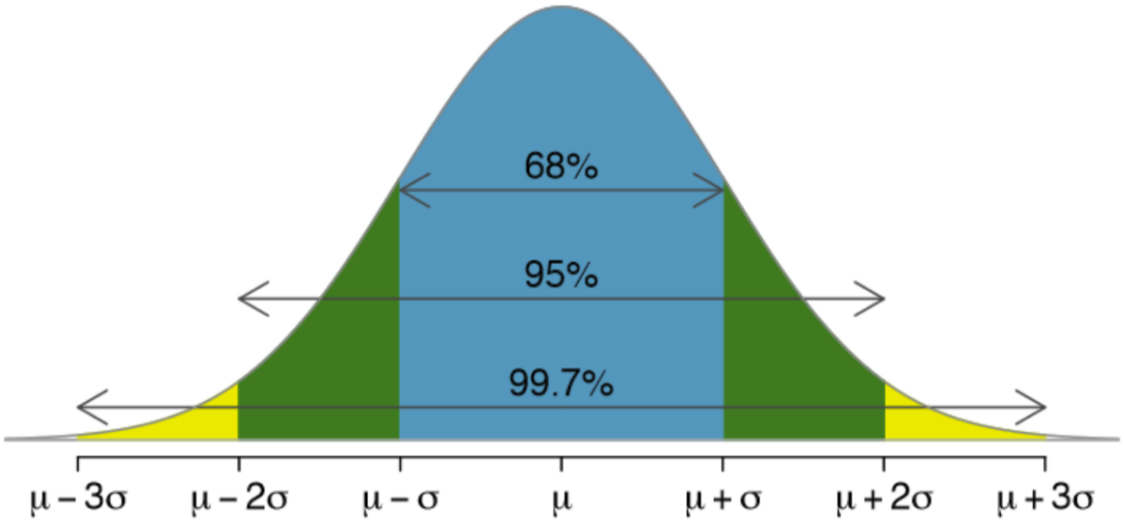
The Empirical Rule and Distribution

In week 2, we discussed the empirical rule or the 68 - 95 - 99.7 rule, which describes how many observations fall within a certain distance from our mean. This distance from the mean is denoted as sigma, or standard deviation (the average distance an observation is from the mean).

The following image may help refresh your memory:



For this tutorial, we will be exploring the number of hours the average college student gets.

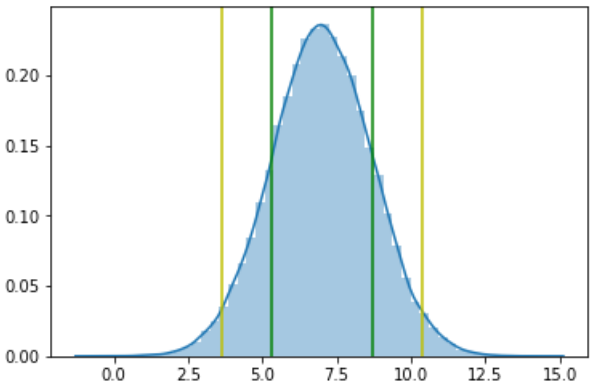
The example used in lecture stated there was a mean of 7 and standard deviation of 1.7 for hours of sleep; we will use these same values.

In [1]:
```
import warnings
warnings.filterwarnings('ignore')
import random
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

random.seed(1738)
```

In [2]:
```
mu = 7

sigma = 1.7

Observations = [random.normalvariate(mu, sigma) for _ in range(100000)]
```

In [3]:
```
sns.distplot(Observations)

plt.axvline(np.mean(Observations) + np.std(Observations), color = "g")
plt.axvline(np.mean(Observations) - np.std(Observations), color = "g")

plt.axvline(np.mean(Observations) + (np.std(Observations) * 2), color = "y")
plt.axvline(np.mean(Observations) - (np.std(Observations) * 2), color = "y")
```

Out[3]: <matplotlib.lines.Line2D at 0x7fe39c84f748>



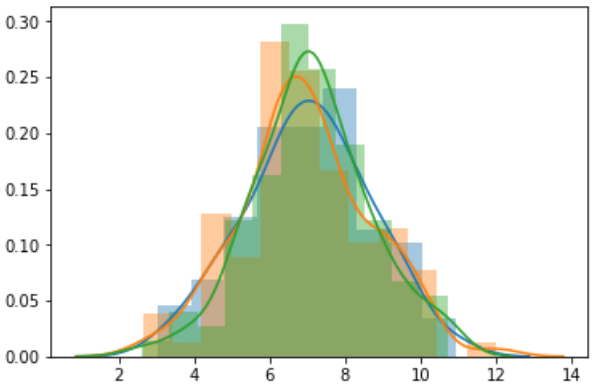
In [4]:
```
pd.Series(Observations).describe()
```

Out[4]:
```
count    100000.000000
mean          7.000626
std           1.693249
min          -0.754203
25%           5.865611
50%           7.003080
75%           8.144851
max          14.595650
dtype: float64
```

In [5]:
```
SampleA = random.sample(Observations, 100)
SampleB = random.sample(Observations, 100)
SampleC = random.sample(Observations, 100)
```

In [6]:
```
fig, ax = plt.subplots()

sns.distplot(SampleA, ax = ax)
sns.distplot(SampleB, ax = ax)
sns.distplot(SampleC, ax = ax)
```

Out[6]: <matplotlib.axes._subplots.AxesSubplot at 0x7fe39c7947f0>



Now that we have covered the 68 - 95 - 99.7 rule, we will take this a step further and discuss the empirical distribution.

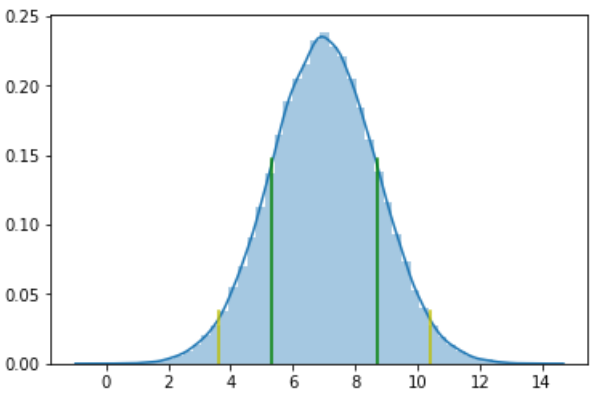
The empirical distribution is a cumulative density function that signifies the proportion of observations that are less than or equal to a certain values.

Lets use the initial image above as an example of this concept:

Now, by using our observations for ours of sleep, we can create an empirical distribution in python that signifies the proportion of observations are observed at a specific number for hours of sleep.

In [7]:
```
mu = 7

sigma = 1.7

Observations = [random.normalvariate(mu, sigma) for _ in range(100000)]

sns.distplot(Observations)
plt.axvline(np.mean(Observations) + np.std(Observations), 0, .59, color = 'g')
plt.axvline(np.mean(Observations) - np.std(Observations), 0, .59, color = 'g')

plt.axvline(np.mean(Observations) + (np.std(Observations) * 2), 0, .15, color
= 'y')
plt.axvline(np.mean(Observations) - (np.std(Observations) * 2), 0, .15, color
= 'y')
```

Out[7]: <matplotlib.lines.Line2D at 0x7fe39eb4f208>



In [8]:
```
from statsmodels.distributions.empirical_distribution import ECDF
import matplotlib.pyplot as plt

ecdf = ECDF(Observations)

plt.plot(ecdf.x, ecdf.y)

plt.axhline(y = 0.025, color = 'y', linestyle='-')
plt.axvline(x = np.mean(Observations) - (2 * np.std(Observations)), color = 'y
', linestyle='-')

plt.axhline(y = 0.975, color = 'y', linestyle='-')
plt.axvline(x = np.mean(Observations) + (2 * np.std(Observations)), color = 'y
', linestyle='-')
```

Out[8]: <matplotlib.lines.Line2D at 0x7fe39ea16e80>