Assignment 2 - Probability and Statistics

Give your Team Details

Group Number: 4

Team members:

1. Dinesh Subramanian
2. S.Aishwarya
3. Vinod A

Import the required Libararies

Write the inference for all the questions

In [55]:

```
import numpy as np
import pandas as pd
import scipy.stats as stat
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings("ignore")

from statsmodels.stats import weightstats as test

import math
import statistics
```

In a library 50 books are selected at random.On an average 1 out of 10 books is a statistics book. . (5 marks)

i) Probability of getting exactly 3 statistics book.

ii) Probability of getting atleast 3 statistics book.

iii) Probability of getting atmost 3 statistics book.

iv) Probability of getting between 3 and 10 statistics book.

v) Plot the distribution graph.

[Note: Think what kind of distribution equation we need to use to solve all the above questions]

In [56]:

```python
# i) Probability of getting exactly 3 statistics book.

n = 50 # books
x = 3 # exactly 3 statistics book.
p = 0.1 # 1 out of 10 books is a statistics book..
# pmf- probability exactly at a point
print("Probability of getting exactly 3 statistics book : ", stat.binom.pmf(x,n,p))
```

```
Probability of getting exactly 3 statistics book :  0.1385651496069605
```

INFERENCE: Probability of getting exactly 3 statistics books out of 50 books is 14%.

In [57]:

```python
# ii) Probability of getting atleast 3 statistics book.

n = 50 # books
x = 3 # atleast 3 statistics book.
p = 0.1 # 1 out of 10 books is a statistics book..
#cdf- cumulative distribution
print("Probability of getting atleast 3 statistics book. : ",1- stat.binom.cdf(x,n,p))
```

```
Probability of getting atleast 3 statistics book. :  0.749706094046692
```

INFERENCE: Probability of getting atleast 3 statistics books out of 50 books is 75%.

In [58]:

```python
# iii) Probability of getting atmost 3 statistics book

n = 50 # books
x = 3 #  atmost 3 statistics book.
p = 0.1 # 1 out of 10 books is a statistics book..
#cdf- cumulative distribution
print(" Probability of getting atmost 3 statistics book : ", stat.binom.cdf(x,n,p))
```

```
 Probability of getting atmost 3 statistics book :  0.250293905953308
```

INFERENCE: Probability of getting atmost 3 statistics books out of 50 books is 25%.

In [59]:

```
# iv) Probability of getting between 3 and 10 statistics book.

n = 50 # books
x ,y= 3,10 #  between 3 and 10 statistics book.
p = 0.1 # 1 out of 10 books is a statistics book..
#cdf- cumulative distribution

print("Probability of getting between 3 and 10 statistics book : ",stat.binom.cdf(y,n,p
```

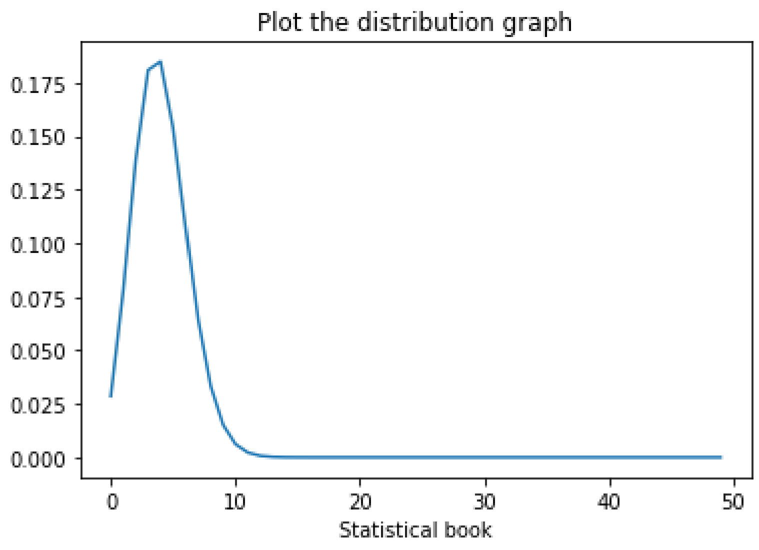
```
Probability of getting between 3 and 10 statistics book :  0.740351492459362
9
```

INFERENCE: Probability of getting between 3 to 10 statistics books out of 50 books is 74%.

In [60]:

```
# v) Plot the distribution graph
n = 50 # books
x = np.arange(1,51)
p = 0.1 # 1 out of 10 books is a statistics book..

prob = stat.binom.pmf(x,n,p)
plt.title("Plot the distribution graph")
plt.xlabel("Statistical book")
plt.plot(prob)
plt.show()
```



INFERENCE: As the Statistics books are taken in random and have the possibility of having either success or failure. So, we are taking Binomial distribution to find the probablity

In a village, the average number of people affected by corona virus in a day is 5.(5 marks)

Find the probability of the following:

(i) exactly 2 persons suffer.

(ii) atleast 2 persons suffer.

(iii) atmost 2 persons suffer.

(iv) between 2 and 5 persons suffer.

(v) Plot the Distribution

[Note: Think what kind of distribution equation we need to use to solve all the above questions]

In [61]:

```python
# (i) exactly 2 persons suffer.

x = 2 #exactly 2 persons
l = 5 #the average number of people affected by corona virus in a day is 5.
# pmf- probability exactly at a point

print("Probability of exactly 2 persons suffer : ", stat.poisson.pmf(x,l))
```

```
Probability of exactly 2 persons suffer :  0.08422433748856832
```

INFERENCE: **Probability of exactly 2 people suffering with corona virus is 8.4%.**

In [62]:

```python
# (ii) atleast 2 persons suffer.

x = 2 #atleast 2 persons suffers
l = 5 #the average number of people affected by corona virus in a day is 5.
#cdf- cumulative distribution

print("Probability of atleast 2 persons suffer : ", 1- stat.poisson.cdf(x,l))
```

```
Probability of atleast 2 persons suffer :  0.8753479805169189
```

INFERENCE: **Probability of atleast 2 people suffering with corona virus is 87%.**

In [63]:

```python
# (iii) atmost 2 persons suffer.

x = 2 #atmost 2 persons suffer
l = 5 #the average number of people affected by corona virus in a day is 5.
#cdf- cumulative distribution

print("Probability of atmost 2 persons suffer : ", stat.poisson.cdf(x,l))
```

```
Probability of atmost 2 persons suffer :  0.12465201948308108
```

INFERENCE: **Probability of atmost 2 people suffering with corona virus is 12%.**

In [64]:

```
# (iv) between 2 and 5 persons suffer

x = 2 #between 2 and 5 persons suffer
y = 5
l = 5 #the average number of people affected by corona virus in a day is 5.
#cdf- cumulative distribution

print("Probability of atmost 2 persons suffer : ", stat.poisson.cdf(y,l) - stat.poissor
```

```
Probability of atmost 2 persons suffer :  0.49130863534998187
```

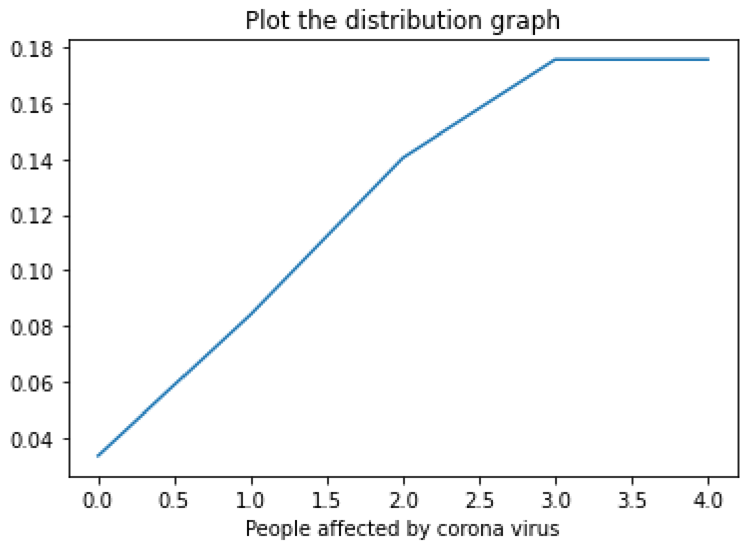
INFERENCE: Probability of 2 to 5 people suffering with corona virus is 49%.

In [65]:

```
# v) Plot the distribution graph

x = np.arange(1,6)
l = 5 #the average number of people affected by corona virus in a day is 5.

prob = stat.poisson.pmf(x,l)
plt.title("Plot the distribution graph")
plt.xlabel("People affected by corona virus")
plt.plot(prob)
plt.show()
```



INFERENCE: As the data of Corona affected people is taken over a period of time with event of occurance. So, we are using the poisson distribution to find the probablity.

The piston pressure in a randomly selected bike is normally distributed with mean value of 28 psi and standard deviation of 0.7 psi. (3 marks)

What is the probability that the pressure for a randomly selected piston:

(a) between 25 and 29 psi

(b) less than 30 psi

(c) greater than 30.5 psi

In [66]:

```
# (a) between 25 and 29 psi

mean=28
std=0.7
z25=(25-28)/0.7
z29=(29-28)/0.7

print("Probability that the pressure for a randomly selected piston between 25 and 29 p

```

```
Probability that the pressure for a randomly selected piston between 25 and
29 psi : 0.9234271668415907
```

INFERENCE: Probability that the pressure for a randomly selected piston is between 25 and 29 psi is 92% .

In [67]:

```
# (b) less than 30 psi

mean=28
std=0.7
z30=(30-mean)/std

print("Probability that the pressure for a randomly selected piston less than 30 psi :'

```

```
Probability that the pressure for a randomly selected piston less than 30 ps
i : 0.9978626330199137
```

INFERENCE: Probability that the pressure for a randomly selected piston is less than 30 psi is 99% .

In [68]:

```
# (c) greater than 30.5 psi

mean=28
std=0.7
z30_5=(30.5-mean)/std

print("Probability that the pressure for a randomly selected piston greater than 30.5 p
```

```
Probability that the pressure for a randomly selected piston greater than 3
0.5 psi : 0.00017751969037349546
```

INFERENCE: Probability that the pressure for a randomly selected piston is greater than 30 psi is 0.017% (extremely low probability).

INFERENCE: Since the normal distribution is a continuous distribution, the area under the curve (cumulative distribution) represents the probabilities. Mean gives us average and standard Deviation tells us how “spread out” the data is. It

is a measure of how far each observed value is from the mean.

A sample of 400 bulbs is manufactued found to have a mean lifetime is 171.38 hrs. Can it reasonably be regarded as a large population with mean more than 173 hrs and standard deviation of 3.3 hrs so that company can make profit(Test at 5% significance level).Will company make profit? Assume data is normally distributed. (2 marks)

In [69]:

```python
H0 = "population_mean = 173 hrs"
H1 = "population_mean >= 173 hrs"

sample = 400
sample_mean = 171.38

# as sample is more than 30, this is called to be large sample test
# therefore we can use z-test

population_mean = 173
population_sd = 3.3/math.sqrt(400)
# Generate a random array of 400 sample having mean 171.38 and sd 3.3
data = population_sd*np.random.randn(400) + population_mean

# print mean and sd
print('mean=%.2f stdv=%.2f' % (np.mean(data), np.std(data)))

# Test at 5% significance level
alpha = 0.05

ztest_Score, p_value= test.ztest(data,value = population_mean, alternative='larger')
print("ztest_Score : ", ztest_Score)

# the function outputs a p_value and z-score corresponding to that value, we compare th
# p-value with alpha, if it is greater than alpha then we do not null hypothesis
# else we reject it.

if(p_value <  alpha):
  print("Reject Null Hypothesis as p value is ", p_value, "\nTherefore,", H1)
else:
  print("Fail to Reject NUll Hypothesis as p value is", p_value, "\nTherefore,", H0)

```

```
mean=173.00 stdv=0.17
ztest_Score :  0.27557521908710264
Fail to Reject NUll Hypothesis as p value is 0.39143717260531036
Therefore, population_mean = 173 hrs
```

The data of weight loss program is given below.Find whether gender has any effect in weight loss. (2 marks)

Weight_loss_Male = [3.69, 4.12, 4.65, 3.19, 4.34, 3.68, 4.12, 4.50, 3.70, 3.09,3.65, 4.73, 3.93, 3.46, 3.28, 4.43, 4.13, 3.62, 3.71, 2.92]

Weight_loss_Female = [2.99, 1.80, 3.79, 4.12, 1.76, 3.50, 3.61, 2.32, 3.67, 4.26, 4.57, 3.01, 3.82, 4.33, 3.40, 3.86]

Calculate the test statistic and p value manually and compare with the inbuilt function.

Note: Assume data is normally distributed

In [70]:

```
male = [ 3.69, 4.12, 4.65, 3.19, 4.34, 3.68, 4.12, 4.50, 3.70, 3.09,
                    3.65, 4.73, 3.93, 3.46, 3.28, 4.43, 4.13, 3.62, 3.71, 2.92]
female = [2.99, 1.80, 3.79, 4.12, 1.76, 3.50, 3.61, 2.32, 3.67, 4.26,
                      4.57, 3.01, 3.82, 4.33, 3.40, 3.86]
```

In [71]:

```
# as samples of male and female are below 30, we are going with small sample test

H0 = "gender has no effect in weight loss (population_mean of weight_loss_male = popula
H1 = "gender has any effect in weight loss (population_mean of weight_loss_male != popu

male_arr = np.array(male)
female_arr = np.array(female)

# Print the variance of both data groups
print(np.var(male_arr), np.var(female_arr))
print("Ratio between variance : ", np.var(female_arr)/np.var(male_arr))

# Here, the ratio is less than 4:1

# therfore, Performing Two-Sample T-Test

statistics_value, p_value = stat.ttest_ind(male_arr,female_arr)

# lets take the los as 5%
alpha = 0.05

if(p_value <  alpha):
  print("Reject Null Hypothesis as p value is ", p_value,"\nTherefore,", H1)
else:
  print("Fail to Reject NUll Hypothesis as p value is", p_value,"\nTherefore,", H0)

```

```
0.25942100000000007 0.680287109375
Ratio between variance :  2.6223286063001834
Fail to Reject NUll Hypothesis as p value is 0.0764604205335295
Therefore, gender has no effect in weight loss (population_mean of weight_lo
ss_male = population_mean of weight_loss_female)
```

The data of weight gain program is given below.Find whether the program is effective. (2 marks)

Weight_before =[52, 56, 61, 47, 58, 52, 56, 60, 52, 46, 51, 62, 54, 50, 48, 59, 56, 51, 52, 44, 52, 45, 57, 60, 45]

Weight_after =[62, 64, 40, 65, 76, 82, 53, 68, 77, 60, 69, 34, 69, 73, 67, 82, 62, 49, 44, 43, 77, 61, 67, 67, 54]

Note: Assume data is normally distributed and have equal variance.

In [72]:

```
Weight_before =[52, 56, 61, 47, 58, 52, 56, 60, 52, 46, 51, 62, 54, 50,
                48, 59, 56, 51, 52, 44, 52, 45, 57, 60, 45]

Weight_after  =[62, 64, 40, 65, 76, 82, 53, 68, 77, 60, 69, 34, 69, 73,
                67, 82, 62, 49, 44, 43, 77, 61, 67, 67, 54]

```

In [73]:

```python
# as samples of weight_before and weight_after are below 30, we are going with small sc

H0 = "program is not effective (population_mean of weight_before = population_mean of w
H1 = "program is effective (population_mean of weight_before != population_mean of weig

weight_before_arr = np.array(Weight_before)
weight_after_arr = np.array(Weight_after)

# Print the variance of both data groups
print(np.var(weight_before_arr), np.var(weight_after_arr))
print("Ratio between variance : ", np.var(weight_after_arr)/np.var(weight_before_arr))

# Here, the ratio is more than 4:1

# Conducting Welch's t-Test by making equal_var = False

statistics_value, p_value = stat.ttest_ind(weight_before_arr,weight_after_arr,equal_var

# lets take the los as 5%
alpha = 0.05

if(p_value <  alpha):
  print("Reject Null Hypothesis as p value is ", p_value,"\nTherefore,", H1)
else:
  print("Fail to Reject NUll Hypothesis as p value is", p_value,"\nTherefore,", H0)

```

```
27.558400000000002 162.08
Ratio between variance :  5.881328379006038
Reject Null Hypothesis as p value is  0.0018213716687280747
Therefore, program is effective (population_mean of weight_before != populat
ion_mean of weight_after)
```

Parkinson Data Statistical Analysis

Attribute Information:

Matrix column entries (attributes):

name - ASCII subject name and recording number

MDVP:Fo(Hz) - Average vocal fundamental frequency

MDVP:Fhi(Hz) - Maximum vocal fundamental frequency

MDVP:Flo(Hz) - Minimum vocal fundamental frequency

MDVP:Jitter(%),MDVP:Jitter(Abs),MDVP:RAP,MDVP:PPQ,Jitter:DDP - Several measures of variation in fundamental frequency

MDVP:Shimmer,MDVP:Shimmer(dB),Shimmer:APQ3,Shimmer:APQ5,MDVP:APQ,Shimmer:DDA - Several measures of variation in amplitude

NHR,HNR - Two measures of ratio of noise to tonal components in the voice status - Health status of the subject (one) - Parkinson's, (zero) - healthy

RPDE,D2 - Two nonlinear dynamical complexity measures

DFA - Signal fractal scaling exponent

spread1,spread2,PPE - Three nonlinear measures of fundamental frequency variation

Reading file

In [74]:

```
data=pd.read_csv('parkinson.csv')
data.shape
```

Out[74]:

```
(195, 24)
```

In [75]:

```
from sklearn.model_selection import train_test_split
```

In [76]:

```
# Input is all the columns except 'Name' and 'Status'
inp=data.drop(columns=['name','status'])
```

In [77]:

```
# Output is to predict the 'Status' value
out=data['status']
```

In [78]:

```
xtrain,xtest,ytrain,ytest=train_test_split(inp,out,test_size=0.3,random_state=40)
```

In [79]:

```
from sklearn import linear_model
from sklearn.metrics import accuracy_score
```

INFERENCE: Here, Logistic regression is used to predict categorical outcomes, unlike linear regression that predicts a continuous outcome. In the above case there are only two outcomes, Status value equal to '0' or '1'(binomial) hence above ML model can be considered as binomial logistic regression. Accuracy_score is used to calculate the accuracy of either the fraction or count of correct prediction. It is used to measure accuracy level of test_value to predicted_value.

Statistical Tests (6 marks)

Find whether MDVP:Fhi(Hz) is helpful in predicting Status? (2 maks)

In [80]:

```
logr1 = linear_model.LogisticRegression()
```

In [81]:

```
logr1.fit(xtrain['MDVP:Fhi(Hz)'].to_numpy().reshape(-1,1), ytrain)
```

Out[81]:

```
LogisticRegression()
```

In [82]:

```
predict = logr1.predict(xtest['MDVP:Fhi(Hz)'].to_numpy().reshape(-1,1))
```

In [83]:

```
accuracy_score(ytest, predict)
```

Out[83]:

```
0.7966101694915254
```

INFERENCE: From the result, we can see that above model as predicted 'Status' using 'MDVP:Fhi(Hz)' feature with 79% accuracy.

Find whether NHR is helpful in predicting Status? (2 marks)

In [84]:

```
logr2 = linear_model.LogisticRegression()
```

In [85]:

```
logr2.fit(xtrain['NHR'].to_numpy().reshape(-1,1), ytrain)
```

Out[85]:

```
LogisticRegression()
```

In [86]:

```
predict = logr2.predict(xtest['NHR'].to_numpy().reshape(-1,1))
```

In [87]:

```
accuracy_score(ytest, predict)
```

Out[87]:

```
0.7796610169491526
```

INFERENCE: From the above result, we can say that the model as predicted 'Status' using 'NHR' feature with 77% accuracy.

Find whether spread2 is helpful in predicting Status? (2 marks)

In [88]:

```
logr3 = linear_model.LogisticRegression()
```

In [89]:

```
logr3.fit(xtrain['spread2'].to_numpy().reshape(-1,1), ytrain)
```

Out[89]:

```
LogisticRegression()
```

In [90]:

```
predict = logr3.predict(xtest['spread2'].to_numpy().reshape(-1,1))
```

In [91]:

```
accuracy_score(ytest, predict)
```

Out[91]:

```
0.7796610169491526
```

INFERENCE: From the above result, we can say that the model as predicted 'Status' using 'spread2' feature with 78% accuracy.

Wine Quality analysis (Optional)

Dataset and Info

Data Set Information:

These data are the results of a chemical analysis of wines grown in the same region in Italy but derived from three different cultivars. The analysis determined the quantities of 13 constituents found in each of the three types of wines.

I think that the initial data set had around 30 variables, but for some reason I only have the 13 dimensional version. I had a list of what the 30 or so variables were, but a.) I lost it, and b.), I would not know which 13 variables are included in the set.

The attributes are (dontated by Riccardo Leardi, riclea '@' anchem.unige.it)

1) Alcohol

2) Malic acid

3) Ash

4) Alcalinity of ash

5) Magnesium

6) Total phenols

7) Flavanoids

8) Nonflavanoid phenols

9) Proanthocyanins

10)Color intensity

11)Hue

12)OD280/OD315 of diluted wines

13)Proline

14) Cultivator

In [92]:

```
data=pd.read_csv("wine.xls"
                 ,names= ["Cultivator", "Alchol", "Malic_Acid", "Ash", "Alcalinity_of_As
              "Magnesium", "Total_phenols", "Falvanoids", "Nonflavanoid_phenols",
              "Proanthocyanins", "Color intensity", "Hue", "OD280", "Proline"])
```

In [93]:

```
data["Cultivator"].value_counts()
```

Out[93]:

```
2    71
1    59
3    48
Name: Cultivator, dtype: int64
```

Stastistical Test (Optional)

Finding whether Nonflavanoid_phenols is helpful in predicting Cultivator?

In [ ]:

```

```

Finding whether Alcalinity_of_Ash is helpful in predicting Cultivator?

In [ ]:

```

```

Finding whether there is relation between Malic_acid and Ash?

In [ ]:

```

```

Covid Dataset Analysis (optional)

Two pharma companies have created the antidote of Corona virus. One Drug (denoted by Old Drug) was created and Finished applying by August. The drug didn't show the result as it was expected. So the New Drug was created and applied. Both the drugs were applied on a set of 200000 people.10000 patients were selected to see if there is any significant effect or which one works better. The patients were divided into 5000 groups (denoted as Patient_id). Each group has patient with similar health condition. Old Drug was applied on first 5000 patients and New Drug was applied on second 5000 patients. The effect of the both the drugs were recorded group wise.

In [ ]:

```

```

Statistical analysis

1. Normal Oxygen Level (denoted as Oxy_level) The average Oxygen level should be 99 .

In [ ]:

```

```

2.Average Pulse rate (denoted as Pulse_rate) of the patient should not be more than 100 .

In [ ]:

```

```

3. Validate the claim that there is no gender discrimination while selecting the sample

In [ ]:

```

```

4. Validate that there was no discrimination based on age group and gender while selecting the patient or in other words the patients were selected randomly.

Age group is described as ,

Age<12 - A

12<Age≤ 25-B

25<Age≤40-C

40<Age≤58-D

58<Age-E

In [ ]:

```

```