```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
```

```
%matplotlib inline
```

```
df = pd.DataFrame({
    'movie': ['A', 'B', 'A', 'C', 'E', 'B', 'E', 'A', 'I', 'C', 'I'],
    'rating': [9,7,9,5,9,3,2,9,6,9,9]

})
```

```
df['rating'].mean()
```

```
6.636363636363637
```

```
df.rating.mean()
```

```
6.636363636363637
```

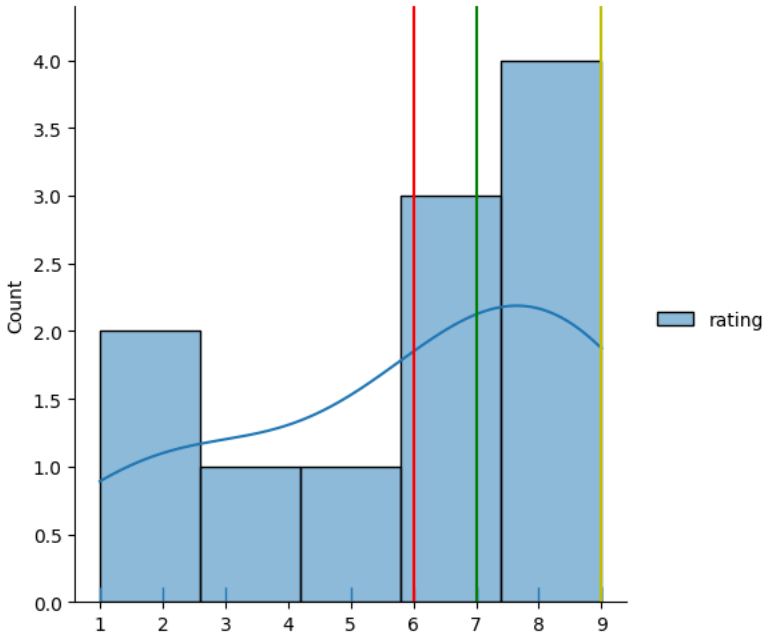
```
#Mean is the average of a dataset's column values
```

```
#Median is the middle number when the values are sorted
```

```
#Mode is the most frequent value
```

```
sns.displot(df, kde=True, rug=True)
plt.axvline(np.mean(df.rating), color='r', linestyle='-')
plt.axvline(np.median(df.rating), color='g', linestyle='-')
plt.axvline(df.rating.mode().values[0], color='y', linestyle='-')
```

```
<matplotlib.lines.Line2D at 0x7fe21c240220>
```



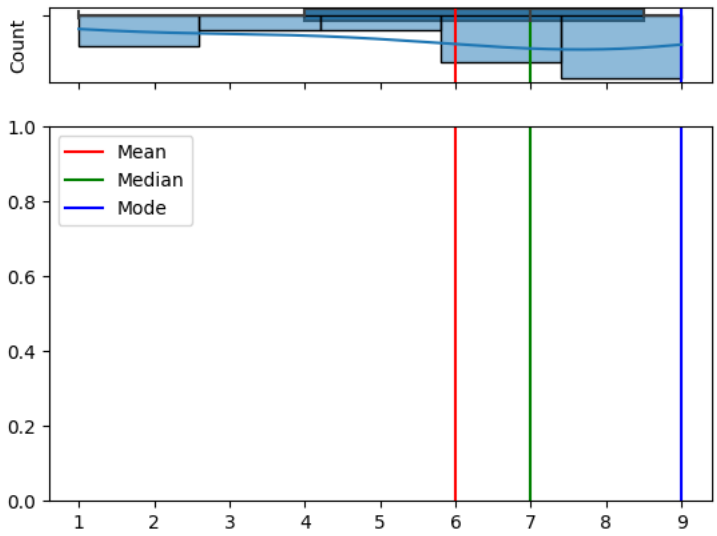

```
f, (ax_box, ax_hist) = plt.subplots(2, sharex=True,
                                    gridspec_kw={'height_ratios':(0.2, 1)})
mean = np.mean(df.rating)
median = np.median(df.rating)
mode = df.rating.mode().values[0]

sns.boxplot(data=df, x='rating', ax=ax_box)
ax_box.axvline(mean, color='r', linestyle='-')
ax_box.axvline(median, color='g', linestyle='-')
ax_box.axvline(mode, color='b', linestyle='-')

sns.histplot(data=df, x='rating', ax=ax_box, kde= True)
```

```
ax_hist.axvline(mean, color='r', linestyle='-', label="Mean")
ax_hist.axvline(median, color='g', linestyle='-', label="Median")
ax_hist.axvline(mode, color='b', linestyle='-', label="Mode")

ax_hist.legend()
ax_box.set(xlabel='')
plt.show()
```




```
df.rating.var()
```

```
8.4
```

```
df.rating.std()
```

```
2.898275349237888
```

```
#The lower the standard deviation, means the values are distributed closely to the mean
```

```
mean = df.groupby(['movie'])['rating'].mean()
std = df.groupby(['movie'])['rating'].std()
```

```
mean
```

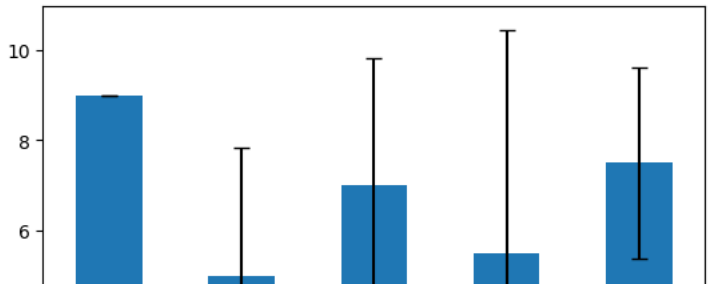
```
movie
A    9.0
B    5.0
C    7.0
E    5.5
I    7.5
Name: rating, dtype: float64
```

```
std
```

```
movie
A    0.000000
B    2.828427
C    2.828427
E    4.949747
I    2.121320
Name: rating, dtype: float64
```

```
fig, ax = plt.subplots()
mean.plot.bar(yerr=std, ax=ax, capsize=4)
```

```
<Axes: xlabel='movie'>
```




```
df1 = pd.DataFrame({'pop_sample':range(20)})
```

```
df1.sample(5).mean()
```

```
pop_sample    9.2
dtype: float64
```



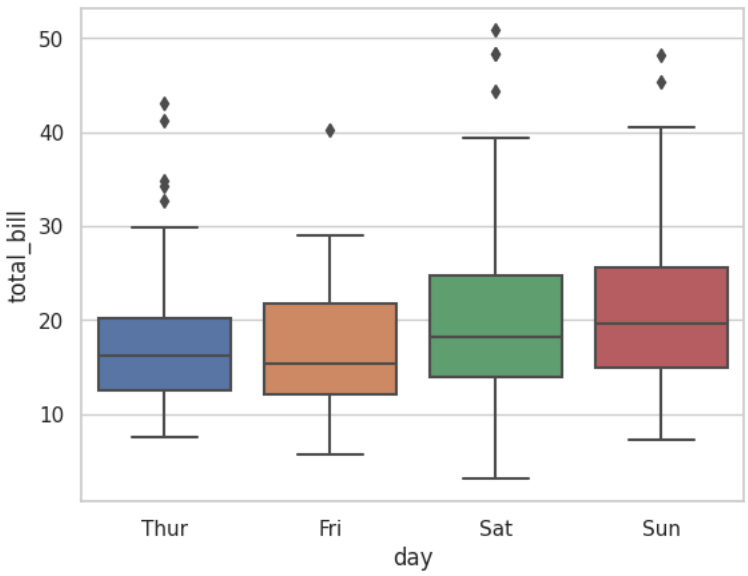
```
df1.mean()
```

```
pop_sample    9.5
dtype: float64
```

```
from scipy import stats
stats.sem(df1)
```

```
array([1.32287566])
```

```
df2 = sns.load_dataset('tips')
sns.set_theme(style='whitegrid')
ax = sns.boxplot(x='day', y='total_bill', data=df2)
```




```
ax = sns.boxplot(x='day', y='total_bill', data=df2)
ax = sns.swarmplot(x='day', y='total_bill', data=df2, color='0.25')
```


```
print(df2['total_bill'].quantile([0.05, 0.25, 0.5, 0.75]))
```

```
0.05     9.5575
0.25    13.3475
0.50    17.7950
0.75    24.1275
Name: total_bill, dtype: float64
```

```
print(df2['total_bill'].quantile(0.75)-df2['total_bill'].quantile(0.25))
```

```
10.779999999999998
```



```
#Co variance is when two variables varies with each other
```

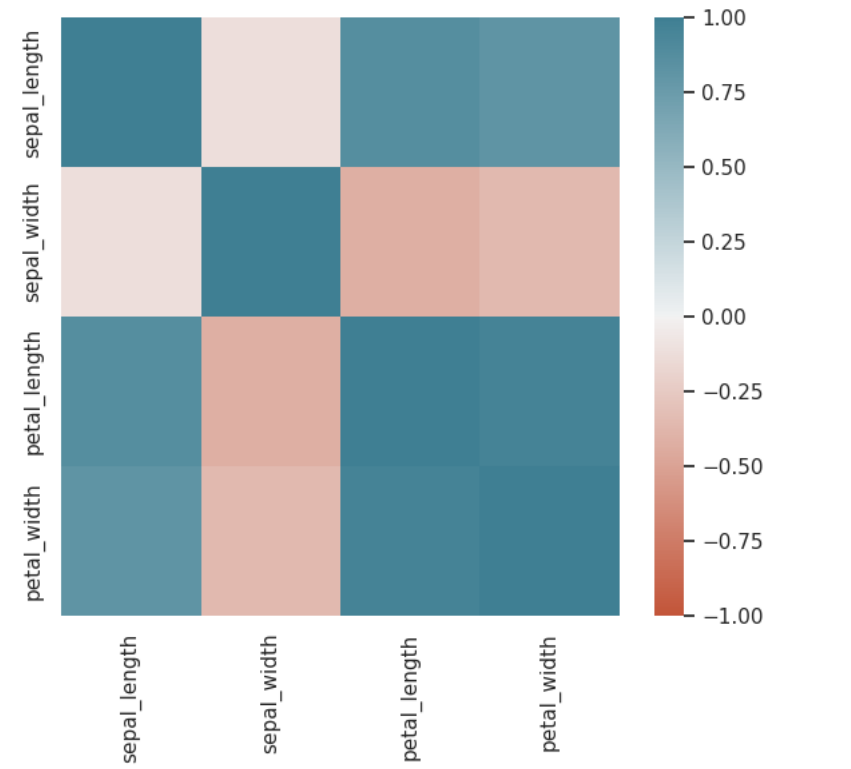
day

```
#Co relation is when change in one variable causes a change in the other variable and vice versa


df3 = sns.load_dataset('iris')
fig, ax= plt.subplots(figsize=(6,6))

ax = sns.heatmap(df3.corr(), vmin=-1, vmax=1,
                 cmap=sns.diverging_palette(20,220, as_cmap=True), ax=ax)
plt.tight_layout()
plt.show()
```

```
<ipython-input-80-5de20dfd8483>:4: FutureWarning: The default value of nume
  ax = sns.heatmap(df3.corr(), vmin=-1, vmax=1,
```

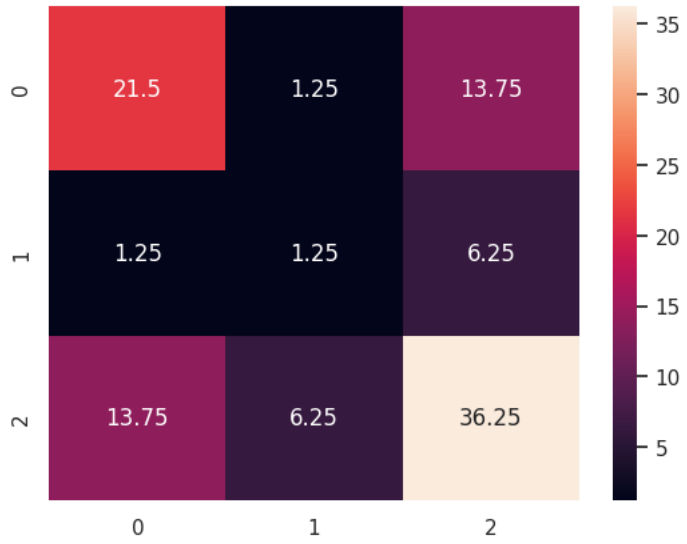



```
a = [11,12,22,11]
b=[7,8,9,10]
c=[10,11,22,23]
arr = np.array([a,b,c])

cov_matrix = np.cov(arr, bias=True)
cov_matrix
```

```
array([[21.5 ,  1.25, 13.75],
       [ 1.25,  1.25,  6.25],
       [13.75,  6.25, 36.25]])
```

```
sns.heatmap(cov_matrix, annot=True, fmt='g')
plt.show()
```

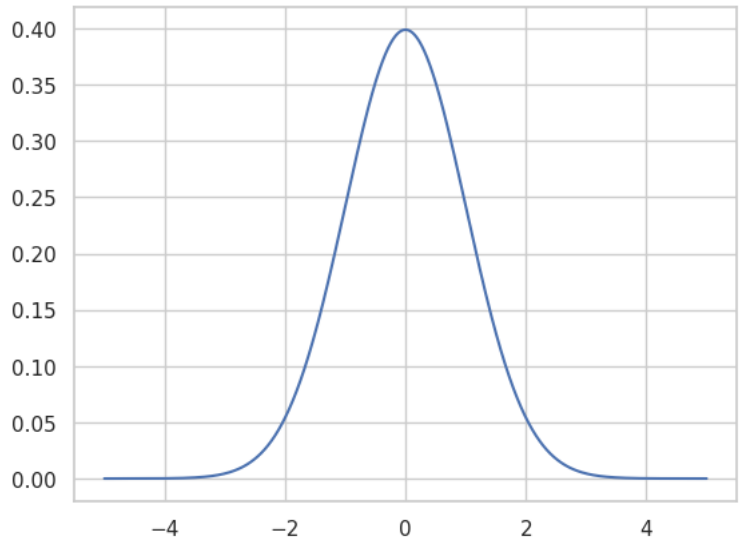



```
df.skew()
```

```
df.kurtosis()
```

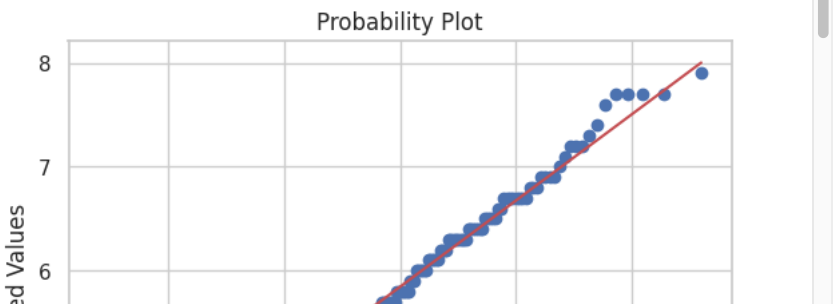
```
<ipython-input-106-c7edf97eb14c>:1: FutureWarning: The default value of numeric_only in DataFrame.kurt is deprecated. In
  df.kurtosis()
rating   -0.447619
dtype: float64
```

```
norm1 = np.arange(-5,5,0.001)
mean = 0.0
std = 1.0
pdf = stats.norm.pdf(norm1, mean, std)
plt.plot(norm1, pdf)
plt.show()
```



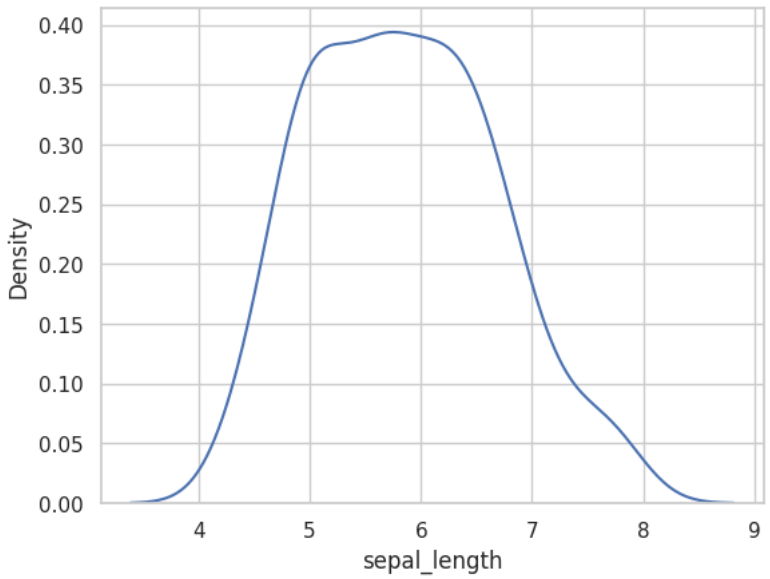

```
import pylab
stats.probplot(df3.sepal_length, plot = pylab)
```

```
((array([-2.60376328, -2.283875  , -2.1005573 , -1.96875864, -1.86428437,
         -1.77691182, -1.70131573, -1.63435332, -1.57400778, -1.51890417,
         -1.46806125, -1.42075308, -1.37642684, -1.33465133, -1.29508341,
         -1.25744533, -1.22150891, -1.18708433, -1.15401181, -1.12215558,
         -1.0913992 , -1.06164202, -1.03279638, -1.00478546, -0.97754152,
         -0.95100448, -0.92512081, -0.89984257, -0.87512664, -0.85093408,
         -0.8272296 , -0.80398107, -0.78115919, -0.75873709, -0.73669013,
         -0.71499557, -0.69363244, -0.67258128, -0.65182406, -0.63134396,
         -0.61112532, -0.59115349, -0.57141472, -0.55189613, -0.53258558,
         -0.51347162, -0.49454346, -0.47579085, -0.45720409, -0.43877397,
         -0.4204917 , -0.40234892, -0.38433762, -0.36645016, -0.3486792 ,
         -0.33101768, -0.31345882, -0.29599609, -0.27862316, -0.26133393,
         -0.24412247, -0.22698303, -0.20991002, -0.19289797, -0.17594158,
         -0.15903562, -0.142175  , -0.12535471, -0.10856981, -0.09181544,
         -0.07508681, -0.05837916, -0.0416878 , -0.02500804, -0.00833524,
          0.00833524,  0.02500804,  0.0416878 ,  0.05837916,  0.07508681,
          0.09181544,  0.10856981,  0.12535471,  0.142175  ,  0.15903562,
          0.17594158,  0.19289797,  0.20991002,  0.22698303,  0.24412247,
          0.26133393,  0.27862316,  0.29599609,  0.31345882,  0.33101768,
          0.3486792 ,  0.36645016,  0.38433762,  0.40234892,  0.4204917 ,
          0.43877397,  0.45720409,  0.47579085,  0.49454346,  0.51347162,
          0.53258558,  0.55189613,  0.57141472,  0.59115349,  0.61112532,
          0.63134396,  0.65182406,  0.67258128,  0.69363244,  0.71499557,
          0.73669013,  0.75873709,  0.78115919,  0.80398107,  0.8272296 ,
          0.85093408,  0.87512664,  0.89984257,  0.92512081,  0.95100448,
          0.97754152,  1.00478546,  1.03279638,  1.06164202,  1.0913992 ,
          1.12215558,  1.15401181,  1.18708433,  1.22150891,  1.25744533,
          1.29508341,  1.33465133,  1.37642684,  1.42075308,  1.46806125,
          1.51890417,  1.57400778,  1.63435332,  1.70131573,  1.77691182,
          1.86428437,  1.96875864,  2.1005573 ,  2.283875  ,
2.60376328]),
  array([4.3, 4.4, 4.4, 4.4, 4.5, 4.6, 4.6, 4.6, 4.6, 4.7, 4.7, 4.8, 4.8,
         4.8, 4.8, 4.8, 4.9, 4.9, 4.9, 4.9, 4.9, 4.9, 5. , 5. , 5. , 5. ,
         5. , 5. , 5. , 5. , 5. , 5. , 5. , 5.1, 5.1, 5.1, 5.1, 5.1, 5.1, 5.1,
         5.1, 5.1, 5.2, 5.2, 5.2, 5.2, 5.3, 5.4, 5.4, 5.4, 5.4, 5.4, 5.4,
         5.5, 5.5, 5.5, 5.5, 5.5, 5.5, 5.5, 5.6, 5.6, 5.6, 5.6, 5.6, 5.6,
         5.7, 5.7, 5.7, 5.7, 5.7, 5.7, 5.7, 5.7, 5.8, 5.8, 5.8, 5.8, 5.8,
         5.8, 5.8, 5.9, 5.9, 5.9, 6. , 6. , 6. , 6. , 6. , 6. , 6.1, 6.1,
         6.1, 6.1, 6.1, 6.1, 6.2, 6.2, 6.2, 6.2, 6.3, 6.3, 6.3, 6.3, 6.3,
         6.3, 6.3, 6.3, 6.3, 6.4, 6.4, 6.4, 6.4, 6.4, 6.4, 6.4, 6.5, 6.5,
         6.5, 6.5, 6.5, 6.6, 6.6, 6.7, 6.7, 6.7, 6.7, 6.7, 6.7, 6.7, 6.7,
         6.8, 6.8, 6.8, 6.9, 6.9, 6.9, 6.9, 7. , 7.1, 7.2, 7.2, 7.2, 7.3,
         7.4, 7.6, 7.7, 7.7, 7.7, 7.7, 7.9])),
 (0.8288552848682113, 5.843333333333334, 0.9900008795634082))
```



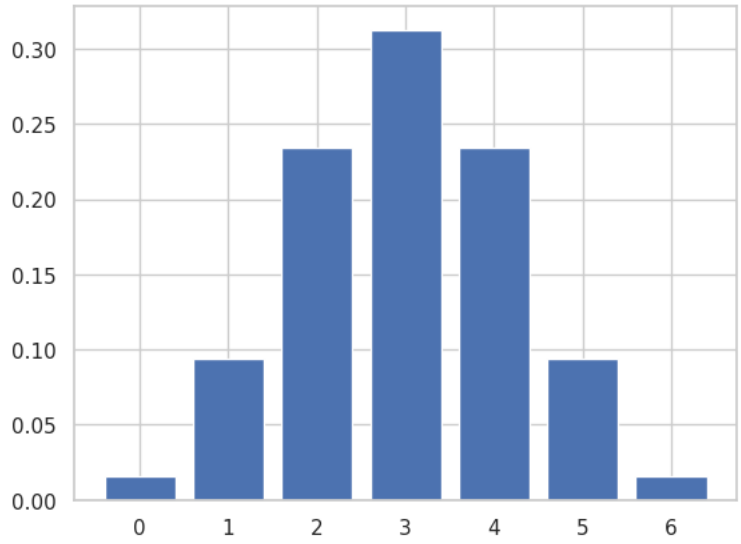

```
sns.kdeplot(df3.sepal_length)
```

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

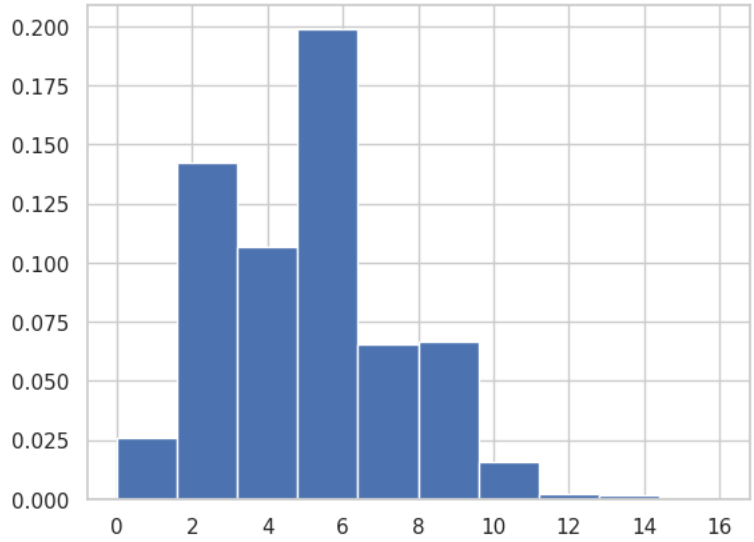


```
#Binomial distribution is discrete distribution


from scipy.stats import binom
n=6
p=0.5
r_value = list(range(n+1))
dist = [binom.pmf(r, n, p) for r in r_value]
plt.bar(r_value, dist)
plt.show()
```



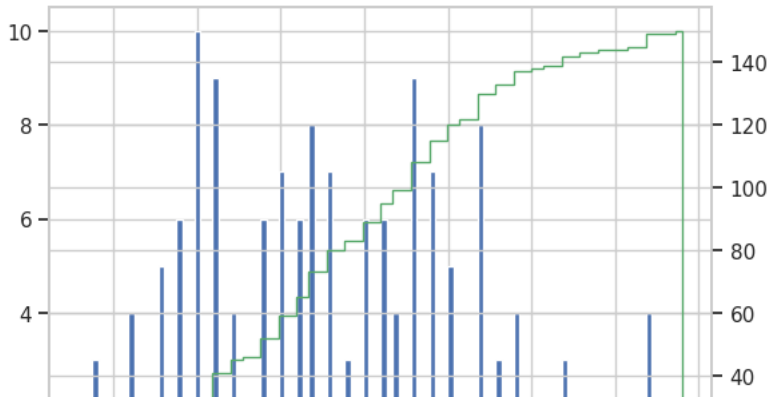

```
s = np.random.poisson(5, 10000)
count, bins, ignored = plt.hist(s, 10, density=True)
plt.show()
```




```
import statsmodels.stats.api as sms
sms.DescrStatsW(df3.sepal_length).tconfint_mean()
```

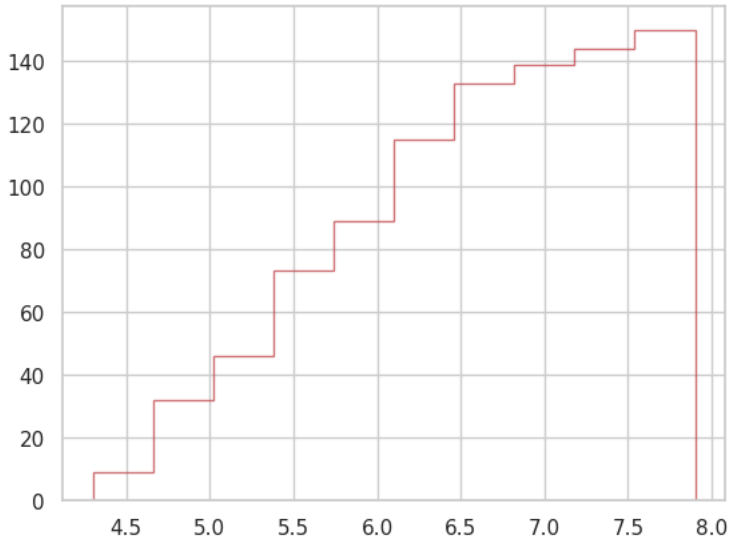
```
(5.709732481507366, 5.976934185159301)
```

```
fig, ax = plt.subplots()
ax2 = ax.twinx()
n, bins, patches = ax.hist(df3.sepal_length, bins=100)
n, bins, patches = ax2.hist(df3.sepal_length, cumulative=1, histtype='step',
                            bins=100, color='g')
```

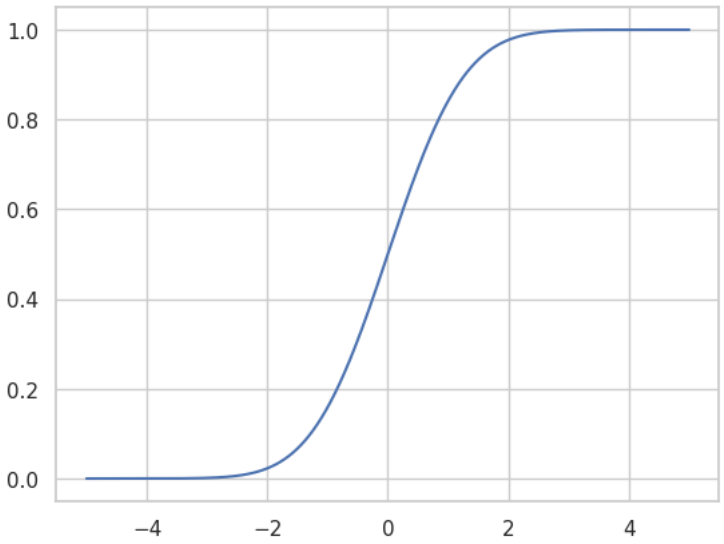



```
plt.hist(df3.sepal_length, cumulative=True, label='CDF', histtype='step',
         alpha=0.8, color='r')
```

```
(array([  9.,  32.,  46.,  73.,  89., 115., 133., 139., 144., 150.]),
 array([4.3 , 4.66, 5.02, 5.38, 5.74, 6.1 , 6.46, 6.82, 7.18, 7.54, 7.9
]),
 [<matplotlib.patches.Polygon at 0x7fe21acb5510>])
```




```
cdf = stats.norm.cdf(norm1)
plt.plot(norm1, cdf)
plt.show()
```




```
ax = sns.displot(df3.sepal_length)
```