# Python equivalent of R statistics package

| **R** | **SciPy** | **Name** |
| --- | --- | --- |
| dnorm() | pdf() | Probability density function (PDF) |
| pnorm() | cdf() | Cumulative density function (CDF) |
| qnorm() | ppf() | Percentile point function (CDF inverse) |
| pnorm(lower.tail = FALSE) | sf() | Complementary CDF (CCDF) or survival function |
| qnorm(lower.tail = FALSE) | isf() | CCDF inverse or inverse survival function |
| rnorm() | rvs() | Random samples |

# Probability Density Function usage or dnorm()

from scipy.stats import norm

import matplotlib.pyplot as plt

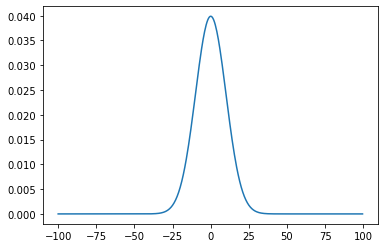
import numpy as np

mean = 0

sd = 10

x = np.arange(-100,100,0.5)

y = norm.pdf(x,loc=mean, scale=sd)

plt.plot(x,y,'-')

# Cumulative Density Function or pnorm()

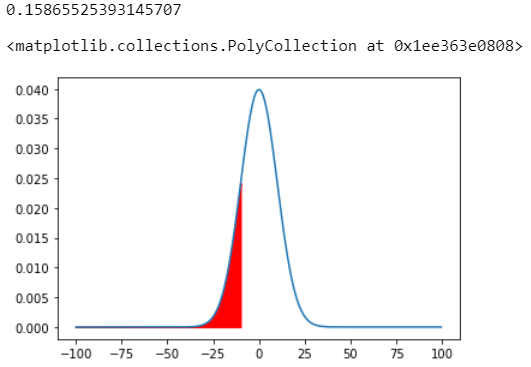
number = -10

prob = norm.cdf(number,loc=mean, scale=sd)

print(prob)

plt.plot(x,y,'-')

plt.fill\_between(x[x<=number],y[x<=number],color='red')



# Complementary CDF or pnorm(lower.tail = FALSE)

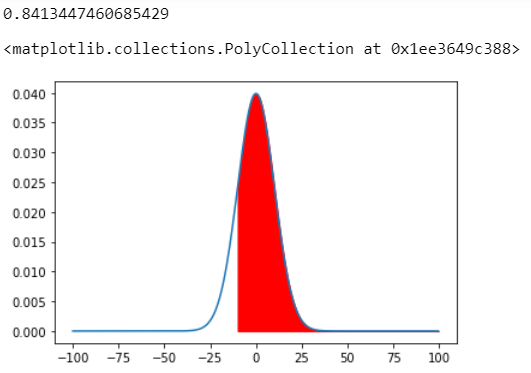
number = -10

prob = norm.sf(number,loc=mean, scale=sd)

print(prob)

plt.plot(x,y,'-')

plt.fill\_between(x[x>=number],y[x>=number],color='red')



# Percentile point function or qnorm()

probability = 0.15865525393145707

number = norm.ppf(probability,loc=mean,scale=sd)

print(number)

-10.0

# Inverse Survival Function or qnorm(lower.tail = False)

probability = 0.8413447460685429

number = norm.isf(probability,loc=mean,scale=sd)

print(number)

-10.0