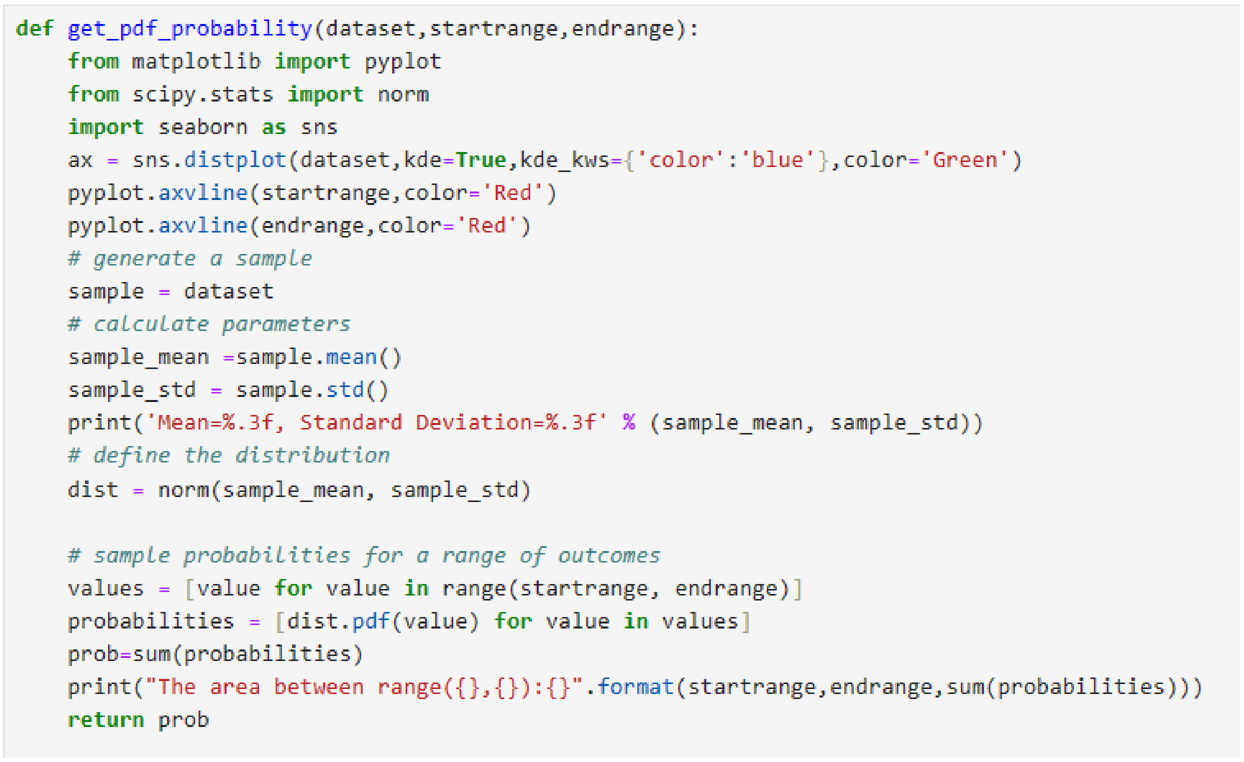
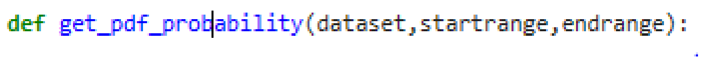
NORMAL DISTRIBUTION CODE EXPLANATION



```python
def get_pdf_probability(dataset,startrange,endrange):
    from matplotlib import pyplot
    from scipy.stats import norm
    import seaborn as sns
    ax = sns.distplot(dataset,kde=True,kde_kws={'color':'blue'},color='Green')
    pyplot.axvline(startrange,color='Red')
    pyplot.axvline(endrange,color='Red')
    # generate a sample
    sample = dataset
    # calculate parameters
    sample_mean =sample.mean()
    sample_std = sample.std()
    print('Mean=%.3f, Standard Deviation=%.3f' % (sample_mean, sample_std))
    # define the distribution
    dist = norm(sample_mean, sample_std)

    # sample probabilities for a range of outcomes
    values = [value for value in range(startrange, endrange)]
    probabilities = [dist.pdf(value) for value in values]
    prob=sum(probabilities)
    print("The area between range({},{}):{}".format(startrange,endrange,sum(probabilities)))
    return prob
```

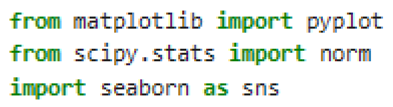


```python
def get_pdf_probability(dataset,startrange,endrange):
```

To create a function **get_pdf_probability(dataset, startrange, endrange)** that calculates the

Total area under the PDF curve within the range [startrange, endrange]. This involves integrating the PDF over the specified range to find the probability

The integral of the PDF over the range [startrange, endrange] gives the probability of the random variable falling within that range.



```python
from matplotlib import pyplot
from scipy.stats import norm
import seaborn as sns
```

Matplotlib -→ It is widely used for creating publication-quality plots and charts.

Pyplot→It is a Module within a Matplotlib. It is often used for simple visualizations, such as line plots, scatter plots, and histograms.

```
from scipy.stats import norm
```

scipy.stats→ It is a module within scipy that provides a wide range of statistical functions and methods, including probability distributions, statistical tests, and regression analysis.

Norm→ It is a function within scipy.stats that provides a normal (Gaussian) probability distribution. It can be used to generate random numbers, calculate probabilities, and fit data to a normal distribution.

Seaborn→It is often used for exploratory data analysis and statistical visualization.

```
ax = sns.distplot(dataset,kde=True,kde_kws={'color':'blue'},color='Green')
```

- dataset: This is the dataset or array of data that you want to visualize.
- kde=True: This parameter specifies that a **kernel density estimate** (KDE) should be plotted along with the histogram.
- kde_kws={'color':'blue'}: This parameter allows you to customize the appearance of the KDE curve. In this case, it sets the color of the KDE curve to blue.
- color='Green': This parameter sets the color of the **histogram** bars to green.

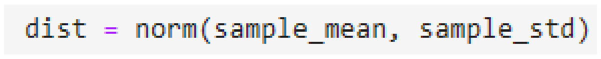
```
pyplot.axvline(startrange,color='Red')
pyplot.axvline(endrange,color='Red')
```

pyplot.axvline()→ is a function from the matplotlib.pyplot module that adds a vertical line to the current axes. The first argument is the x-position of the line, and the second argument is the color of the line.

The first line adds a vertical line at the position startrange with a red color, and the **second line** adds a vertical line at the position endrange with a red color as well. These lines can be used to highlight specific regions or values on a plot.

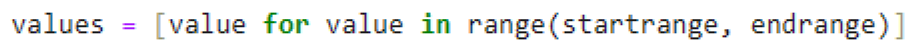
```
sample = dataset
# calculate parameters
sample_mean =sample.mean()
sample_std = sample.std()
print('Mean=%.3f, Standard Deviation=%.3f' % (sample_mean, sample_std))
```

- The first line calculates the mean of the sample using the **mean()** method, which returns the average value of the sample.
- The second line calculates the standard deviation of the sample using the **std()** method, which returns the square root of the variance of the sample.
- The **%.3f** format specifier in the print() statement specifies that the output should be formatted with **3 decimal places**.



```
dist = norm(sample_mean, sample_std)
```

It is used to define a **normal distribution** with specific parameters based on the mean and standard deviation calculated from a sample dataset.



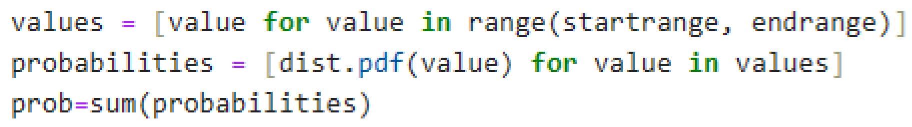
```
values = [value for value in range(startrange, endrange)]
```

In Traditional way we can write as,

values=[]

for value in range(startrange,endrange):

    values.append(values)



```
values = [value for value in range(startrange, endrange)]
probabilities = [dist.pdf(value) for value in values]
prob=sum(probabilities)
```

The **range()** function is used to generate a sequence of values from startrange to endrange.

The **dist.pdf()** function is then used to calculate the **probability density function** (PDF) values for each value in the values list. The resulting probabilities are stored in the probabilities list using list comprehension

Finally, the **sum()** function is used to add up the probabilities in the probabilities list, resulting in the total probability of the range of values.