

**ANL252 (Online)**

**Python for Data Analytics**

# **Tutor-Marked Assignment**

**July 2021 Presentation**

**Submitted by:**

|  |  |
| --- | --- |
| **Name** | **PI No.** |
| **Geekger Ee** | **W2071283** |

**Tutorial Group: ­­­­­­­­­­­­ T 09**

**Instructor’s Name: A/Prof. Munish Kumar**

**Submission Date: 08/08/2021**



1

# Importing math packages to use

from math import exp, sqrt, pi

1. #Asking for mean Input with numeric validation and automatic setting

#Loops until the if or else condition is fulfilled, error will print a text

while True:

try:

mean= (input("Mean value can be any value between –∞ to +∞: "))

if mean == "":

mean = float(0)

else:

mean= float(mean)

break

except ValueError:

print("Please enter a numeric input.")



* When entering anything that is non-numeric they will prompt the user to input numeric input.



* When user press enter without providing any input the program sets mean to 0

#asking for variance Input with validations and automatic setting

#loops until the if or else condition is fulfilled, two different types of error depending on what the error is which prints a text

while True:

try:

variance= (input("Variance value must be more larger than 0: "))

if variance == "":

variance = float(1)

else:

variance = float(variance)

if variance <= 0:

raise Exception

break

except ValueError:

print("Please enter a numeric input")

except Exception:

print("Variance should not be smaller or equal to 0")



* If the number is smaller or equal to 0 it prints a text informing the user to input a bigger input.



* If the input is not numeric, user will get a text to input a numeric input.

1. #For user to input the value of x with validations

#Loops until the input entered is numeric else erorr will happen and print a text

while True:

try:

x= float(input("X value can be any value between –∞ to +∞: "))

break

except ValueError:

print("Please enter a numeric input.")



* If the input is not numeric, user will get a text to input a numeric input.

1. #the formula for probability density function with variables based on what was inputed earlier

def pdf(mean,variance,x):

answer = 1 / (sqrt(2\*pi)\*variance)\*exp((-0.5)\*(((x-mean)/variance)\*\*2))

return(answer)

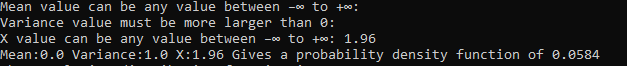
1. # funciton displaying the answer by printing the saved variable which called the function of pdf with what the user input

def displaypdf():

answer =pdf(mean, variance, x)

print(f"Mean:{mean} Variance:{variance} X:{x} Gives a probability density function of {round(answer,4)}")

displaypdf() #calling the function which displays the result



#function for cumulative distribution function using while loop while increasing the value of a per loop and storing the pdf value

def cdf(mean, variance, x):

#declaring the values of a and alpha

a=-100.00

alpha =0.01

temp=0

while a<=x:

answer =pdf(mean, variance, a)

a= round(a+alpha,2)

temp=temp+answer

take= alpha\*(temp)

return(take)

#function displaying the answer which is printed to the user

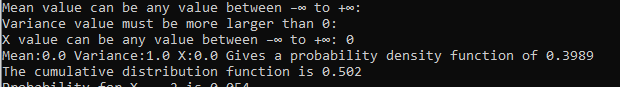
def displaycdf():

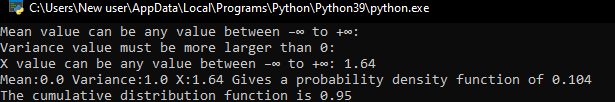
take=cdf(mean, variance, x)

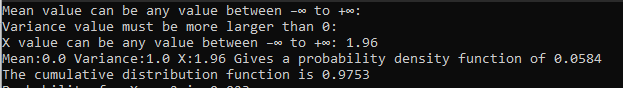
print(f"The cumulative distribution function is {round(take,4)}")

displaycdf()

Results when K = 0

  
  
  
Results when K = 1.64



Results when K=1.96  


My program uses a while loop that only ends when the value of a is more than or equal to x/k. When a is not more than or equal to x/k. The next step we do is calculate the probability density function of the current value of a and store it in a variable answer. Next, the program then gives a a new value where we add a and alpha together as the value of a as we know that a is a number close to negative infinity The program then rounds the value of a by 2 decimal places due to the precision error with python as it is unable to represent all numbers. We do this to ensure that the loop doesn’t break in advance due to precision errors. The calculated pdf value is then store in a temp variable where temp equals to temp plus the pdf value. This saves the previous value of the pdf of the different values of a. Lastly, once the loop breaks all we do is multiply alpha and the temp variable and return it so that we can call the function again.

1. #storing the dictionary from -5 to 5

z=-5

dict = {}

while z<=5:

key = round(z,2)

val = cdf(mean, variance, key)

z = round(z+0.1,2)

dict[key] = round(val, 4)

#displaying the diction from -2 to 2 by increment of 0.5

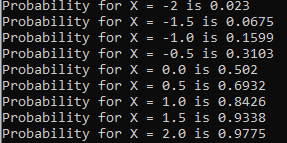
z=-2

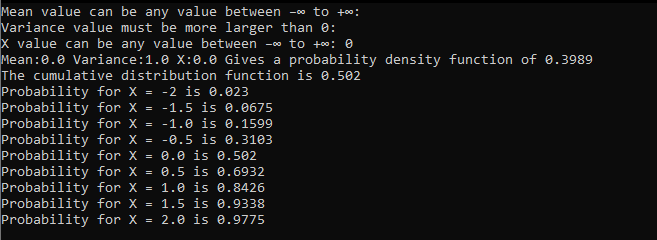
while z<=2:

key= round(z,1)

z=z+0.5

print(f"Probability for X = {key} is {dict[key]}")





* Program running as a whole