



San Francisco Bay University

**MATH208 - Probability and Statistics  
2023 Fall Homework #4**

**Due day:  
12/7/2023**

**Instruction:**

- 1. Homework answer sheet should contain the original questions and corresponding answers.**
  - 2. Answer sheet must be in PDF file format with Github links for the programming questions, but MS Word file can't be accepted. As follows is the answer sheet name format.**  
*<course\_id>\_week<week\_number>\_StudentID\_FirstName\_LastName.pdf*
  - 3. The program name in Github must follow the format like**  
*<course\_id>\_week<week\_number>\_q<question\_number>\_StudentID\_FirstName\_LastName*
  - 4. If the calculation in Excel is needed, the original file must be provided.**
  - 5. Show screenshot of all running results, including the system date/time.**
  - 6. The calculation process must be **printed** if needed, handwriting can't be accepted.**
  - 7. Only accept homework submission uploaded via Canvas.**
  - 8. Overdue homework submission can't be accepted.**
  - 3. Takes academic honesty and integrity seriously (Zero Tolerance of Cheating & Plagiarism)**
- 
1. Assuming that the lengths of American anchovies appease the normal distribution with the mean  $\mu = 10.3cm$  and standard deviation  $\sigma = 0.65cm$ , please find the percentages of the lengths in the population of American anchovies.

a. Less than 9cm.



The screenshot shows a Jupyter Notebook titled 'Untitled43.ipynb' in a Google Colab environment. The notebook contains three questions. Question 1 asks for the percentage of anchovies with lengths less than 9 cm. The code calculates the z-score for 9 cm and uses the standard normal CDF to find the percentage, which is 2.28%. Question 2 asks for the percentage of anchovies with lengths between 9.5 cm and 10.6 cm. The code calculates z-scores for both values and uses the standard normal CDF to find the percentage, which is 56.86%. Question 3 asks for the top 20 percentile, which is calculated as 0.8.

```
#Question 1.
from scipy.stats import norm

avg_len = 10.3
std = 0.65

z_score_for_9cm = (9 - avg_len) / std

percentage_less_than_9cm = norm.cdf(z_score_for_9cm) * 100

print(f"The percentage of anchovies with lengths less than 9 cm is: {percentage_less_than_9cm:.2f}%")

#
The percentage of anchovies with lengths less than 9 cm is: 2.28%

[12] # Question 2.
zscore_9_5cm = (9.5 - avg_len) / std
zscore_10_6cm = (10.6 - avg_len) / std

percentage_between_9_5_and_10_6cm = (norm.cdf(zscore_10_6cm) - norm.cdf(zscore_9_5cm)) * 100

print(f"The percentage of anchovies with lengths between 9.5 cm and 10.6 cm is: {percentage_between_9_5_and_10_6cm:.2f}%")

The percentage of anchovies with lengths between 9.5 cm and 10.6 cm is: 56.86%

#Question 3.
zscore_top20_per = norm.ppf(0.8)

0.8 completed at 12:51 PM
```

b. Between 9.5cm and 10.6cm.

This screenshot is identical to the one above, showing the same Jupyter Notebook with the same code and output for the three questions.

```
#Question 1.
from scipy.stats import norm

avg_len = 10.3
std = 0.65

z_score_for_9cm = (9 - avg_len) / std

percentage_less_than_9cm = norm.cdf(z_score_for_9cm) * 100

print(f"The percentage of anchovies with lengths less than 9 cm is: {percentage_less_than_9cm:.2f}%")

#
The percentage of anchovies with lengths less than 9 cm is: 2.28%

[12] # Question 2.
zscore_9_5cm = (9.5 - avg_len) / std
zscore_10_6cm = (10.6 - avg_len) / std

percentage_between_9_5_and_10_6cm = (norm.cdf(zscore_10_6cm) - norm.cdf(zscore_9_5cm)) * 100

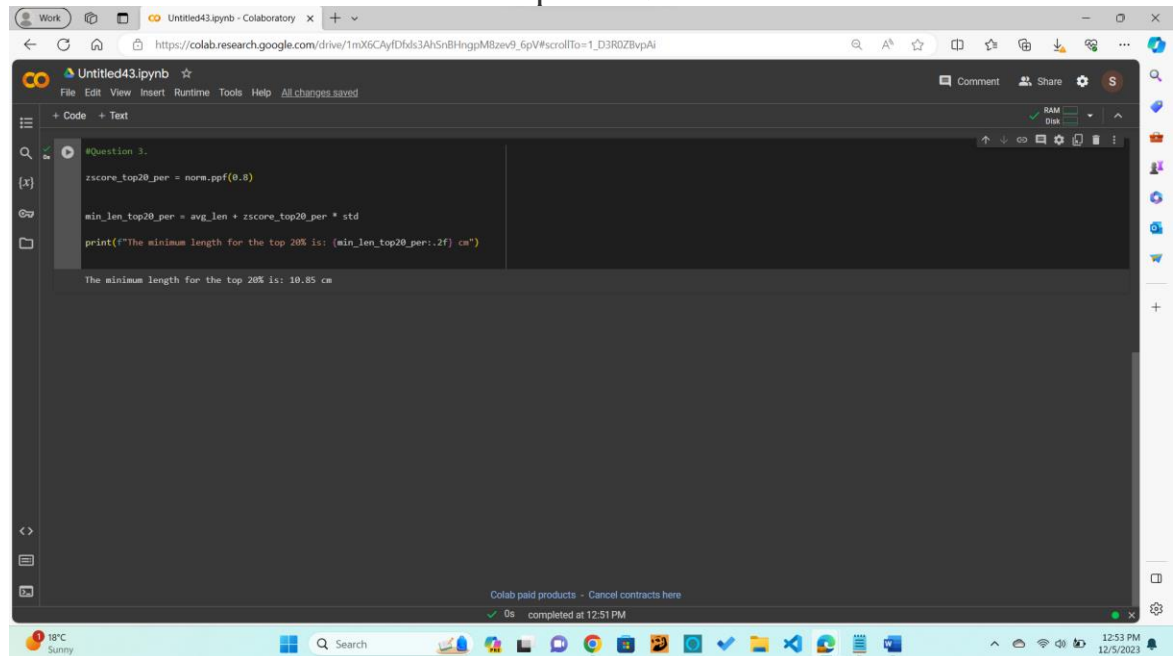
print(f"The percentage of anchovies with lengths between 9.5 cm and 10.6 cm is: {percentage_between_9_5_and_10_6cm:.2f}%")

The percentage of anchovies with lengths between 9.5 cm and 10.6 cm is: 56.86%

#Question 3.
zscore_top20_per = norm.ppf(0.8)

0.8 completed at 12:51 PM
```

- c. What is the minimum length if a restaurant claimed that the lengths of the sold anchovies are in the top of 20%?



The screenshot shows a Google Colab notebook titled 'Untitled43.ipynb'. The code cell contains the following Python code:

```
#Question 3.
zscore_top20_per = norm.ppf(0.8)

min_len_top20_per = avg_len + zscore_top20_per * std

print(f"The minimum length for the top 20% is: {min_len_top20_per:.2f} cm")
```

The output of the code is: "The minimum length for the top 20% is: 10.85 cm". The notebook interface includes a menu bar (File, Edit, View, Insert, Runtime, Tools, Help), a toolbar with icons for file operations, and a status bar at the bottom showing the temperature (18°C) and time (12:53 PM, 12/5/2023).

Hint: Python functions or functions in Excel for the calculation of each question is preferred.

2. If the random variables  $X$  and  $Y$  are normal distributions with  $\mu = 10$  &  $\sigma = 3$  and  $\mu = 15$  &  $\sigma = 8$ , namely,  $X \sim N(10, 3)$  and  $Y \sim N(15, 8)$ , and they are **independent**, what is the probability distribution and statistical parameters of :

→

(1)  $X + Y$  :

$$\mu_x + \mu_y = 10 + 15$$
$$\text{Mean} = 25$$

$$\sigma_{x+y}^2 = 3^2 + 8^2$$
$$= 9 + 64$$
$$\text{Variance} = 73$$

The probability distribution is:  $N(25, \sqrt{73})$

(2)  $X - Y$  :

$$\mu_x - \mu_y = 10 - 15$$
$$\text{Mean} = -5$$

$$\begin{aligned}\sigma_{x-y}^2 &= 3^2 + 8^2 \\ &= 9 + 64 \\ \text{Variance} &= 73\end{aligned}$$

The probability distribution is :  $N(-5, \sqrt{73})$

(3)  $3X$  :

$$\begin{aligned}\mu_{3x} &= 3 \cdot \mu_x \\ \text{Mean} &= 3 \cdot 10 = 30\end{aligned}$$

$$\begin{aligned}\sigma_{3x}^2 &= (3 \cdot \sigma_x)^2 \\ &= (3 \cdot 3)^2 \\ \text{Variance} &= 81\end{aligned}$$

The probability distribution is :  $N(30, 9)$

(4)  $4X + 5Y$ :

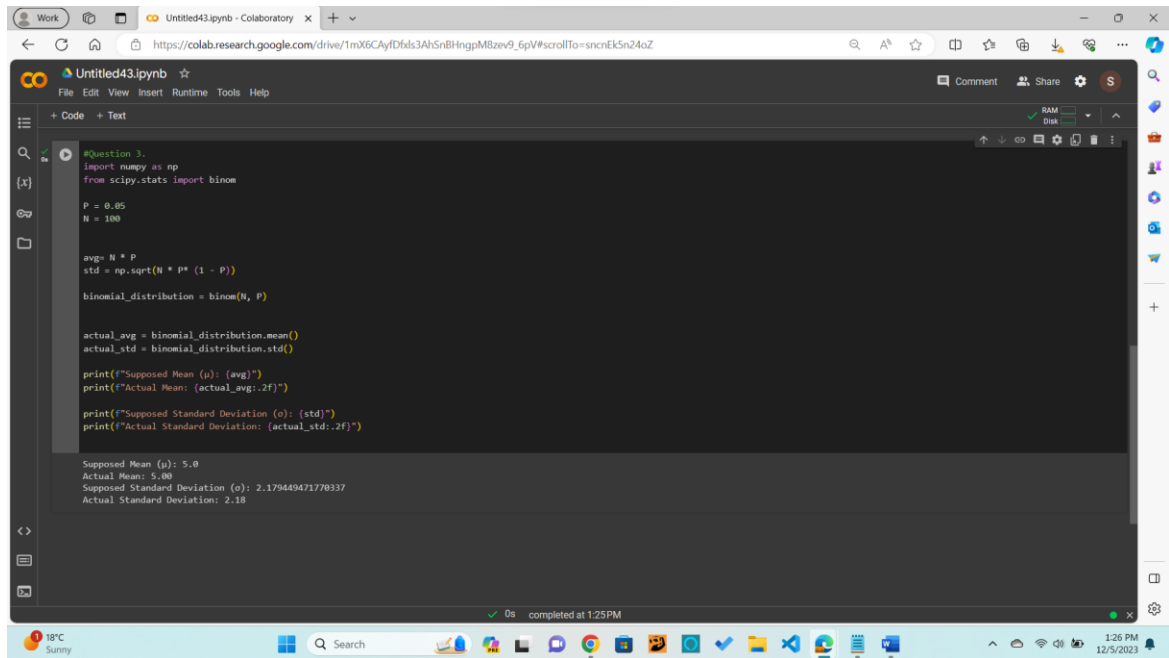
$$\begin{aligned}\mu_{4x+5y} &= 4 \cdot \mu_x + 5 \cdot \mu_y \\ &= 4 \cdot 10 + 5 \cdot 15 \\ &= 40 + 75 \\ \text{Mean} &= 115\end{aligned}$$

$$\begin{aligned}\sigma_{4x+5y}^2 &= (4 \cdot \sigma_x)^2 + (5 \cdot \sigma_y)^2 \\ &= (4 \cdot 3)^2 + (5 \cdot 8)^2 \\ &= 144 + 200 \\ &= 344\end{aligned}$$

$$\text{Variance} = 73$$

The probability distribution is:  $N(115, \sqrt{344})$

3. For the students in Engineering School, please write Python program to verify the mean  $\mu = np$  and standard deviation  $\sigma = \sqrt{npq}$  for  $p=0.05$  and selecting any  $n$  greater than 50 in binomial distribution.  
For Business school students, verify in Excel.  
→



The screenshot shows a Google Colab notebook titled 'Untitled43.ipynb'. The code is as follows:

```
#Question 3
import numpy as np
from scipy.stats import binom

P = 0.05
N = 100

avg = N * P
std = np.sqrt(N * P * (1 - P))

binomial_distribution = binom(N, P)

actual_avg = binomial_distribution.mean()
actual_std = binomial_distribution.std()

print(f"Supposed Mean (μ): {avg}")
print(f"Actual Mean: {actual_avg:.2f}")

print(f"Supposed Standard Deviation (σ): {std}")
print(f"Actual Standard Deviation: {actual_std:.2f}")
```

The output of the code is displayed below the code cell:

```
Supposed Mean (μ): 5.0
Actual Mean: 5.00
Supposed Standard Deviation (σ): 2.179449471770337
Actual Standard Deviation: 2.18
```

The notebook interface includes a file explorer on the left, a top bar with 'File Edit View Insert Runtime Tools Help', and a bottom status bar showing '0s completed at 1:25 PM'.

4. In general, if  $np > 5$  and  $nq > 5$  in binomial distribution, binomial probabilities can be approximated using the normal distribution. Please select any big enough  $n$  and  $p$ 's values to verify in Python program or Excel and plot the histogram.

```
Untitled43.ipynb - Colaboratory
https://colab.research.google.com/drive/1mX6CAyFDf6s3Ah5nB8HngpM8zev9_6pV#scrollTo=ufr6oBZ4Utz

File Edit View Insert Runtime Tools Help Saving...

+ Code + Text

#Question 4.
import numpy as np
import matplotlib.pyplot as plt
from scipy.stats import binom, norm

N = 50
P = 0.3

avg_binomial = N * P
std_binomial = np.sqrt(N * P * (1 - P))

if N * P > 5 and N * (1 - P) > 5:
    print("Given the condition np > 5 and nq > 5, binomial distribution can be approximated by normal distribution.")
else:
    print("Given the condition np > 5 and nq > 5, binomial distribution cannot be approximated correctly by normal distribution.")

b_exp = binom.rvs(N, P, size=1000)
x_vals = np.arange(0, N + 1)

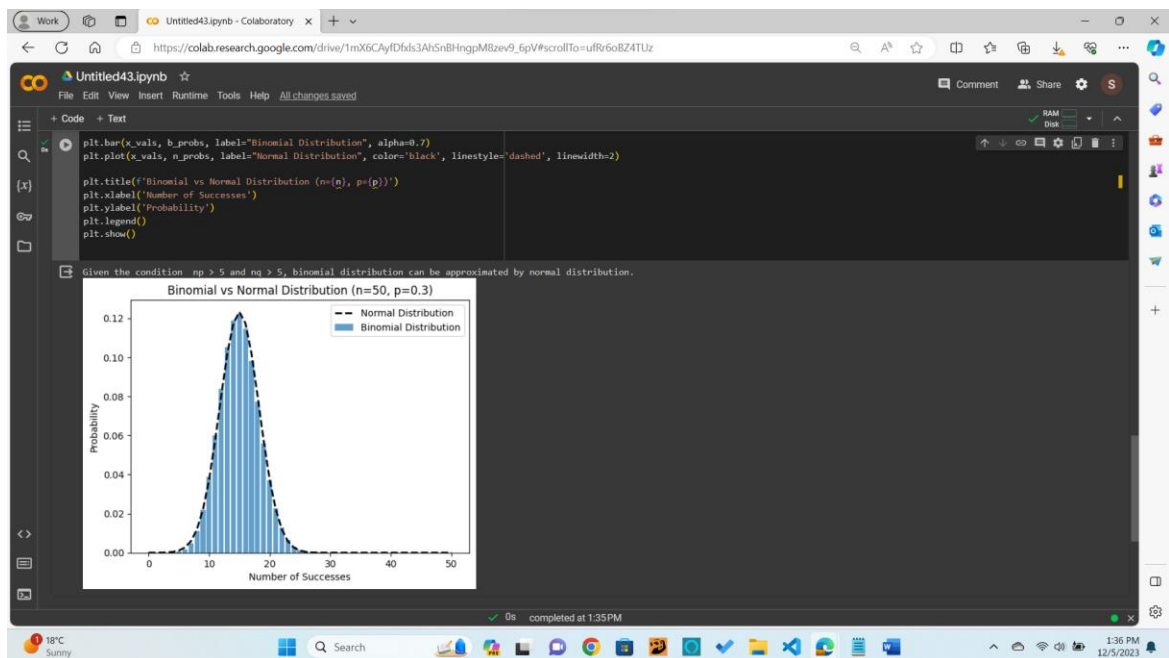
b_probs = binom.pmf(x_vals, N, P)

n_probs = norm.pdf(x_vals, avg_binomial, std_binomial)

plt.bar(x_vals, b_probs, label="Binomial Distribution", alpha=0.7)
plt.plot(x_vals, n_probs, label="Normal Distribution", color='black', linestyle='dashed', linewidth=2)

plt.title("Binomial vs Normal Distribution (n=50, p=0.3)")
plt.xlabel("Number of Successes")
plt.ylabel("Probability")
plt.legend()

0s completed at 1:35PM
```



5. In coin tossing experiments, please find the probability of the exact 6 heads from 12 tossing by **ONLY** using the normal distribution method.  
→

$$\mu = np$$

$$\sigma = \sqrt{npq}$$

Using normal distribution to find the probability of exactly 6 heads, we get:

$$P(X=k) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$$

Substituting the values , we get:

$$\mu = 12 * 0.5$$

$$= 6$$

$$\sigma = \sqrt{12 \times 0.5 \times 0.5}$$

$$= \sqrt{3}$$

$$P(X=6) = \frac{1}{\sqrt{3}\sqrt{2\pi}} \exp\left(-\frac{(6-6)^2}{2 \times 3}\right)$$

$$P(X=6) \frac{1}{\sqrt{6\pi}}$$

Calculating probability , we get:

$$P(X=6) \sim \frac{1}{\sqrt{6\pi}} \sim 0.1508$$

The probability of getting exactly 6 heads in 12 coins toss is 0.1508

6. Given that the defective rate of a product of the batteries in a manufacturing company is 6%, 150 batteries are randomly selected from the population. Please find the probability of 12 or more defective ones in them by **ONLY** using the normal distribution method.

→

$$p = 0.06$$

$$n = 150$$

$$\mu = NP$$

$$= 150 * 0.06$$

$$Mean = 9$$

Calculating q , we get:

$$q = 1 - p$$

$$= 1 - 0.06$$

$$q = 0.94$$

Calculating std, we get:

$$\begin{aligned}\sigma &= \sqrt{npq} \\ &= \sqrt{150 \times 0.06 \times 0.94} \\ &= \sqrt{8.46} \sim 2.91\end{aligned}$$

Calculating the z – score, we get:

$$\begin{aligned}Z &= \frac{X - \mu}{\sigma} \\ &= \frac{12 - 9}{2.91} \sim 1.03\end{aligned}$$

Using z-score to find  $P(X < 12)$  using a standard normal distribution :

$$\begin{aligned}P(X \geq 12) &= 1 - P(X < 12) \\ &= 1 - 0.8485 \\ P(X \geq 12) &\sim 0.1515\end{aligned}$$

The probability of having 12 or more defective batteries out of 150 is 0.1515 or 15.15%.

7. For the students in Engineering School, please write a Python program by calling functions in the following link to create 100 random numbers in T distribution with  $df = 10$  (degree of freedom) and calculate the mean  $\mu$  and standard deviation  $\sigma$ . After that, the 30 samples will be randomly selected from these random numbers in each sampling group. A total of 15 sampling groups should be created. Based on Central Limit Theorem (CLT), the mean value  $\bar{x}$  in total 15 sampling group is roughly the mean  $\mu$  of 100 random numbers and  $\sigma_{\bar{x}} = \frac{\sigma}{\sqrt{n}}$ . Please verify it and plot the histogram, which should be normal distribution. For Business school students, complete the above process in Excel.





```
Untitled43.ipynb - Colaboratory
https://colab.research.google.com/drive/1mX6CAyFDf6s3Ah5nBHngpM8zev9_6pV#scrollTo=N586H4a9N_TJ

File Edit View Insert Runtime Tools Help
+ Code + Text
#Questions 7.
import numpy as np
import matplotlib.pyplot as plt
from scipy.stats import t

dfr = 10

rand_nums = t.rvs(dfr, size=100)

first_mean = np.mean(rand_nums)
first_std = np.std(rand_nums)

grp_of_num = 15
sample_each_grp = 30

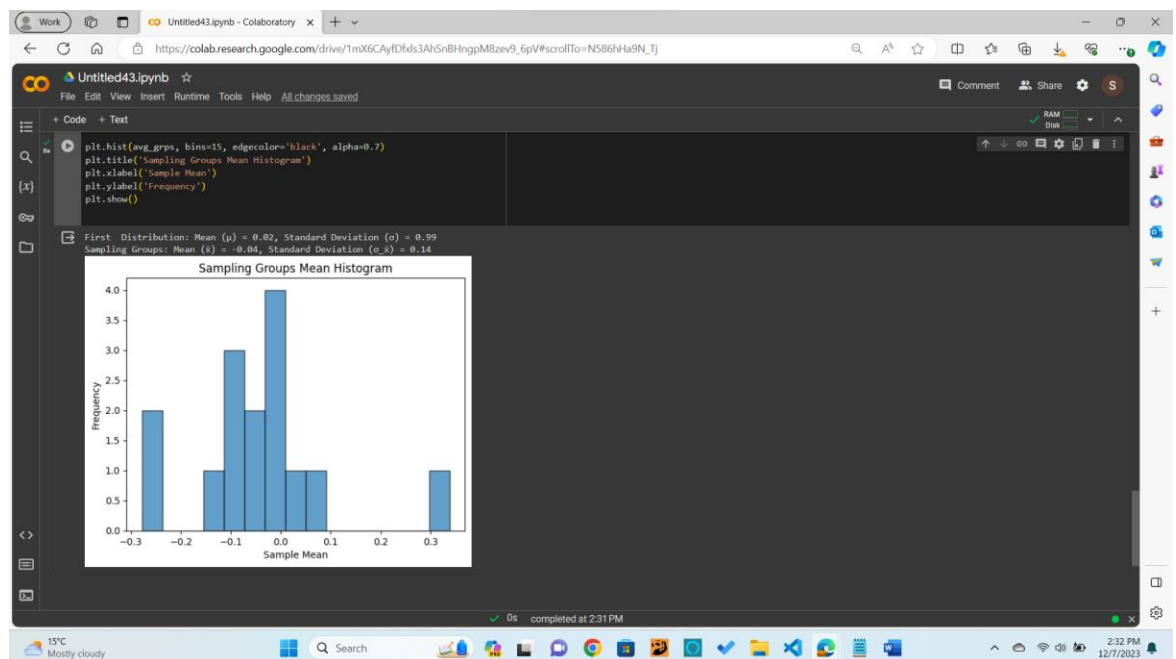
avg_grps = np.zeros(grp_of_num)

for i in range(grp_of_num):
    rand_samp = np.random.choice(rand_nums, size=sample_each_grp, replace=False)
    avg_grps[i] = np.mean(rand_samp)

avg_samp_grps = np.mean(avg_grps)
std_samp_grps = np.std(avg_grps, ddof=1)

print(f"First Distribution: Mean ( $\mu$ ) = {first_mean:.2f}, Standard Deviation ( $\sigma$ ) = {first_std:.2f}")
print(f"Sampling Groups: Mean ( $\bar{x}$ ) = {avg_samp_grps:.2f}, Standard Deviation ( $\sigma_{\bar{x}}$ ) = {std_samp_grps:.2f}")

plt.hist(avg_grps, bins=15, edgecolor='black', alpha=0.7)
plt.title('Sampling Groups Mean Histogram')
plt.xlabel('Sample Mean')
```



Python function link:

<https://www.statology.org/t-distribution-python/>

Excel function link:

<https://support.microsoft.com/en-us/office/t-dist-function-4329459f-ae91-48c2-bba8-1ead1c6c21b2>

