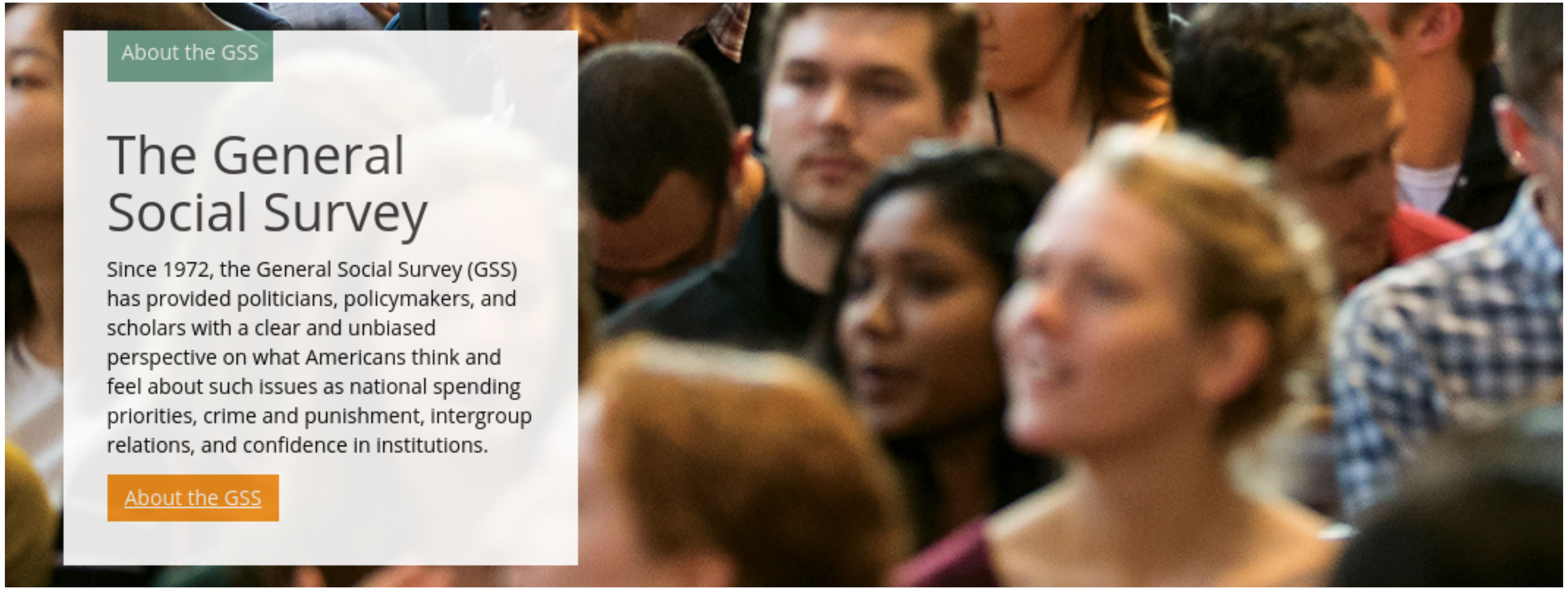
GSS

- Annual sample of U.S. population.
- Asks about demographics, social and political beliefs.
- Widely used by policy makers and researchers.



About the GSS

The General Social Survey

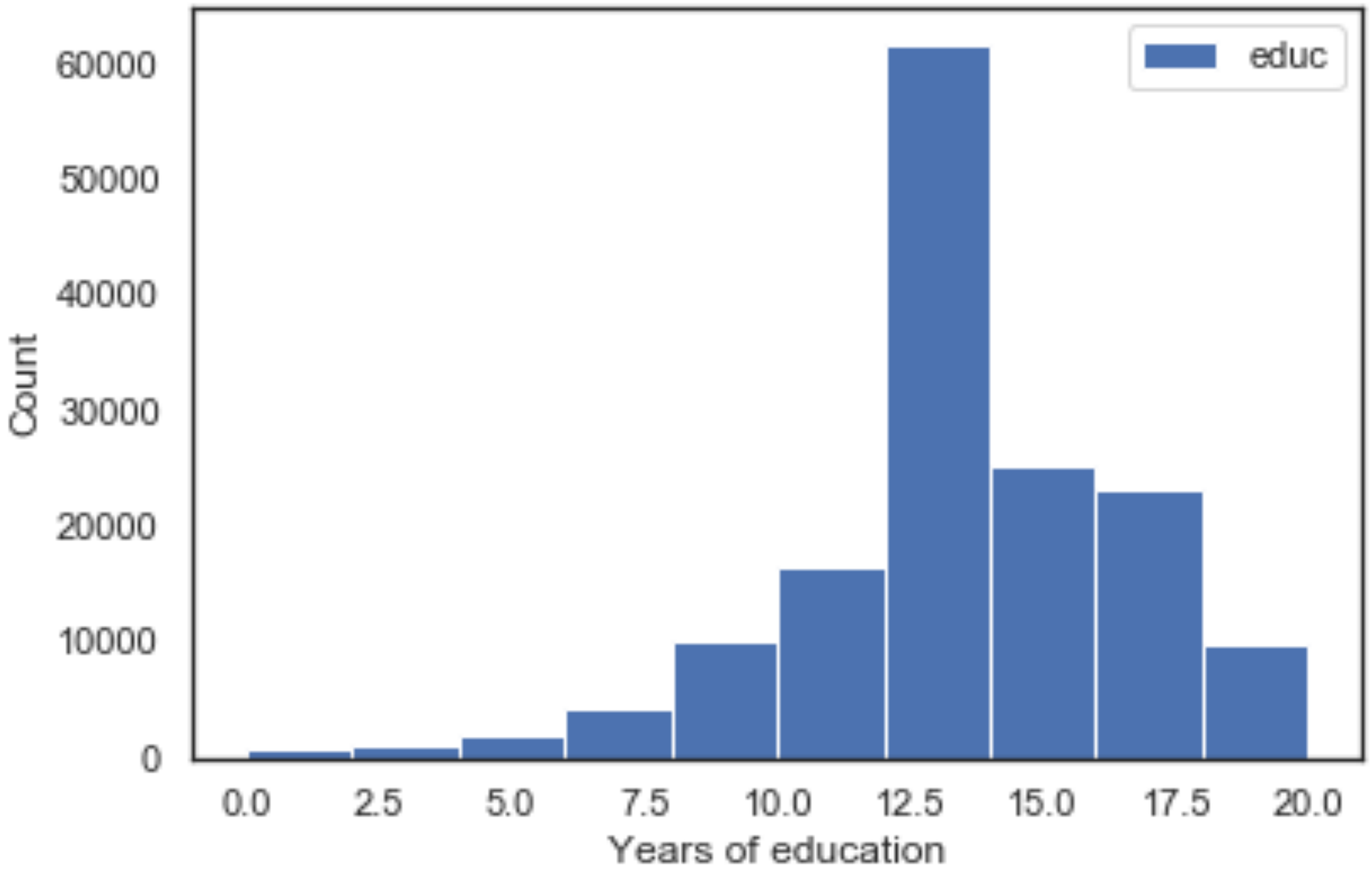
Since 1972, the General Social Survey (GSS) has provided politicians, policymakers, and scholars with a clear and unbiased perspective on what Americans think and feel about such issues as national spending priorities, crime and punishment, intergroup relations, and confidence in institutions.

Read the data

```
gss = pd.read_hdf('gss.hdf5', 'gss')


gss.head()
```

```
   year  sex   age  cohort  race  educ  realinc  wtssall
0  1972    1  26.0  1946.0     1  18.0  13537.0   0.8893
1  1972    2  38.0  1934.0     1  12.0  18951.0   0.4446
2  1972    1  57.0  1915.0     1  12.0  30458.0   1.3339
3  1972    2  61.0  1911.0     1  14.0  37226.0   0.8893
4  1972    1  59.0  1913.0     1  12.0  30458.0   0.8893
```

```python
educ = gss['educ']
plt.hist(educ.dropna(), label='educ')
plt.show()
```



Based on the histogram we can see the General distribution and the central tendency.

- Peak is at 12 year of education
- But the histogram is not the best way to visualise this distribution.
- An alternative is PMF.
- PMF contain the unique values in dataset and how often they apper.

PMF

```python
pmf_educ = Pmf(educ, normalize=False)
pmf_educ.head()
```

```
0.0    566
1.0    118
2.0    292
3.0    686
4.0    746
Name: educ, dtype: int64
```

PMF

```
pmf_educ[12]
```

```
47689
```

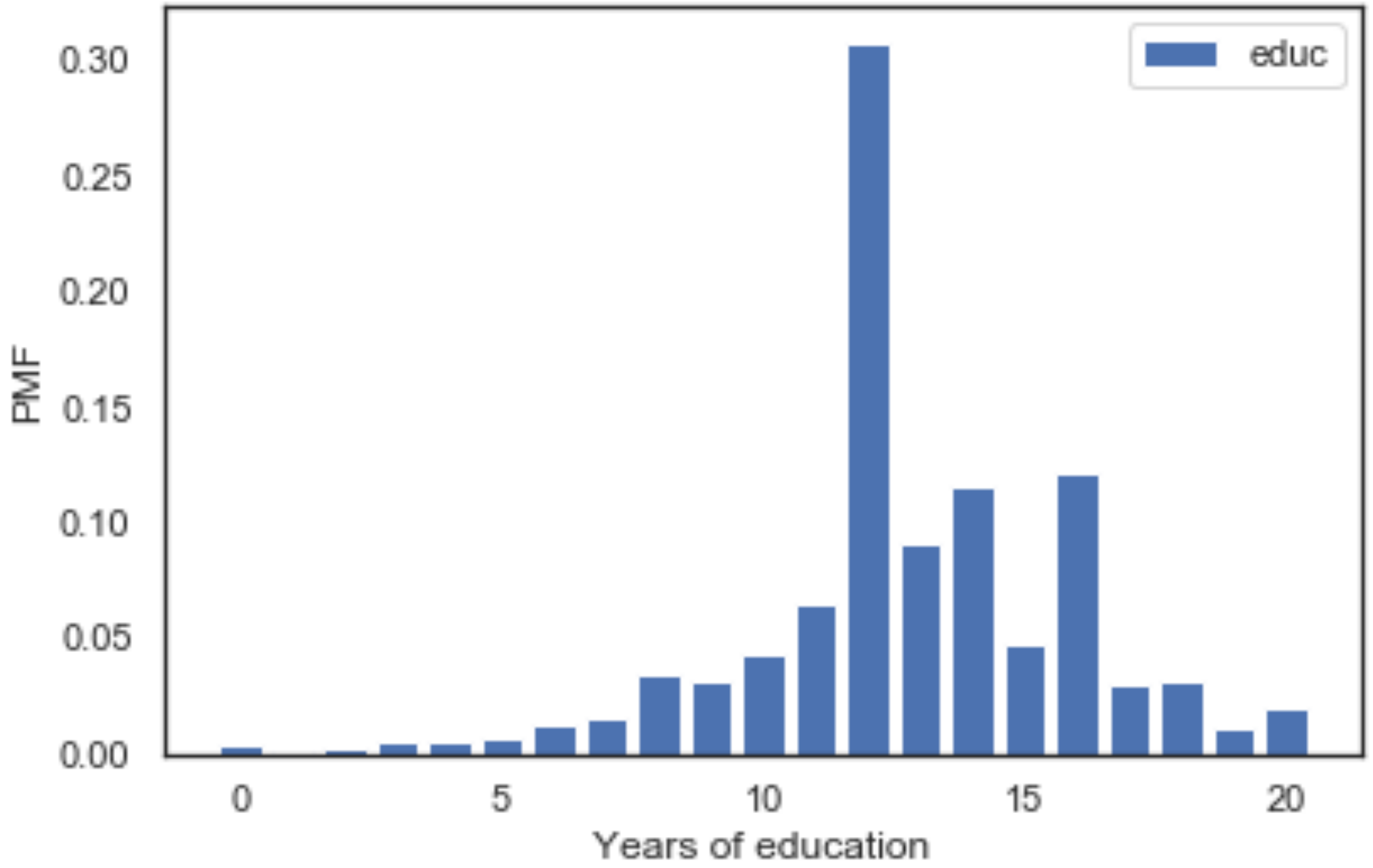
```
pmf_educ = Pmf(educ, normalize=True)

pmf_educ.head()
```

```
0.0    0.003663
1.0    0.000764
2.0    0.001890
3.0    0.004440
4.0    0.004828
Name: educ, dtype: int64
```

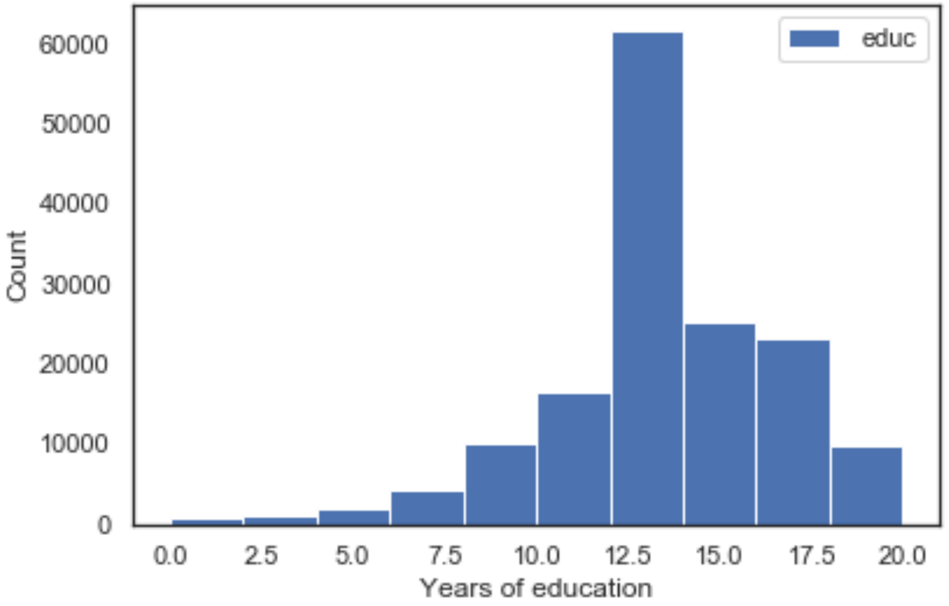
```
pmf_educ[12]
```

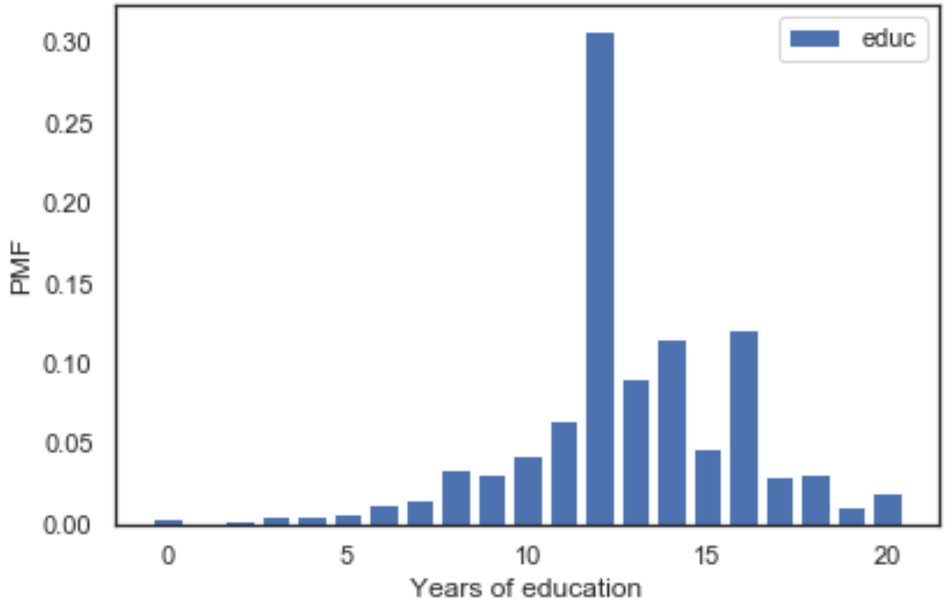
```
0.30863869940587907
```

```python
pmf_educ.bar(label='educ')
plt.xlabel('Years of education')
plt.ylabel('PMF')
plt.show()
```



Histogram vs. PMF





Let's make some PMFs!

EXPLORATORY DATA ANALYSIS IN PYTHON

From PMF to CDF

If you draw a random element from a distribution:

- PMF (Probability Mass Function) is the probability that you get exactly x
- CDF (Cumulative Distribution Function) is the probability that you get a value <= x

for a given value of x.

Example

PMF of {1, 2, 2, 3, 5}

PMF(1) = 1/5

PMF(2) = 2/5

PMF(3) = 1/5

PMF(5) = 1/5

CDF is the cumulative sum of the PMF.

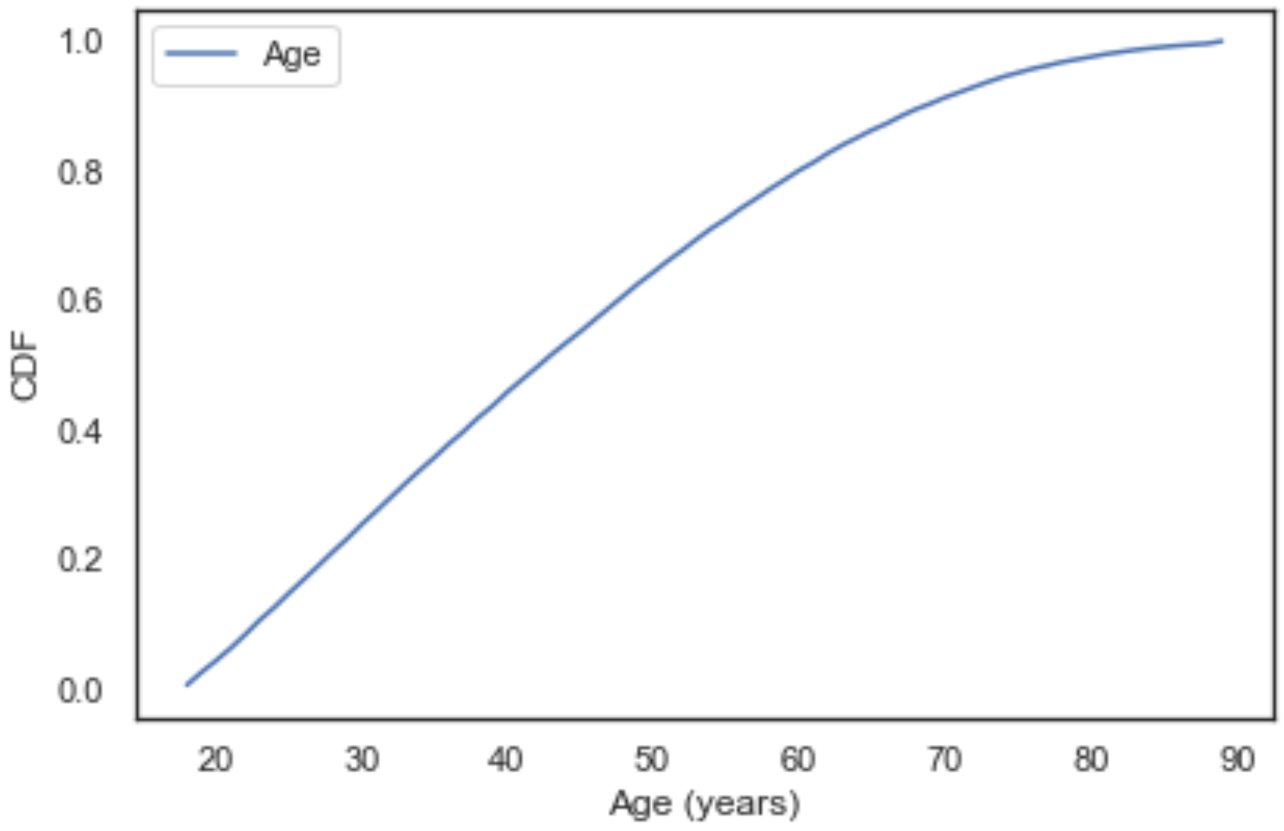
CDF(1) = 1/5

CDF(2) = 3/5

CDF(3) = 4/5

CDF(5) = 1

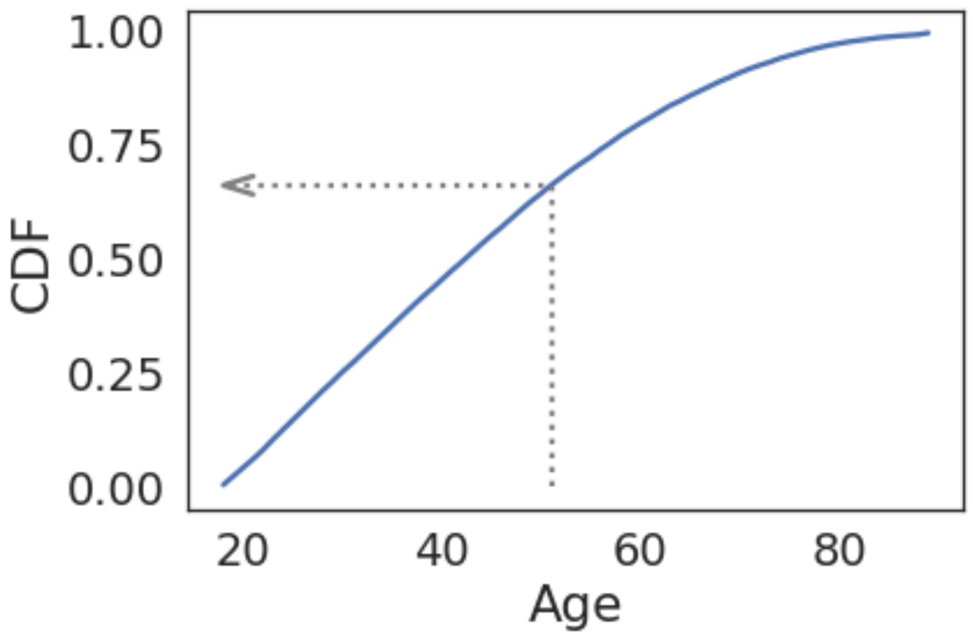
```
cdf = Cdf(gss['age'])
cdf.plot()
plt.xlabel('Age')
plt.ylabel('CDF')
plt.show()
```



Evaluating the CDF

```python
q = 51
p = cdf(q)
print(p)
```

```
0.66
```



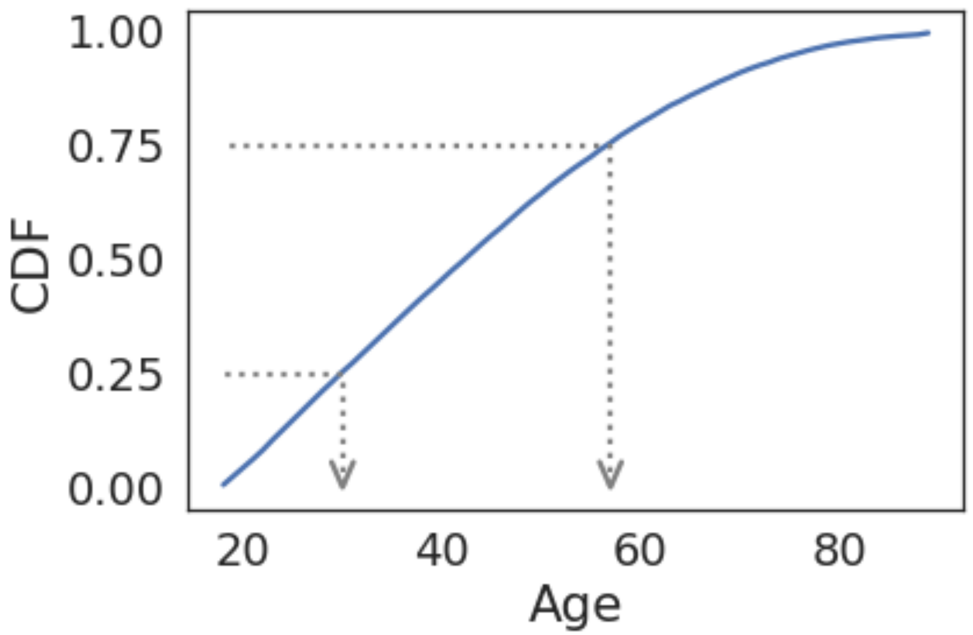
Evaluating the inverse CDF

```python
p = 0.25
q = cdf.inverse(p)
print(q)
```

```
30
```

```python
p = 0.75
q = cdf.inverse(p)
print(q)
```

```
57
```



The CDF is an invertible function, which means that if you have a probability, p, you can look up the corresponding quantity, q.

- 0.25, which returns 30. That means that 25% of the respondents are age 30 or less.

- 0.75, which returns 57, so 75% of the respondents are 57 or younger.

The distance from the 25th to the 75th percentile is called the interquartile range, or IQR. It measures the spread of the distribution,

- similar to standard deviation or variance.

- Because it is based on percentiles, it doesn't get thrown off by extreme values or outliers.

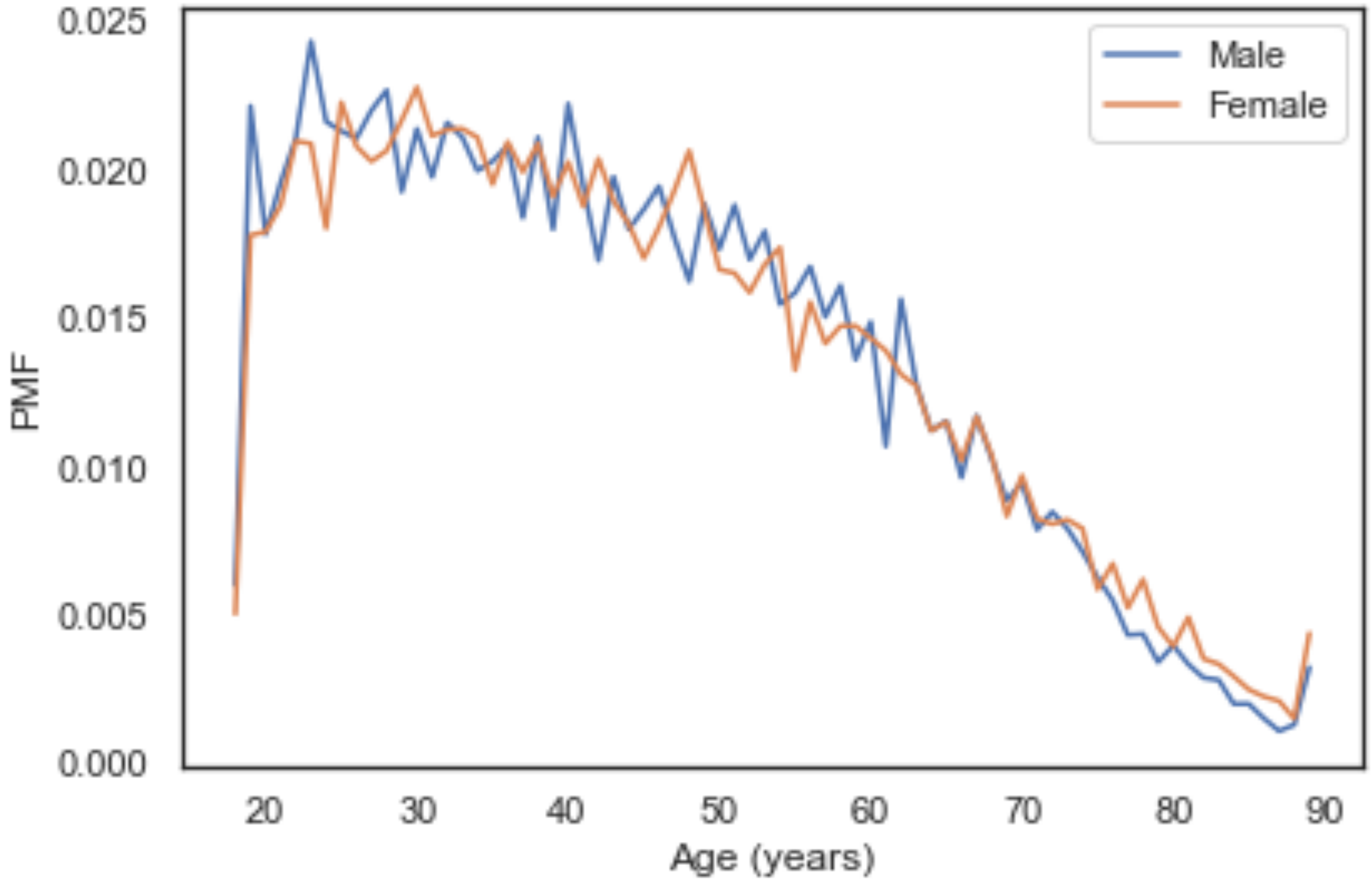
- IQR can be more "robust" than variance, which means it works well even if there are errors in the data or extreme values.

Let's practice!

EXPLORATORY DATA ANALYSIS IN PYTHON

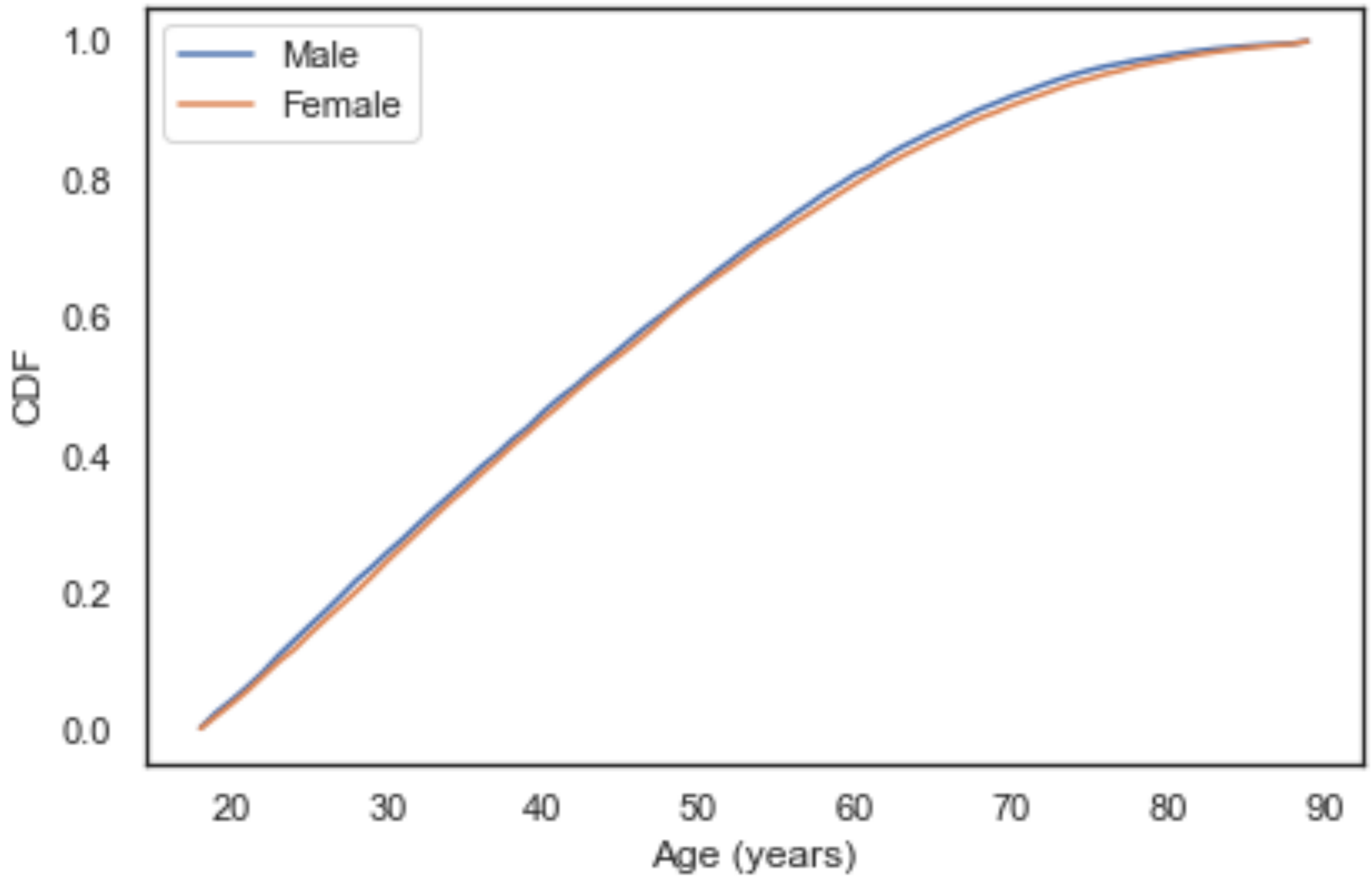
Multiple PMFs

```
male = gss['sex'] == 1
age = gss['age']
male_age = age[male]
female_age = age[~male]
Pmf(male_age).plot(label='Male')
Pmf(female_age).plot(label='Female')
plt.xlabel('Age (years)')
plt.ylabel('Count')
plt.show()
```


0.025
0.020
0.015
0.010
0.005
0.000
PMF
20
30
40
50
60
70
80
90
Age (years)
Male
Female

Multiple CDFs

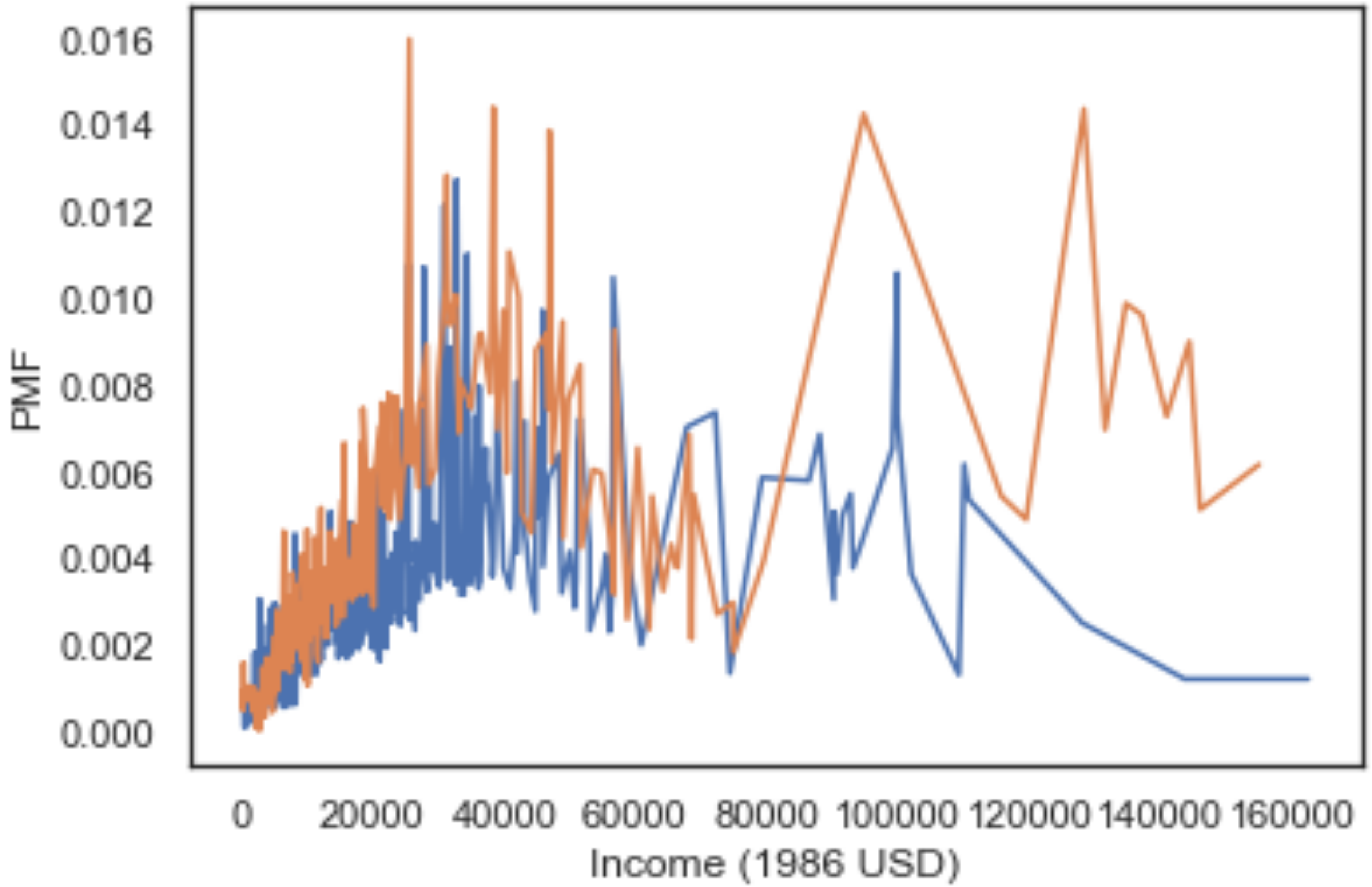
```python
Cdf(male_age).plot(label='Male')
Cdf(female_age).plot(label='Female')

plt.xlabel('Age (years)')
plt.ylabel('Count')
plt.show()
```



In general, CDFs are smoother than PMFs. Because they smooth out randomness, we can often get a better view of real differences between distributions.

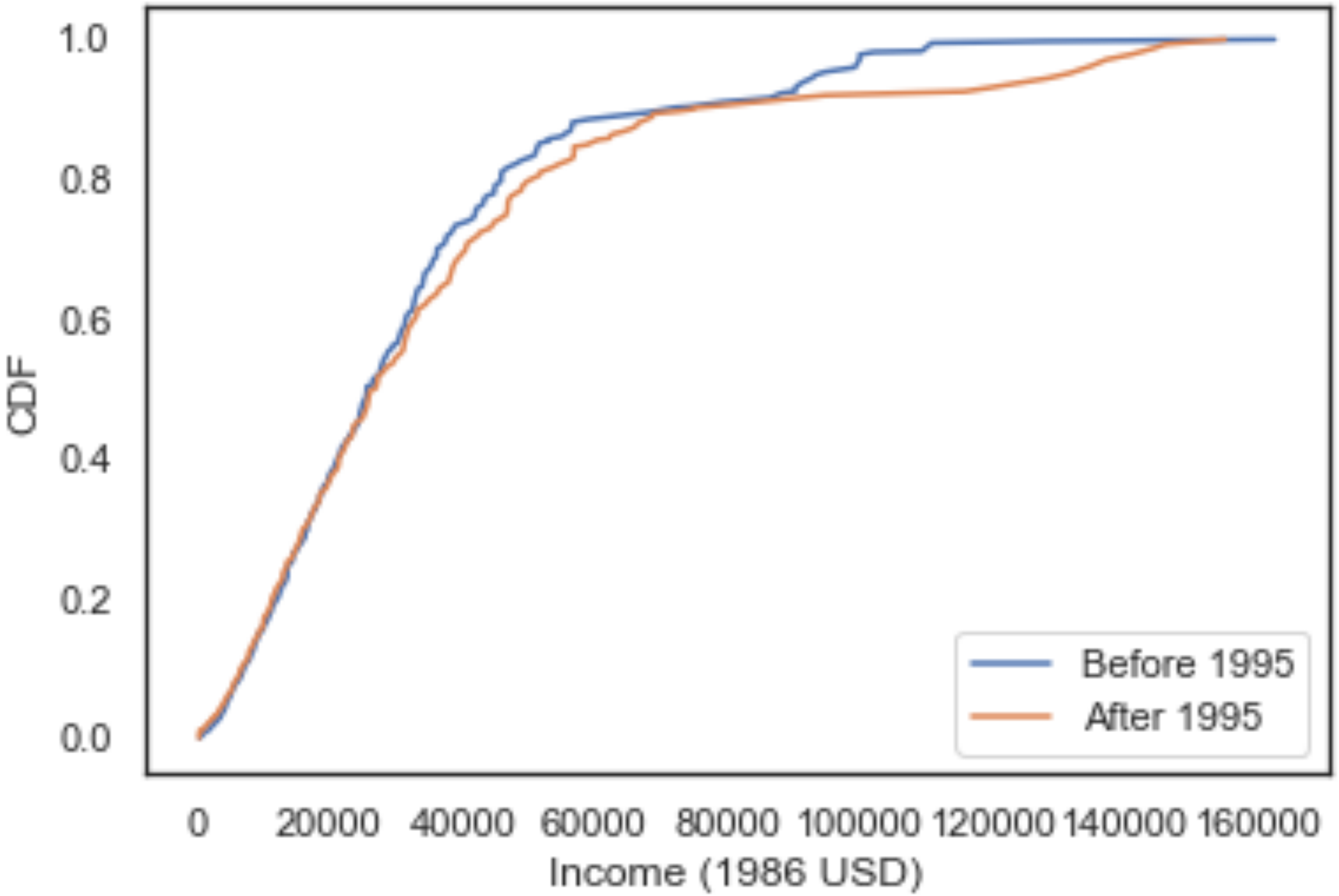
Income distribution

```
income = gss['realinc']
pre95 = gss['year'] < 1995
Pmf(income[pre95]).plot(label='Before 1995')
Pmf(income[~pre95]).plot(label='After 1995')
plt.xlabel('Income (1986 USD)')
plt.ylabel('PMF')
plt.show()
```


0.016
0.014
0.012
0.010
0.008
0.006
0.004
0.002
0.000
PMF
0
20000
40000
60000
80000
100000
120000
140000
160000
Income (1986 USD)

Income CDFs

```
Cdf(income[pre95]).plot(label='Before 1995')
Cdf(income[~pre95]).plot(label='After 1995')
```

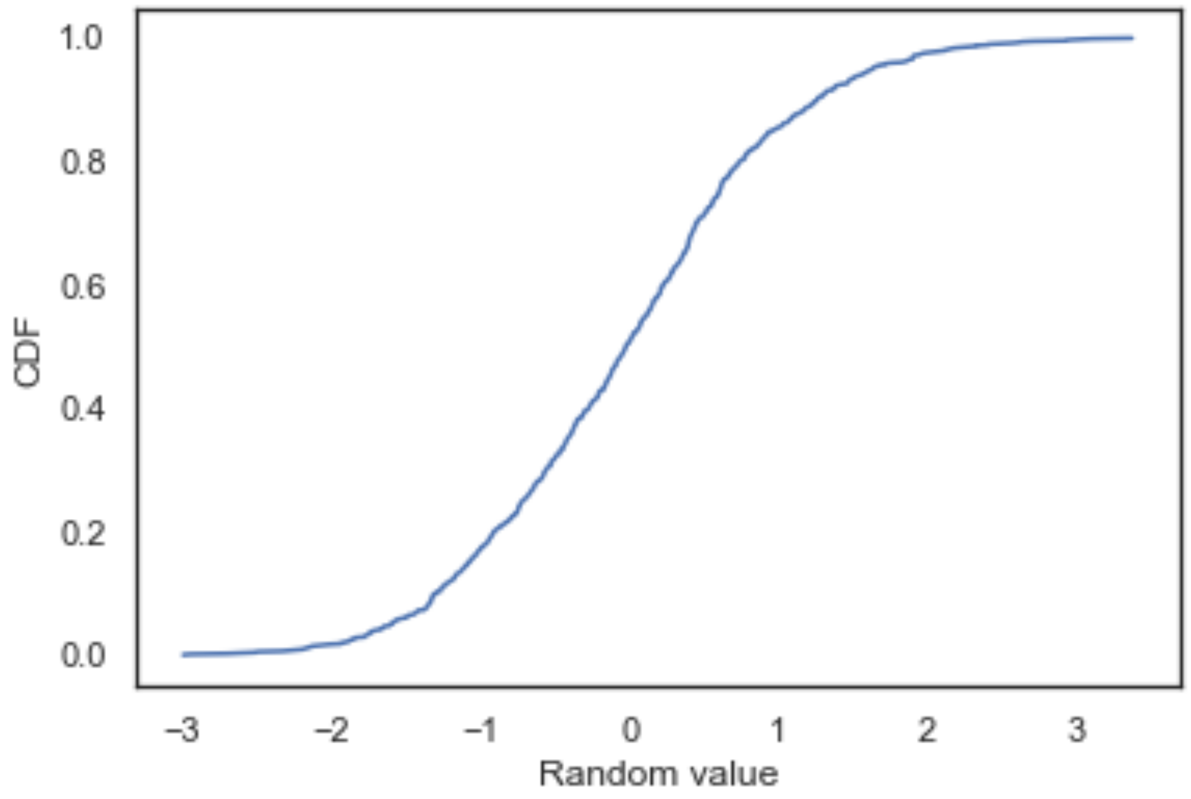

1.0
0.8
0.6
0.4
0.2
0.0
CDF
0
20000
40000
60000
80000
100000
120000
140000
160000
Income (1986 USD)
Before 1995
After 1995

Let's practice!

EXPLORATORY DATA ANALYSIS IN PYTHON

The normal distribution

```
sample = np.random.normal(size=1000)
Cdf(sample).plot()
```



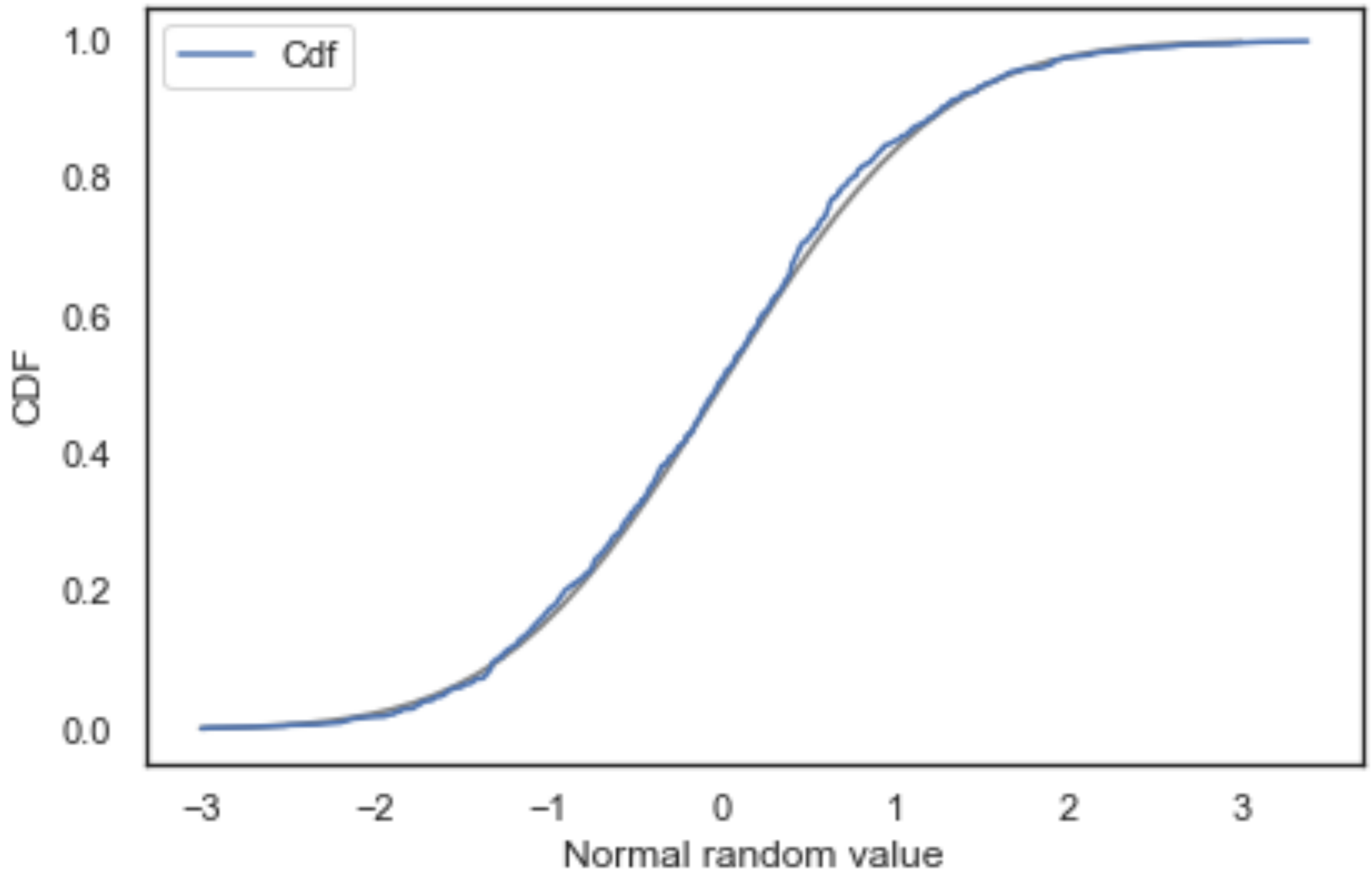
The normal CDF

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

```
xs = np.linspace(-3, 3)
ys = norm(0, 1).cdf(xs)
```

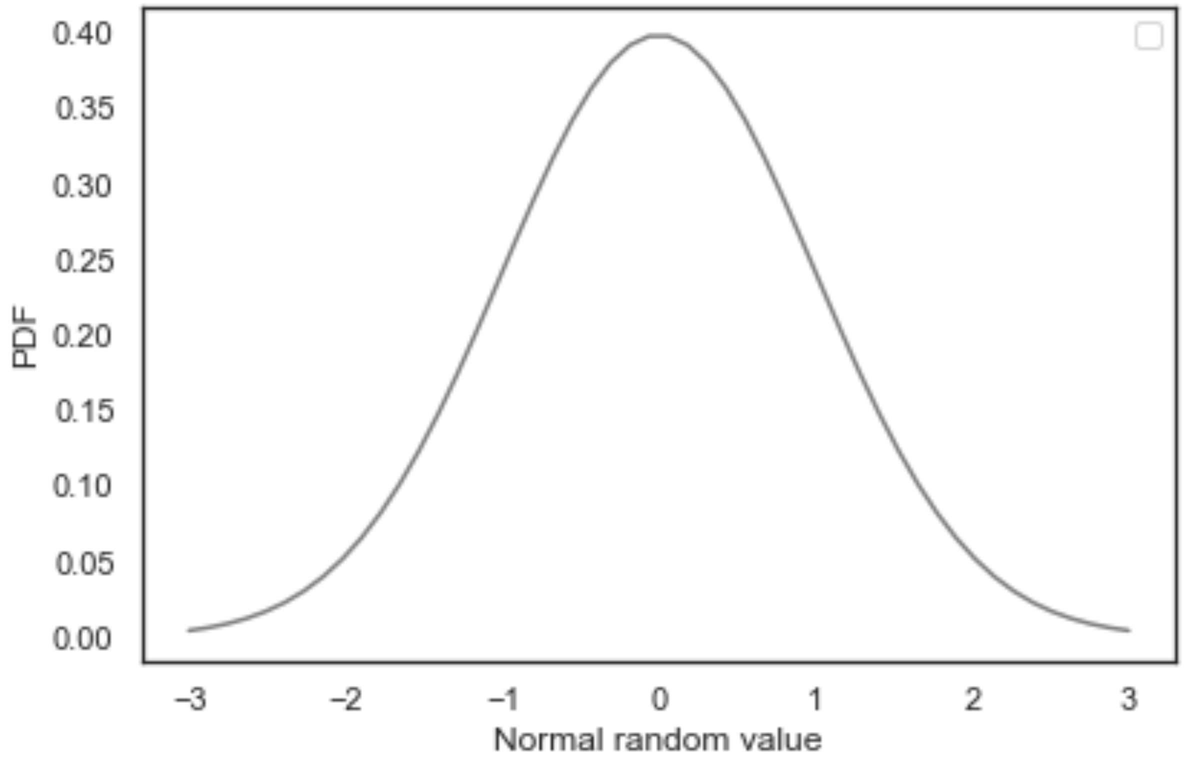
```
plt.plot(xs, ys, color='gray')
```

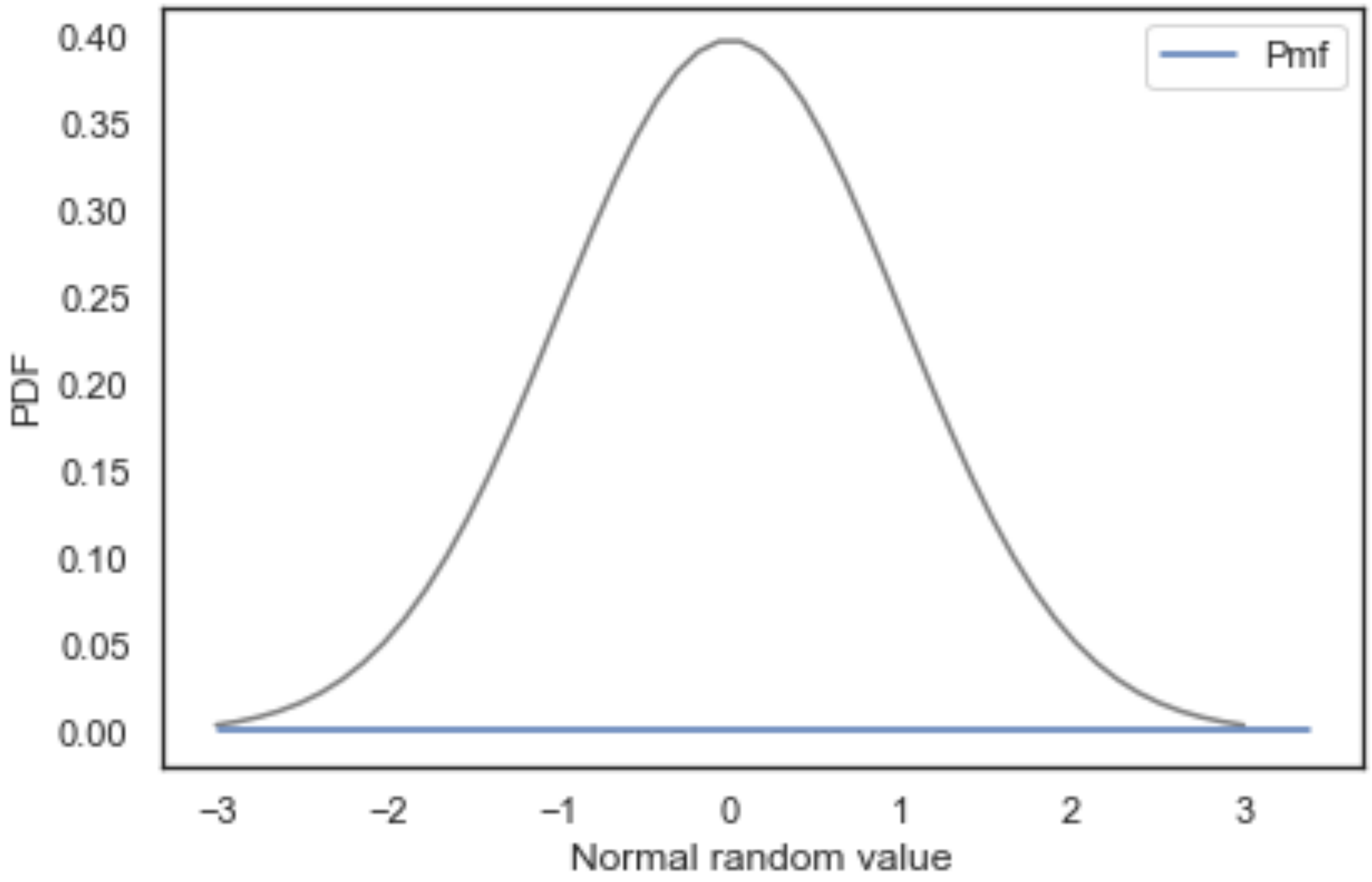
```
Cdf(sample).plot()
```


Cdf
1.0
0.8
0.6
0.4
0.2
0.0
CDF
−3
−2
−1
0
1
2
3
Normal random value

The bell curve

```
xs = np.linspace(-3, 3)
ys = norm(0,1).pdf(xs)
plt.plot(xs, ys, color='gray')
```



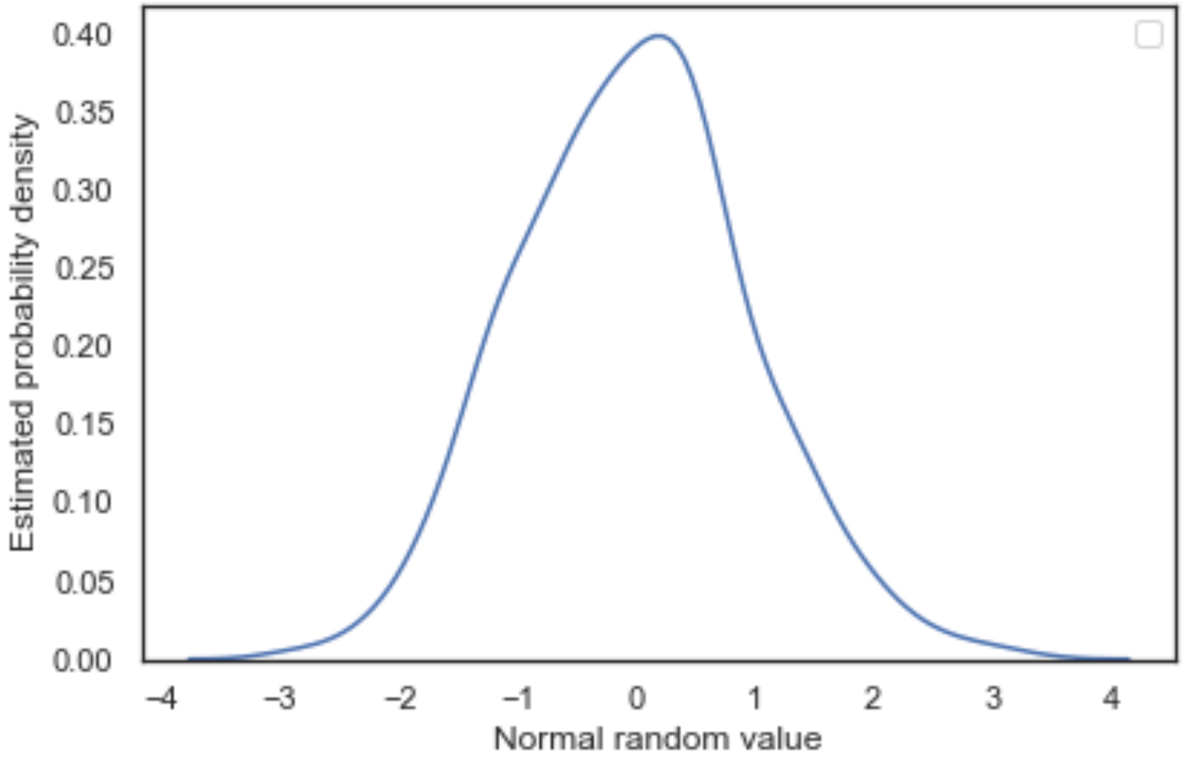


Unfortunately, if we compare this PDF to the PMF of the sample, it doesn't work very well.

Here's what it looks like. The PMF of the sample is a flat line across the bottom. In the random sample, every value is unique, so they all have the same probability, one in 1000. However, we can use the points in the sample to estimate the PDF of the distribution they came from. This process is called kernel density estimation, or KDE. It's a way of getting from a PMF, a probability mass function, to a PDF, a probability density function.

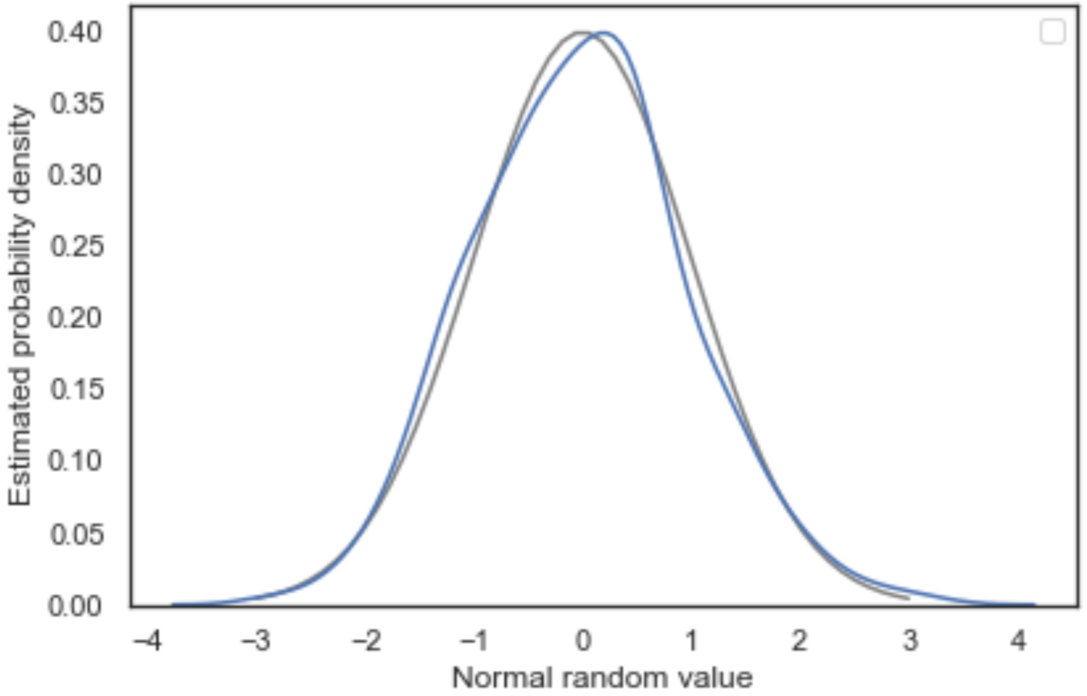
KDE plot

```
import seaborn as sns
sns.kdeplot(sample)
```



KDE and PDF

```
xs = np.linspace(-3, 3)
ys = norm.pdf(xs)
plt.plot(xs, ys, color='gray')
sns.kdeplot(sample)
```



PMF, CDF, KDE

- Use CDFs for exploration.
- Use PMFs if there are a small number of unique values.
- Use KDE if there are a lot of values.

Let's practice!

EXPLORATORY DATA ANALYSIS IN PYTHON