Setup

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

[06 Points] Task 01 - Gaussian Random Variable

Gaussian/Normal random variables are the most common type of random variable. They are characterized by a mean μ and a standard deviation σ. The probability density function of a Gaussian/Normal random variable is given by

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

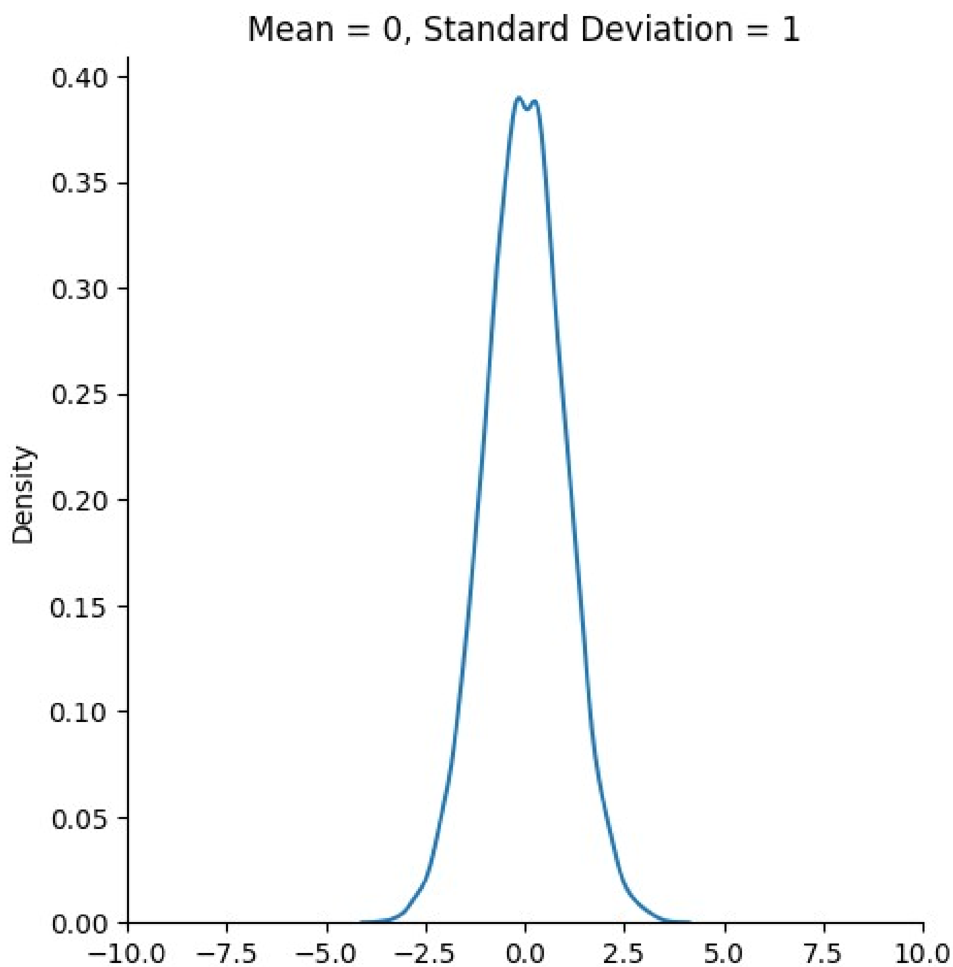
The standard normal distribution is the distribution of the Normal random variable with mean 0 and standard deviation 1 which is denoted by $ X \sim \mathcal{N}(0,1)$.

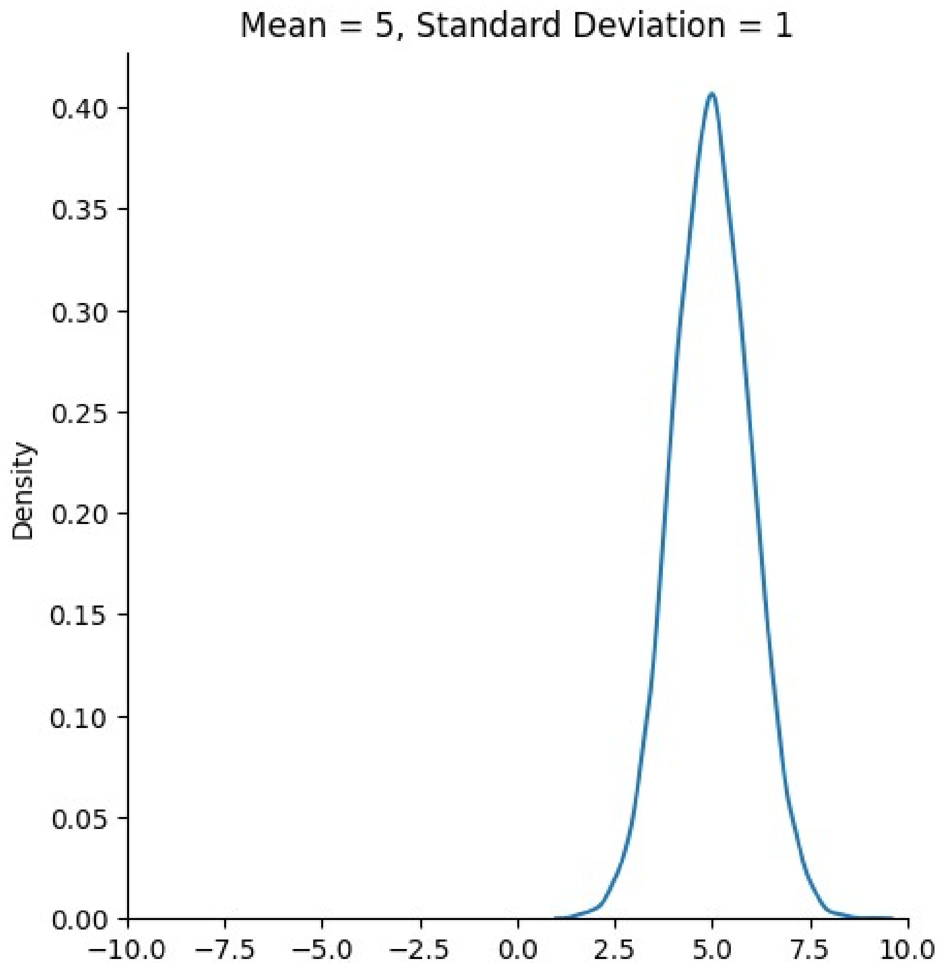
The following code generates 10,000 samples of standard Gaussian random variable X and plots an approximate PDF of X (based on its 10,000 samples) using the displot() function of Seaborn library.

Modify the code to plot the approximate PDF of Gaussian random variable X for the following values of mean: 0, 5, -5 (keeping the standard deviation fixed at 1). Then, plot the approximate PDF of Gaussian random variable X for the following values of standard deviation: 1, 2, 4 (keeping the mean fixed at 0). Comment on the changes in the PDF shape for different values of mean and standard devivation.

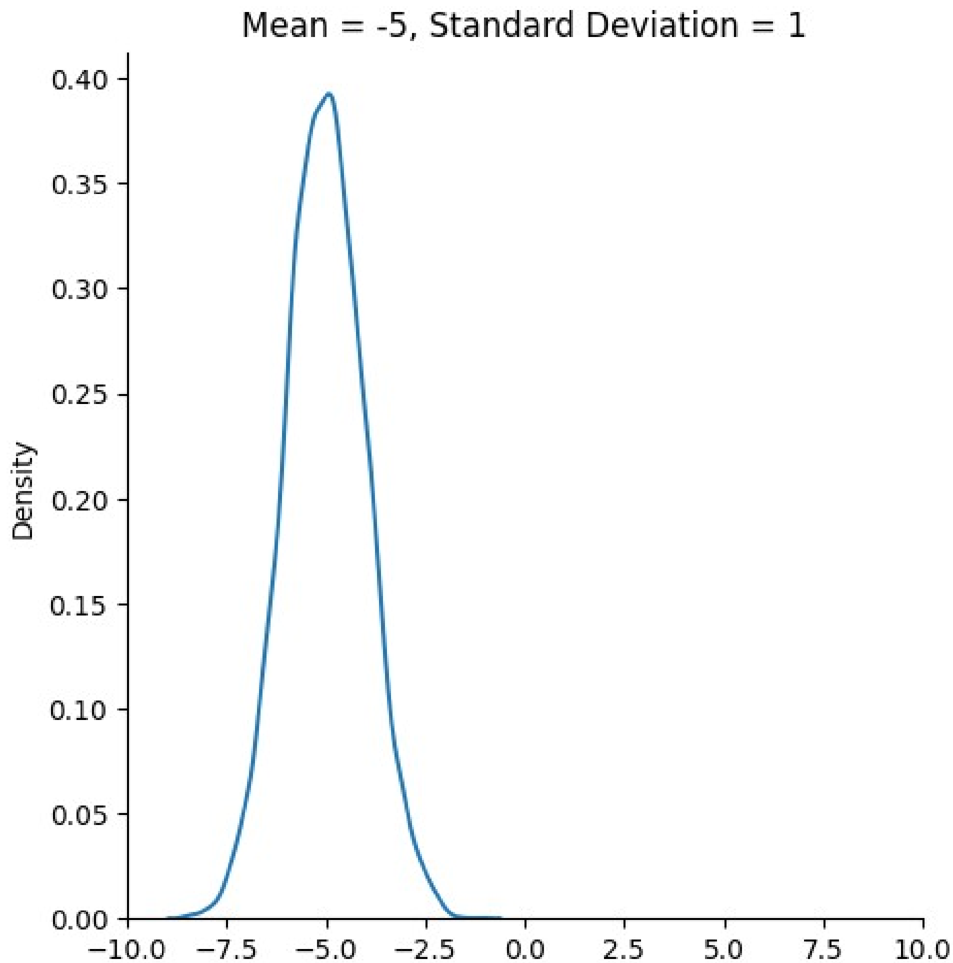
```
# sample_size = 10000

# x_mean0_sd1_sample = np.random.normal(0, 1, sample_size)


# sns.displot(data=x_mean0_sd1_sample,kind="kde").set(title='Mean = 0,
Standard Deviation = 1')
# plt.xlim(-10, 10)

'''Breeha Qasim bq08283 and Hammad Malik hm08298'''

sample_size = 10000

# Samples with mean 0, 5, -5 and standard deviation 1
for mean in [0, 5, -5]:
    samples = np.random.normal(mean, 1, sample_size)
    sns.displot(samples, kind="kde").set(title=f'Mean = {mean},
```

```
Standard Deviation = 1')
    plt.xlim(-10, 10)

# Samples with mean 0 and standard deviation 1, 2, 4
for sd in [1, 2, 4]:
    samples = np.random.normal(0, sd, sample_size)
    sns.displot(samples, kind="kde").set(title=f'Mean = 0, Standard
Deviation = {sd}')
    plt.xlim(-10, 10)
```

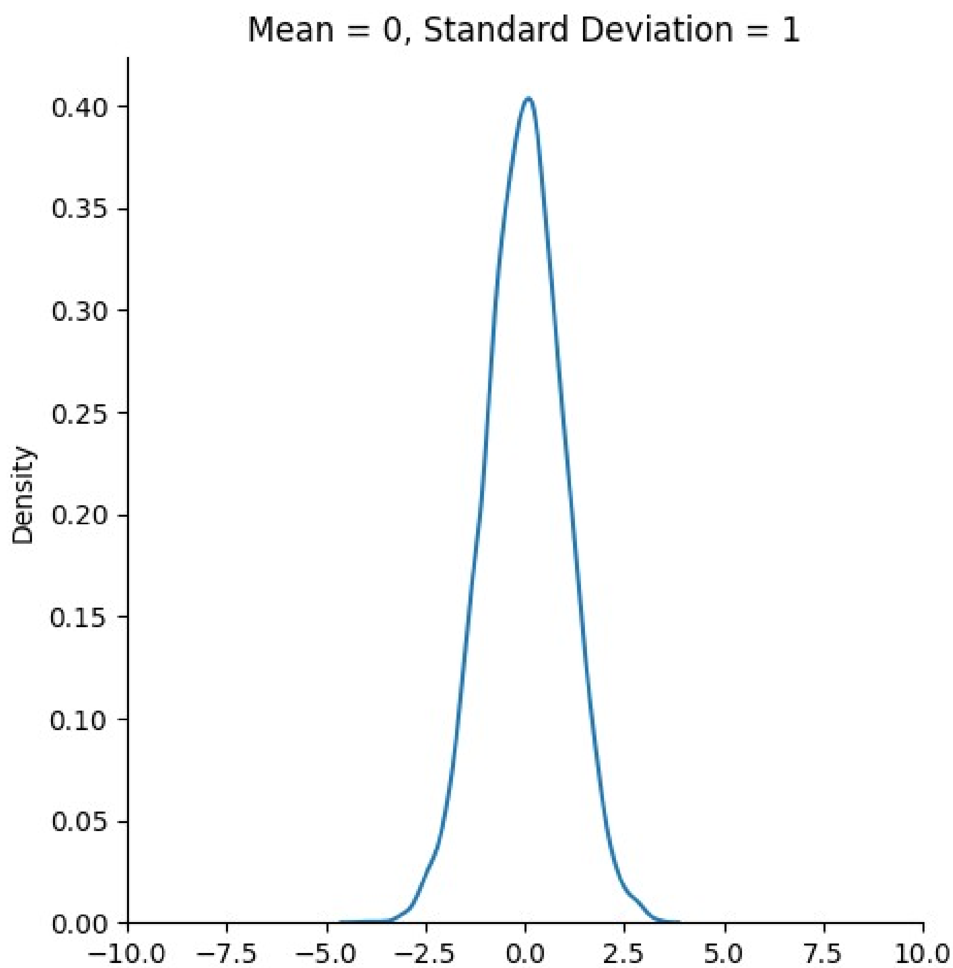


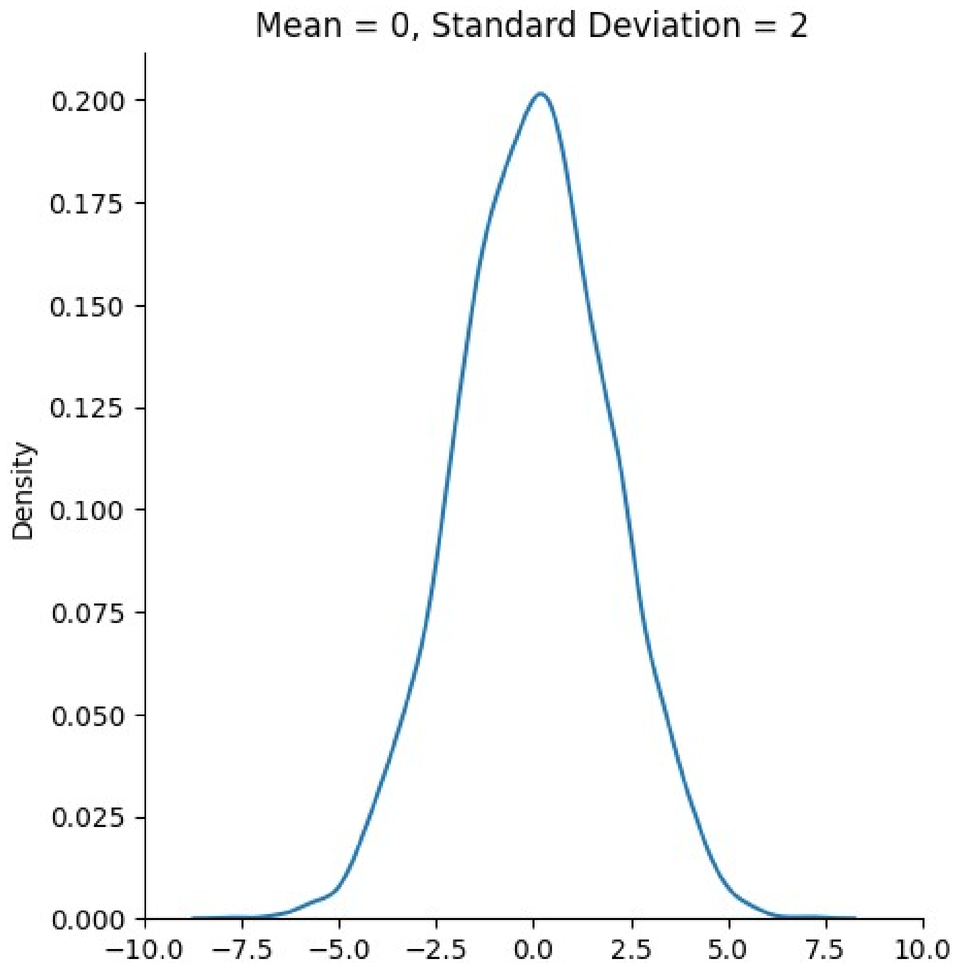


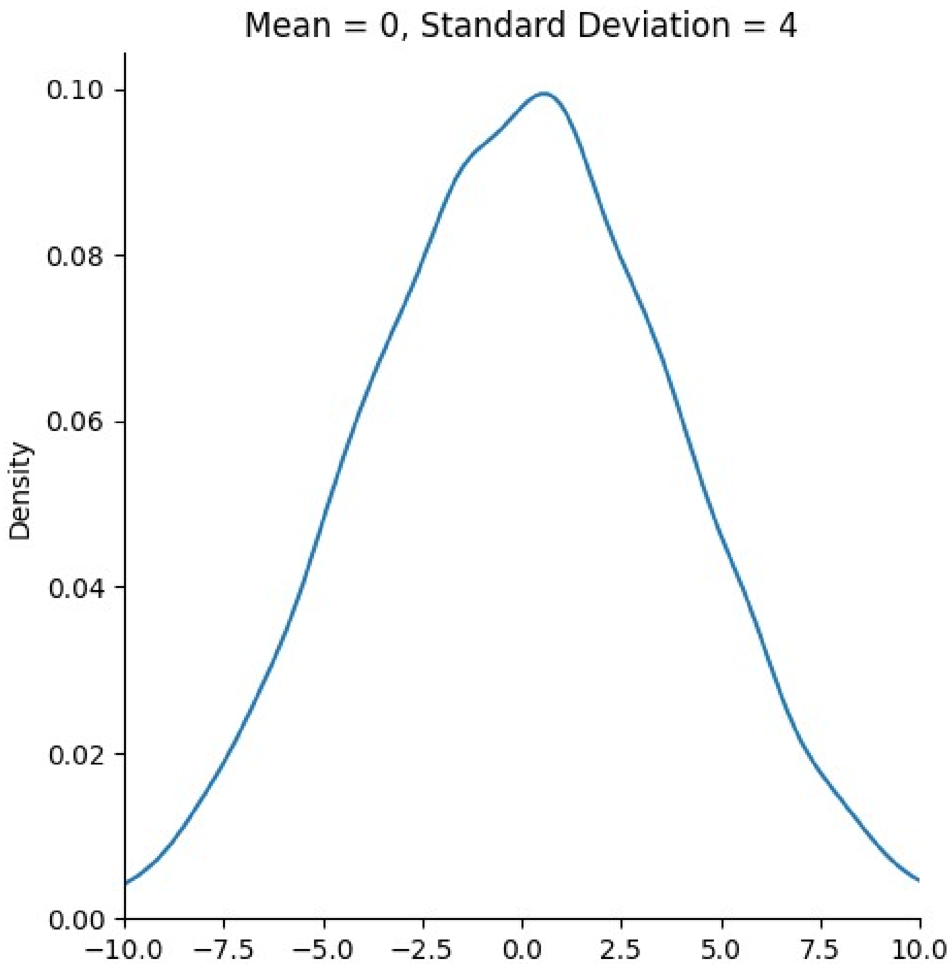
Mean = 5, Standard Deviation = 1
Density
0.40
0.35
0.30
0.25
0.20
0.15
0.10
0.05
0.00
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5
10.0



Mean = -5, Standard Deviation = 1
Density
0.40
0.35
0.30
0.25
0.20
0.15
0.10
0.05
0.00
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5
10.0


Mean = 0, Standard Deviation = 1
Density
0.40
0.35
0.30
0.25
0.20
0.15
0.10
0.05
0.00
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5
10.0


Mean = 0, Standard Deviation = 2
Density
0.200
0.175
0.150
0.125
0.100
0.075
0.050
0.025
0.000
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5
10.0



Comments

Breeha Qasim bq08283 and Hammad Malik hm08298

Changing the mean shifts the peak of the probability density function (PDF) along the x-axis without altering the shape. Increasing or decreasing the mean moves the entire distribution right or left, respectively.

Varying the standard deviation changes the spread of the PDF. A larger standard deviation results in a flatter and wider curve, indicating a larger range of values. A smaller standard deviation leads to a steeper and narrower curve, signifying that the values are more concentrated around the mean.

[20 Points] Task 02 - Functions of Gaussian Random Variable

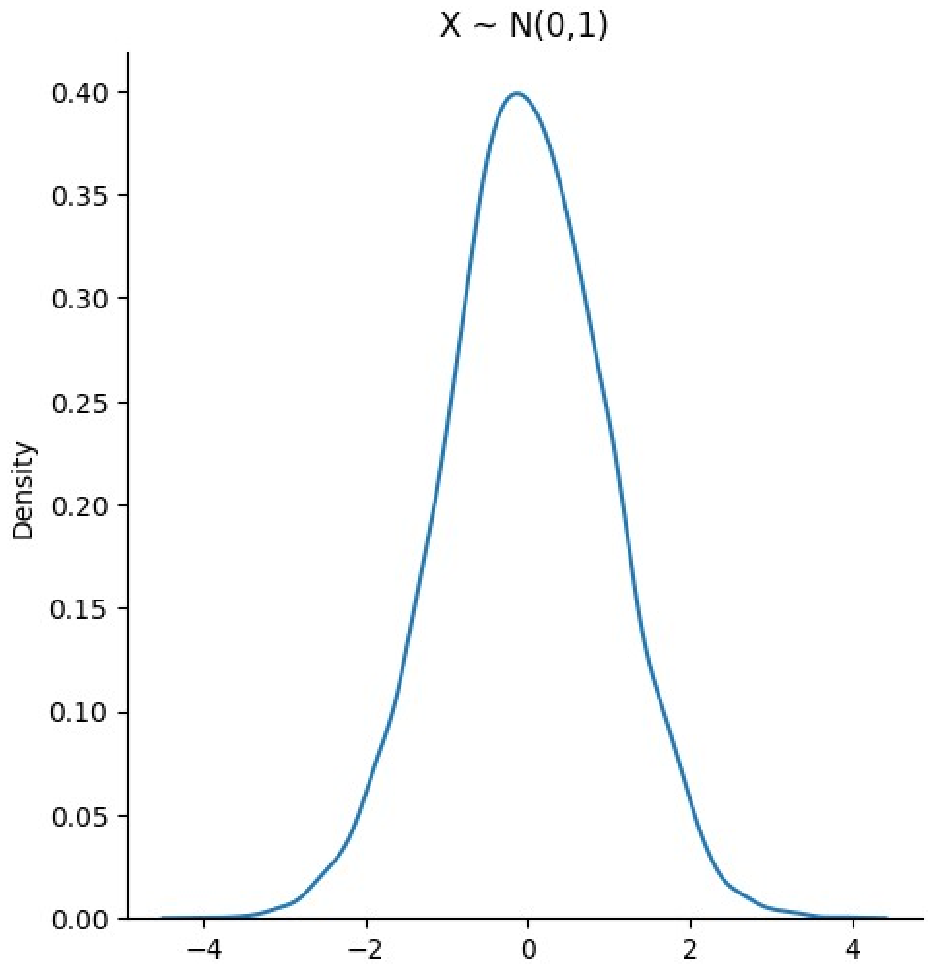
In this task, you are required to comment on the PDF shape of different functions of a Gaussian random variable.

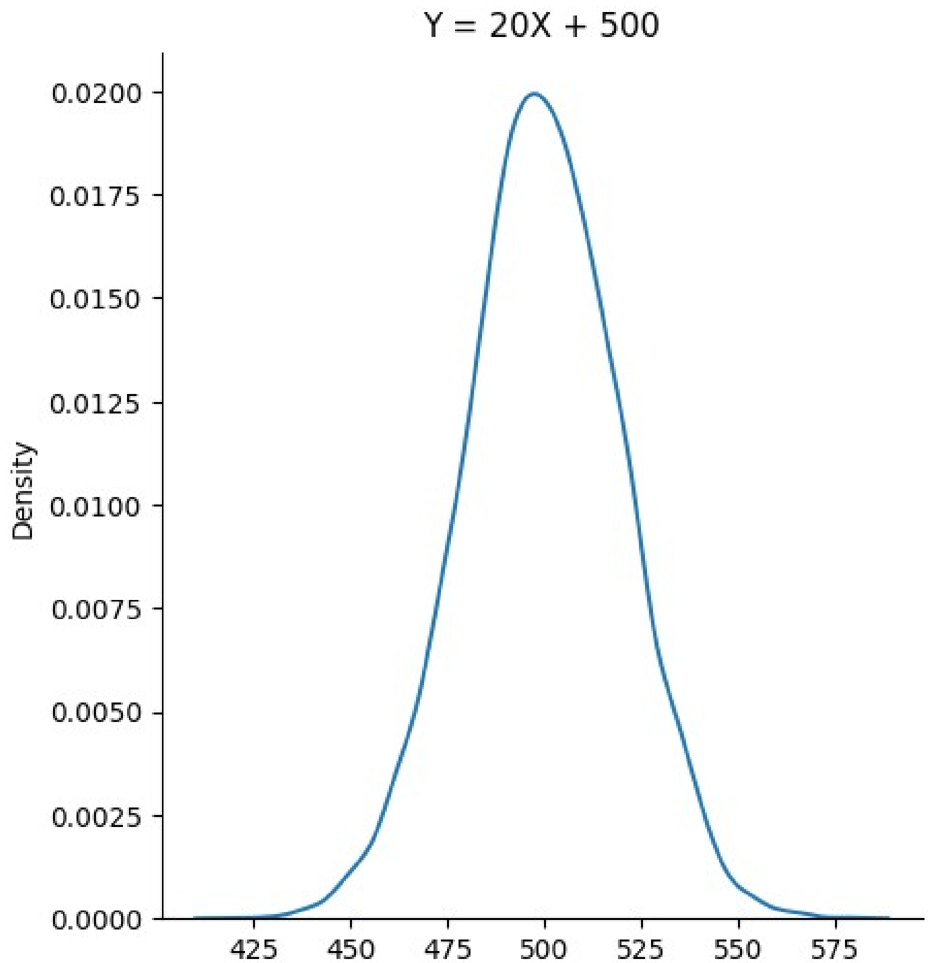
The following code generates 10,000 samples of standard Gaussian random variable X and plots an approximate PDF of X (based on its 10,000 samples) using the displot() function of Seaborn library.

Modify the code to plot the approximate PDF of $Y = 1.8X + 32$ and $Z = X^2 + X + 32$. Comment on the PDF shapes of Y and Z, compared to X.

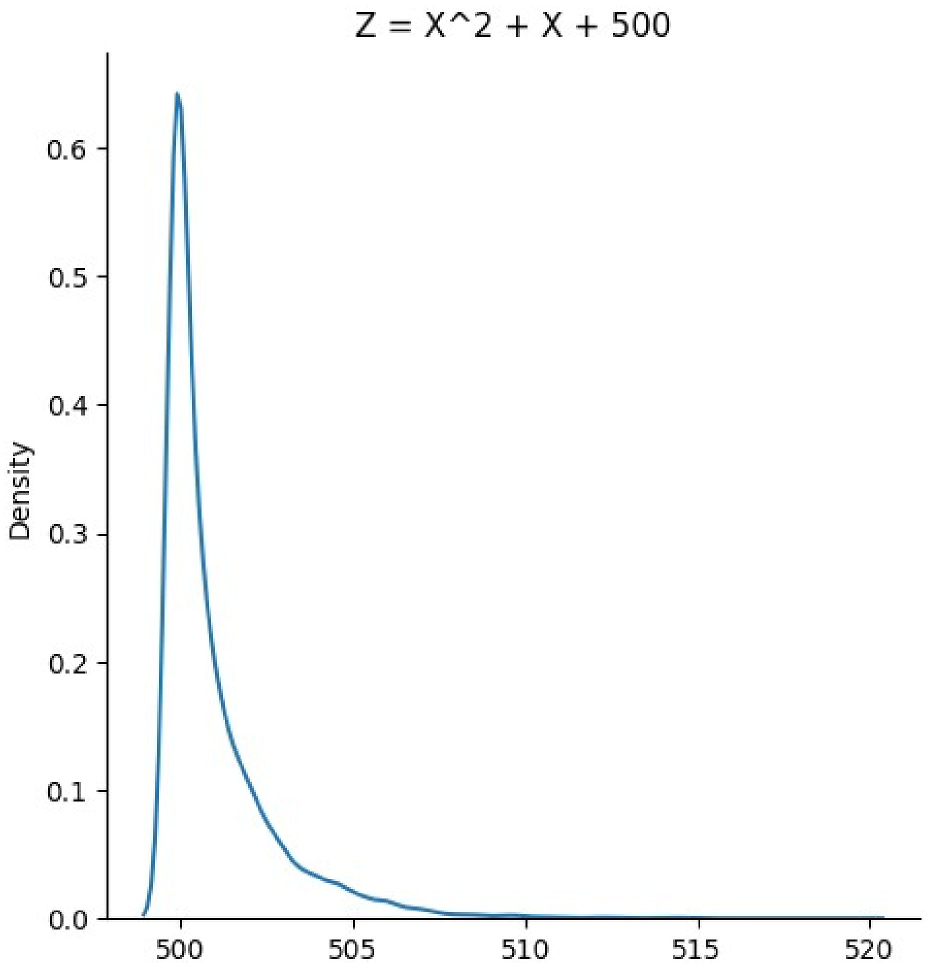
```
'''Breeha Qasim bq08283 and Hammad Malik hm08298'''

sample_size = 10000

# X ~ N(0,1)
x_sample = np.random.normal(loc=0, scale=1, size=sample_size)
sns.displot(x_sample, kind="kde").set(title='X ~ N(0,1)')

# Y = 20X + 500
Y = np.add(np.multiply(20, x_sample), 500)
sns.displot(Y, kind="kde").set(title='Y = 20X + 500')

# Z = X^2 + X + 500
Z = np.add(np.add(np.power(x_sample, 2), x_sample), 500)
sns.displot(Z, kind="kde").set(title='Z = X^2 + X + 500')
```

```
<seaborn.axisgrid.FacetGrid at 0x798acf8479a0>
```


X ~ N(0,1)
Density
0.40
0.35
0.30
0.25
0.20
0.15
0.10
0.05
0.00
−4
−2
0
2
4



Y = 20X + 500
Density
0.0200
0.0175
0.0150
0.0125
0.0100
0.0075
0.0050
0.0025
0.0000
425
450
475
500
525
550
575



Comments

Breeha Qasim bq08283 and Hammad Malik hm08298

The probability density function (PDF) of X represents a standard normal distribution, centered at a mean of 0 with a standard deviation of 1. Its shape is the classic symmetrical bell curve often associated with normal distributions.

In contrast, the PDF of Y retains the bell shape, yet it is centered at a mean of 500 with a dispersion that is 20 times greater than that of X. The transformation Y = 20X + 500 expands the range of X, leading to a broader distribution of Y's values.

The PDF of Z, on the other hand, does not follow a normal distribution and exhibits a rightward skew with a "U" shaped appearance. The quadratic transformation Z = (X^2) + X + 500 imparts the rightward skewness due to the X squared component, while the addition of 500 shifts the entire distribution rightward.

[24 points] Task 03 - Sum of Independent and Identically-Distributed Random Variables

Suppose $X_1, X_2, \cdots, X_n$ are independent random variables with the same underlying distribution. In this case, we say that the X_i are independent and identically distributed or i.i.d. In particular, the X_i, all have the same mean μ and standard deviation σ.

Let S_n be the sum of n i.i.d random variables:

$$S_n = X_1 + X_2 + \cdots + X_n$$

[6 points] Part A: Uniform Random Variable

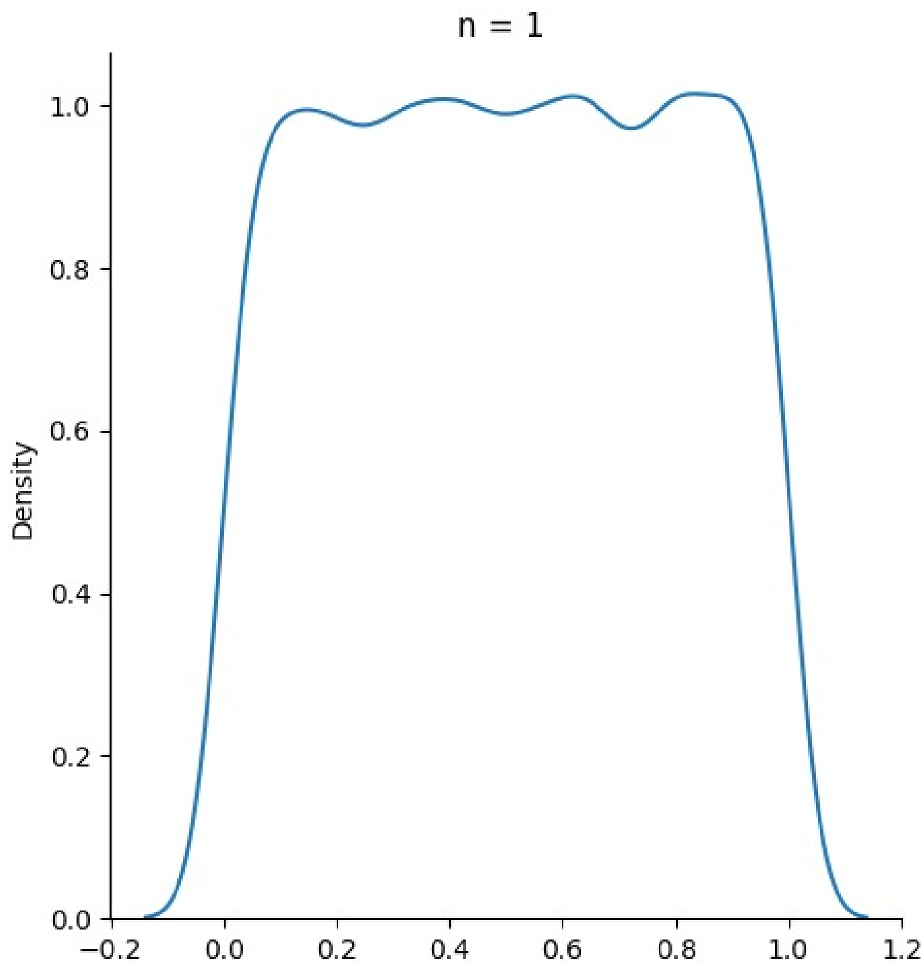
In this part, we consider X_is to be continuous uniform random variables. The following code generates 10,000 samples of X_1 random variable and plots an approximate PDF of X_1 (based on its 10,000 samples) using the displot() function of Seaborn library.

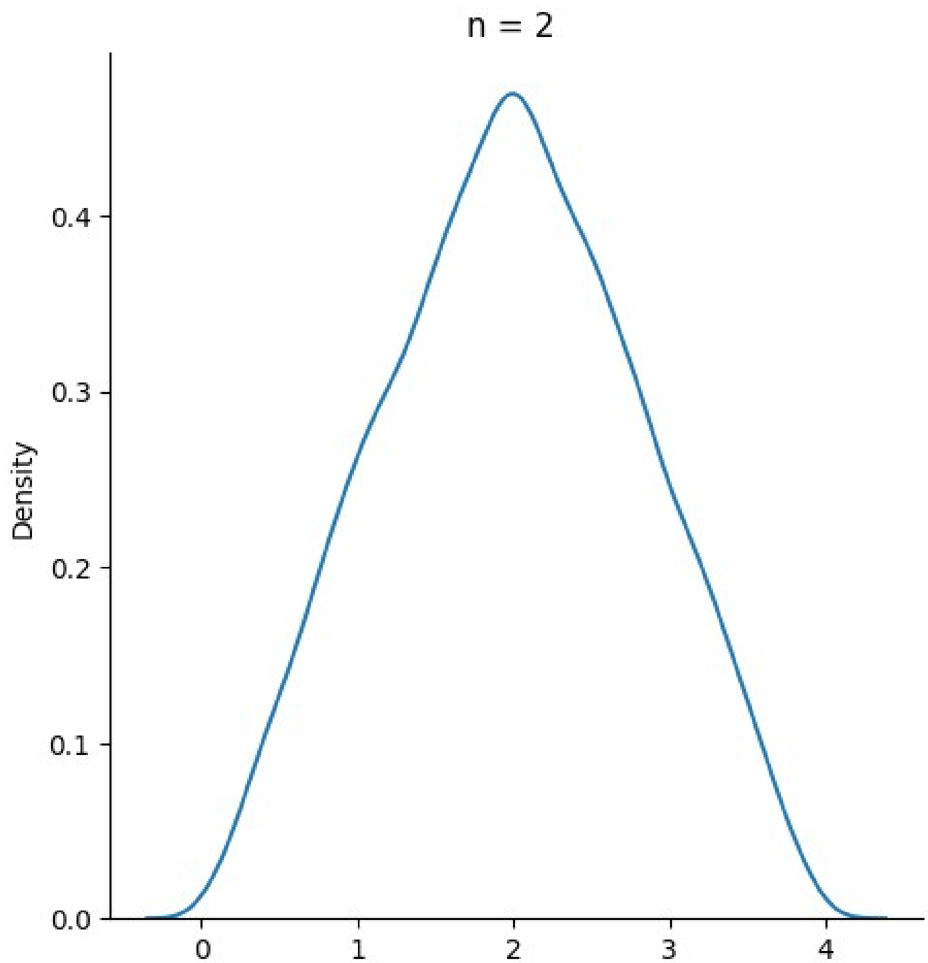
Modify the given code to also generate 10,000 samples of $X_2, X_3, \cdots, X_n$, and plot the approximate PDF of S_n using the displot() function of Seaborn library for the following values of n: 1,2,3,5,10,50,100.

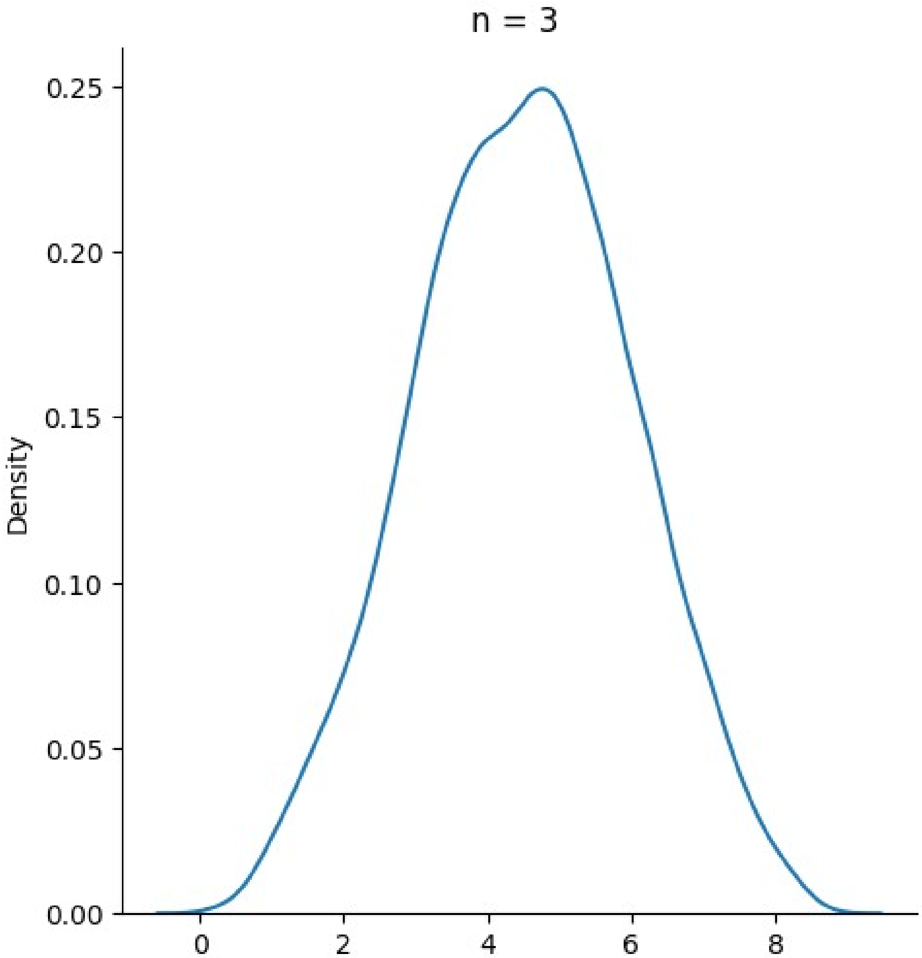
For example, for n=2, you have to plot the approximate PDF of $S_2 = X_1 + X_2$ and for n=3, you have to plot the approximate PDF of $S_3 = X_1 + X_2 + X_3$

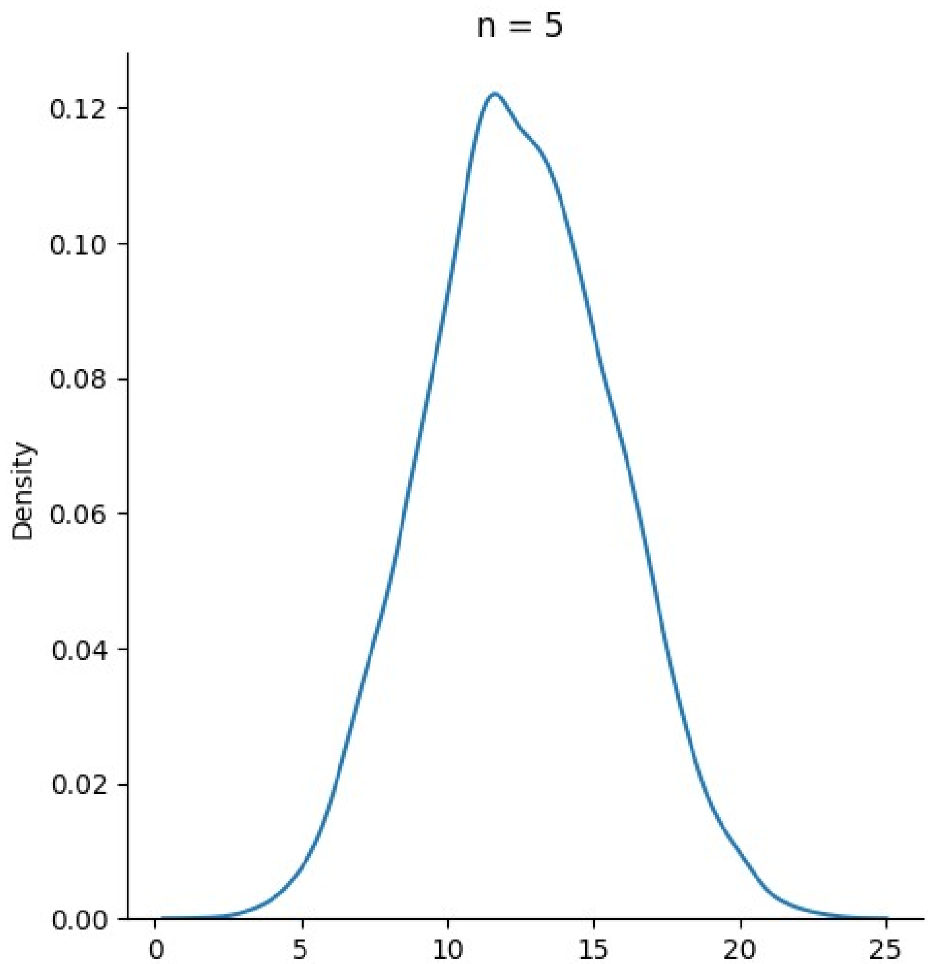
Comment on the evolution of PDF shape of S_n as n increases.

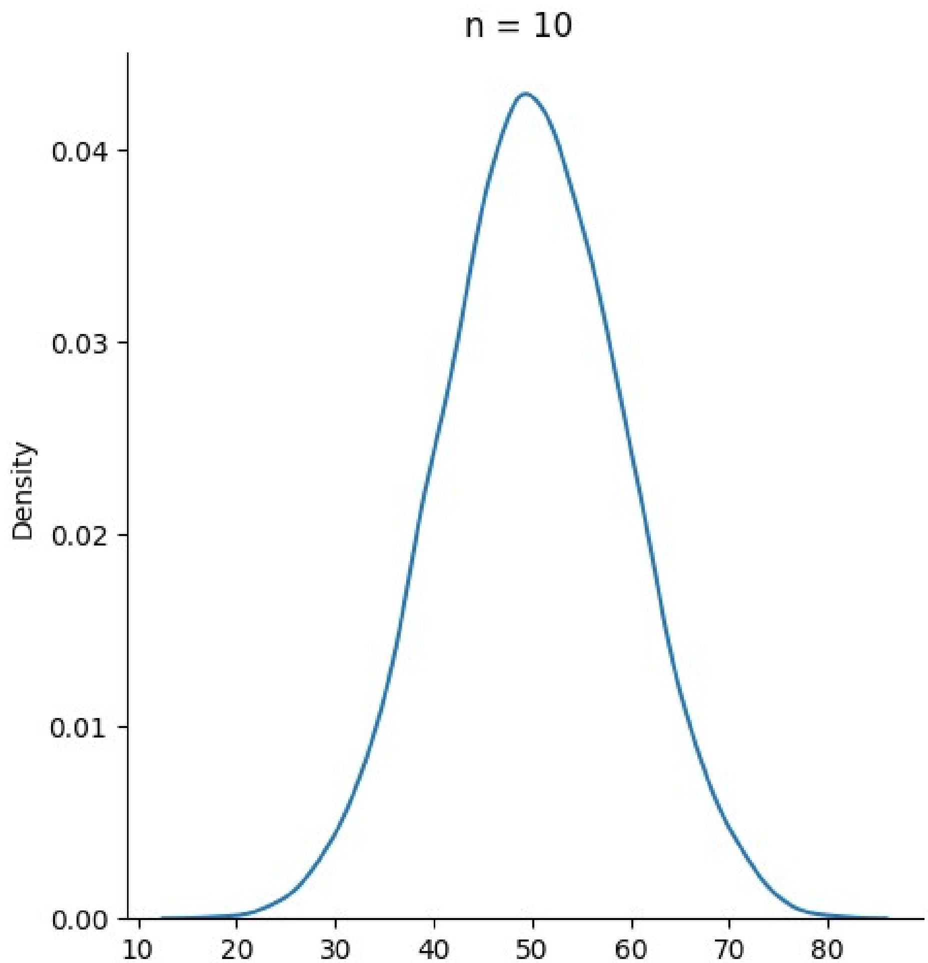
```
'''Breeha Qasim bq08283 and Hammad Malik hm08298'''

sample_size = 10000
df = pd.DataFrame()
n_values = [1, 2, 3, 5, 10, 50, 100] # Different sample sizes to
generate sums for

for n in n_values:
    # Generate 'count' samples from a uniform distribution and sum
them up
    uniform_samples = np.array([np.random.uniform(0, n, sample_size)
for i in range(1, n + 1)]).sum(axis=0)
    df[f'x{n} sample'] = uniform_samples
    sns.displot(uniform_samples, kind="kde").set(title= f'n = {n} ') #
Plotting the pdf of Sn where Sn = X1 + X2 + X3 + ... + Xn
```

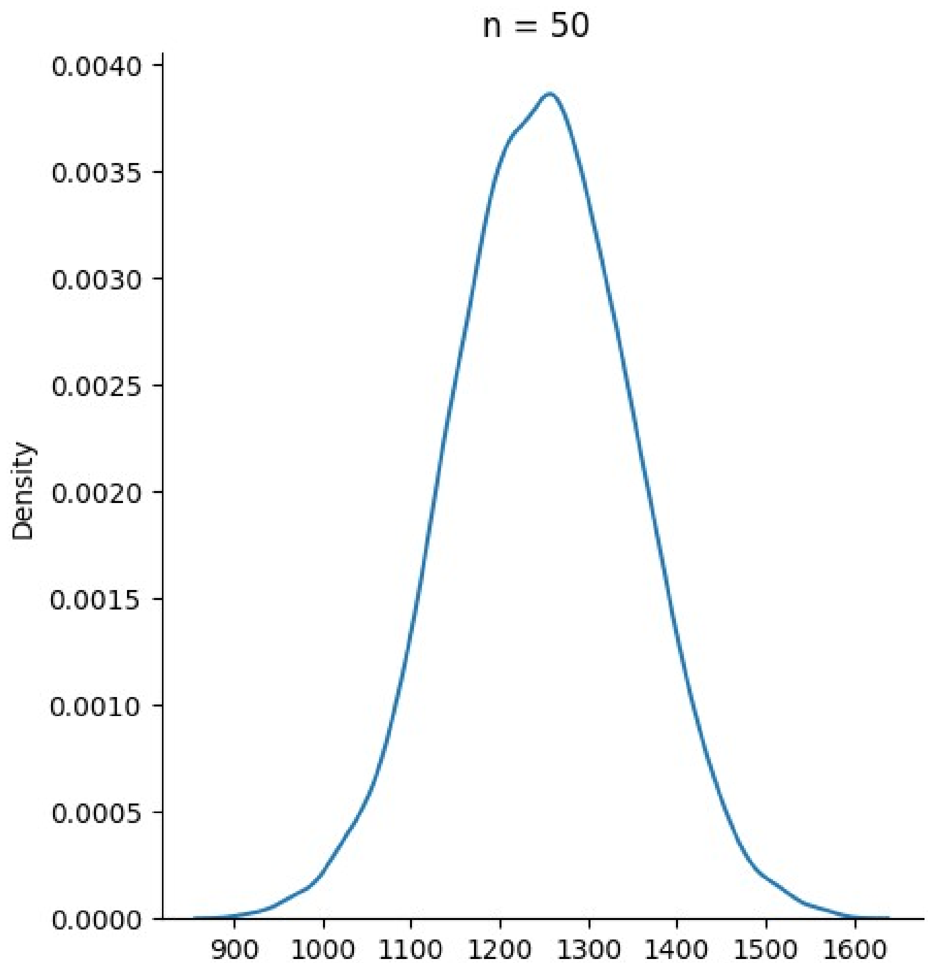

n = 1
Density
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
0.0
0.2
0.4
0.6
0.8
1.0
1.2


n = 2
Density
0.0
0.1
0.2
0.3
0.4
0
1
2
3
4

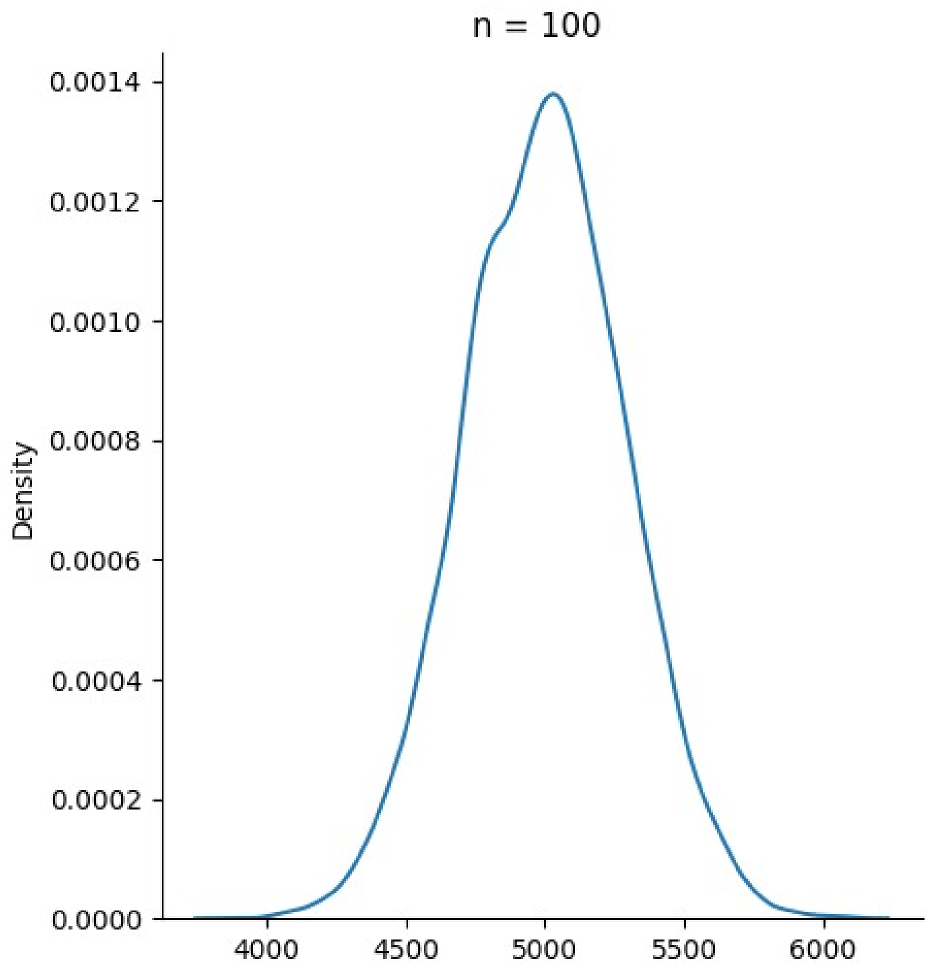

n = 3
Density
0.25
0.20
0.15
0.10
0.05
0.00
0
2
4
6
8


n = 5
Density
0.00
0.02
0.04
0.06
0.08
0.10
0.12
0
5
10
15
20
25


n = 10
Density
0.00
0.01
0.02
0.03
0.04
10
20
30
40
50
60
70
80



n = 50
Density
0.0040
0.0035
0.0030
0.0025
0.0020
0.0015
0.0010
0.0005
0.0000
900
1000
1100
1200
1300
1400
1500
1600



Comments

Breeha Qasim bq08283 and Hammad Malik hm08298

When examining the initial graph where n = 1, we observe a plateau-like peak indicative of a uniform distribution, as at this stage S_n consists solely of X_1. As the value of 'n' is incremented, a transformation in the graph's contour becomes evident, progressively approximating the distinct bell-shaped curve associated with the Gaussian distribution, particularly noticeable in the $x\,100$ sample graph. This phenomenon aligns with the Central Limit Theorem, which predicts that the aggregation of a substantial number of independent and identically distributed variables will converge towards a normal distribution, irrespective of the original variables' distributions. Additionally, with the rise in 'n', the distribution's mean progresses to the right due to the inclusion of more variables. Concurrently, the peak diminishes and the spread widens, maintaining the area under the curve constant at 1, thereby ensuring the probability density function remains properly normalized.

[6 points] Part B: Exponential Random Variable

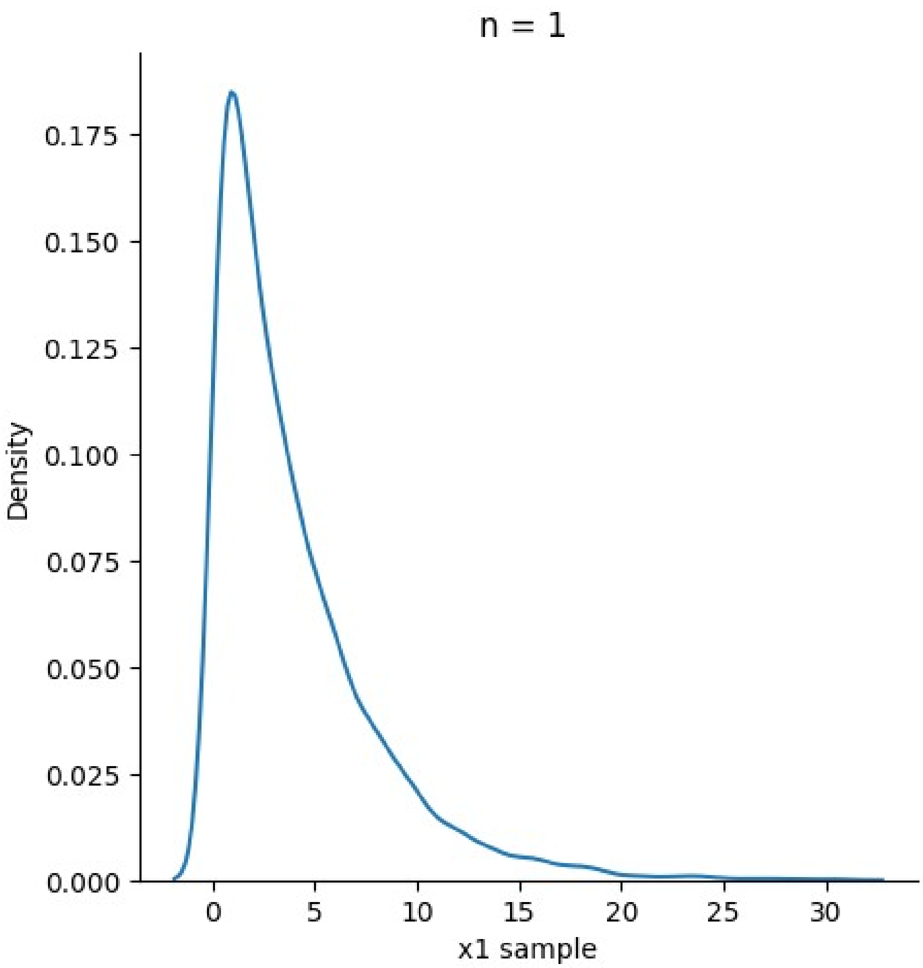
In this part, we consider X_is to be exponential random variables. The following code generates 10,000 samples of X_1 random variable and plots an approximate PDF of X_1 (based on its 10,000 samples) using the displot() function of Seaborn library.

Modify the given code to generate 10,000 samples of $X_2, X_3, \cdots, X_n$, and plot the approximate PDF of S_n using the displot() function of Seaborn library for the following values of n: 1,2,3,5,10,50,100.

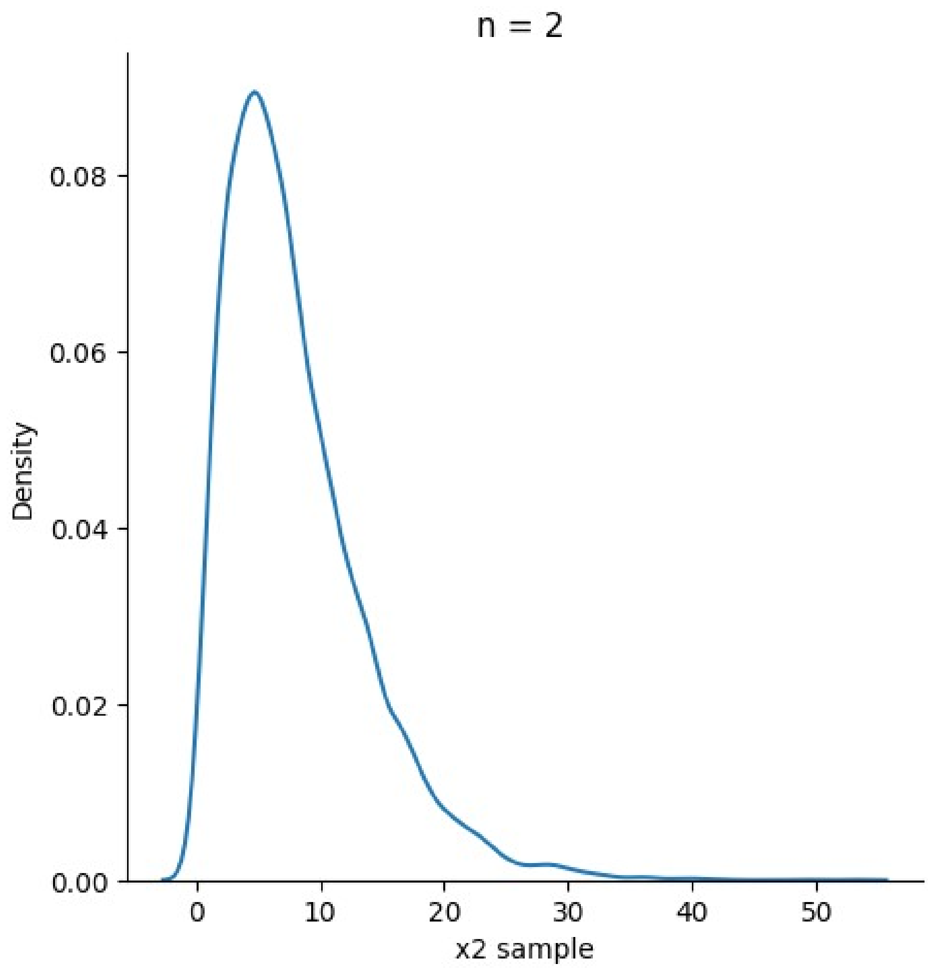
For example, for n=2, you have to plot the approximate PDF of $S_2 = X_1 + X_2$ and for n=3, you have to plot the approximate PDF of $S_3 = X_1 + X_2 + X_3$

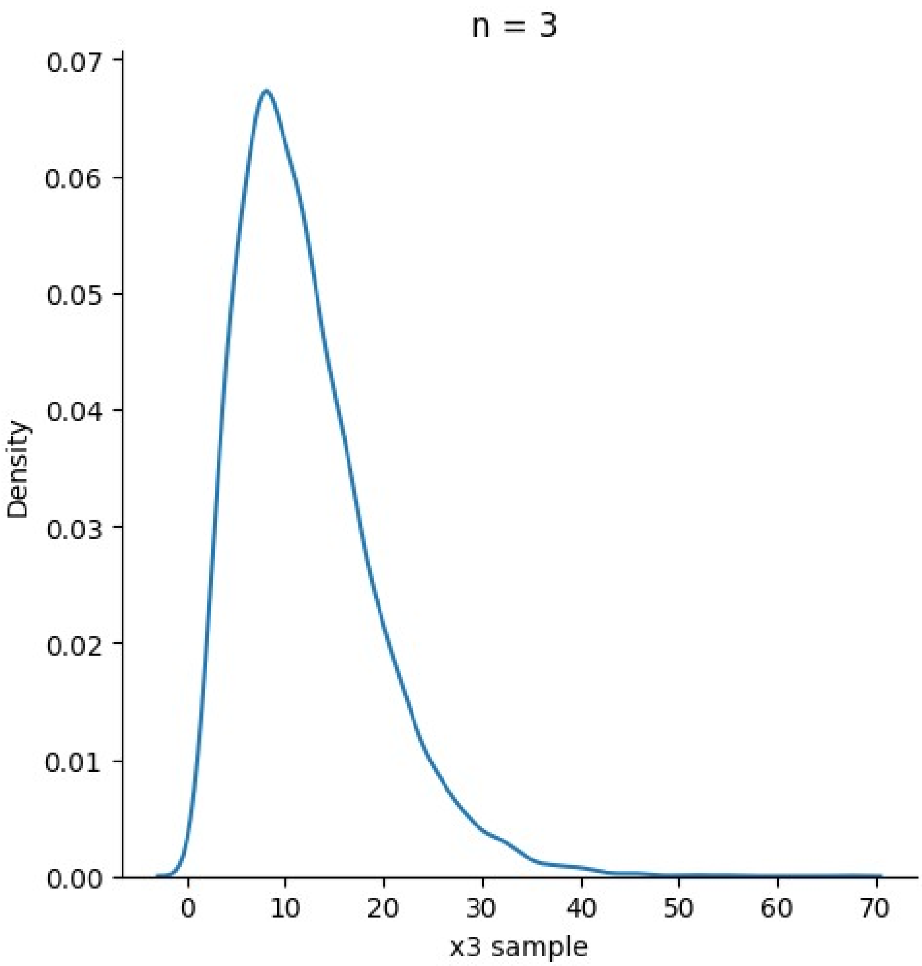
Comment on the evolution of PDF shape of S_n as n increases.

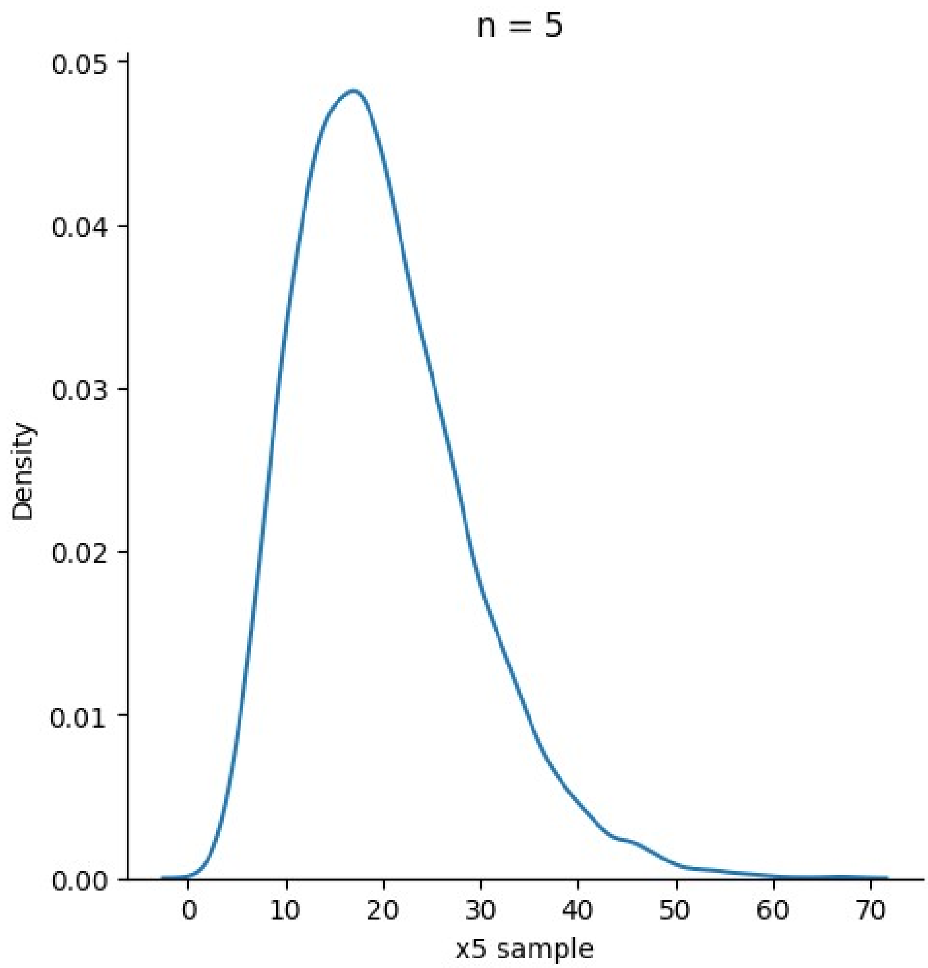
```
'''Breeha Qasim bq08283 and Hammad Malik hm08298'''

sample_size = 10000
rate = 0.25 #Rate of exponential random variables
df = pd.DataFrame()
n_values = [1, 2, 3, 5, 10, 50, 100]

for n in n_values:
    x_sample = np.sum([np.random.exponential(1 / rate, sample_size)
for i in range(n)], axis=0)
    df[f'x{n} sample'] = x_sample
    sns.displot(data=df, x=f'x{n} sample', kind="kde").set(title= f'n
= {n} ')
```

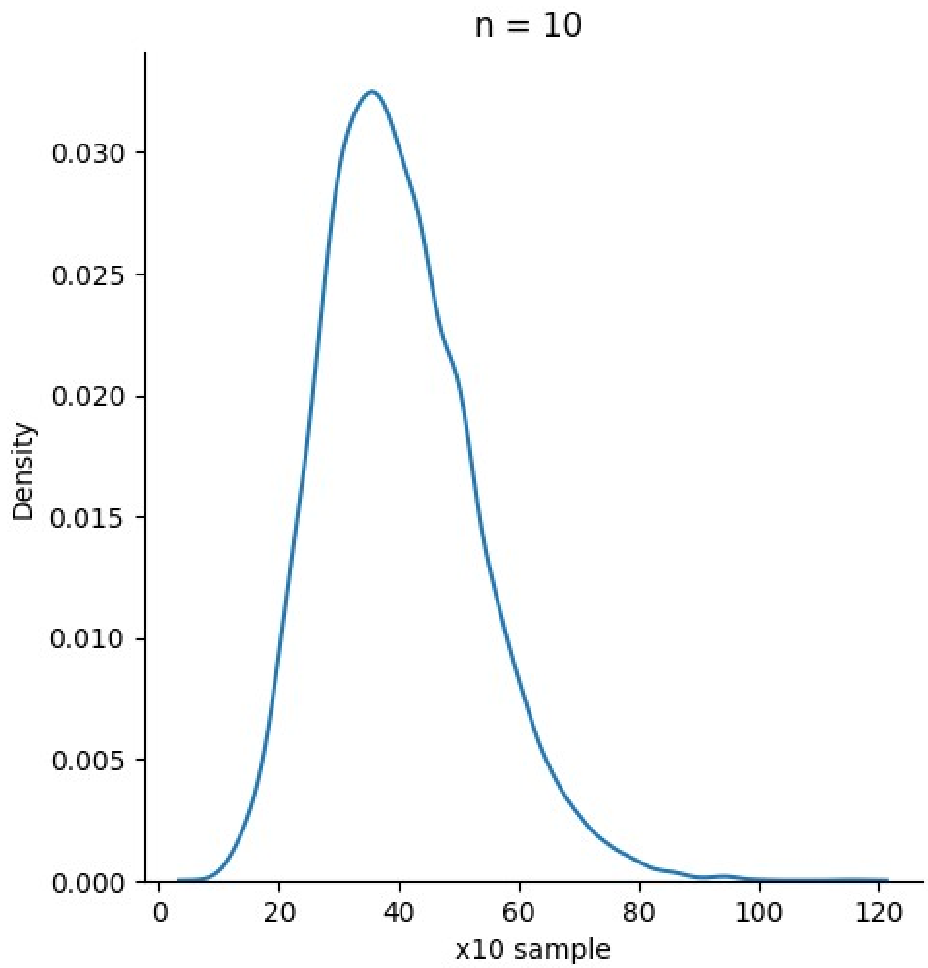


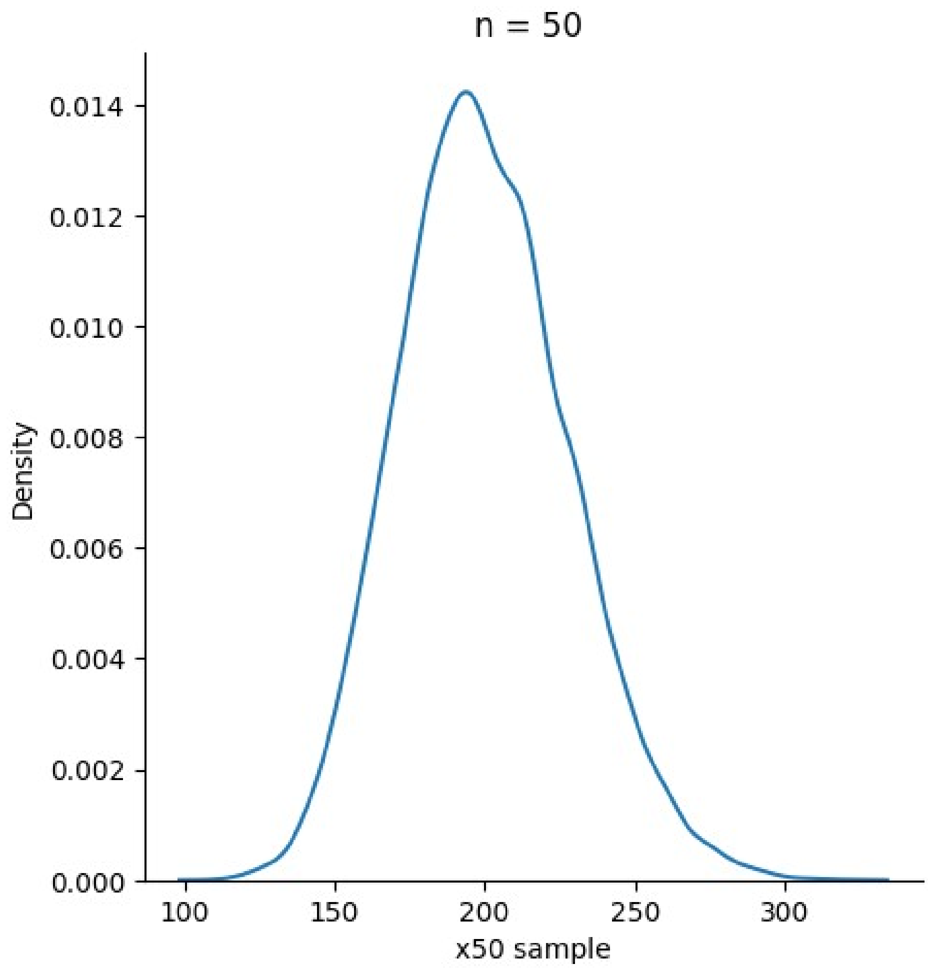
n = 1
Density
0.000
0.025
0.050
0.075
0.100
0.125
0.150
0.175
0
5
10
15
20
25
30
x1 sample


n = 2
Density
0.00
0.02
0.04
0.06
0.08
0
10
20
30
40
50
x2 sample

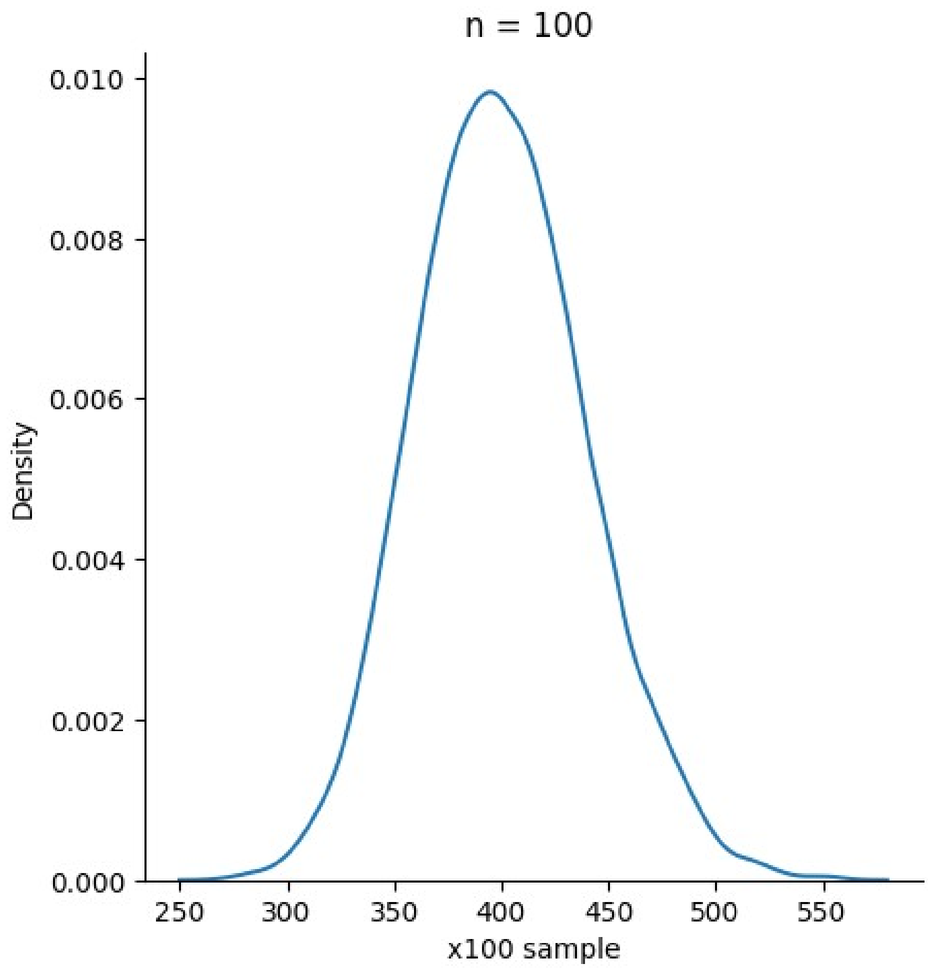

n = 3
0.07
0.06
0.05
0.04
0.03
0.02
0.01
0.00
Density
0
10
20
30
40
50
60
70
x3 sample


n = 5
0.05
0.04
0.03
0.02
0.01
0.00
Density
0
10
20
30
40
50
60
70
x5 sample


n = 10
Density
0.000
0.005
0.010
0.015
0.020
0.025
0.030
0
20
40
60
80
100
120
x10 sample



n = 50
Density
0.014
0.012
0.010
0.008
0.006
0.004
0.002
0.000
100
150
200
250
300
x50 sample



Comments

Breeha Qasim bq08283 and Hammad Malik hm08298

Initially, when n is set to 1, the probability density function (PDF) S_n mirrors what you would expect from an Exponential Random Variable. With an increase in n, the PDF S_n starts to take on the characteristics of a Gaussian curve, notably its bell shape. As n becomes larger, there's a noticeable shift in the mean towards higher values, and the curve's peak becomes less pronounced, while the spread of the distribution broadens. This is to maintain the total area under the curve at 1, preserving the properties of a valid PDF.

This transformation aligns with the Central Limit Theorem, which posits that the accumulated sum of a substantial number of independent random variables, each with similar distributions, will converge towards a normal distribution, regardless of the original distribution of the individual variables.

[6 points] Part C: Binomial Random Variable

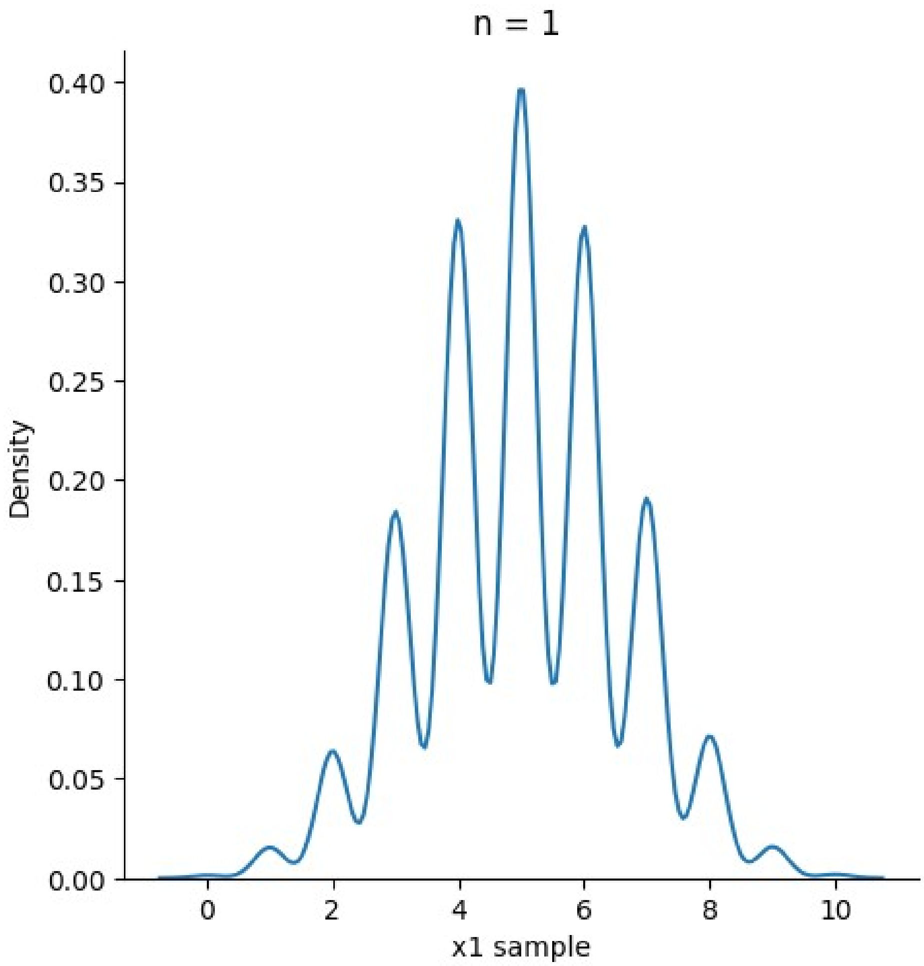
In this part, we consider X_is to be binomial random variables. The following code generates 10,000 samples of X_1 random variable and plots an approximate PDF of X_1 (based on its 10,000 samples) using the displot() function of Seaborn library.

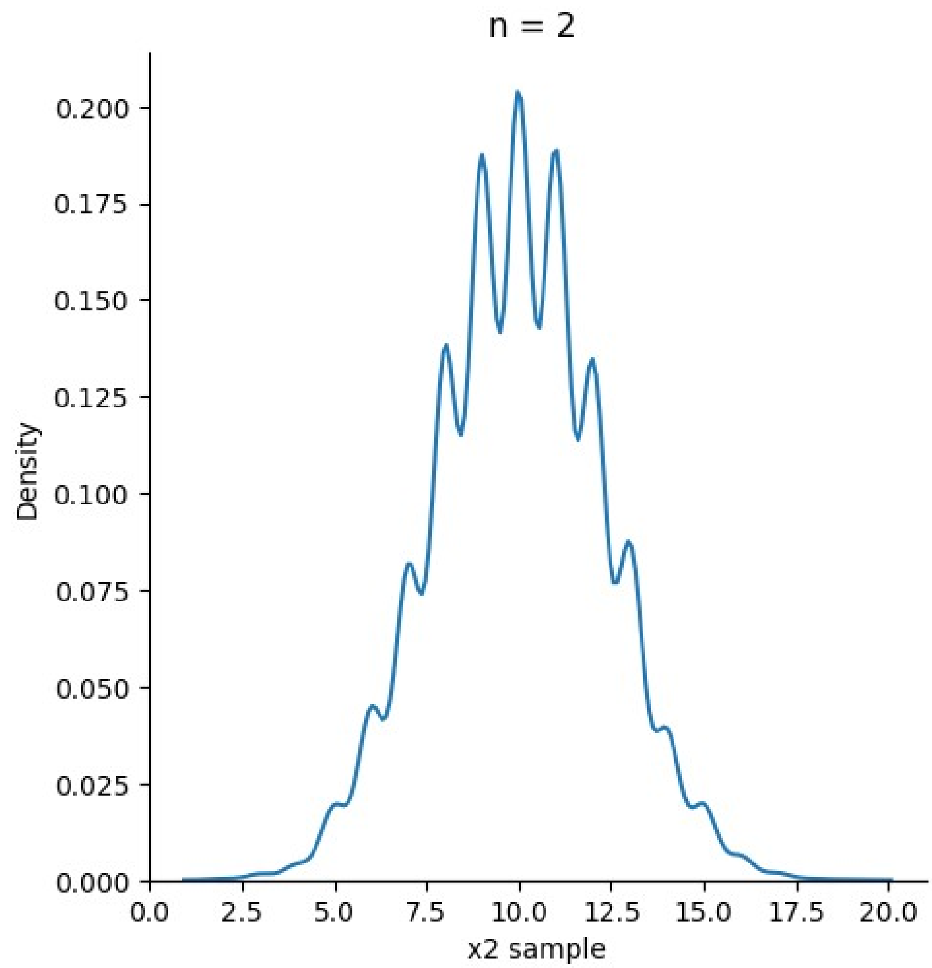
Modify the given code to generate 10,000 samples of $X_2, X_3, \cdots, X_n$, and plot the approximate PDF of S_n using the displot() function of Seaborn library for the following values of n: 1,2,3,5,10,50,100.

For example, for n=2, you have to plot the approximate PDF of $S_2 = X_1 + X_2$ and for n=3, you have to plot the approximate PDF of $S_3 = X_1 + X_2 + X_3$

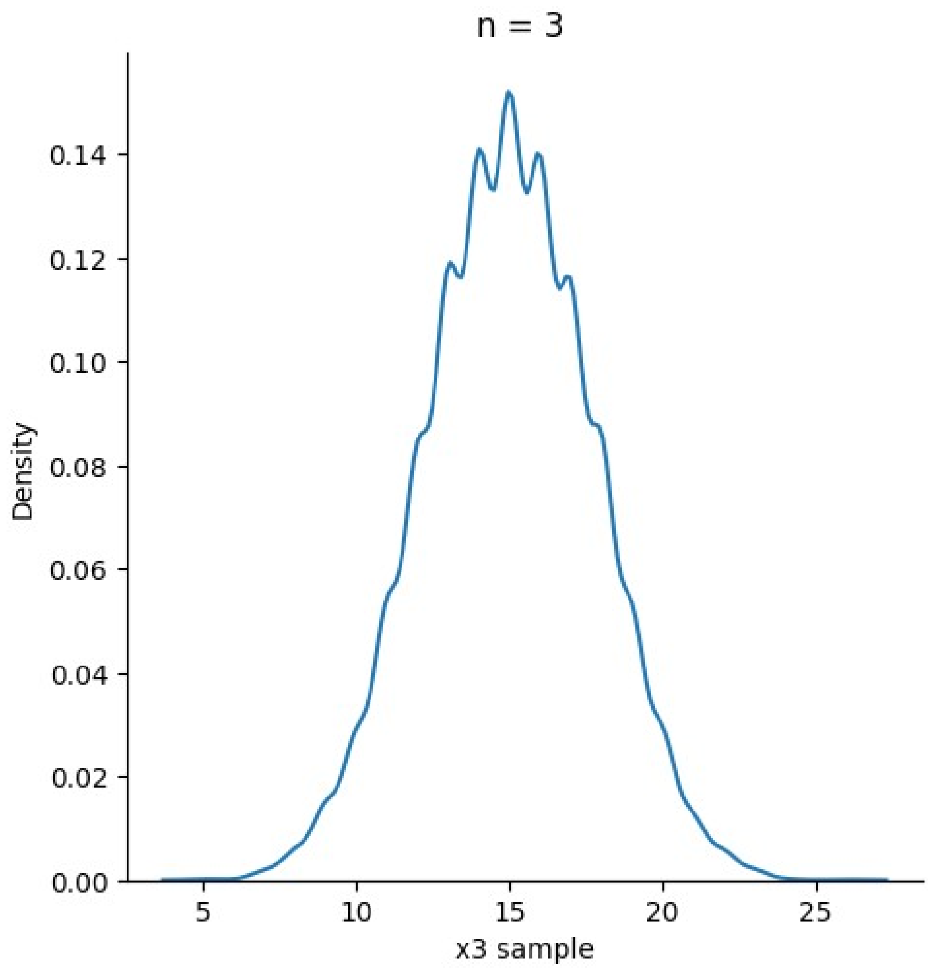
Comment on the evolution of PDF shape of S_n as n increases.

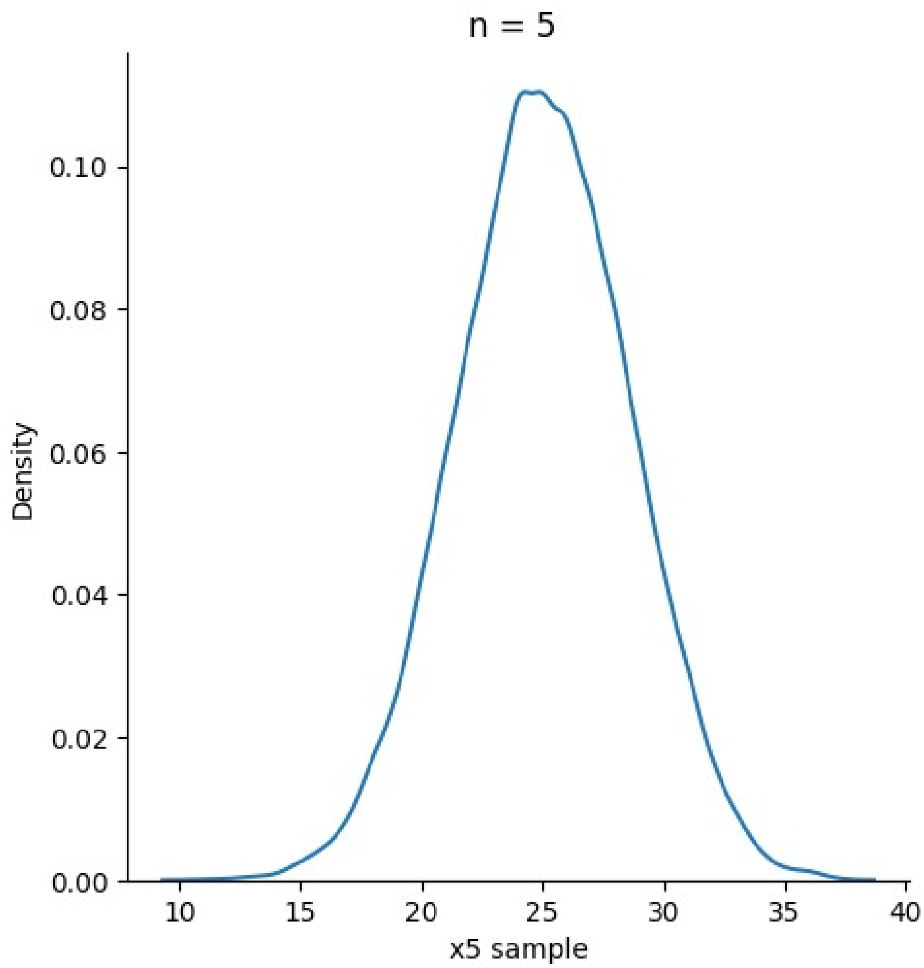
```
'''Breeha Qasim bq08283 and Hammad Malik hm08298'''

sample_size = 10000
df = pd.DataFrame()
n_values = [1, 2, 3, 5, 10, 50, 100]

for n in n_values:
    x_sample = sum(np.random.binomial(10, 0.5, size=sample_size) for i
in range(n))
    df[f'x{n} sample'] = x_sample
    sns.displot(df[f'x{n} sample'], kind="kde").set(title= f'n = {n}
')
```


n = 1
0.40
0.35
0.30
0.25
0.20
0.15
0.10
0.05
0.00
Density
0
2
4
6
8
10
x1 sample

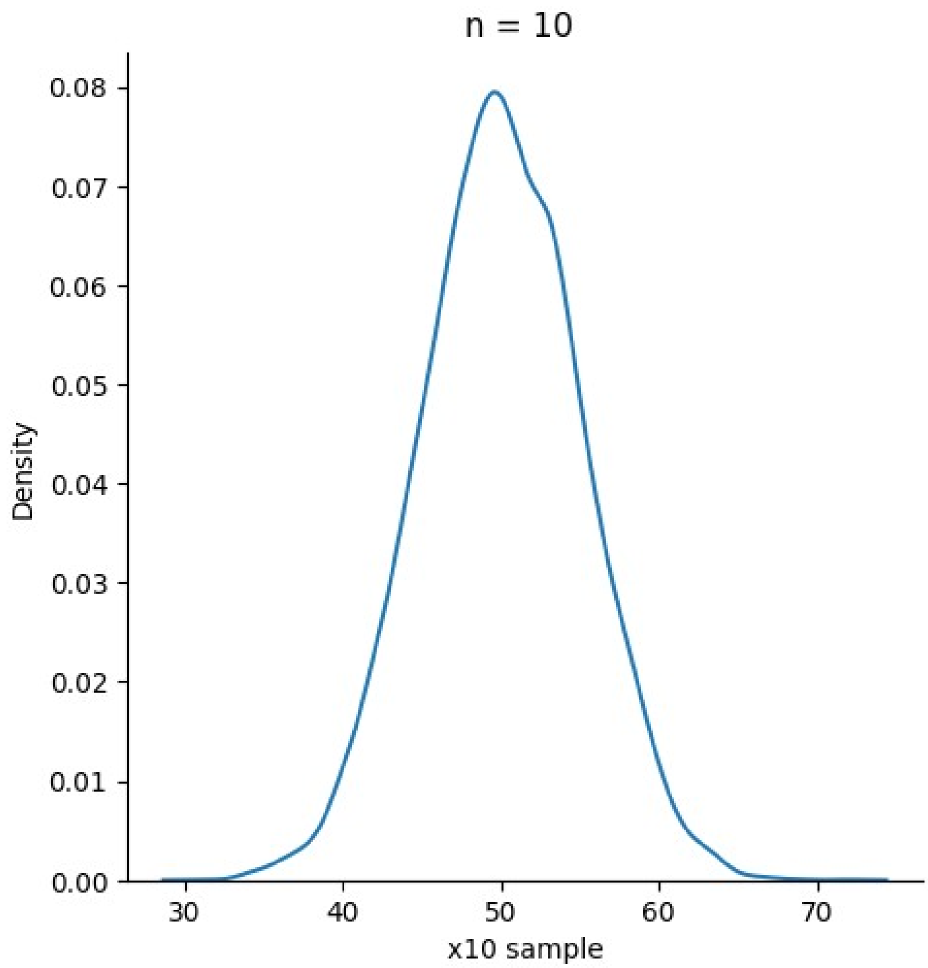


n = 2
Density
0.200
0.175
0.150
0.125
0.100
0.075
0.050
0.025
0.000
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0
x2 sample

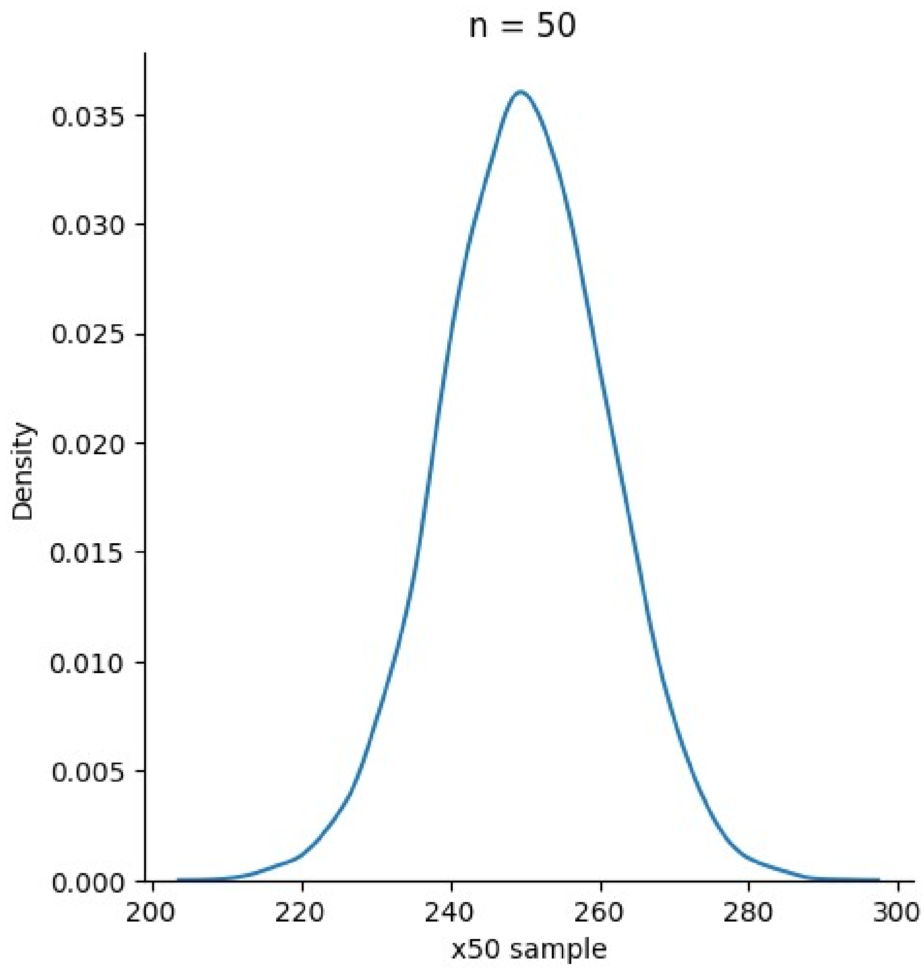

n = 3
Density
0.00
0.02
0.04
0.06
0.08
0.10
0.12
0.14
5
10
15
20
25
x3 sample

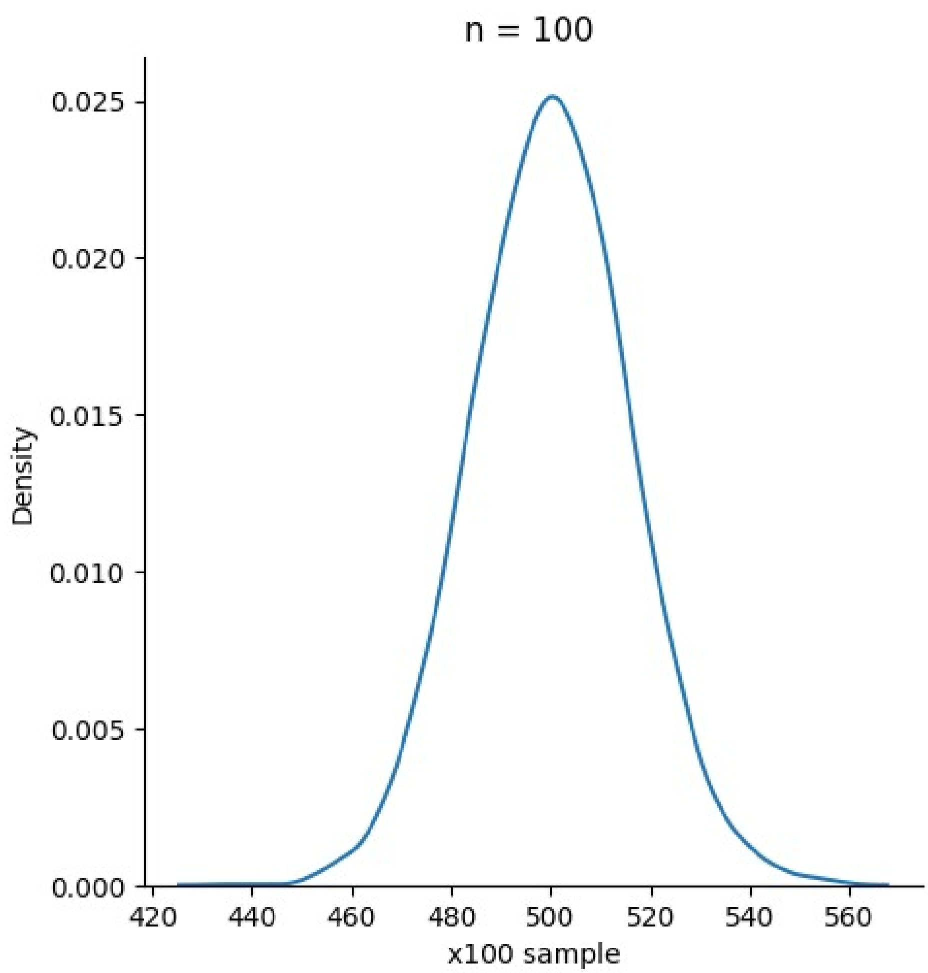


n = 5
Density
0.00
0.02
0.04
0.06
0.08
0.10
10
15
20
25
30
35
40
x5 sample



n = 10
Density
0.08
0.07
0.06
0.05
0.04
0.03
0.02
0.01
0.00
30
40
50
60
70
x10 sample


n = 50
Density
0.035
0.030
0.025
0.020
0.015
0.010
0.005
0.000
200
220
240
260
280
300
x50 sample



Comments

Breeha Qasim bq08283 and Hammad Malik hm08298

Initially, for $n=1$, the probability density function (PDF) S_n exhibits a simpler form with limited peaks, characteristic of a binomial distribution with x possible outcomes. With an increment in n, there's a notable increase in the number of peaks which also begin to draw nearer to one another. Upon reaching higher values of n, the PDF S_n starts to take the form of a bell-shaped curve, synonymous with the Normal Distribution. Additionally, as n grows, the mean of the distribution shifts rightward, the multiple peaks merge into a singular peak, the peak's height diminishes, and the spread or variance grows, ensuring the total area under the curve remains one—this maintains the PDF's legitimacy. This phenomenon aligns with the Central Limit Theorem, which asserts that the aggregate of a substantial quantity of independent, uniformly distributed random variables will tend to a normal distribution, irrespective of the distribution of the individual variables.

[6 points] Part D: Poisson Random Variable

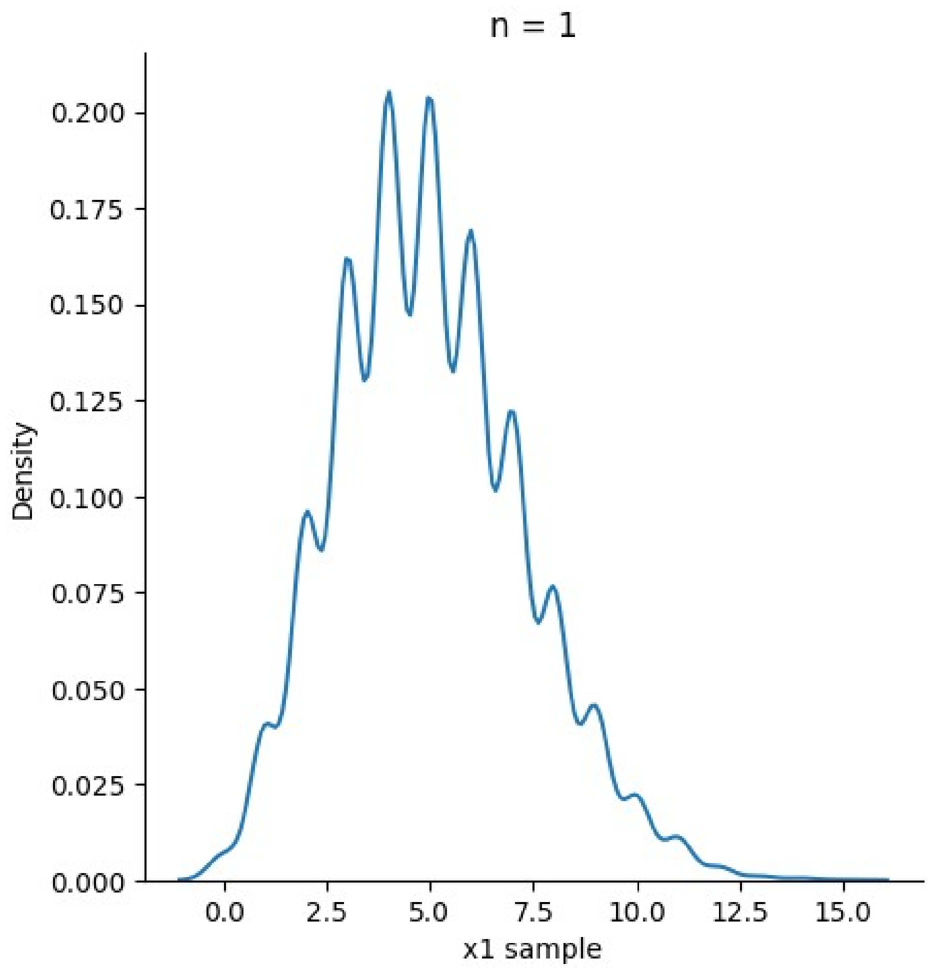
In this part, we consider X_is to be poisson random variables. The following code generates 10,000 samples of X_1 random variable and plots an approximate PDF of X_1 (based on its 10,000 samples) using the displot() function of Seaborn library.

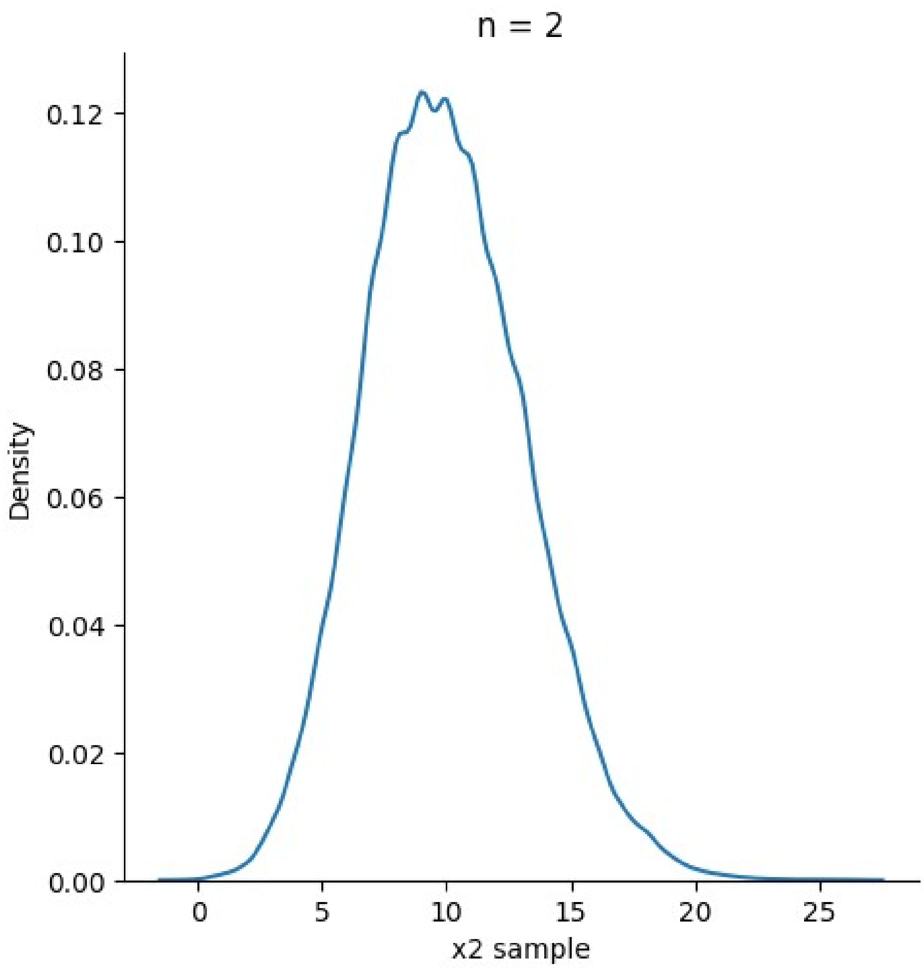
Modify the given code to generate 10,000 samples of $X_2, X_3, \cdots, X_n$, and plot the approximate PDF of S_n using the displot() function of Seaborn library for the following values of n: 1,2,3,5,10,50,100.

For example, for n=2, you have to plot the approximate PDF of $S_2 = X_1 + X_2$ and for n=3, you have to plot the approximate PDF of $S_3 = X_1 + X_2 + X_3$

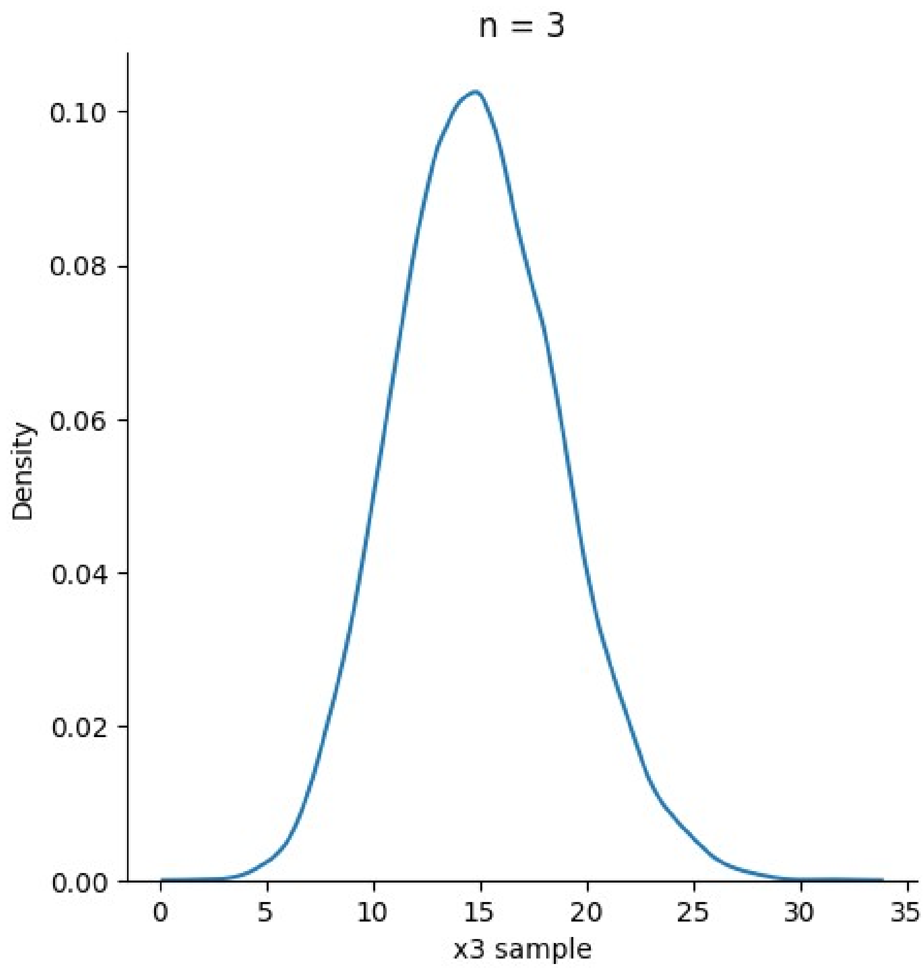
Comment on the evolution of PDF shape of S_n as n increases.

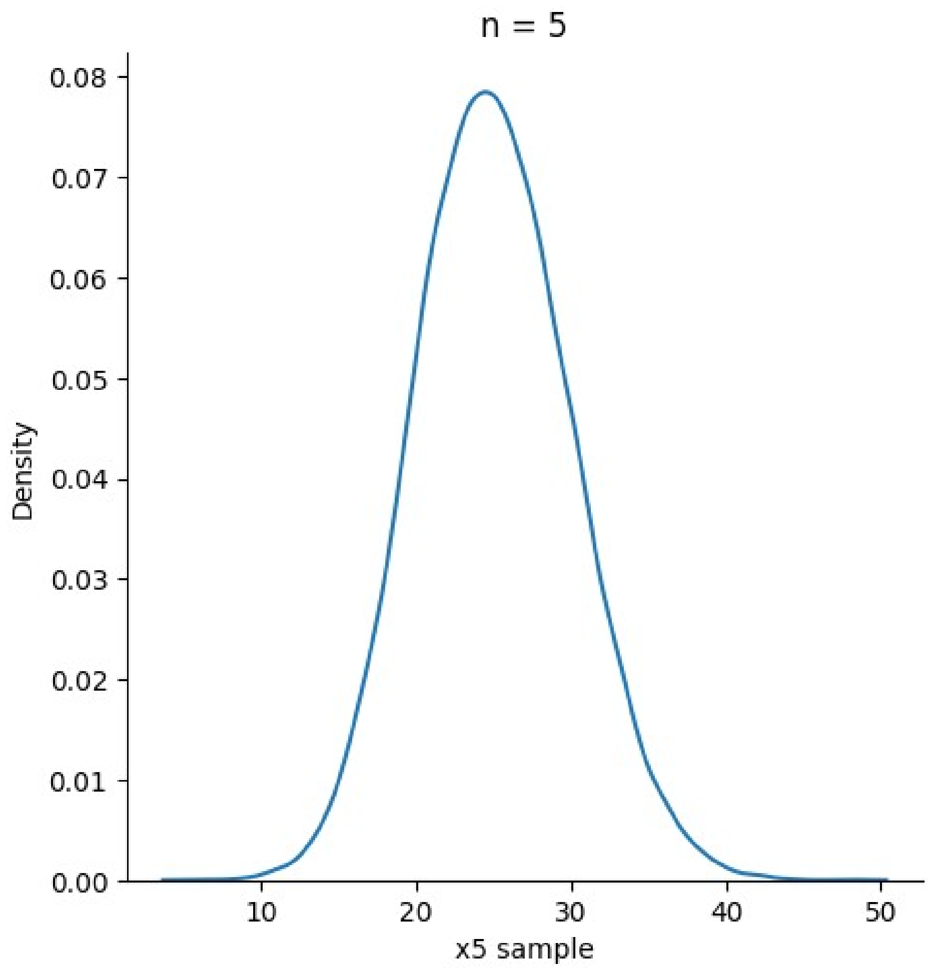
```
'''Breeha Qasim bq08283 and Hammad Malik hm08298'''

# sample_size = 10000

# df = pd.DataFrame()
# x1_sample = np.random.poisson(5,size=sample_size)

# i = 1
# col = f'x{i} sample'
# df[col] = x1_sample

# sns.displot(data=df[col], kind="kde").set(title='n =1')

sample_size = 10000
lambda_val = 5
number_samples = [1, 2, 3, 5, 10, 50, 100]

for n in number_samples:
    # Generate n samples of Poisson distributed random variables and
sum them
    samples = sum(np.random.poisson(lambda_val, size=(sample_size, n))
for i in range(n))
    df[f'x{n} sample'] = samples
    # Plot the approximate PDF of the sum
    sns.displot(df[f'x{n} sample'], kind="kde").set(title=f'n = {n}')
```

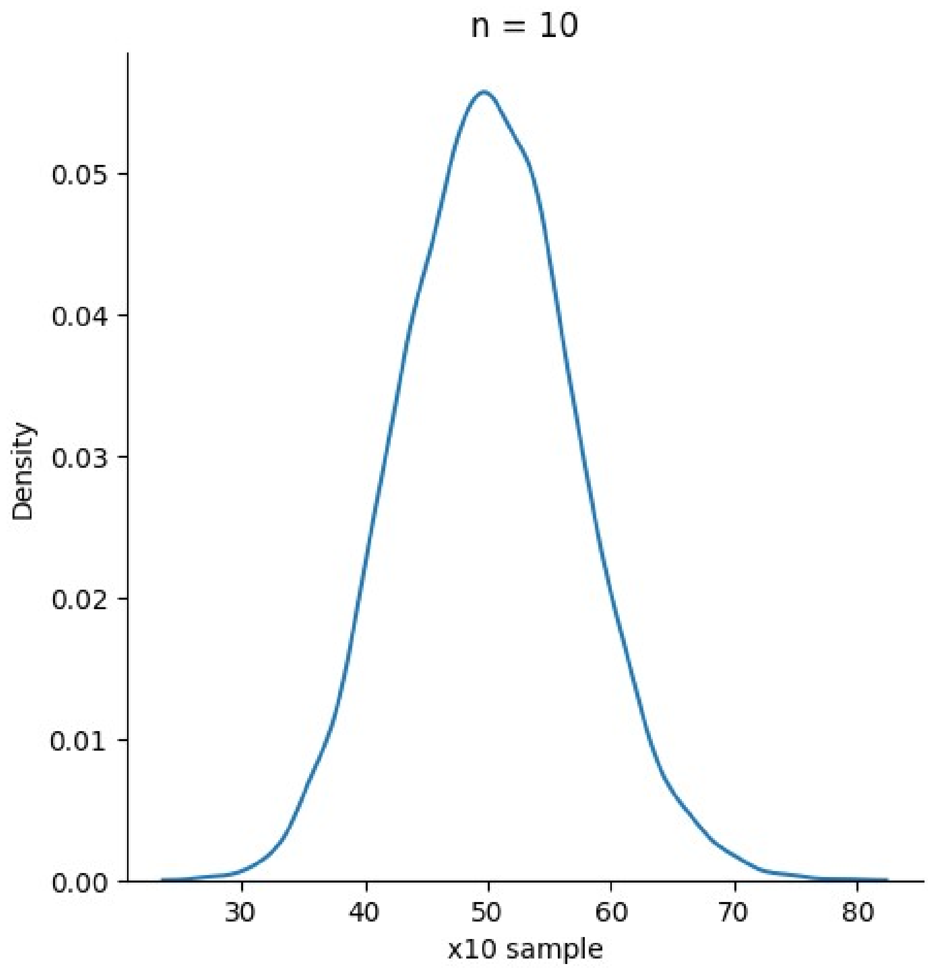

n = 1
Density
0.200
0.175
0.150
0.125
0.100
0.075
0.050
0.025
0.000
0.0
2.5
5.0
7.5
10.0
12.5
15.0
x1 sample

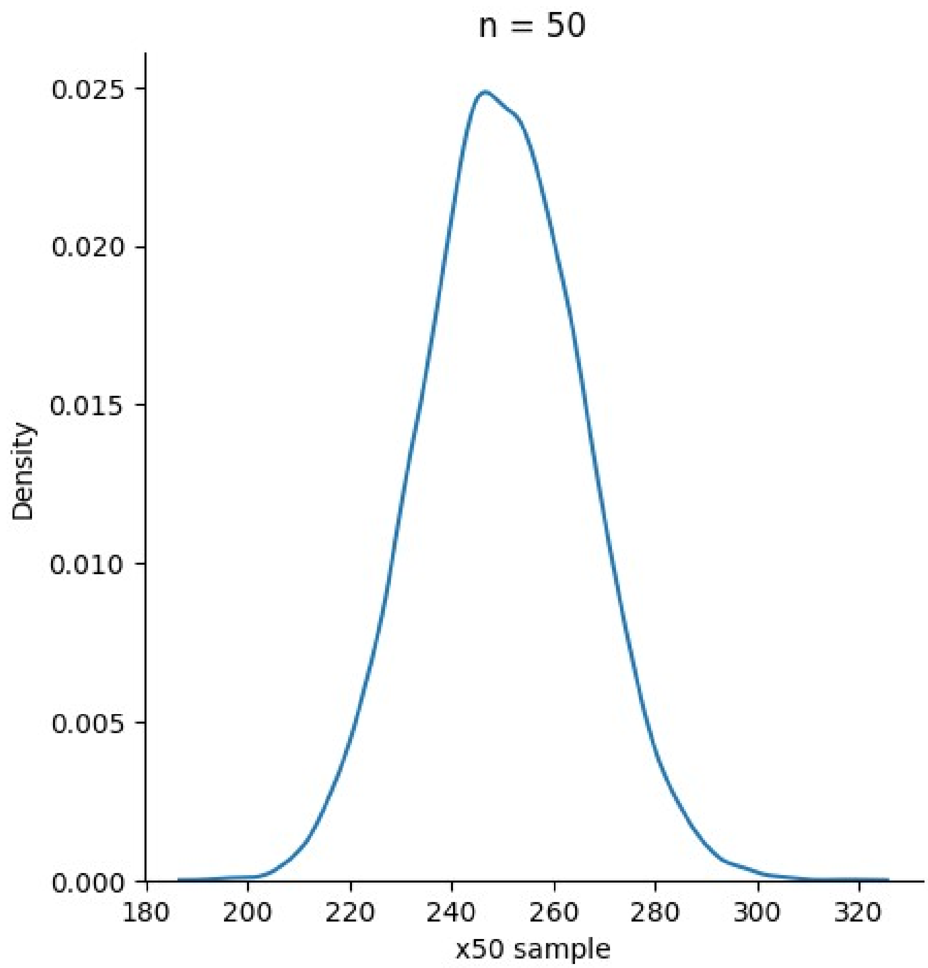


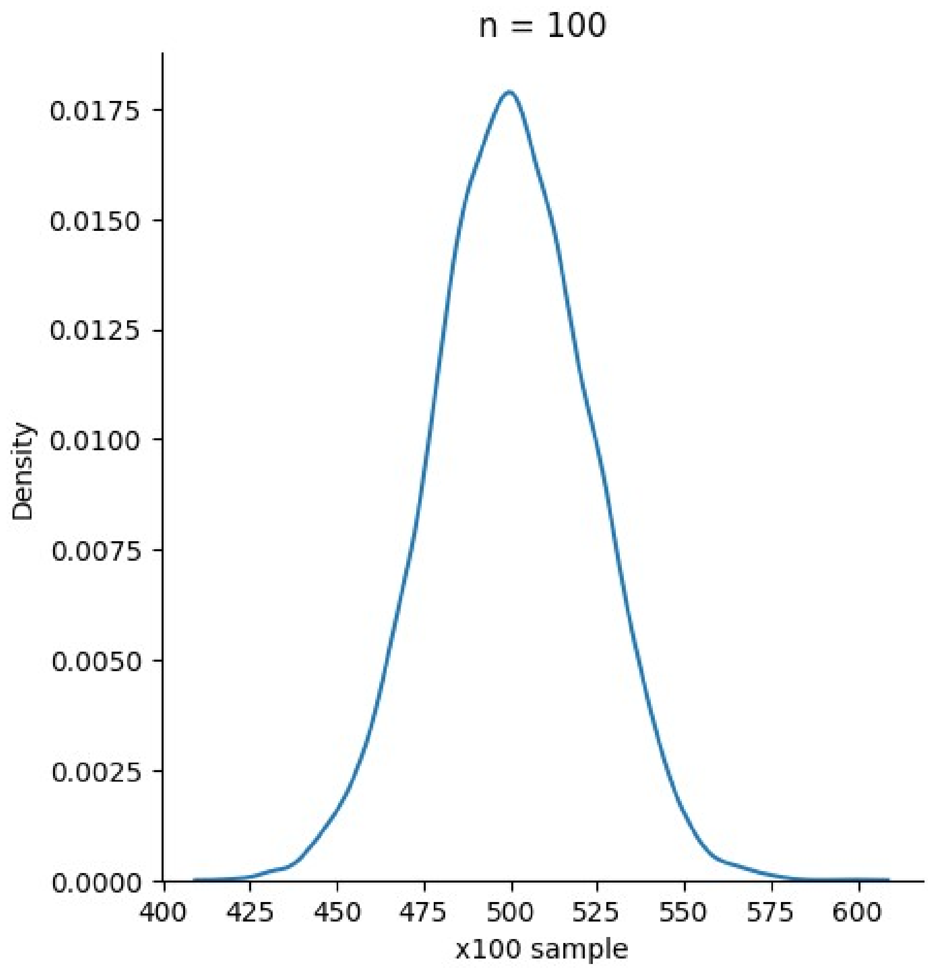
n = 2
Density
0.12
0.10
0.08
0.06
0.04
0.02
0.00
0
5
10
15
20
25
x2 sample


n = 3
Density
0.10
0.08
0.06
0.04
0.02
0.00
0
5
10
15
20
25
30
35
x3 sample


n = 5
Density
0.00
0.01
0.02
0.03
0.04
0.05
0.06
0.07
0.08
10
20
30
40
50
x5 sample


n = 10
Density
0.00
0.01
0.02
0.03
0.04
0.05
30
40
50
60
70
80
x10 sample


n = 50
Density
0.025
0.020
0.015
0.010
0.005
0.000
180
200
220
240
260
280
300
320
x50 sample



Comments

Breeha Qasim bq08283 and Hammad Malik hm08298

The code generates sums of Poisson-distributed random variables for given values of n, plotting their Kernel Density Estimates (KDEs) to approximate their Probability Density Functions (PDFs). As n increases, these sums transition from a Poisson to a more bell-shaped, normal-like distribution, illustrating the Central Limit Theorem. The increasing n leads to a smoother distribution with a rightward shift in mean and broader variance, while still maintaining the defining properties of a PDF.