```python
def
get_pdf_probability(dataset,startrang
e,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)
```


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

🔍 Step-by-step meaning (in very simple words)

1️⃣ Draws a distribution graph

- Uses seaborn to plot the dataset's histogram + KDE curve.
- Draws two red vertical lines at:
  - `startrange`
  - `endrange`

This shows visually where the range lies.

2️⃣ Calculates mean and standard devation

```python
sample_mean = sample.mean()
sample_std.std()
```

This finds:

- **Average value of the dataset**
- **Spread (std deviation)** of the **dataset**

3️⃣ Creates a normal distribution

```python
dist = norm(sample_std)
```

This assumes:

- **“The dataset behaves like a**
- **normal (bell curve) distribution.**

4️⃣ Computes PDF values inside the range

```python
probabilities = []
for value in values:
    probabilities.append(dist.pdf(value))
```

For every integer between `startrange` and `endrange`, it calculates:

5️⃣ Returns the total PDF sum between the range

python

```python
prob = sum(probabilities)
return prob
```

This is **not the true probability** but an **approximation** using PDF values.