CLick here to download IPYTHON notes for this chapter EDA

```
import numpy as np
# consider you have an array like this
a= [1, 2, 3, 5, 10,11, 12, 13, 14, 15, 21,23, 25, 26, 27, 29, 30,31, 35, 51]
# here we have decided to group all these numbers into 5 bins
# i.e bins = 5
# the minimum number in the array is 1
# the maximum number in the array is 51
# the width of each bin is calculated as = ((max - min) / bins)
# width of each bin = (51-1)/5 = 10
# Since we got each bin with as 10, we can choose the bin edges like this
# 1 ...... 11 ....... 21 ........ 31 ....... 41 ....... 51
# |---10---|----10----|----10-----|----10----|----10----|
# so we have found out the bin edges now
# to find thte counts we calcuate how many number of points fall into each bin
# therefore the count of a bin = number of elements of a such that left_bin_egde<=ai 5 [1,2,3,5,10]
# ii. number of elements belongs to the 2nd bin 11<=x<21 => 5 [11,12,13,14,15]
# iii. number of elements belongs to the 3rd bin 21<=x<31 => 7 [21,23,25,26,27,29,30]
# iii. number of elements belongs to the 3rd bin 21<=x<31 => 7 [21,23,25,26,27,29,30]
# iv. number of elements belongs to the 4th bin 31<=x<41 => 2 [31,35]
# v. number of elements belongs to the 5th bin 41<=x<=51 => 1 [51]

# note: from the documentation: https://docs.scipy.org/doc/numpy/reference/generated/numpy.histogram.html
# All but the last (righthand-most) bin is half-open i.e [1,2,3,4], the bins are [1,2), [2,3), [3,4]

# [1,10) = 1,2,3,4,5,6,7,8,9 means includig 1 and but not 10. its half open bracket

print('='*30, "explaining 'bin edges and counts",'='*30)
counts,bins = np.histogram(a, bins=5)

print("bin edges :",bins)
print("counts per each bin :",counts)

# density: bool, optional
# If False, the result will contain the number of samples in each bin.
# If True, the result is the value of the probability density function at the bin, normalized such that the integral over the range is 1.
# Note that the sum of the histogram values will not be equal to 1 unless bins of unity width are chosen;
# it is not a probability mass function.

# and from the source code
#if density:
#        db = np.array(np.diff(bin_edges), float)
#        return n/db/n.sum(), bin_edges

# here the n => number of elements for each bin
n = counts
# and db = difference between bin edges
db = np.array(np.diff(bins))
# n.sum() number of all the elemnts


print('='*30, "explaining 'density=True' parameter",'='*30)
print("manual calculated densities for each bin",counts/db/counts.sum())

counts, bins = np.histogram(a, bins=5, density=True)

print("bin edges :",bins)
print("counts per each bin using density=True:",counts)

print('='*30, "explaining counts/sum(counts)",'='*30)
# pleasen note that the documentation says when you have density=True,
# "that the sum of the histogram values will not be equal to 1"

# this is simple logic we used, to make the whole sum=1, we have divided each element by the number of whole elements

counts, bins = np.histogram(a, bins=5, density=True)

print("bin edges :",bins)
# sum(counts) = summ of all the elements in the counts array = [0.025 + 0.025 + 0.035 + 0.01 + 0.005] = 0.1
# counts/sum(counts) = devide every element of counts=[0.025/0.1, 0.025/0.1, 0.035/0.1, 0.01/0.1, 0.005/0.1] = [0.25 0.25 0.35 0.1  0.05]
print("counts per each bin using density=True:",counts/sum(counts))
```

The output of this above program

```
============================== explaining 'bin edges and counts ==============================
bin edges : [ 1. 11. 21. 31. 41. 51.]
counts per each bin : [5 5 7 2 1]
============================== explaining 'density=True' parameter ==============================
manual calculated densities for each bin [0.025 0.025 0.035 0.01  0.005]
bin edges : [ 1. 11. 21. 31. 41. 51.]
counts per each bin using density=True: [0.025 0.025 0.035 0.01  0.005]
============================== explaining counts/sum(counts) ==============================
bin edges : [ 1. 11. 21. 31. 41. 51.]
counts per each bin using density=True: [0.25 0.25 0.35 0.1  0.05]
```

you can find the link for this program here: https://ideone.com/IqCwsI