Central Limit Theorem — Explained with Examples

Clearly explained with implementation

Soner Yıldırım May 18, 2020 · 5 min read ★

Photo by Luís Perdigão on Unsplash

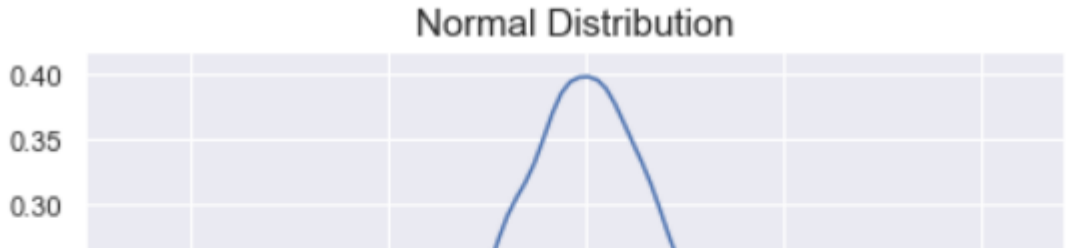
The central limit theorem (CLT) is a fundamental and widely used theorem in the field of statistics. Before we go in detail on CLT, let's define some terms that will make it easier to comprehend the idea behind CLT.

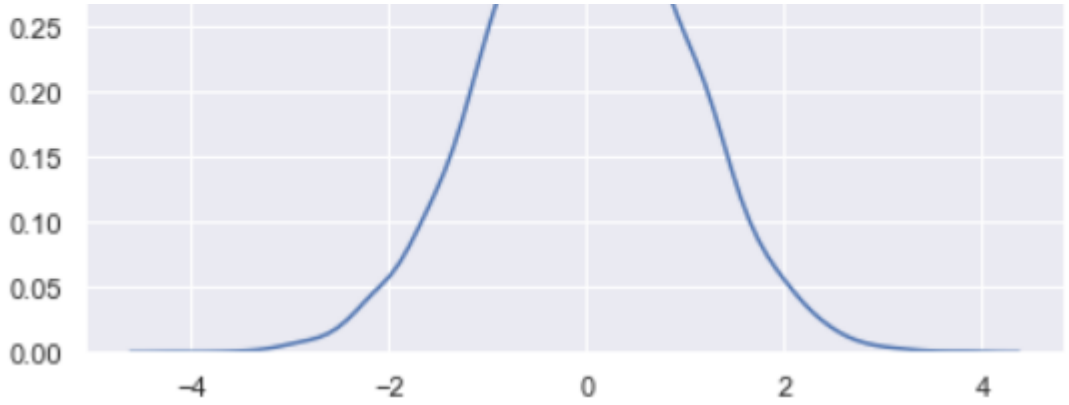
Basic concepts

- **Population** is all elements in a group. For example, college students in US is a population that includes all of the college students in US. 25-year-old people in Europe is a population that includes all of the people that fits the description.

It is not always feasible or possible to do analysis on population because we cannot collect all the data of a population. Therefore, we use samples.

- **Sample** is a subset of a population. For example, 1000 college students in US is a subset of "college students in US" population.
- **Probability distribution**: A function that shows the probabilities of the outcomes of an event or experiment. Consider rolling a dice example. There are 6 possible outcomes (1,2,3,4,5,6). If the dice in unbiased, the probability of observing each number on top side is equal so the probability distibution is a discrete uniform distribution.
- **Normal (Gaussian) distribution**: A probability distribution that looks like a bell:





Two terms that describe a normal distribution are **mean** and **standard deviation**. Mean is the average value that has the highest probability to be observed. **Standard deviation** is a measure of how spread out the values are. As standard deviation increases, the normal distribution curve gets wider.

Central Limit Theorem

Normal distribution is used to represent random variables with unknown distributions. Thus, it is widely used in many fields including natural and social sciences. The reason to justify why it can used to represent random variables with unknown distributions is the **central limit theorem (CLT)**.

According to the **CLT**, as we take more samples from a distribution, the sample averages will tend towards a **normal distribution** regardless of the

population distribution.

Consider a case that we need to learn the distribution of the heights of all 20-year-old people in a country. It is almost impossible and, of course not practical, to collect this data. So, we take samples of 20-year-old people across the country and calculate the average height of the people in samples. According to the CLT, as we take more samples from the population, sampling distribution will get close to a normal distribution.

Why is it so important to have a normal distribution? Normal distribution is described in terms of mean and standard deviation which can easily be calculated. And, if we know the mean and standard deviation of a normal distribution, we can compute pretty much everything about it.

Examples to prove CLT

Let's go over a few examples and prove that CLT is true. We will use python libraries to create populations, samples, and plots. As always, we start with importing related libraries:

```
import numpy as np
import pandas as pd
```

```
#Data visualization
import matplotlib.pyplot as plt
import seaborn as sns
sns.set(style='darkgrid')


%matplotlib inline
```

We first define a function that will create random samples from a distribution. We can use **sample** function of pandas that will select random elements without replacement.

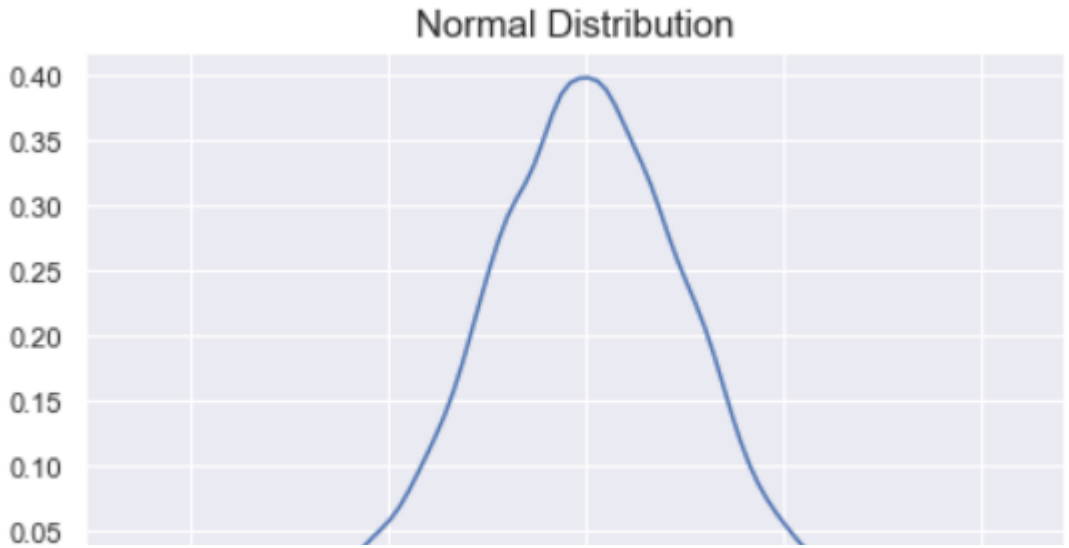
```
def random_samples(population, sample_qty, sample_size):
    sample_means = []
    for i in range(sample_qty):
        sample = population.sample(n=sample_size)
        sample_mean = np.array(sample).mean()
        sample_means.append(sample_mean)
    return sample_means
```


```
def random_samples(population, sample_qty, sample_size):
    sample_means = []
    for i in range(sample_qty):
        sample = population.sample(n=sample_size)
        sample_mean = np.array(sample).mean()
        sample_means.append(sample_mean)
    return sample_means
```

We just need to input a population, how many samples we need (sample_qty), and the how many observations each sample includes (sample_size). Then the function will pick samples and calculate their means. The returned list will include the sample means.

Let's first define a population that actually has a **normal distribution**. We use np.random.randn function to create an array with a size of 10000 and a normal distribution.

```
norm = list(np.random.randn(10000))
plt.figure(figsize=(8,5))
plt.title("Normal Distribution", fontsize=18)
sns.distplot(norm, hist=False)
```

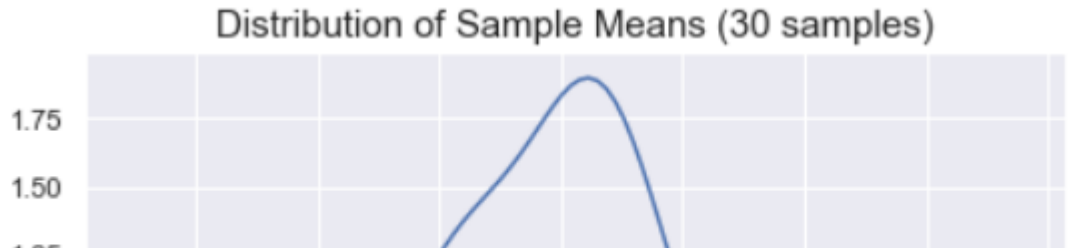


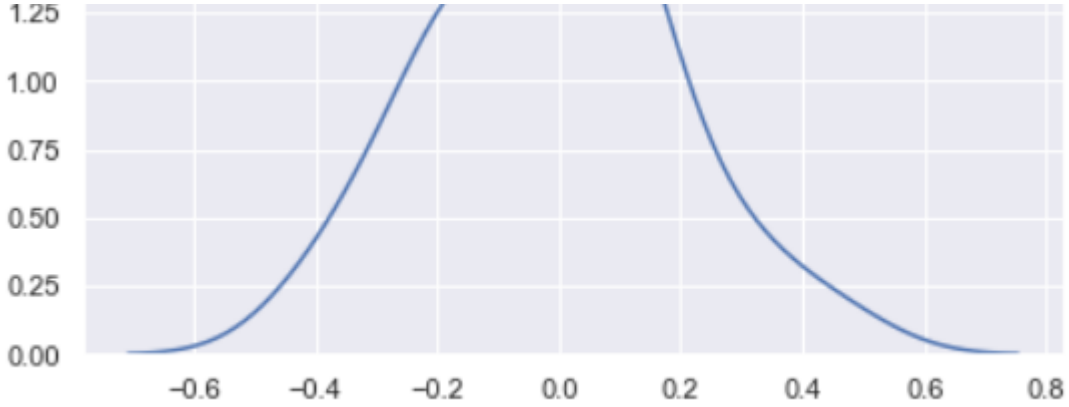
Now we take 30 samples from this population and each sample includes 30 values. Please note that we need to convert the population to pandas series because sample function will not accept numpy arrays.

```
population = pd.Series(norm)
samples_from_normal = random_samples(population, 30, 30)
plt.figure(figsize=(7,4))
plt.title("Distribution of Sample Means - Normal Distribution",
fontsize=15)
sns.distplot(samples_from_normal, hist=False)
```

```
population = pd.Series(norm)
samples_from_normal = random_samples(population, 30, 30)
plt.figure(figsize=(7,4))
plt.title("Distribution of Sample Means (30 samples)", fontsize=15)
sns.distplot(samples_from_normal, hist=False)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0xbad9883198>
```





Sampling distribution from a normal distribution

The sampling distribution (distribution of sample means) looks pretty close to a normal distribution. As we take more samples with larges size, sampling distribution will look more “normal”.

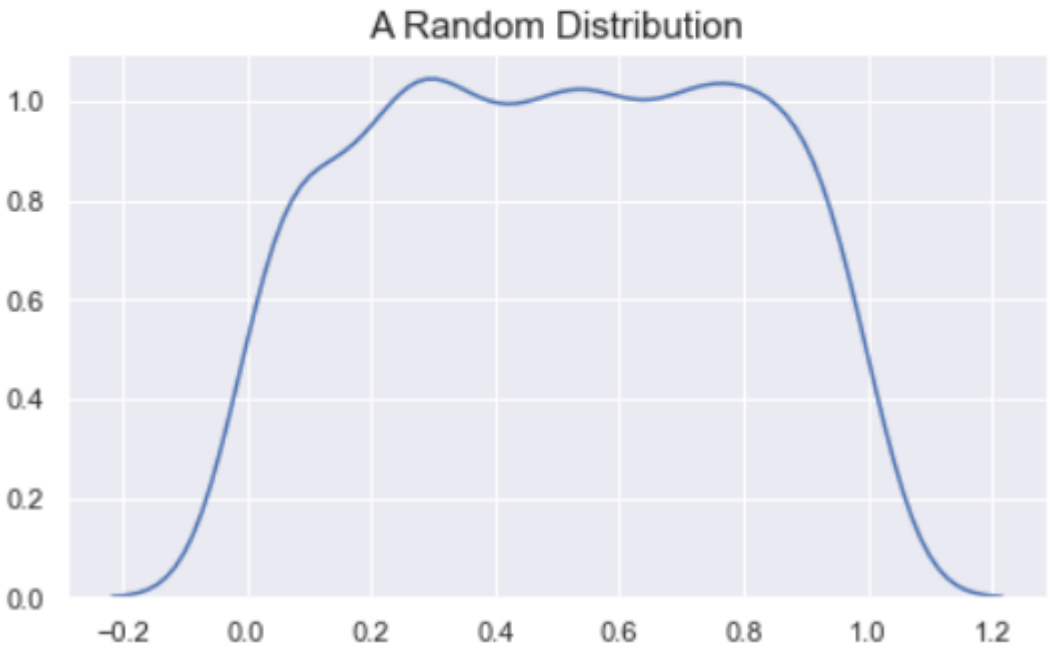
Let’s apply the same procedure to a population with **random distribution**. We first create an array with 1000 random numbers:

```
random = np.random.random(1000)
plt.figure(figsize=(7,4))
plt.title("A Random Distribution", fontsize=15)
sns.distplot(random, hist=False)
```


```
random = np.random.random(1000)
plt.figure(figsize=(7,4))
```

```
plt.title("A Random Distribution", fontsize=15)
sns.distplot(random, hist=False)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0xbad9a9e710>
```

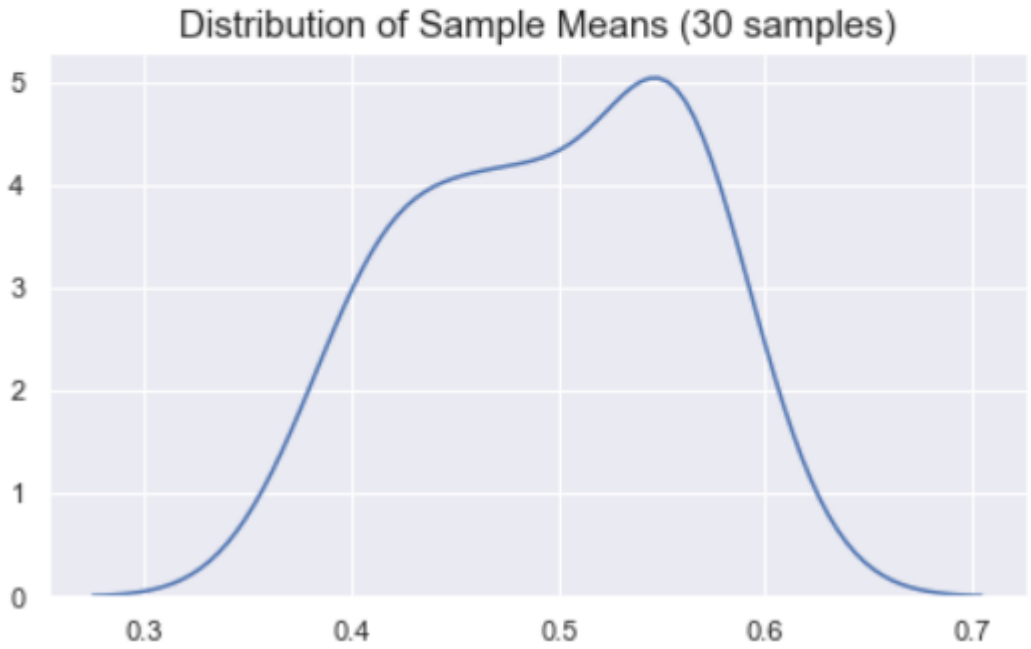


Let's see how sampling distribution will look like with 30 samples with 30 values each:

```
population = pd.Series(random)
samples_from_normal = random_samples(population, 30, 30)
plt.figure(figsize=(7,4))
plt.title("Distribution of Sample Means (30 samples)", fontsize=15)
sns.distplot(samples_from_normal, hist=False)
```

```
population = pd.Series(random)
samples_from_normal = random_samples(population, 30, 30)
plt.figure(figsize=(7,4))
plt.title("Distribution of Sample Means (30 samples)", fontsize=15)
sns.distplot(samples_from_normal, hist=False)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0xbad9a4c6d8>
```



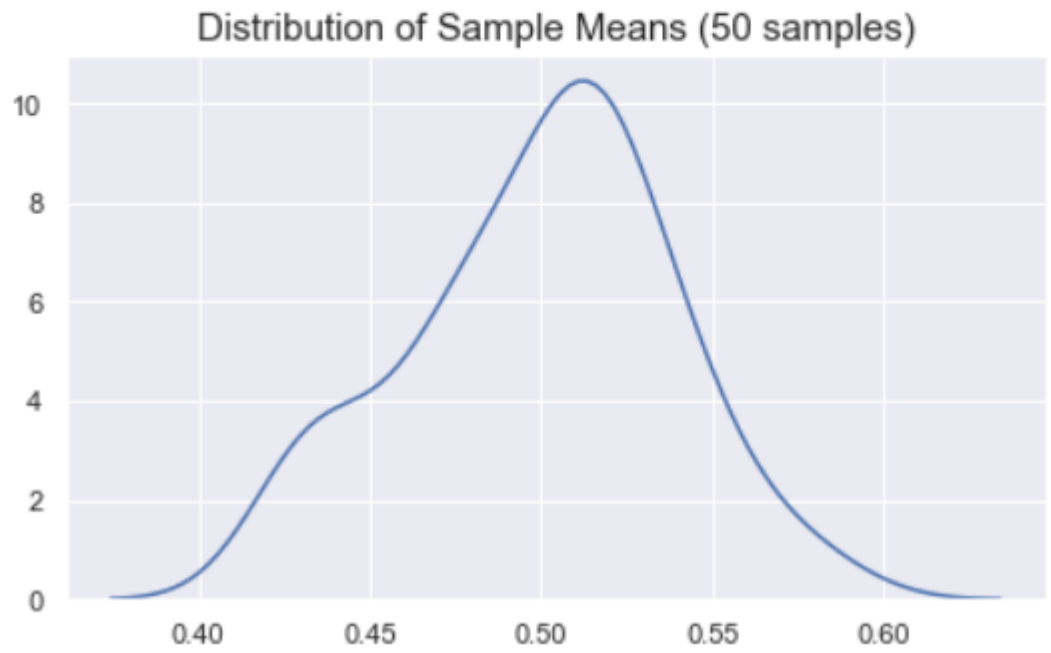
Sampling distribution from a random distribution (30 samples)

It is getting close to a normal distribution. We now try with 50 samples and also increase the sample size to 50:

```
population = pd.Series(random)
samples_from_normal = random_samples(population, 50, 50)
```

```
plt.figure(figsize=(7,4))
plt.title("Distribution of Sample Means (50 samples)", fontsize=15)
sns.distplot(samples_from_normal, hist=False)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0xbadaca8d30>
```



Sampling distribution from a random distribution (50 samples)

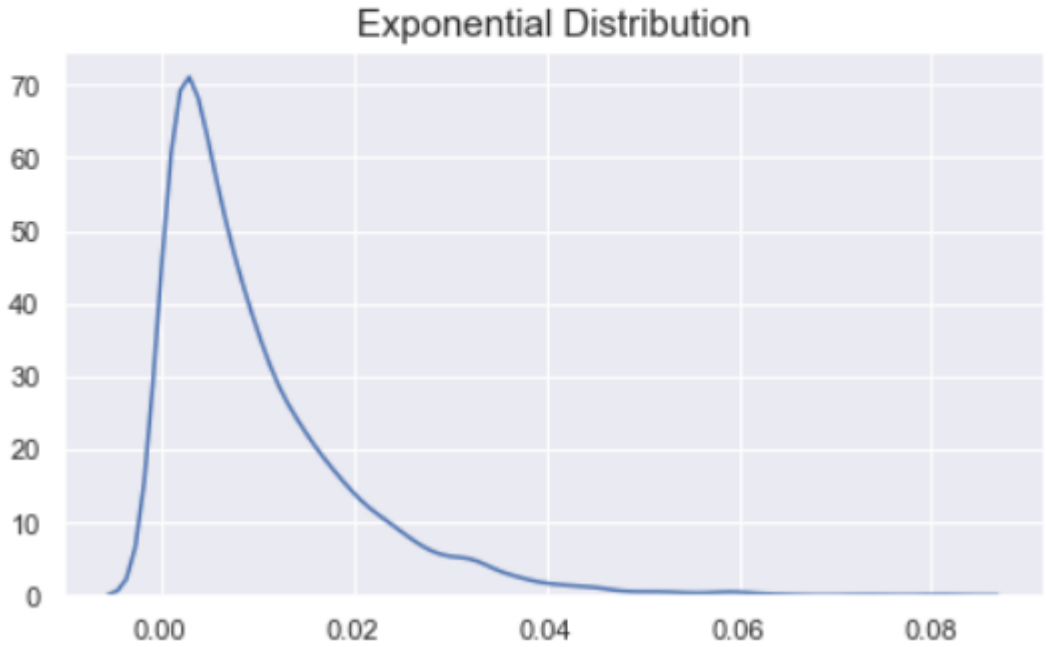
It definitely looks more “normal”. I added the code as texts so you can just copy-paste and try out with different sample quantity and sizes.

We can also try the **exponential distribution** and see CLT applies:

```
exp = np.random.exponential(0.01, 5000)
plt.figure(figsize=(7,4))
plt.title("Exponential Distribution", fontsize=15)
sns.distplot(exp, hist=False)
```


```
exp = np.random.exponential(0.01, 5000)
plt.figure(figsize=(7,4))
plt.title("Exponential Distribution", fontsize=15)
sns.distplot(exp, hist=False)
```


```
<matplotlib.axes._subplots.AxesSubplot at 0xbadad28320>
```

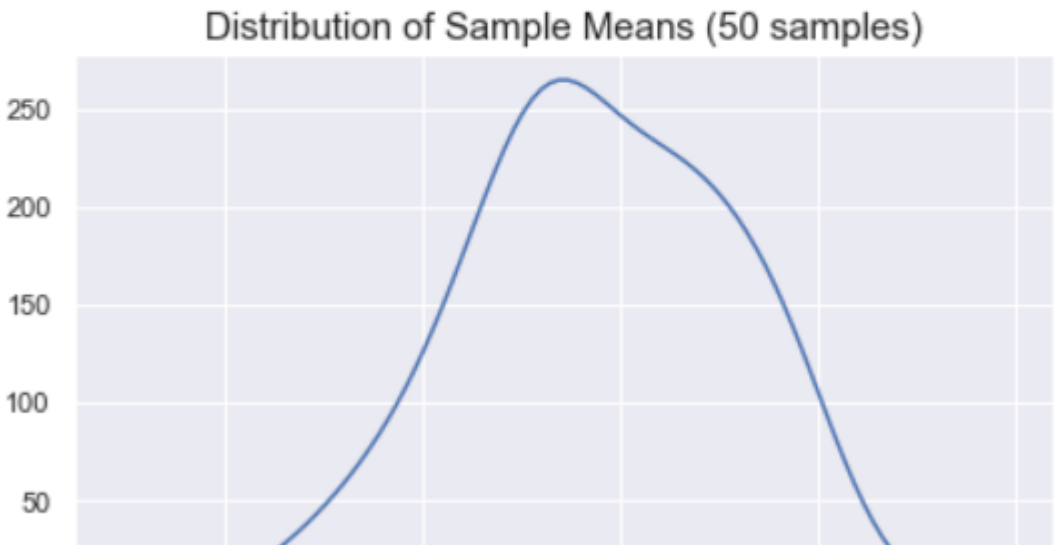


If we randomly take 50 samples with a size of 50, the distribution of the sample means look like:

```
population = pd.Series(exp)
samples_from_normal = random_samples(population, 50, 50)
plt.figure(figsize=(7,4))
plt.title("Distribution of Sample Means (50 samples)", fontsize=15)
sns.distplot(samples_from_normal, hist=False)
```

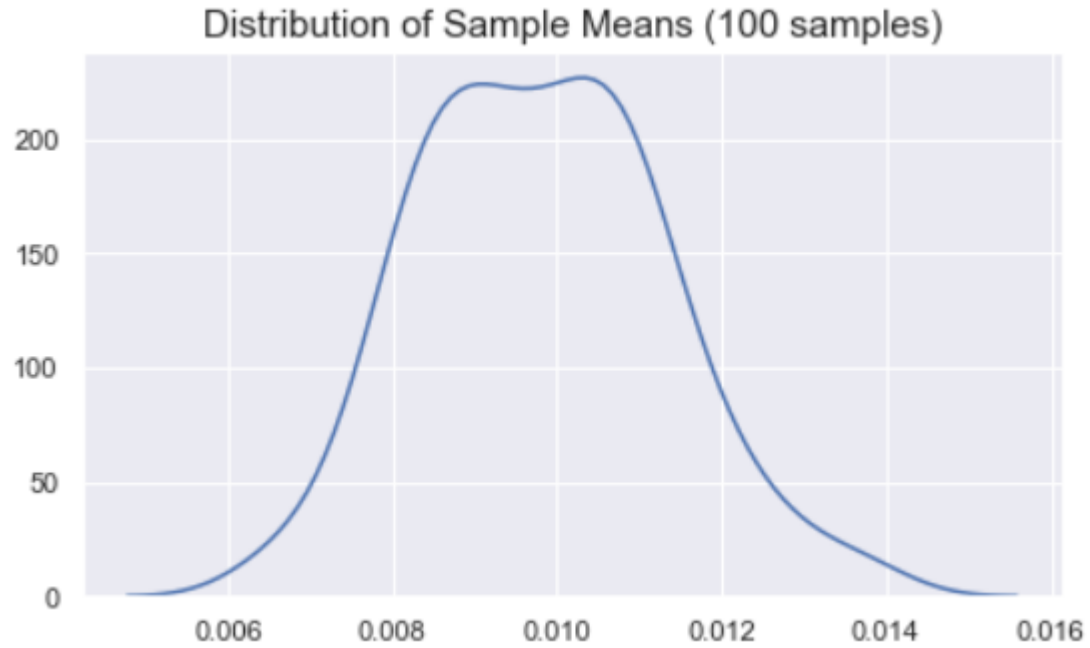
```
population = pd.Series(exp)
samples_from_normal = random_samples(population, 50, 50)
plt.figure(figsize=(7,4))
plt.title("Distribution of Sample Means (50 samples)", fontsize=15)
sns.distplot(samples_from_normal, hist=False)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0xbadae07c88>
```



Sampling distribution from an exponential distribution (50 samples)

It looks more like a normal distribution than an exponential distribution. With 100 samples, normality is more prominent:



Sampling distribution from an exponential distribution (100 samples)

As we have seen in the examples, regardless of the population distribution, the distribution of sample means get closer to a normal distribution as we take more samples. This is exactly what central limit theorem states.

Thank you for reading. Please let me know if you have any feedback.