STAT 207 Homework 6 [50 points] - Solutions

Random Variables and Defined Distributions

Due: Friday, March 3 by noon (11:59 am) CST

Package Imports

Run the cell provided below to import packages needed for this assignment.

You may also need to read in additional packages below.

In [1]:
```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import numpy as np
from scipy.stats import geom
from scipy.stats import norm
from scipy.stats import expon
```

Case Study 1: Slot Machine Gambling [7 points]

Suppose a gambler has a strategy to keep playing a slot matchine until they win a round. After the gambler **wins** a round, they stop playing for the day. The probability of **winning** any given slot machine round is 0.02.

a) On average, how many rounds of the slot machine will the gambler play before stopping for the day?

In [2]:
```
geom.mean(0.02)
```

Out[2]:
```
50.0
```

b) Suppose that it costs $\$1$ to play each round of the slot machine. The gambler always brings $\$100$ with them to play the slot machine each day. If they run out of this $\$100$ (i.e. they play the game more than 100 times), then they go to borrow money from their friend and keep playing the slot machine until they win. What percent of days does does the gambler need to borrow money from this friend?

In [3]:
```
(1 - geom.cdf(100, 0.02)) * 100
```

Out[3]:
```
13.261955589475317
```

Case Study 2: SAT Score Scholarship Analysis [13 points]

Suppose the distribution of SAT scores for the seniors at a local high school (just math and verbal) has an average of 1000, a standard deviation of 100, and follows a normal distribution.

a) What is the probability that randomly selected senior from the high school scored more than 1250?

In [4]:
```
mean = 1000
sd = 100
1 - norm.cdf(1250, loc = mean, scale = sd)
```

Out[4]:
```
0.006209665325776159
```

b) What percent of seniors at the school score between a 900 and 950?

In [5]:
```
norm.cdf(950, loc = mean, scale = sd) - norm.cdf(900, loc = mean, scale = sd)
```

Out[5]:
```
0.1498822847945298
```

c) Suppose that all seniors at the school who score higher than a 1200 get a scholarship from the county. Suppose that we randomly select a student that we KNOW got the scholarship. GIVEN that we know that the student has the scholarship, what is the probability that this student's SAT score was less than 1300?

Hint: You might want to consider some of our basic probability rules.

In [6]:
```
given_prop = 1 - norm.cdf(1200, loc = mean, scale = sd) # prop'n who get scholarship
and_prop = norm.cdf (1300, loc = mean, scale = sd) - norm.cdf(1200, loc = mean, scale = sc
    # prop'n who get scholarship AND score less than 1300, i.e. between 1200 and 1300
and_prop / given_prop
```

Out[6]:
```
0.9406641669285728
```

Case Study 3: Farm Workers Salaries [5 points]

Suppose the standard deviation of the hourly wage of a farm worker in Illinois is 4 dollars per hour and the distribution of hourly farm worker wages follows the normal distribution. Suppose that we know that 25% of farm workers in Illinois make at least 15 dollars per hour.

What is the average hourly wage of farm workers in Illinois?

In [7]:
```
z = norm.ppf(0.75)
15 - z * 4
```

Out[7]:
```
12.302040999215674
```

If we know that 25% of farm workers make at least 15 dollars per hour, then that means that the 75th percentile is 15. First, we can find the z-score that corresponds to the 75th percentile. Then, we can use the other information (x and sd) to solve for the population mean from the z-score formula ($z = \frac{x-\mu}{\sigma}$).

Finally, we can confirm that this is a reasonable value based on the Empirical Rule. We expect 15 to be within 1 standard deviation of the mean, since it is between the 50th and 84th percentiles. This is true for the value that we have calculated.

Case Study 4: Wheelchair Basketball [15 points]

Suppose that you are a manager for a wheelchair basketball team with 12 players on the roster. The length of time that an athletic wheelchair can function before needing to be serviced follows an exponential distribution (accessible within Python through the `expon` set of methods from `scipy.stats`). For this exponential distribution, you should set the scale parameter to 7 weeks.

a) Using Python, generate a set of 12 randomly selected draws from the exponential distribution defined above. You can think of each draw as the amount of time before a wheelchair needs to be serviced for a single player on the roster.

Hint: I recommend using a random_state here, so that you can recreate your draws, if needed.

In [8]:
```
chair_times = expon.rvs(scale = 7, size = 12, random_state = 38)
chair_times
```

Out[8]:
```
array([ 3.40034917, 13.74819754, 20.20182418,  8.49403611,  7.02885897,
        6.51914274,  1.56311482,  3.43016319,  2.09812353,  0.5427595 ,
        2.30883858,  4.03918367])
```

b) Calculate the minimum, mean, median, standard deviation, and estimated probability of a wheelchair not needing to be serviced for the 16 weeks of the semester based on this particular set of draws from the exponential distribution, i.e. the time until a wheelchair needs to be serviced for each of the 12 players on the roster.

Hint: You may want to use the tools demonstrated in the tutorial below to help perform these calculations.

Tutorial for working with arrays.

We don't always have objects in Python that allow us to use our typical methods for calculating summary values, including means, medians, and standard deviations. If you find that your first attempt at a calculation doesn't work, try adjusting your code to the following format.

In [9]:
```
x = np.array([1, 2, 3, 4, 5])
# x.median() # doesn't work -- try it out my removing the first "#" to see the error messa
np.median(x)
```

Out[9]:
```
3.0
```

In [10]:
```
# minimum
np.min(chair_times)
```

Out[10]:
```
0.5427594965995712
```

In [11]:
```
# mean
np.mean(chair_times)
```

Out[11]:
```
6.114549334062411
```

In [12]:
```
# median
np.median(chair_times)
```

Out[12]:
```
3.7346734296682342
```

In [13]:
```
# std
```

```
np.std(chair_times)
```

Out[13]:
```
5.512417615935697
```

In [14]:
```
# estimated probability
np.mean(chair_times >= 16)
```

Out[14]:
```
0.08333333333333333
```

Note that all of these could also be calculated by creating a data frame and using many of the functions that we have regularly used so far this semester.

c) Based on your random sample, what is the first time that a player would need their wheelchair serviced?

How many players would need their wheelchair serviced during the semester?

Based on my random sample, the first player would need their wheelchair serviced within 0.54 weeks (~3-4 days). 11 (all except 1) players would need their wheelchair serviced before the semester ends.

d) For the theoretical exponential distribution with a scale of 7, calculate and report the mean, median, standard deviation, and probability of a wheelchair not needing to be serviced for the 16 weeks of the semester.

Hint: The exponential distribution is a continuous random variable and has many of the same functions as other continuous random variables we have discussed in class. You can also find documentation for this distribution online.

In [15]:
```
expon.mean(scale = 7)
```

Out[15]:
```
7.0
```

In [16]:
```
expon.median(scale = 7)
```

Out[16]:
```
4.852030263919617
```

In [17]:
```
expon.std(scale = 7)
```

Out[17]:
```
7.0
```

In [18]:
```
1 - expon.cdf(16, scale = 7)
```

Out[18]:
```
0.10170139230422681
```

e) How close are the mean, median, standard deviation, and probability values between **4c** and **4d**?

Most of the values estimated from my random sample of 12 are pretty close to the population values from **4d**. The means are within 1, the observed median is a little more than 1 less than theoretical, the observed standard deviation is about 1.5 less than theoretical, and the probability values are similar, with about 2% difference.

Note that I only have a random sample of 12, so variability is reasonable.

Case Study 5: Central Limit Theorem [10 points]

Consider the population of counties provided in the county.csv file. We will examine the **per_capita_income** variable from this data.

a) Read in the data.

Note that you will need to clean the data before you can perform your calculations. The phrase 'data unavailable' represents missing data in this csv file. You may assume that any counties with missing data are not included in our population for this calculation.

In [19]:
```
df = pd.read_csv('county.csv', na_values = ['data unavailable'])
df = df.dropna()
df.shape
```

Out[19]:
```
(2559, 15)
```

In [20]:
```
## OR, just drop missing values for my variable of interest:

df1 = pd.read_csv('county.csv', na_values = ['data unavailable'])
df1 = df1['per_capita_income'].dropna()
df1.shape
```

Out[20]:
```
(3139,)
```

b) From the information about the population of all counties in the US, calculate the *theoretical* standard error of the sampling distribution for the sample mean (of 50 counties) per capita income (**per_capita_income**).

In [21]:
```
df['per_capita_income'].std() / np.sqrt(50)
```

Out[21]:
```
878.0516938109902
```

In [22]:
```
## OR
df1.std() / (50 ** (1/2))
```

Out[22]:
```
888.2772461200067
```

c) Are the conditions for the Central Limit Theorem met? That is, will the Central Limit Theorem apply to the *theoretical* standard error of the sampling distribution for the sample mean (of 50 counties) per capita income?

Note: You may make reasonable assumptions when needed.

The conditions for the Central Limit Theorem are:

- Population is Normally distributed and/or the sample size is at least 30
- The sample size is less than 10% of the population size
- We have a random sample of counties

Are these conditions met?

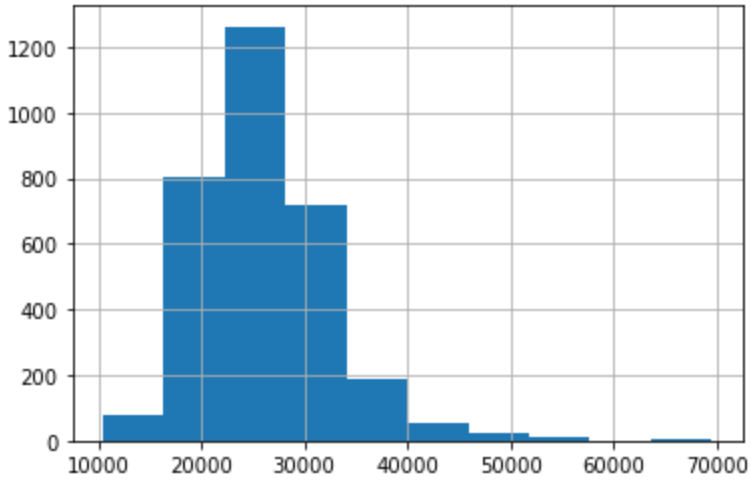
- Our sample size is at least 30. Looking at the histogram below, the population distribution is right skewed, so Normality of the population is not reasonable. However, we can use the large sample size to

bypass the Normal population requirement.

- The sample size (50) is less than 10% of the population size (~3000). We would need a population of at least 500 counties, and we have many more.
- We can assume that our sample is a random sample of counties.

In [23]:
```
df1.hist()
```

Out[23]:
```
<AxesSubplot:>
```



Remember to keep all your cells and hit the save icon above periodically to checkpoint (save) your results on your local computer. Once you are satisified with your results restart the kernel and run all (Kernel -> Restart & Run All). **Make sure nothing has changed**. Checkpoint and exit (File -> Save and Checkpoint + File -> Close and Halt). Follow the instructions on the Homework 4 Canvas Assignment to submit your notebook to GitHub.