**Problem Statement 1: Marks of Year 8 Students**

**Data:**

marks = [6, 7, 5, 7, 7, 8, 7, 6, 9, 7, 4, 10, 6, 8, 8, 9, 5, 6, 4, 8]

**Calculations:**

**Python Code**

import numpy as np

from scipy import stats

# Data

marks = [6, 7, 5, 7, 7, 8, 7, 6, 9, 7, 4, 10, 6, 8, 8, 9, 5, 6, 4, 8]

# Mean

mean\_marks = np.mean(marks)

# Median

median\_marks = np.median(marks)

# Mode

mode\_marks = stats.mode(marks)[0][0]

# Standard Deviation

std\_marks = np.std(marks)

print(f"Mean: {mean\_marks}")

print(f"Median: {median\_marks}")

print(f"Mode: {mode\_marks}")

print(f"Standard Deviation: {std\_marks}")

**Problem Statement 2: Number of Calls from Motorists**

**Data:**

calls = [28, 122, 217, 130, 120, 86, 80, 90, 140, 120, 70, 40, 145, 113, 90, 68, 174, 194, 170,

100, 75, 104, 97, 75, 123, 100, 75, 104, 97, 75, 123, 100, 89, 120, 109]

**Calculations:**

**Python Code**

# Data

calls = [28, 122, 217, 130, 120, 86, 80, 90, 140, 120, 70, 40, 145, 113, 90, 68, 174, 194, 170,

100, 75, 104, 97, 75, 123, 100, 75, 104, 97, 75, 123, 100, 89, 120, 109]

# Mean

mean\_calls = np.mean(calls)

# Median

median\_calls = np.median(calls)

# Mode

mode\_calls = stats.mode(calls)[0][0]

# Standard Deviation

std\_calls = np.std(calls)

print(f"Mean: {mean\_calls}")

print(f"Median: {median\_calls}")

print(f"Mode: {mode\_calls}")

print(f"Standard Deviation: {std\_calls}")

**Problem Statement 3: Gym Workouts**

**Data:**

x = [0, 1, 2, 3, 4, 5]

f\_x = [0.09, 0.15, 0.40, 0.25, 0.10, 0.01]

**Calculations:**

**Python Code**

# Data

x = np.array([0, 1, 2, 3, 4, 5])

f\_x = np.array([0.09, 0.15, 0.40, 0.25, 0.10, 0.01])

# Mean

mean\_workouts = np.sum(x \* f\_x)

# Variance

variance\_workouts = np.sum(f\_x \* (x - mean\_workouts)\*\*2)

print(f"Mean number of workouts: {mean\_workouts}")

print(f"Variance: {variance\_workouts}")

**Problem Statement 4: Drilled Hole Diameter**

Given PDF: ( f(d) = 20e^{-20(d - 12.5)}, d \geq 12.5 )

**Proportion of parts with diameter > 12.6:**

**Python Code**

from scipy.integrate import quad

# Calculate the proportion of parts > 12.6

def pdf(d):

return 20 \* np.exp(-20 \* (d - 12.5))

proportion\_scrap, \_ = quad(pdf, 12.6, np.inf)

print(f"Proportion of parts with diameter > 12.6 mm: {proportion\_scrap}")

**CDF when the diameter is 11 mm:**

Since the PDF is defined for ( d \geq 12.5 ), the CDF at 11 mm is zero.

**Python Code**

cdf\_11mm = 0

print(f"CDF at 11 mm: {cdf\_11mm}")

**Problem Statement 5: Faulty LEDs**

Given: ( p = 0.30 ), ( n = 6 ), ( k = 2 )

**Calculations:**

**Python Code**

from scipy.stats import binom

# Probability of having 2 faulty LEDs in a sample of 6

p = 0.30

n = 6

k = 2

prob\_2\_faulty = binom.pmf(k, n, p)

mean\_faulty = n \* p

std\_faulty = np.sqrt(n \* p \* (1 - p))

print(f"Probability of 2 faulty LEDs: {prob\_2\_faulty}")

print(f"Mean number of faulty LEDs: {mean\_faulty}")

print(f"Standard Deviation: {std\_faulty}")

**Problem Statement 6: Gaurav and Barakha**

For Gaurav: ( n = 8 ), ( p = 0.75 ) For Barakha: ( n = 12 ), ( p = 0.45 )

**Calculations:**

**Python Code**

# Gaurav's calculations

n\_guarav = 8

p\_guarav = 0.75

# Probability of solving 5 questions correctly

prob\_guarav\_5 = binom.pmf(5, n\_guarav, p\_guarav)

prob\_guarav\_4 = binom.pmf(4, n\_guarav, p\_guarav)

prob\_guarav\_6 = binom.pmf(6, n\_guarav, p\_guarav)

# Barakha's calculations

n\_barakha = 12

p\_barakha = 0.45

# Probability of solving 5 questions correctly

prob\_barakha\_5 = binom.pmf(5, n\_barakha, p\_barakha)

prob\_barakha\_4 = binom.pmf(4, n\_barakha, p\_barakha)

prob\_barakha\_6 = binom.pmf(6, n\_barakha, p\_barakha)

print(f"Gaurav - Probability of 5 correct: {prob\_guarav\_5}")

print(f"Gaurav - Probability of 4 correct: {prob\_guarav\_4}")

print(f"Gaurav - Probability of 6 correct: {prob\_guarav\_6}")

print(f"Barakha - Probability of 5 correct: {prob\_barakha\_5}")

print(f"Barakha - Probability of 4 Sure, let's continue with Problem Statement 6, where we left off.

```python

print(f"Barakha - Probability of 4 correct: {prob\_barakha\_4}")

print(f"Barakha - Probability of 6 correct: {prob\_barakha\_6}")

**Problem Statement 7: Customer Arrivals**

Given a rate of 72 customers per hour, we need to find the probability of k customers arriving in 4 minutes. The rate for 4 minutes is ( \lambda = 72 \times \frac{4}{60} = 4.8 ).

**Calculations:**

**Python Code**

from scipy.stats import poisson

# Rate for 4 minutes

lambda\_4min = 72 \* (4 / 60)

# a) 5 customers

prob\_5\_customers = poisson.pmf(5, lambda\_4min)

# b) not more than 3 customers

prob\_not\_more\_than\_3 = poisson.cdf(3, lambda\_4min)

# c) more than 3 customers

prob\_more\_than\_3 = 1 - poisson.cdf(3, lambda\_4min)

print(f"Probability of 5 customers: {prob\_5\_customers}")

print(f"Probability of not more than 3 customers: {prob\_not\_more\_than\_3}")

print(f"Probability of more than 3 customers: {prob\_more\_than\_3}")

**Problem Statement 8: Errors in Typing**

Given:

* Typing rate: 77 words per minute
* Error rate: 6 errors per hour
* Words in the report: 455

**Calculations:**

**Python Code**

# Error rate per word

error\_rate\_per\_word = 6 / (77 \* 60)

# Number of words in the report

words\_in\_report = 455

# Expected number of errors in the report

lambda\_report = error\_rate\_per\_word \* words\_in\_report

# Probability of 2 errors

prob\_2\_errors = poisson.pmf(2, lambda\_report)

print(f"Probability of 2 errors in 455-word report: {prob\_2\_errors}")

**For 1000 words and 255 words:**

**Python Code**

# For 1000 words

words\_1000 = 1000

lambda\_1000 = error\_rate\_per\_word \* words\_1000

prob\_2\_errors\_1000 = poisson.pmf(2, lambda\_1000)

# For 255 words

words\_255 = 255

lambda\_255 = error\_rate\_per\_word \* words\_255

prob\_2\_errors\_255 = poisson.pmf(2, lambda\_255)

print(f"Probability of 2 errors in 1000-word report: {prob\_2\_errors\_1000}")

print(f"Probability of 2 errors in 255-word report: {prob\_2\_errors\_255}")

**Problem Statement 9: Diameter of Hole Drilled**

Given PDF: ( f(d) = 20e^{-20(d - 12.5)}, d \geq 12.5 )

**Proportion of parts with diameter > 12.6:**

**Python Code**

# Proportion of parts > 12.6

from scipy.integrate import quad

def pdf(d):

return 20 \* np.exp(-20 \* (d - 12.5))

proportion\_scrap, \_ = quad(pdf, 12.6, np.inf)

print(f"Proportion of parts with diameter > 12.6 mm: {proportion\_scrap}")

**CDF for diameter of 11 mm:**

Since the PDF is defined for ( d \geq 12.5 ), the CDF at 11 mm is zero.

**Python Code**

cdf\_11mm = 0

print(f"CDF at 11 mm: {cdf\_11mm}")

**Problem Statement 10: Standard Normal Distribution**

**a) Compute the probabilities:**

**Python Code**

from scipy.stats import norm

# a) P(Z > 1.26)

prob\_z\_greater\_1\_26 = 1 - norm.cdf(1.26)

# P(Z < -0.86)

prob\_z\_less\_neg\_0\_86 = norm.cdf(-0.86)

# P(Z > -1.37)

prob\_z\_greater\_neg\_1\_37 = 1 - norm.cdf(-1.37)

# P(-1.25 < Z < 0.37)

prob\_z\_between\_neg\_1\_25\_and\_0\_37 = norm.cdf(0.37) - norm.cdf(-1.25)

# P(Z ≤ -4.6)

prob\_z\_less\_equal\_neg\_4\_6 = norm.cdf(-4.6)

print(f"P(Z > 1.26): {prob\_z\_greater\_1\_26}")

print(f"P(Z < -0.86): {prob\_z\_less\_neg\_0\_86}")

print(f"P(Z > -1.37): {prob\_z\_greater\_neg\_1\_37}")

print(f"P(-1.25 < Z < 0.37): {prob\_z\_between\_neg\_1\_25\_and\_0\_37}")

print(f"P(Z ≤ -4.6): {prob\_z\_less\_equal\_neg\_4\_6}")

**b) Find z such that P(Z > z) = 0.05**

**Python Code**

z\_0\_05 = norm.ppf(1 - 0.05)

print(f"Value of z such that P(Z > z) = 0.05: {z\_0\_05}")

**c) Find z such that P(−z < Z < z) = 0.99**

**Python Code**

z\_0\_99 = norm.ppf(0.995)

print(f"Value of z such that P(−z < Z < z) = 0.99: {z\_0\_99}")

**Problem Statement 11: Current Flow in Copper Wire**

Given: Mean = 10 mA, Variance = 4 (mA)^2

**Calculations:**

**Python Code**

mean\_current = 10

std\_current = np.sqrt(4)

# a) Probability that current measurement > 13 mA

prob\_greater\_13 = 1 - norm.cdf(13, mean\_current, std\_current)

# b) Probability that current measurement is between 9 and 11 mA

prob\_between\_9\_and\_11 = norm.cdf(11, mean\_current, std\_current) - norm.cdf(9, mean\_current, std\_current)

# c) Current measurement which has a probability of 0.98

current\_0\_98 = norm.ppf(0.98, mean\_current, std\_current)

print(f"Probability that current measurement > 13 mA: {prob\_greater\_13}")

print(f"Probability that current measurement is between 9 and 11 mA: {prob\_between\_9\_and\_11}")

print(f"Current measurement with a probability of 0.98: {current\_0\_98}")

**Problem Statement 12: Shaft Diameter**

Given: Mean = 0.2508 inch, Std = 0.0005 inch, Specifications: 0.2500 ± 0.0015 inch

**Calculations:**

**Python Code**

### Problem Statement 12: Shaft Diameter

Given: Mean = 0.2508 inch, Std = 0.0005 inch, Specifications: 0.2500 ± 0.0015 inch

#### Calculations:

# Given data

mean\_diameter = 0.2508

std\_diameter = 0.0005

lower\_spec = 0.2500 - 0.0015

upper\_spec = 0.2500 + 0.0015

# Proportion of shafts within specifications

proportion\_within\_specs = norm.cdf(upper\_spec, mean\_diameter, std\_diameter) - norm.cdf(lower\_spec, mean\_diameter, std\_diameter)

# If the process is centered at 0.2500

mean\_centered = 0.2500

proportion\_centered\_specs = norm.cdf(upper\_spec, mean\_centered, std\_diameter) - norm.cdf(lower\_spec, mean\_centered, std\_diameter)

print(f"Proportion of shafts within specifications (mean = 0.2508): {proportion\_within\_specs}")

print(f"Proportion of shafts within specifications (mean = 0.2500): {proportion\_centered\_specs}")