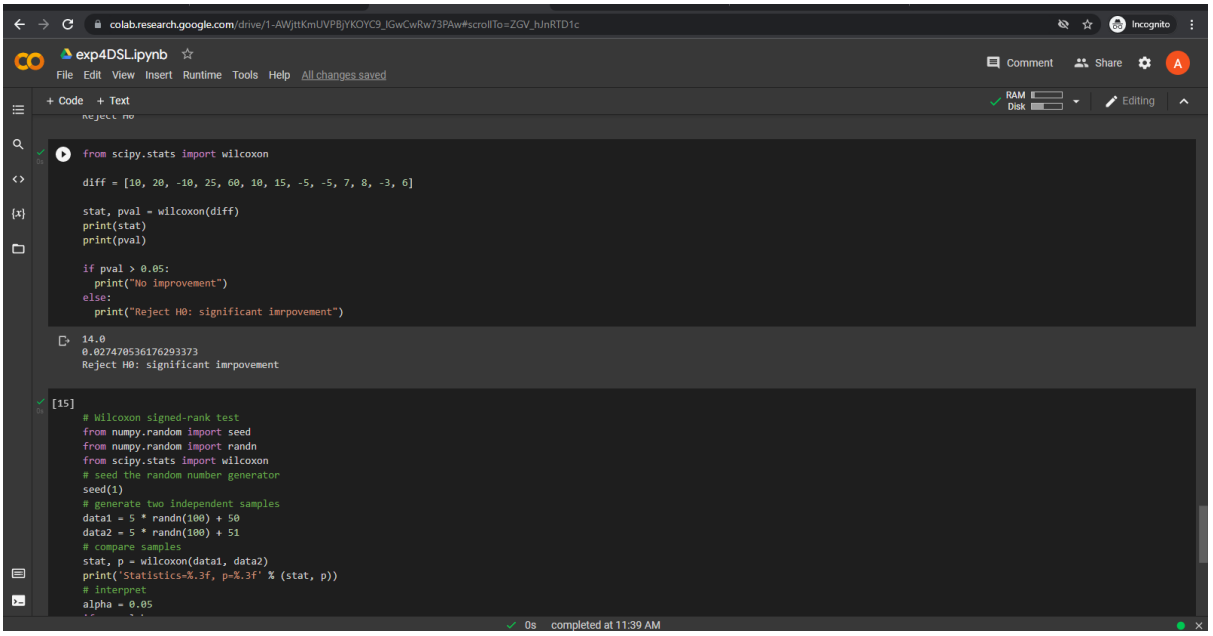
- Student's t-test

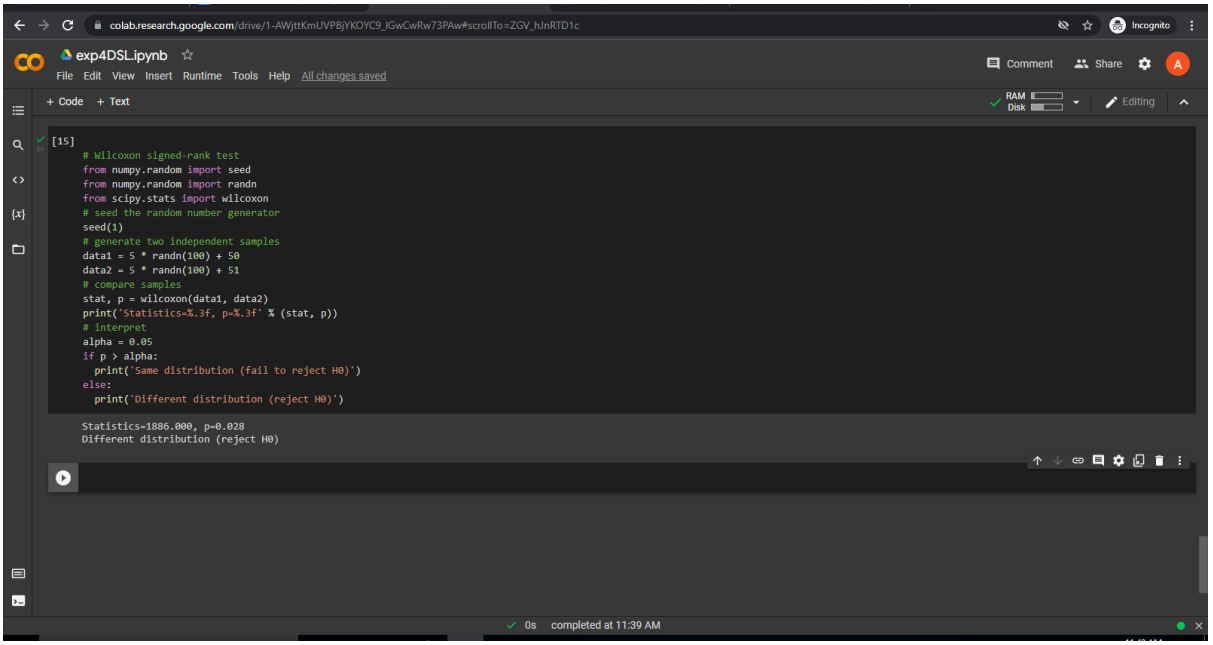


Nonparametric Statistical Hypothesis Tests
- Mann-Whitney U Test




colab.research.google.com/drive/1-AWjttKmUVPBjYKOYC9_lGwCwRw73PAw#scrollTo=SiKQosbBS3Ds
Incognito
exp4DSL.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
RAM
Disk
Editing
[12]
else:
print('Different distribution (reject H0)')
Statistics=4025.000, p=0.009
Different distribution (reject H0)
from scipy.stats import mannwhitneyu
X = [3, 1, 4, 6, 2, 5]
Y = [2, 3, 5, 4, 1, 6]
Z = [7, 9, 10, 3, 8, 5]
stat_xy,pval_xy = mannwhitneyu(X,Y)
print(pval_xy)
stat_xy,pval_xz = mannwhitneyu(X,Z)
print(pval_xz)
if pval_xy > 0.05:
print("XY performace same")
else:
print("Reject H0")
if pval_xz > 0.05:
print("XZ performace same")
else:
print("Reject H0")
0.46775383722680897
0.02228779822860464
XY performace same
Reject H0
0s completed at 11:38 AM


colab.research.google.com/drive/1-AWjttKmUVPBjYKOYC9_lGwCwRw73PAw#scrollTo=ZGV_hJnRTD1c
Incognito
exp4DSL.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
RAM
Disk
Editing
from scipy.stats import wilcoxon
diff = [10, 20, -10, 25, 60, 10, 15, -5, -5, 7, 8, -3, 6]
stat, pval = wilcoxon(diff)
print(stat)
print(pval)
if pval > 0.05:
print("No improvement")
else:
print("Reject H0: significant imrpovement")
14.0
0.027470536176293373
Reject H0: significant imrpovement
[15]
# Wilcoxon signed-rank test
from numpy.random import seed
from numpy.random import randn
from scipy.stats import wilcoxon
# seed the random number generator
seed(1)
# generate two independent samples
data1 = 5 * randn(100) + 50
data2 = 5 * randn(100) + 51
# compare samples
stat, p = wilcoxon(data1, data2)
print('Statistics=%.3f, p=%.3f' % (stat, p))
# interpret
alpha = 0.05
0s completed at 11:39 AM



B. Questions:

Q.1) what is a normality test?

Q.2) What is the difference between parametric statistical methods and non-parametric statistical methods?

Allan Rodrigues TE IT A-59

ST. FRANCIS INSTITUTE OF TECHNOLOGY
(Engineering College)

DATE : PAGE NO. :

Exp 4 - DSL

Q.8

B.

1) What is normality test?

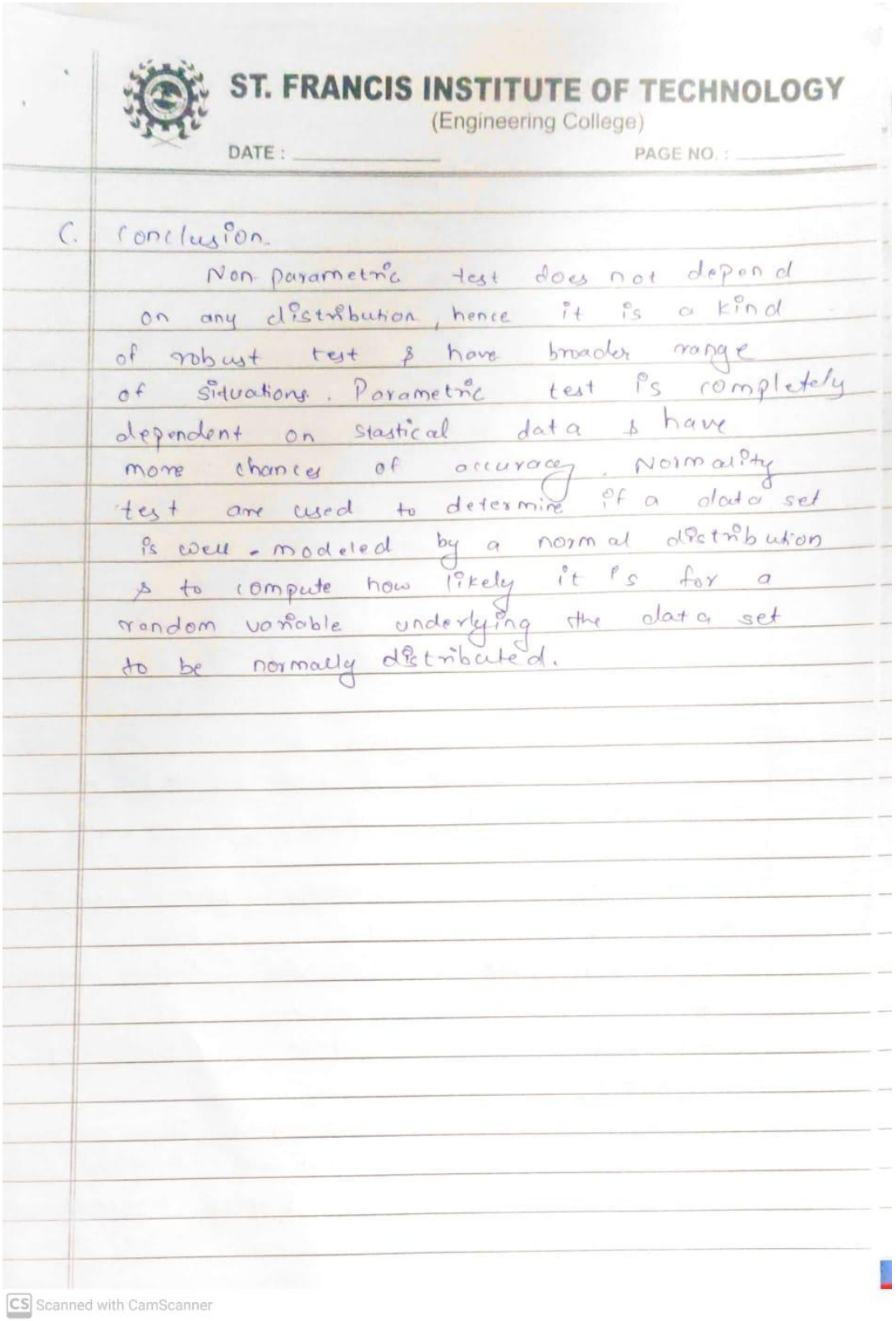
→ It is a test for the normality of your data. But what does it mean, Normality refers to a specific statistical distribution called a normal distribution or sometimes the Gaussian distribution or bell-shaped curve. The normal distribution is a symmetrical continuous distribution defined by the mean and standard deviation of the data

2) What is the difference between parametric statistical method & non-parametric statistical method?

Properties	Parametric	Non parametric
Assumptions	Yes	NO
Central Tendency Value	Mean value	Median value
Correlation	Pearson	Spearman
Probabilistic distribution	Normal	Arbitrary
Population knowledge	Requires	Does not require
Used for	Interval data	Nominal data
Examples	z-test, t-test etc	Kruskal-wallis, Mann-Whitney

C. Conclusion:

Write the significance of the topic studied in the experiment.



ST. FRANCIS INSTITUTE OF TECHNOLOGY
(Engineering College)
DATE : PAGE NO. :

C. Conclusion.

Non parametric test does not depend on any distribution, hence it is a kind of robust test & have broader range of situations. Parametric test is completely dependent on statical data & have more chances of accuracy. Normality test are used to determine if a data set is well-modeled by a normal distribution & to compute how likely it is for a random variable underlying the data set to be normally distributed.

D. References:

1)https://machinelearningmastery.com/statistical-hypothesis-tests-in-python-cheat-sheet/
