```
import pandas as pd
```

```
import random
```

```
L = []
for i in range(10000):
  a = random.randint(1,6)
  b = random.randint(1,6)

  L.append(a + b)
```

```
len(L)
```

```
10000
```

```
L[:5]
```

```
[7, 10, 2, 10, 6]
```

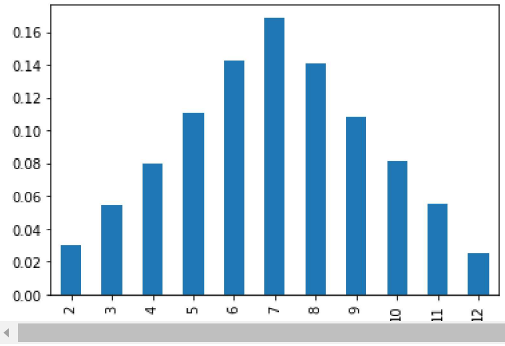
```
s = (pd.Series(L).value_counts()/pd.Series(L).value_counts().sum()).sort_index()
```

```
import numpy as np
np.cumsum(s)
```

```
2     0.0304
3     0.0852
4     0.1651
5     0.2762
6     0.4192
7     0.5879
8     0.7289
9     0.8375
10    0.9190
11    0.9743
12    1.0000
dtype: float64
```

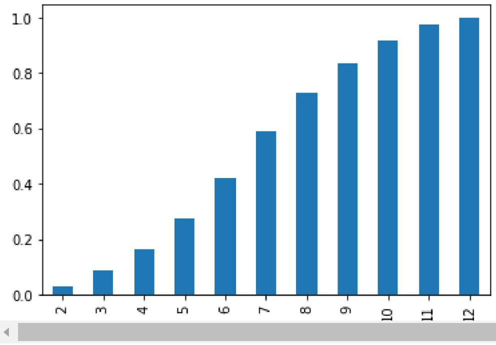
```
s.plot(kind='bar')
```

```
<AxesSubplot:>
```



```
np.cumsum(s).plot(kind='bar')
```

```
<AxesSubplot:>
```



Parametric Density Estimation

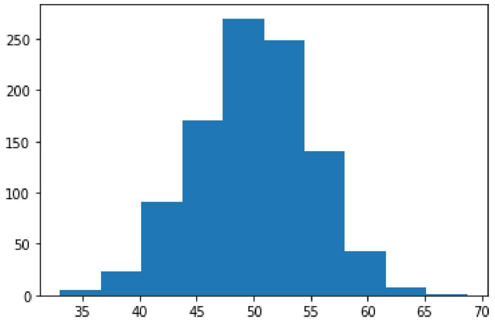
```
import matplotlib.pyplot as plt
import numpy as np
from numpy.random import normal

sample = normal(loc=50, scale=5,size=1000)
```

```
sample.mean()
```

```
50.03089373243247
```

```
# plot histogram to understand the distribution of data
plt.hist(sample,bins=10)
```

```
(array([  5.,  23.,  91., 170., 270., 249., 140.,  43.,   8.,   1.]),
 array([33.05302216, 36.61930009, 40.18557801, 43.75185594, 47.31813387,
        50.8844118 , 54.45068973, 58.01696766, 61.58324559, 65.14952352,
        68.71580144]),
 <BarContainer object of 10 artists>)
```



```
# calculate sample mean and sample std dev
sample_mean = sample.mean()
sample_std = sample.std()
```

```
# fit the distribution with the above parameters

from scipy.stats import norm
dist = norm(60, 12)
```

```
values = np.linspace(sample.min(),sample.max(),100)
```

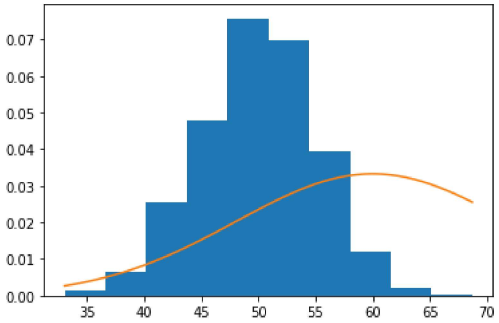
```
sample.max()
```

```
68.71580144383803
```

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

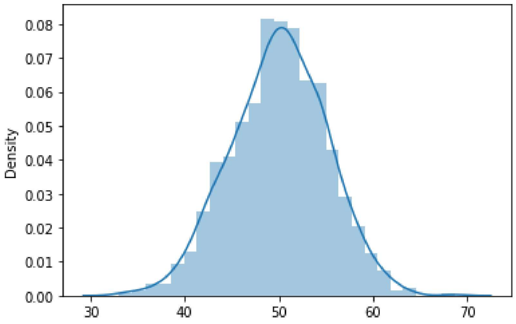
```
# plot the histogram and pdf
plt.hist(sample,bins=10,density=True)
plt.plot(values,probabilities)
```

```
[<matplotlib.lines.Line2D at 0x7f41e713d640>]
```



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

```
/usr/local/lib/python3.9/dist-packages/seaborn/distributions.py:2619: FutureWarning: `distplot` is a deprecated function and will be rem
  warnings.warn(msg, FutureWarning)
<AxesSubplot:ylabel='Density'>
```



KDE

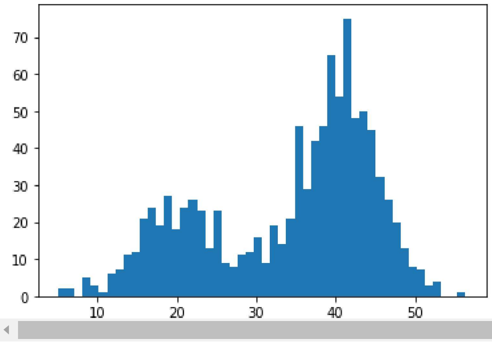
```
# generate a sample
sample1 = normal(loc=20, scale=5, size=300)
sample2 = normal(loc=40, scale=5, size=700)
sample = np.hstack((sample1, sample2))
```

```
sample
```

```
array([25.37101508, 23.17282846, 23.19266408, 21.23591442, 19.99585223,
       26.05479481, 24.87635573, 22.5707997 , 24.53847252, 17.19733365,
       15.62925657, 18.86415649, 19.00022272, 18.59746903, 13.39775335,
       14.44637505, 16.50879004, 22.05320241, 16.91522729, 24.87998461,
       25.38985776, 27.60649904, 20.60436429, 20.51424303, 16.82990467,
       16.72124091, 15.82440899, 19.21818665, 22.43898545, 19.05112432,
       21.23007028, 22.35301007, 14.54516627, 13.68508601, 21.93909454,
       16.50490442, 18.52882481, 22.6361671 , 22.06974787, 22.80384678,
       16.02872418, 14.45938868, 19.39410043, 23.67978479, 30.33476097,
       17.75143584, 18.41254234, 22.79632257, 14.69758962, 24.0195535 ,
       19.72296554, 18.34844409, 22.4203516 , 17.6989518 , 23.95721438,
       22.29306234, 13.23571722, 17.7655658 , 26.32972605, 15.46106693,
       21.51581129, 22.84867011, 19.72513232, 24.73696242, 21.79223535,
        8.22274972, 15.86279733, 20.42353259, 25.27510655, 11.79336859,
       30.98321687, 25.27164993, 20.99753848, 26.67515231, 24.7271303 ,
       21.63150605, 24.60940313, 23.30878058, 28.05151889, 21.17020424,
       16.78264853, 14.12669305, 11.80739263,  5.05186622, 21.39726354,
       15.61932563, 16.59017009, 20.04965707, 23.11173277, 14.57859971,
       26.12455937, 26.26216464, 21.22182602,  8.85059506, 17.19364332,
       22.29817619, 19.17735695, 14.89720989, 17.36085747, 17.1183832 ,
       20.28324735, 27.05545087, 21.09241359, 21.88932091, 24.12244744,
       14.2880955 , 22.71175357, 16.03407841, 21.16002229, 17.25168091,
       12.35880113, 17.99983483, 17.76747721, 19.64827737, 20.868073  ,
       22.56885729, 20.10785454, 21.38082444, 21.90100254, 29.08848779,
       24.74629218, 13.50222425, 23.88186078, 21.85681186,  7.02675044,
        9.90320294, 18.69286498,  9.15045863, 25.57540596, 14.18278419,
       21.73176782, 23.1606919 , 17.90438272, 24.96182011, 16.57844909,
        6.36002823, 24.78961174, 11.86592833, 15.3407537 , 25.08009891,
       11.34006611, 22.97390008, 28.27499148, 15.55016342, 16.34681504,
       23.50536342, 22.29213397, 20.23312824, 23.11627322, 20.61688122,
        9.57667934, 26.78338359, 21.10275124, 32.10644825, 17.09923323,
       27.98466852, 20.32323469, 14.10503853, 18.12318603, 17.43011757,
       21.07648969, 18.67818737, 17.81757834,  9.30223607, 18.48552739,
       18.70204873, 25.68824933, 19.05019273, 16.39032404, 26.08934326,
       20.17006534,  9.12645349,  5.89423269, 16.35405488, 23.27501581,
       16.82084574, 29.74043569, 20.18629982, 18.24601079, 23.18352563,
       18.90359697,  8.63121699, 18.2930157 , 11.5290024 , 16.39303508,
       18.92919987, 14.57333024, 19.52109113, 17.53375919, 25.87628953,
       23.55675677, 11.33288285, 16.74051487, 18.46771288, 28.79462526,
       21.37614571, 28.04604062, 26.34210096, 24.68440547, 28.47095162,
       15.94117231, 19.35134081, 14.5701244 , 15.71938091, 23.60508574,
       15.46047783, 19.90686679, 12.55894732, 15.46622532, 21.4263793 ,
       18.17069968, 27.28927082, 12.54958259, 14.91019411, 24.65403664,
       18.00419714, 14.41376697, 23.77078508, 21.75974396, 20.89187443,
       19.42516299, 21.89752599, 20.93304196, 23.90513584, 21.43002363,
       18.52895782, 16.71201353, 13.67725951, 18.30366448, 23.03085079,
       19.33705698, 25.27806926, 20.11656414, 25.26448695, 33.38601471,
        13.96621707, 15.89733303, 20.83892597, 16.27569864, 28.68295975,
        25.75488628, 22.939394  , 25.45781151, 20.29437577, 27.07100449,
        24.42457018, 22.31363222, 22.7558104 , 22.8726237 , 18.47619897,
        20.99851326, 18.91547169, 28.88932465, 19.45021749, 18.93285244,
        21.69607542, 15.77282225, 15.68924639, 15.57018335, 28.02897941,
        17.99334961, 23.86003035, 14.18082463, 16.93595427, 19.2814389 ,
        22.15848311, 25.4594286 , 22.50721518, 14.13010351, 25.03236024,
        28.84136287, 24.17320625, 15.77171299, 19.78258207, 13.21617116,
        22.33858876, 13.2282553 , 18.63374997, 16.51538315, 21.46124808,
        12.29998606, 10.49452423, 21.90382576, 18.08726767, 16.61968217,
```

```
# plot histogram bins=50
plt.hist(sample,bins=50)
```

```
(array([ 2.,  2.,  0.,  5.,  3.,  1.,  6.,  7., 11., 12., 21., 24., 19.,
        27., 18., 24., 26., 23., 13., 23.,  9.,  8., 11., 12., 16.,  9.,
        19., 14., 21., 46., 29., 42., 46., 65., 54., 75., 48., 50., 45.,
        32., 26., 20., 13.,  8.,  7.,  3.,  4.,  0.,  0.,  1.]),
 array([ 5.05186622,  6.07863826,  7.1054103 ,  8.13218234,  9.15895438,
        10.18572641, 11.21249845, 12.23927049, 13.26604253, 14.29281457,
        15.31958661, 16.34635865, 17.37313068, 18.39990272, 19.42667476,
        20.4534468 , 21.48021884, 22.50699088, 23.53376291, 24.56053495,
        25.58730699, 26.61407903, 27.64085107, 28.66762311, 29.69439514,
        30.72116718, 31.74793922, 32.77471126, 33.8014833 , 34.82825534,
        35.85502737, 36.88179941, 37.90857145, 38.93534349, 39.96211553,
        40.98888757, 42.0156596 , 43.04243164, 44.06920368, 45.09597572,
        46.12274776, 47.1495198 , 48.17629184, 49.20306387, 50.22983591,
        51.25660795, 52.28337999, 53.31015203, 54.33692407, 55.3636961 ,
        56.39046814]),
 <BarContainer object of 50 artists>)
```



```
from sklearn.neighbors import KernelDensity

model = KernelDensity(bandwidth=5, kernel='gaussian')

# convert data to a 2D array
sample = sample.reshape((len(sample), 1))

model.fit(sample)
```

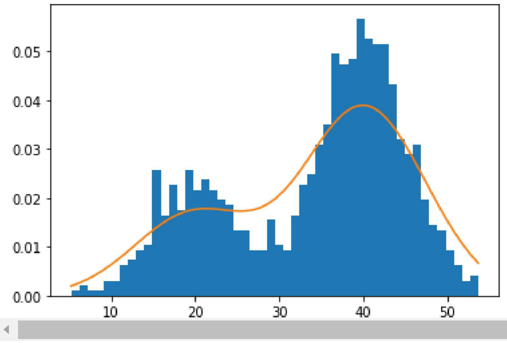
```
▾      KernelDensity
KernelDensity(bandwidth=5)
```

```
values = np.linspace(sample.min(),sample.max(),100)
values = values.reshape((len(values), 1))
```

```
probabilities = model.score_samples(values)
probabilities = np.exp(probabilities)
```

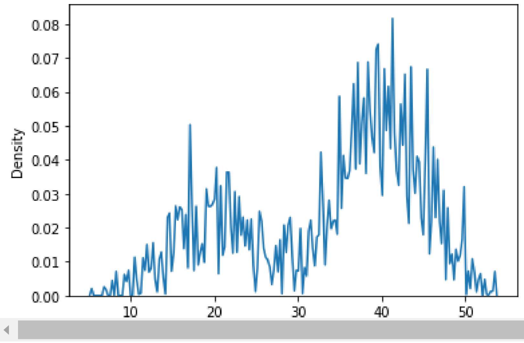
`score_samples(values)` returns the log-density estimate of the input samples values. This is because the `score_samples()` method of the KernelDensity class returns the logarithm of the probability density estimate rather than the actual probability density estimate.

```
plt.hist(sample, bins=50, density=True)
plt.plot(values[:], probabilities)
plt.show()
```



```
sns.kdeplot(sample.reshape(1000),bw_adjust=0.02)
```

```
<AxesSubplot:ylabel='Density'>
```



```
import seaborn as sns
```

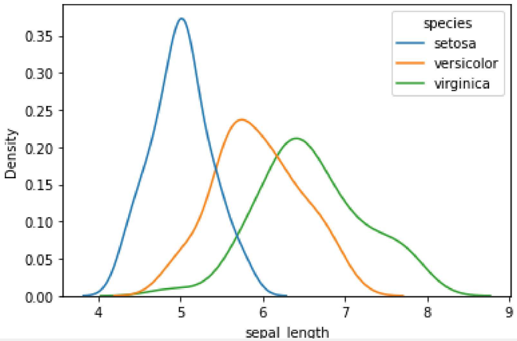
```
df = sns.load_dataset('iris')
```

```
df.head()
```

	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	setosa
1	4.9	3.0	1.4	0.2	setosa
2	4.7	3.2	1.3	0.2	setosa
3	4.6	3.1	1.5	0.2	setosa
4	5.0	3.6	1.4	0.2	setosa

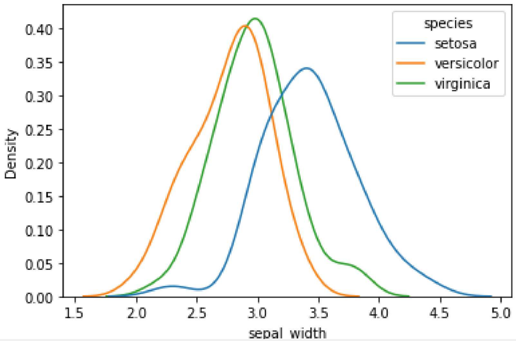
```
sns.kdeplot(data=df,x='sepal_length',hue='species')
```

```
<Axes: xlabel='sepal_length', ylabel='Density'>
```



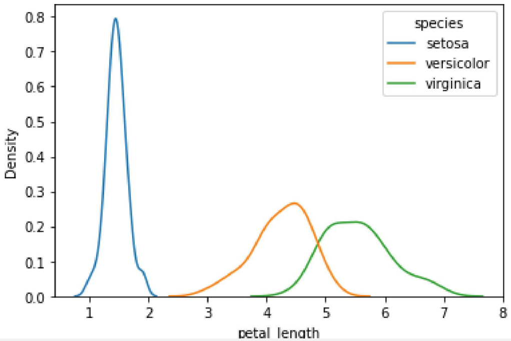
```
sns.kdeplot(data=df,x='sepal_width',hue='species')
```

```
<Axes: xlabel='sepal_width', ylabel='Density'>
```



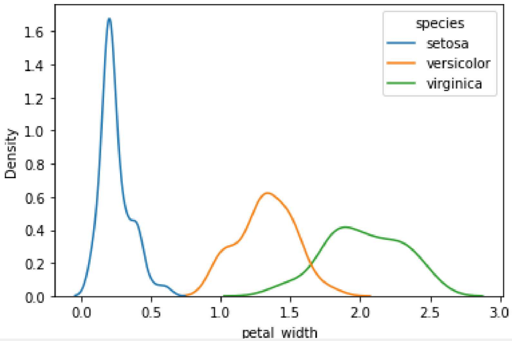
```
sns.kdeplot(data=df,x='petal_length',hue='species')
```

```
<Axes: xlabel='petal_length', ylabel='Density'>
```



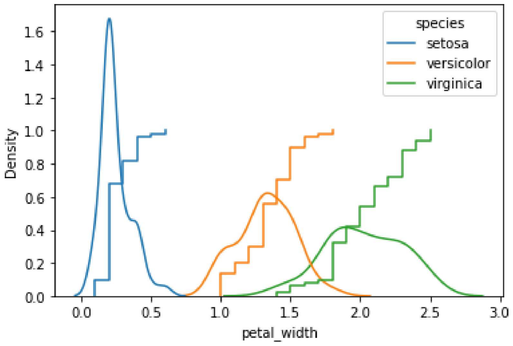
```
sns.kdeplot(data=df,x='petal_width',hue='species')
```

```
<Axes: xlabel='petal_width', ylabel='Density'>
```



```
sns.kdeplot(df['petal_width'],hue=df['species'])
sns.ecdfplot(data=df,x='petal_width',hue='species')
```

```
<AxesSubplot:xlabel='petal_width', ylabel='Density'>
```



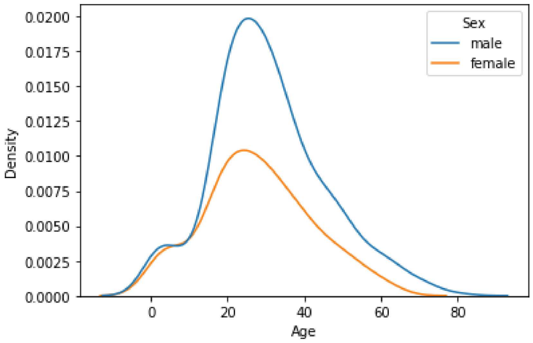
```
titanic = pd.read_csv('https://raw.githubusercontent.com/datasciencedojo/datasets/master/titanic.csv')
```

```
titanic.head()
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
				Futrelle, Mrs. Jacques Heath (Lily May								

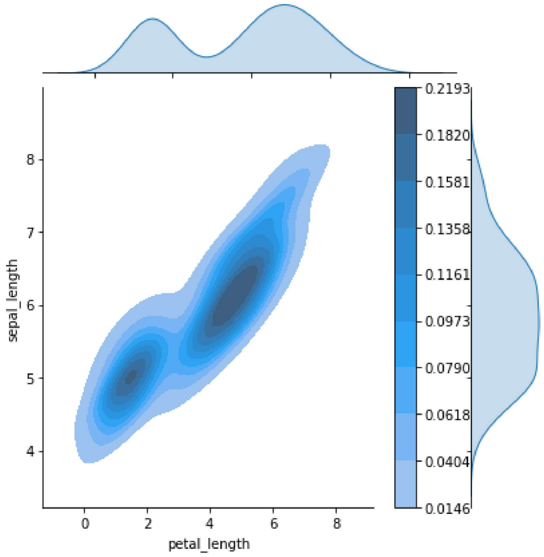
```
# code here
sns.kdeplot(data=titanic,x='Age',hue='Sex')
```

```
<Axes: xlabel='Age', ylabel='Density'>
```



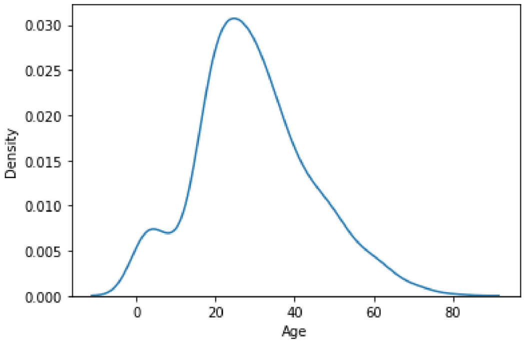
```
 sns.jointplot(data=df, x="petal_length", y="sepal_length", kind="kde",fill=True,cbar=True)
```

```
<seaborn.axisgrid.JointGrid at 0x7f41e974e7c0>
```



```
sns.kdeplot(titanic['Age'])
```

```
<Axes: xlabel='Age', ylabel='Density'>
```



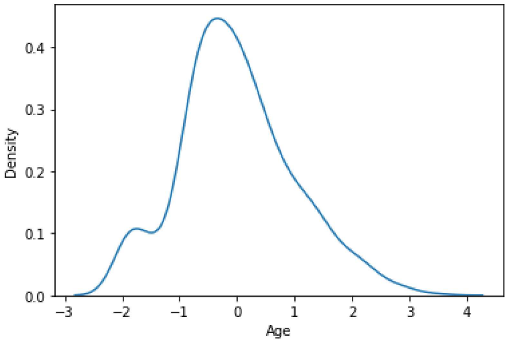
```
titanic['Age'].mean()
```

```
29.69911764705882
```

```
x = (titanic['Age'] - titanic['Age'].mean())/titanic['Age'].std()
```

```
sns.kdeplot(x)
```

```
<Axes: xlabel='Age', ylabel='Density'>
```



```
x.mean()
```

```
2.338621049070358e-16
```

```
x.std()
```

```
1.0
```

```
titanic['Age'].skew()
```

```
0.38910778230082704
```

```
titanic['Age'].mean() + 3*titanic['Age'].std()
```

```
73.27860964406094
```

```
titanic['Age'].mean() - 3*titanic['Age'].std()
```

```
-13.880374349943303
```

```
titanic[titanic['Age'] > 73]
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
630	631	1	1	Barkworth, Mr. Algernon Henry Wilson	male	80.0	0	0	27042	30.000	A23	S
851	852	0	3	Svensson, Mr. Johan	male	74.0	0	0	347060	7.775	NaN	S

```
titanic['Age'].max()
```

```
80.0
```