Assignment3

February 22, 2021

1 Assignment 3

Enter Your Name Here

1.1 Instructions

1. Download the jupyter notebook file (this should have a.ipynb extension) as an HTML file (this will have a .html extension) by clicking the **Files** tab and selecting "download as HTML (.html)";
2. Upload both the jupter notebook file (**Assignment3.ipynb**) and the HTML file (**Assignment3.html**) to Canvas page.
3. The files you upload to the Canvas page should be updated with commands you provide to answer each of the questions below. You can edit this file directly to produce your final solutions.

1.2 Background: The Normal Distribution

Recall from your probability class that a random variable X is normally-distributed with mean μ and variance σ^2 (denoted $X \sim N(\mu, \sigma^2)$) if it has a probability density function, or *pdf*, equal to

$$f(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}, \quad -\infty < x < \infty.$$

In *Python* we can simulate $N(\mu, \sigma^2)$ random variables using the `numpy.random.normal()` function. For example,

[2]:

```
import numpy as np
np.random.seed(1) # Please don't remove this code!
# seed function is used to save the state of a random function,
# so that it can generate same random numbers on multiple executions of the␣
 ↪code
# on the same machine or on different machines (for a specific seed value).
# The seed value is the previous value number generated by the generator.
# For the first time when there is no previous value, it uses current system␣
 ↪time.

np.random.normal(loc = 10, scale = 3, size = 5)
```

[2]:
```
array([14.87303609,  8.16473076,  8.41548474,  6.78109413, 12.59622289])
```

outputs 5 normally-distributed random variables with mean equal to 10 and standard deviation (this is σ) equal to 3.

1.3 Tasks

1.3.1 Sample means as sample size increases

1. Generate 100 random draws rom the standard normal distribution and save them in an array named **normal100**. Calculate the mean and standard deviation of **normal100**. In words explain why these values aren't exactly equal to 0 and 1.

[1]:
```
# You'll want to type your response here. Your response should look like:
# normal100 =
# Of course, your answer should not be commented out.
```

The mean and standard deviation aren't exactly equal to 0 and 1 because they are calculated using from random samples.

2. The function `matplotlib.pyplot.hist()` is a base *Python* graphing function that plots a histogram of its input. Use `matplotlib.pyplot.hist()` with your array of standard normal random variables from question 1 to produce a histogram of the standard normal distribution.

   The Jupyter Notebook has two ways to get help. Place the cursor inside the parenthesis of the function, hold down shift , and press tab, or type a function name with a question mark after it.

   If coded properly, these plots will be automatically embedded in your output file.

[ ]:
```

```

3. Repeat question 1 except change the number of draws to be 10, 1000, 10,000, and 100,000 and store the results in arrays called **normal10**, **normal1000**, **normal10000**, **normal100000**.

[ ]:
```

```

4. We want to compare the means of our four random draws. Create a list called **sample_means** that has as its first element the mean of **normal10**, its second element the mean of **normal100**, its third element the mean of **normal1000**, its fourth element the mean of **normal10000**, and its fifth element the mean of **normal100000**. After you have created the **sample_means** vector, print the contents of the vector and use the `len()` function to find the length of this list. (it should be five). There are, of course, multiple ways to create this list. Finally, explain in words the pattern we are seeing with the means in the **sample_means** list.

[ ]:
```

```

As the sample size increases, the mean of the sample is becoming closer and closer to 0, which is the mean of the random variables making up the sample. As made explicit in question 10 below, this is because the sample mean is distributed as $N(0, 1/n)$, where n is th sample size.

1.4 Sample distribution of the sample mean

5. Let's push this a little farther. Generate 1 million random draws from a normal distribution with $\mu = 3$ and $\sigma^2 = 4$ and save them in an array named **normal1mil**. Calculate the mean and standard deviation of **normal1mil**.

[ ]:

```
```

6. Find the mean of all the entries in **normal1mil** that are greater than 3. You may want to generate a new vector first which identifies the elements that fit the criteria.

[ ]:

```
```

7. Create an 2D array **normal1mil_mat** from the 1D array **normal1mil** that has 10,000 columns (and therefore should have 100 rows).

[ ]:

```
```

8. Calculate the mean of the 1234^{th} column.

[ ]:

```
```

9. Use the `numpy.mean()` functions to calculate the *means* of each column of **normal1mil_mat**. Remember, **numpy.mean?** will give you help documents about this function. Save the array of column means with an appropriate name as it will be used in the next task.

[ ]:

```
```

10. Finally, produce a histogram of the column means you calculated in task 9. What is the distribution that this histogram approximates (i.e. what is the distribution of the sample mean in this case)?

[ ]:

```
```