**How to Plot a Histogram in Python using Matplotlib**

import matplotlib.pyplot as plt

x = [value1, value2, value3,....]

plt.hist(x, bins = number of bins)

plt.show()

Still not sure how to plot a histogram in Python?

If so, I’ll show you the full steps to plot a histogram in Python using a simple example.

**Steps to plot a histogram in Python using Matplotlib**

**Step 1: Install the Matplotlib package**

pip install matplotlib

**Step 2: Collect the data for the histogram**

For example, let’s say that you have the following data about the age of 100 individuals:

|  |
| --- |
| **Age** |
| 1,1,2,3,3,5,7,8,9,10, 10,11,11,13,13,15,16,17,18,18, 18,19,20,21,21,23,24,24,25,25, 25,25,26,26,26,27,27,27,27,27, 29,30,30,31,33,34,34,34,35,36, 36,37,37,38,38,39,40,41,41,42, 43,44,45,45,46,47,48,48,49,50, 51,52,53,54,55,55,56,57,58,60, 61,63,64,65,66,68,70,71,72,74, 75,77,81,83,84,87,89,90,90,91 |

Later you’ll see how to plot the histogram based on the above data.

**Step 3: Determine the number of bins**

Next, determine the number of bins to be used for the histogram.

For simplicity, let’s set the number of bins to 10. At the end of this guide, I’ll show you another way to derive the bins.

**Step 4: Plot the histogram in Python using matplotlib**

You’ll now be able to plot the histogram based on the template that you saw at the beginning of this guide:

import matplotlib.pyplot as plt

x = [value1, value2, value3,....]

plt.hist(x, bins = number of bins)

plt.show()

And for our example, this is the complete Python code after applying the above template:

import matplotlib.pyplot as plt

x = [1,1,2,3,3,5,7,8,9,10,

10,11,11,13,13,15,16,17,18,18,

18,19,20,21,21,23,24,24,25,25,

25,25,26,26,26,27,27,27,27,27,

29,30,30,31,33,34,34,34,35,36,

36,37,37,38,38,39,40,41,41,42,

43,44,45,45,46,47,48,48,49,50,

51,52,53,54,55,55,56,57,58,60,

61,63,64,65,66,68,70,71,72,74,

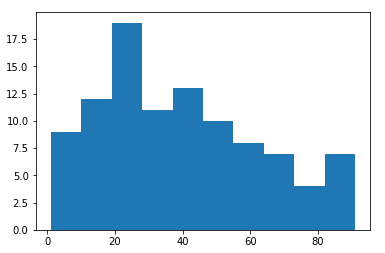
75,77,81,83,84,87,89,90,90,91

]

plt.hist(x, bins=10)

plt.show()

Run the code, and you’ll get the histogram below:



That’s it! You should now have your histogram in Python.

If needed, you can further [style your histogram](https://matplotlib.org/users/style_sheets.html). One way to style your histogram is by adding this syntax towards the end of the code:

plt.style.use('ggplot')

And for our example, the code would look like this:

import matplotlib.pyplot as plt

x = [1,1,2,3,3,5,7,8,9,10,

10,11,11,13,13,15,16,17,18,18,

18,19,20,21,21,23,24,24,25,25,

25,25,26,26,26,27,27,27,27,27,

29,30,30,31,33,34,34,34,35,36,

36,37,37,38,38,39,40,41,41,42,

43,44,45,45,46,47,48,48,49,50,

51,52,53,54,55,55,56,57,58,60,

61,63,64,65,66,68,70,71,72,74,

75,77,81,83,84,87,89,90,90,91

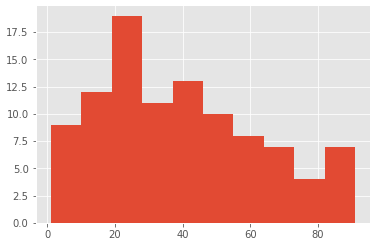
]

plt.style.use('ggplot')

plt.hist(x, bins=10)

plt.show()

Run the code, and you’ll get this styled histogram:



Just by looking at the histogram, you may have noticed the positive Skewness.

You can derive the skew in Python by using the [scipy](https://docs.scipy.org/doc/scipy/reference/stats.html) library.

This is the code that you can use to derive the skew for our example:

from scipy.stats import skew

x = [1,1,2,3,3,5,7,8,9,10,

10,11,11,13,13,15,16,17,18,18,

18,19,20,21,21,23,24,24,25,25,

25,25,26,26,26,27,27,27,27,27,

29,30,30,31,33,34,34,34,35,36,

36,37,37,38,38,39,40,41,41,42,

43,44,45,45,46,47,48,48,49,50,

51,52,53,54,55,55,56,57,58,60,

61,63,64,65,66,68,70,71,72,74,

75,77,81,83,84,87,89,90,90,91

]

print (skew(x))

Once you run the code in Python, you’ll get the following Skew:

*0.4575278444409153*

**Additional way to determine the number of bins**

Originally, we set the number of bins to 10 for simplicity.

Alternatively, you may derive the bins using the following formulas:

* **n** = number of observations
* **Range** = maximum value – minimum value
* **# of intervals** =  √n
* **Width of intervals** =  Range / (# of intervals)

These formulas can then be used to create the frequency table followed by the histogram.

Recall that our dataset contained the following 100 observations:

|  |
| --- |
| **Age** |
| 1,1,2,3,3,5,7,8,9,10, 10,11,11,13,13,15,16,17,18,18, 18,19,20,21,21,23,24,24,25,25, 25,25,26,26,26,27,27,27,27,27, 29,30,30,31,33,34,34,34,35,36, 36,37,37,38,38,39,40,41,41,42, 43,44,45,45,46,47,48,48,49,50, 51,52,53,54,55,55,56,57,58,60, 61,63,64,65,66,68,70,71,72,74, 75,77,81,83,84,87,89,90,90,91 |

Using our formulas:

* n = number of observations = **100**
* Range = maximum value – minimum value = 91 – 1 = **90**
* # of intervals =  √n = √100 = **10**
* Width of intervals =  Range / (# of intervals) = 90/10 = **9**

Based on this information, the frequency table would look like this:

|  |  |
| --- | --- |
| **Intervals (bins)** | **Frequency** |
| **0**-9 | 9 |
| **10**-19 | 13 |
| **20**-29 | 19 |
| **30**-39 | 15 |
| **40**-49 | 13 |
| **50**-59 | 10 |
| **60**-69 | 7 |
| **70**-79 | 6 |
| **80**-89 | 5 |
| **90**–**99** | 3 |

Note that the starting point for the first interval is 0, which is very close to the minimum observation of 1 in our dataset. If, for example, the minimum observation was 20 in another dataset, then the starting point for the first interval should be 20, rather than 0.

For the *bins*in the Python code below, you’ll need to specify the values highlighted in blue, rather than a particular number (such as 10, which we used before). Don’t forget to include the last value of 99.

This is how the Python code would look like:

import matplotlib.pyplot as plt

x = [1,1,2,3,3,5,7,8,9,10,

10,11,11,13,13,15,16,17,18,18,

18,19,20,21,21,23,24,24,25,25,

25,25,26,26,26,27,27,27,27,27,

29,30,30,31,33,34,34,34,35,36,

36,37,37,38,38,39,40,41,41,42,

43,44,45,45,46,47,48,48,49,50,

51,52,53,54,55,55,56,57,58,60,

61,63,64,65,66,68,70,71,72,74,

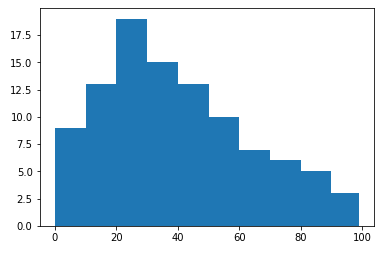
75,77,81,83,84,87,89,90,90,91

]

plt.hist(x, bins=[0,10,20,30,40,50,60,70,80,90,99])

plt.show()

Run the code, and you’ll get the following histogram:



You’ll notice that the histogram is similar to the one we saw earlier. The positive skew is also apparent.

**Post navigation**

[← Previous Post](https://datatofish.com/delete-ms-access-python/)

[Next Post →](https://datatofish.com/descriptive-statistics-pandas/)

Tutorials

* [Python Tutorials](https://datatofish.com/python-tutorials/)
* [R Tutorials](https://datatofish.com/r-tutorials/)
* [Julia Tutorials](https://datatofish.com/julia-tutorials/)
* [Batch Scripts Tutorials](https://datatofish.com/batch-scripts-tutorials/)
* [MS Access Tutorials](https://datatofish.com/ms-access-2016-tutorials/)
* [Excel Tutorials](https://datatofish.com/ms-excel-2016-tutorials/)

Copyright