

**ANL252**

**Python for Data Analytics**

# **Tutor-Marked Assignment**

**July 2021 Presentation**

**Submitted by:**

|  |  |
| --- | --- |
| **Name** | **PI No.** |
| **CHUNG WEI TAT** | **Y2070638** |

**Tutorial Group: ­­­­­­­­­­­­ T 09**

**Submission Date: 15/08/2021**

**Question 1**

**(a)**

import math

**(b)**

quit\_ask\_mean = False

quit\_ask\_variance = False

quit\_ask\_x = False

mean = 0 # default value to 0

variance = 1 # default value to 1

x = 0 # declare for global use

while(quit\_ask\_mean == False):

  mean = input("Enter mean of distribution. It can be any value between minus infinity (-∞) and plus infinity (+∞).")

  if mean == "":

    # if user key blank, end loop setting mean = 0 & variance = 1

    mean = 0

    variance = 1

    quit\_ask\_mean = True

    quit\_ask\_variance = True

    print("Empty field detected. Setting mean = 0, variance = 1.")

break



  else:

    try:

        # float type is infinity

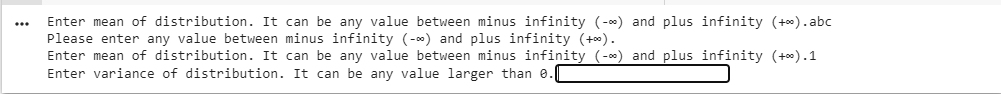
        mean = float(mean) # check if entered is a float

    except ValueError:

        print("Please enter any value between minus infinity (-∞) and plus infinity (+∞).")

    else:

        quit\_ask\_mean = True # if float then continue



while(quit\_ask\_variance == False):

  variance = input("Enter variance of distribution. It can be any value larger than 0.")

  if variance == "":

    # if user key blank, end loop setting variance = 1

    variance = 1

    quit\_ask\_variance = True

    print("Empty field detected. Setting variance = 1.")

break



  else:

    try:

        variance = float(variance) # check if entered is a float

    except ValueError:

        print("Please enter any number larger than 0.")

    else:

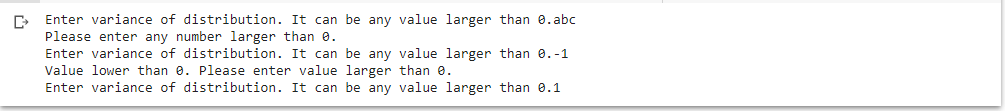
        # if larger than 0 then continue else repeat

        if (variance > 0):

          quit\_ask\_variance = True

        else:

          print("Value lower than 0. Please enter value larger than 0.")



**(c)**

while(quit\_ask\_x == False):

  try:

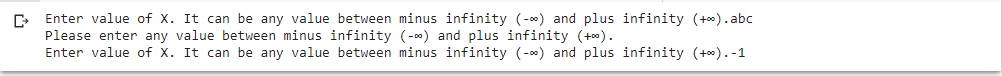
      x = float(input("Enter value of X. It can be any value between minus infinity (-∞) and plus infinity (+∞)."))

  except ValueError:

      print("Please enter any value between minus infinity (-∞) and plus infinity (+∞).")

  else:

      quit\_ask\_x = True



**(d)**

# user\_val = x in equation

# formula broken down in sections

def compute\_pdf(v, m, user\_val):

  pdf\_a =  1/(math.sqrt(2\*math.pi\*v))

  pdf\_b = math.pow(user\_val-m, 2)

  pdf\_c = math.exp(-(pdf\_b)/(2\*v))

  # pdf

  return (pdf\_a \* pdf\_c)

pdf = compute\_pdf(variance, mean, x)

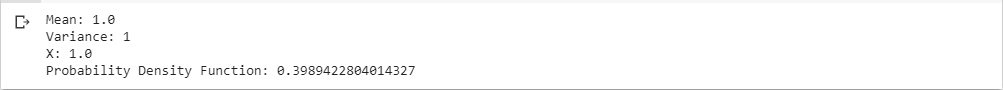
**(e)**

print("Mean: {0}".format(mean))

print("Variance: {0}".format(variance))

print("X: {0}".format(x))

print("Probability Density Function: {0}".format(pdf))



**(f)**

def count\_steps(begin, end, step):

  # get num of steps in integer w/o negative sign

  steps = (begin - end)/step

  return int(max(steps, -(steps)))

# v = variance, m = mean

def compute\_pxk(v, m, alpha, a, k):

  summation = 0

  # get num of steps in integer w/o negative sign

  num\_of\_steps = count\_steps(a, k, alpha)

  # increase step from alpha till k

  for i in range(0, num\_of\_steps):

    sum = compute\_pdf(v, m, (k + i\*alpha))

    summation = summation + sum

  return alpha \* summation

# assume a = -100

result\_1 = compute\_pxk(variance, mean, 0.01, -100, 0) # when k = 0

result\_2 = compute\_pxk(variance, mean, 0.01, -100, 1.64) # when k = 1.64

result\_3 = compute\_pxk(variance, mean, 0.01, -100, 1.96) # when k = 1.96

print (result\_1)

print (result\_2)

print (result\_3)



**(g)**

We want to test the 3 values of k so putting the computation in a user-defined function to make things easy.

For loop uses only integer so to get the number of steps, the end number is subtracted from the beginning number to get the difference and divided by alpha to get the number of times the alpha needs to run to reach the end number. A user-defined function for calculating steps is created for ease of use.

Summation is declared at the beginning of the function to store and add on.

Referencing with mean = 0 and variance = 1, it is sent to compute\_pdf which is the function to calculate pdf. pdf also requires x. x in this case is the value of each step down from k. Thus getting the current step in for loop to multiply alpha to determine the number remaining that needs to be sent and calculate.

The number returned from computing pdf is added to the sum every time until the loops end.

Finally, the returning value is alpha multiple summation as stated in the formula.

**(h)**

# run for loop from q (f) to replace x = k

prob\_dict = {}

dict\_steps = count\_steps(-2, 2, 0.5)

for i in range(0, dict\_steps):

  current\_val = (-2 + i\*0.5)

  # assume a = -100

  current\_val\_result = compute\_pxk(variance, mean, 0.01, -100, current\_val) # connected with the program

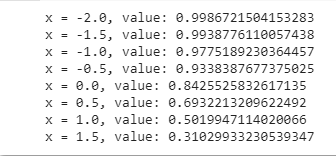
  new\_dict = {current\_val: current\_val\_result}

  prob\_dict = {\*\*prob\_dict, \*\*new\_dict}

# print out all probabilities

for x, value in list(prob\_dict.items()):

  print("x = {0}, value: {1}".format(x, value))



Attached Python file

