```python
def get_pdf_probability(ds,startrange,endrange):
    from matplotlib import pyplot
    from scipy.stats import norm
    import seaborn as sns
    ax=sns.distplot(ds,kde=True,kde_kws={'color':'blue'},color='green')
    pyplot.axvline(startrange,color='Red')
    pyplot.axvline(endrange,color='Red')
    #generate a sample
    sample=ds
    #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
```

1. Import Libraries:

* matplotlib.pyplot : Used for plotting and visualization.

* scipy.stats.norm: Provides the normal distribution for calculating probabilities.

* seaborn : Used for statistical data visualization, specifically for creating the distribution plot.

2. Function:

* startrange: The starting value of the range.

* endrange: The ending value of the range.

3. Visualization (Distribution Plot):

* ax = sns.distplot(ds, kde=True, kde_kws={‘color’: ‘blue’}, color=’green’)

* Creates a distribution plot (histogram with kernel density estimation) of the data sample ds using seaborn.distplot.

* kde=True enables the kernel density estimate

* kde_kws={‘color’: ‘blue’} sets the color of the KDE curve to blue.

* color=’green’ sets the color of the histogram bars to green.

* pyplot.axvline(startrange, color=’Red’)

* Draws a vertical red line at the startrange value on the plot.

* pyplot.axvline(endrange, color=’Red’)

* Draws a vertical red line at the endrange value on the plot.

4. Calculate Sample Statistics:

* sample_mean = sample.mean()

* Calculates the mean (average) of the sample.

* sample_std = sample.std()

* Calculates the standard deviation of the sample.

* print(‘Mean=%.3f, Standard Deviation=%.3f’ % (sample_mean, sample_std))

* Prints the calculated mean and standard deviation to the console, formatted to three decimal places.

5. Define Normal Distribution:

* dist = norm(sample_mean, sample_std)

* Creates a normal distribution object (dist) using the calculated sample_mean and sample_std as its parameters.

6. Calculate Probabilities:

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

* Creates a list of integer values within the specified range (from startrange to endrange – 1).

* probabilities = [dist.pdf(value) for value in values]