Suvo_Ganguli_Assignment_2.1

January 21, 2024

1 Assignment 2.1

Name: Subhabrata (Suvo) Ganguli

Date: Jan 21, 2024

For this assignment, you will refer to the textbook to solve the practice exercises. **Use Python to answer any coding problems (not R, even if indicated in your textbook).** Use Jupyter Notebook, Google Colab, or a similar software program to complete your assignment. Submit your answers as a **PDF or HTML** file. As a best practice, always label your axes and provide titles for any graphs generated on this assignment. Round all quantitative answers to 2 decimal places.

1.1 Problem # 2.1.

For the rain simulation example in Section 2.1.1, but with probability of rain 0.30 on any given day, simulate the outcome (a) on the next day, (b) the next 10 days. (c) Simulate the proportion of days of rain for the next (i) 100 days, (ii) 10,000 days, (iii) 1,000,000 days. Use the simulation to explain the long-run relative frequency definition of probability.

(a) See code below

[118]:

```
import random
import numpy as np

p_rain = 0.3

print('True = rain, False = no rain')

# next day
next_day = random.random() < p_rain
print('Next day: ')
print(next_day)
```

```
True = rain, False = no rain
Next day:
False
```

(b) See code below.

As number of days increases, the proportion converges to the probability value set (0.3)

[119]:

```python
# next 10 days
for i in np.arange(10):
    print('Day ' + str(i+1) + ':')
    print(random.random() < p_rain)
```

```
Day 1:
False
Day 2:
False
Day 3:
False
Day 4:
True
Day 5:
True
Day 6:
False
Day 7:
True
Day 8:
False
Day 9:
False
Day 10:
False
```

(c) See code below

As the number of days increases, the proportion converges to the probability value modeled (0.3)

[120]:

```python
def prop_day(n):

    p_rain = 0.30

    val = 0
    for i in np.arange(n):
        val = val + (random.random() < p_rain)

    prop_day_out = val / n

    return prop_day_out

prop_rain_100 = prop_day(100)
prop_rain_10000 = prop_day(10000)
prop_rain_1000000 = prop_day(1000000)

print("Proportion of rainy days for the next 100 days:", prop_rain_100)
print("Proportion of rainy days for the next 10,000 days:", prop_rain_10000)
print("Proportion of rainy days for the next 1,000,000 days:",␣
 ↪prop_rain_1000000)
```

```
Proportion of rainy days for the next 100 days: 0.32
Proportion of rainy days for the next 10,000 days: 0.2941
Proportion of rainy days for the next 1,000,000 days: 0.300291
```

1.2 Problem # 2.2.

Data analysts often implement statistical inference methods by setting the probability of a correct inference equal to 0.95. Let A denote the event that an inference for the population about men is correct. Let B represent the event of a corresponding inference about women being correct. Suppose that these are independent events.

(a) Find the probability that (i) *both* inferences are correct, (ii) *neither* inference is correct.

(b) Construct the probability distribution for Y = number of correct inferences.

(c) With what probability would each inference need to be correct in order for the probability to be 0.95 that *both* are correct?

(a)

P(A) = probability of correct inference for men = 0.95

P(B) = probability of correct inference for women = 0.95

Therefore:

Probability of correct inference for both men and women = P(AB) = 0.95 x 0.95 = 0.9025

Probability that neither inference is correct = 1 - (P(A) + P(B) - P(AB)) = 1 - (0.95x2 - 0.9025) = 0.0025

Let us construct Y as follows:

Y = 0: Probability that neither inference is correct

Y = 1: Probability that one inference is correct and the other is not

Y = 2: Probability that both inferences are correct

P(Y=0) = 0.0025

P(Y=1) = 2x0.95x0.05 = 0.095

P(Y=2) = 0.9025

(c) Let probability of correct inference = p

Therefore, P(A)*P(B) = pˆ2 = 0.95

p = 0.9747

1.3 Problem # 2.4.

A wine connoisseur is asked to match five glasses of red wine with the bottles from which they came, representing five different grape types.

(a) Set up a sample space for the five guesses.

(b) With random guessing, find the probability of getting all five correct.

(a) The code is given below. There are 120 permutations.

[121]:

```python
# read in the permutations library
from itertools import permutations

# Set up sample space
wine_guesses = permutations([1,2,3,4,5])

# Find all permutations of the sample space
prob = 0
for i in wine_guesses:
    prob+=1
print('There are {} possible permutations'.format(prob))
print('The probability of getting all 5 guesses correct is {}'.format(round(1/
 ↪prob, 4)))
```

```
There are 120 possible permutations
The probability of getting all 5 guesses correct is 0.0083
```

(b) The probability that all guesses are correct is 0.0083

1.4 Problem # 2.15.

Each week an insurance company records Y = number of payments because of a home burning down. State conditions under which we would expect Y to approximately have a Poisson distribution.

The conditions under which we would expect Y to approximate a Poission distribution are:

(a) all events are independent of each other

(b) the rate of events through time is constant, and

(c) events cannot occur simultaneously.

1.5 Problem # 2.16.

Each day a hospital records the number of people who come to the emergency room for treatment.

(a) In the first week, the observations from Sunday to Saturday are 10, 8, 14, 7, 21, 44, 60. Do you think that the Poisson distribution might describe the random variability of this phenomenon adequately. Why or why not?

(b) Would you expect the Poisson distribution to better describe, or more poorly describe, the number of weekly admissions to the hospital for a rare disease? Why?

(a) To some extent, the random fluctuation of the number of persons visiting the emergency room could be described by the Poisson distribution, which is commonly used to characterize random events like dice rolls. It is impossible to predict how many individuals will end up at the emergency room because it is a random event. Therefore, it is possible that the Poisson distribution could appropriately capture the random variability of this occurrence.

When the following requirements are satisfied, the Poisson distribution is the appropriate choice: 1. The events are independent. 2. The frequency of the occurrences is unchanging. 3. Two events cannot occur at the same time.

If the events (people arriving to the emergency department) are independent, the events occur at a constant rate, and two events cannot occur simultaneously, then the Poisson distribution is an appropriate model for the number of individuals that arrive to the emergency room.

In “real life,” the requirements necessary for the Poisson distribution are not always satisfied. For example, it may be that the number of patients who visit the emergency room is not necessarily a completely independent variable. In the event that there is an accident, it is possible that a greater number of people will present themselves at the emergency room in comparison to when there is no accident.

(b) The weekly rates of admissions in the hospital due to the rare disease must be very low compared to the weekly rates of people who come to hospital for all types of disease.

Given that the admission events of rare diseases are likely independent, the events occur at a constant rate, and the rate is small, then the Poisson distribution is an appropriate model for the number of individuals that arrive to the emergency room with rare disease.

1.6 Problem # 2.17.

An instructor gives a course grade of B to students who have total score on exams and homeworks between 800 and 900, where the maximum possible is 1000. If the total scores have approximately a normal distribution with mean 830 and standard deviation 50, about what proportion of the students receive a B?

Based on the code below, the z-scores for 800 and 900 are -0.6 and 1.4.

Using the z-score table:

Percentage corresponding to -0.6 = 1 - 0.7257 = 0.2743

Percentage corresponding to 1.4 = 0.9192

Therefore, proportion of students receiving a B = 0.9192 - 0.2743 = 0.6449

[122]:

```python
import scipy.stats as sp

z1 = (800 - 830)/50
z2 = (900 - 830)/50

print(z1,z2)
```

```
-0.6 1.4
```

1.7 Problem # 2.20.

Create a data file with the income values in the `Income` data file at the text website.

(a) Construct a histogram or a smooth-curve approximation for the *pdf* of income in the corresponding population by plotting results using the density function in R (explained in Exercise 1.18).

(b) Of the probability distributions studied in this chapter, which do you think might be most appropriate for these data? Why? Plot the probability function of that distribution having the same mean and standard deviation as the income values. Does it seem to describe the income distribution well?

(a) The histogram is plotted below.

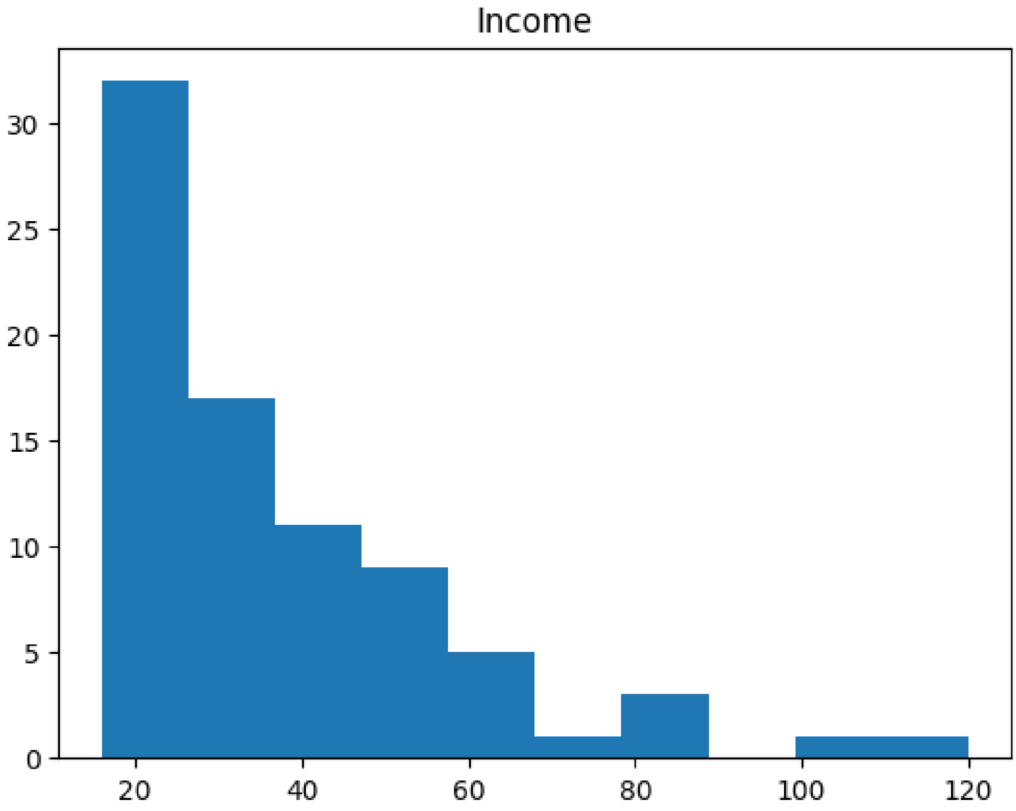
[123]:

```python
# Include library
import matplotlib.pyplot as plt
import pandas as pd

# Read in the data file
data = pd.read_csv('Income.dat', sep='\s+')

# See the data
print(data.head())

# Plot
fig, ax = plt.subplots(1,1)
ax.hist(data = data, x = 'income')
ax.set_title('Income')
```

```
   income  education race
0      16         10    B
1      18          7    B
2      26          9    B
3      16         11    B
4      34         14    B
```

[123]:

```
Text(0.5, 1.0, 'Income')
```

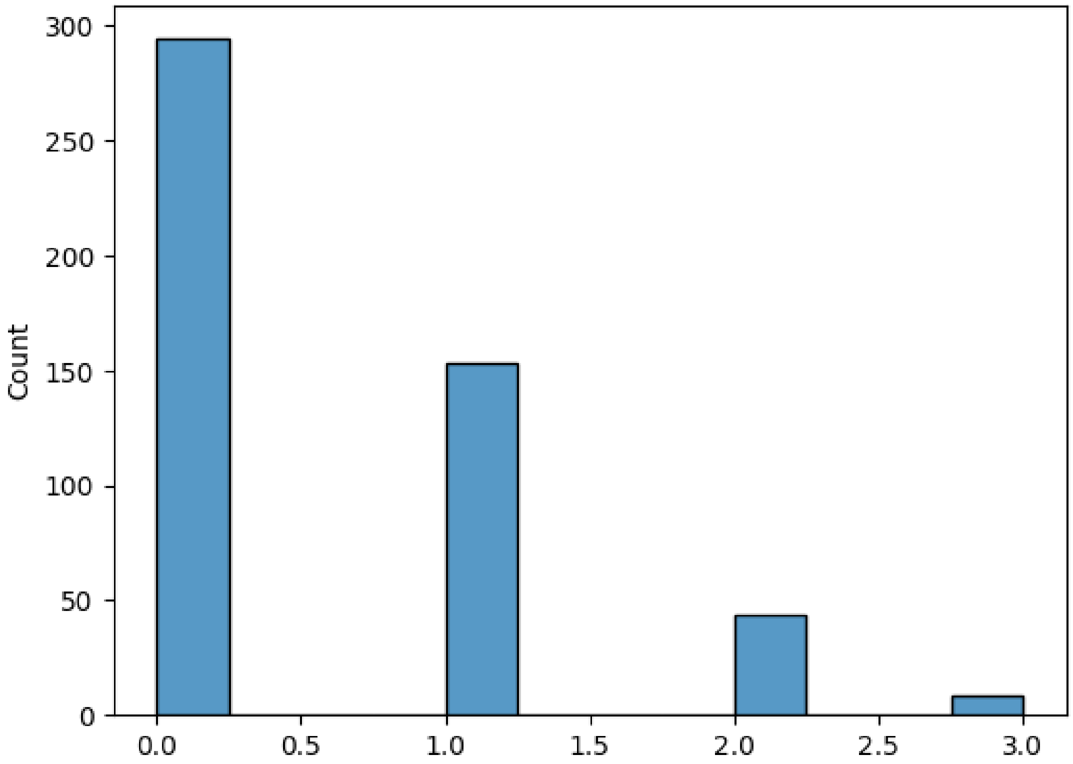


(b) Based on the shape of the histogram, a good representation of the data will using a Poisson distribution. This is because the data is skewed towards the right.

We can verify the above by plotting a Poisson distribution with lambda = 0.5 and comparing with the histogram plotted earlier.

[128]:
```
# import library
from numpy import random
import seaborn as sns

# plot poission distribution histrogram
sns.histplot(random.poisson(lam=0.5, size=500), kde=False)
```

[128]:
```
<Axes: ylabel='Count'>
```



1.8 Problem # 2.21.

Plot the gamma distribution by fixing the shape parameter $k = 3$ and setting the scale parameter $= 0.5, 1, 2, 3, 4, 5$. What is the effect of increasing the scale parameter? (See also Exercise 2.48.)

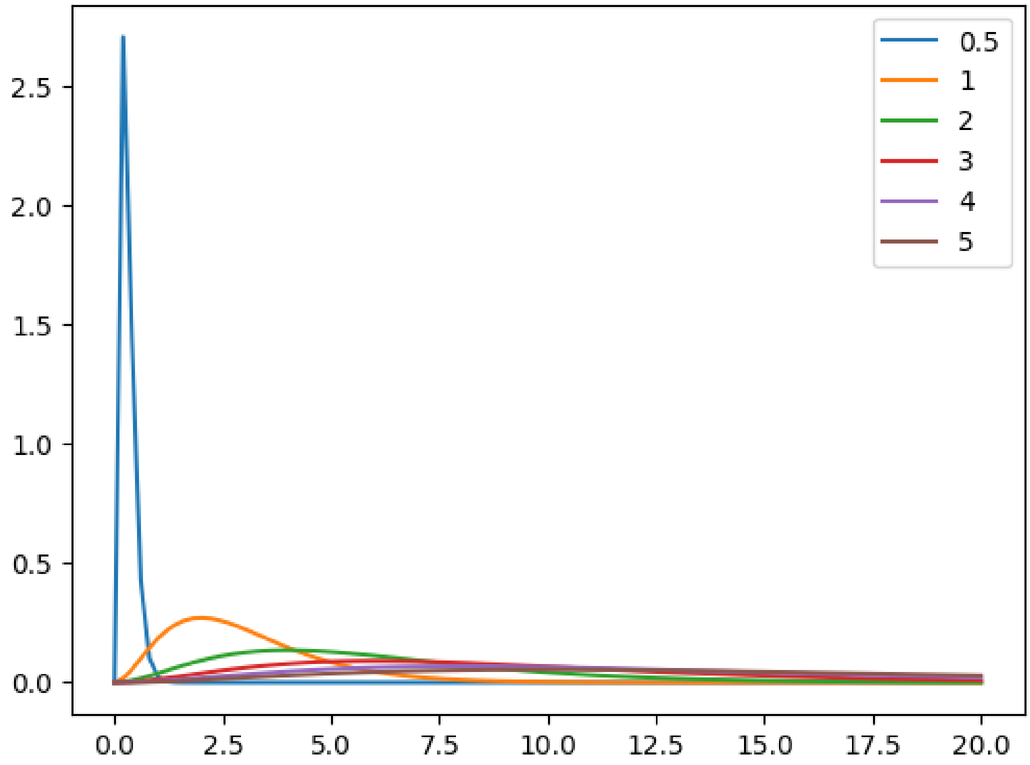
See code and output below.

As the scale parameter increases, the curve flattens and peak moves towards the right.

[125]:

```
#define x-axis values
x = np.linspace (0, 20, 100)

k = 3
scl = [0.1, 1, 2, 3, 4, 5]

#calculate pdf of Gamma distribution for each x-value
for i in np.arange(len(scl)):
    y = sp.gamma.pdf(x, a=k, scale=scl[i])

    #create plot of Gamma distribution
    plt.plot(x, y)

plt.legend(['0.5','1','2','3','4','5'])
```

```
#display plot
plt.show()
```



1.9 Problem # 2.22.

Consider the mammogram diagnostic example in Section 2.1.4.

(a) Show that the joint probability distribution of diagnosis and disease status is as shown in Table 2.6. Given that a diagnostic test result is positive, explain how this joint distribution shows that the 12% of incorrect diagnoses for the 99% of women not having breast cancer swamp the 86% of correct diagnoses for the 1% of women actually having breast cancer.

(b) The first test for detecting HIV-positive status had a sensitivity of 0.999 and specificity of 0.9999. Explain what these mean. If at that time 1 in 10,000 men were truly HIVpositive, find the positive predictive value. Based on this example, explain the potential disadvantage of routine diagnostic screening of a population for a rare disease.

TABLE 2.6 Joint probability distribution for disease status and diagnosis of breast cancer mammogram, based on conditional probabilities in Table 2.1

Disease Status	**Diagnosis from Mammogram**		
	Positive (+)	Negative (-)	**Total**
Yes (D)	0.0086	0.0014	0.01
No (D^c)	0.1188	0.8712	0.99

(a) Define the events:

D = An event that women is having breast cancer

Dc = An even that women is not having breast cancer

P = Positive test

N = Negative test

The probability of incorrect diagnoses given that a women not having breast cancer is

p(P/Dc) = p(P and Dc)/p(Dc) = 0.1188/0.99 = 0.12 –> 12%

The probability of correct diagnoses given that a women actually having breast cancer is

p(p/D) = p(P and D)/p(D) = 0.0086/0.01 = 0.86 –> 86%

(b) Define the events

D = An event that a person has HIV

Dc = An even that a person does not have HIV

P = Positive test

N = Negative test

The sensitivity of the test is 0.999, means that the probability that a person's test is positive given that the person is having HIV is 0.999. This means

p(P/D) = 0.999 p(N/D) = 1 - 0.999 = 0.001

The specificity of the test is 0.9999 means that the probability that a person's test is negative given that the person is not having HIV is 0.9999. This means

p(P/Dc) = 0.9999 p(N/Dc) = 0.0001

Given p(D) = 1/10000 = 0.0001 p(N) = 1-0.0001 = 0.9999

The positive productive value indicates the probability that a person actually have HIV given that the test is test is positive. In the given terms we have to find p(D/P)

p(D/P) = p(p/D)xp(D) / (p(P/D)xp(D) + p(P/Dc)xp(Dc))

= 0.999x0.0001 / (0.999x0.0001 + 0.0001x0.9999) = 0.4997

Since, p(P/Dc)=1-0.999=0.0001, the disadvantage of the routine test is that there is 0.01% chance that the routine test is positive even the person is not having HIV.

1.10 Problem # 2.27.

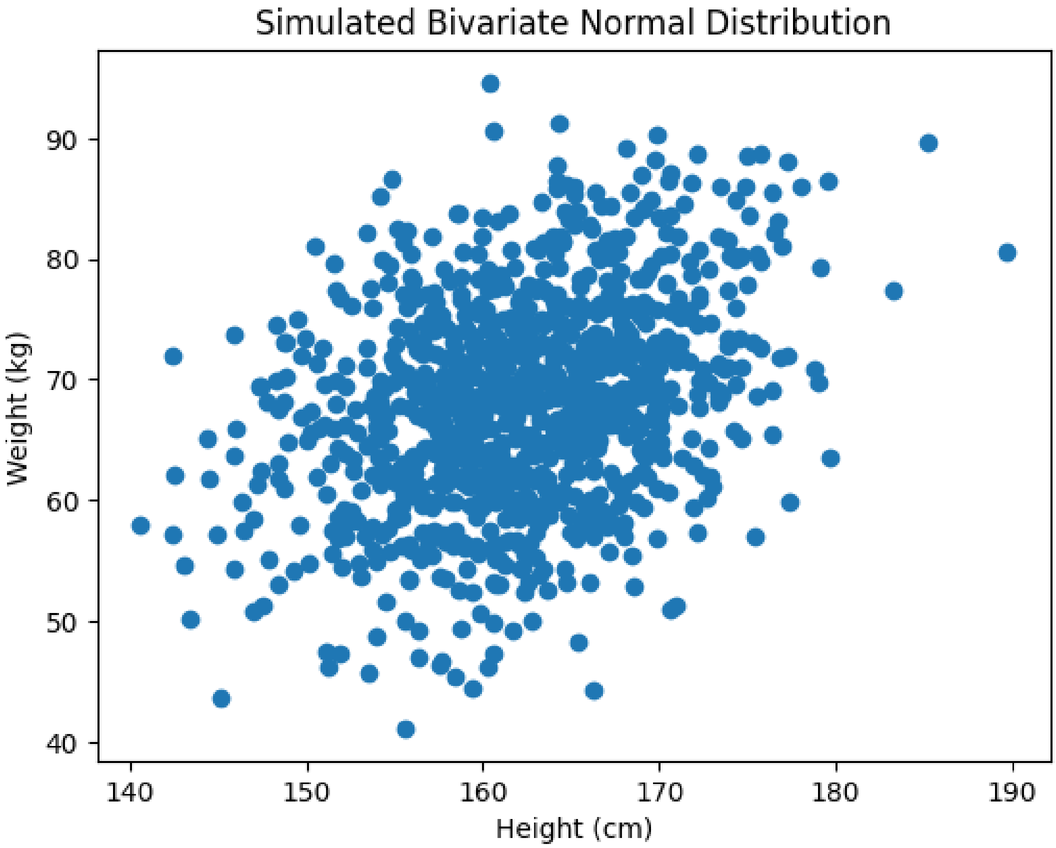
The distribution of X = heights *(cm)* of women in the U.K. is approximately $N(162, 7^2)$. Conditional on $X = x$, suppose Y = weight *(kg)* has a $N(3.0 + 0.40\text{x}, 8^2)$ distribution. Simulate and plot 1000 observations from this approximate bivariate normal distribution. Approximate the marginal means and standard deviations for X and Y. Approximate and interpret the correlation.

See code and output below.

A positive correlation indicates that taller women tend to have higher weights. The value 0.33 means there is relatively medium correlation between the height and the weight of the women.

[126]:

```
# Parameters for X distribution
mean_X = 162
std_X = 7

# Parameters for Y distribution conditional on X
mean_Y = lambda x: 3.0 + 0.40 * x
std_Y = 8

# Simulate 1000 observations
np.random.seed(1)
X = np.random.normal(mean_X, std_X, 1000)
Y = np.random.normal(mean_Y(X), std_Y)

# Plot the observations
plt.scatter(X, Y)
plt.xlabel('Height (cm)')
plt.ylabel('Weight (kg)')
plt.title('Simulated Bivariate Normal Distribution')
plt.show()

# Approximate marginal means and standard deviations
mean_X_approx = np.mean(X)
std_X_approx = np.std(X)
mean_Y_approx = np.mean(Y)
std_Y_approx = np.std(Y)

print('Mean X approx = ' + str(mean_X_approx))
print('Std X approx = ' + str(std_X_approx))
print('Mean Y approx = ' + str(mean_Y_approx))
print('Std Y approx = ' + str(std_Y_approx))

# Approximate correlation
corr = np.corrcoef(X, Y)[0, 1]

print('Correlation coef = ' + str(corr))
```



```
Mean X approx = 162.2716873331172
Std X approx = 6.867028937525482
Mean Y approx = 68.1272784804035
Std Y approx = 8.742822939380673
Correlation coef = 0.3347796720017178
```