Lab 01 - Getting Started

Name: Nick Satriano Class: CSCI 349 - Intro to Data Mining Semester: 2023SP Instructor: Brian King

1) Did you read the syllabus? All of it? Do you understand and agree to abide by the cheating rules? Write a sentence clearly indicating your commitment to not cheating.

I have read the syllabus and I do agree with it and will abide by the cheating rules.

2) What are you hoping to get out of this course?

I am hoping to gain a lot of hands-on experience with data science tools. I'm excited to learn more about machine learning and refine my data analysis skills.

3) Print the Python version (available in sys package)

In [11]:
```
import sys
print(sys.version)
```

```
3.9.16 (main, Jan 11 2023, 16:16:36) [MSC v.1916 64 bit (AMD64)]
```

4) Create a Python list of 10000 random integers in the range 1 to 100 using the random package. Name the list x_list

In [6]:
```
import random

x_list = [random.choice(range(1, 101)) for i in range(10000)]
```

5) What is the minimum value of x_list? What is the max value? What is the mode?

In [10]:
```
from statistics import mode
print("The min of the list is: {}." .format(min(x_list)))
print("The max of the list is: {}." .format(max(x_list)))
print("The mode of the list is: {}." .format(mode(x_list)))
```

```
The min of the list is: 1.
The max of the list is: 100.
The mode of the list is: 13.
```

6) Write a function to take a list of numbers as a parameter, and return the average of the list. Then, use your function to report the average value of x_list, printed as a float with 2 places of precision.

In [7]:
```
def avg(alist):
    sum = 0
    for i in alist:
        sum += i
    ret = sum / len(alist)
    return ret
```

```
print("The average is: %.2f" % avg(x_list))
```

```
The average is: 50.94
```

7) Create a list called x_hist that represents a histogram, i.e. a distribution of the numerical data, of x_list. Each entry in x_hist should contain the frequency of data over a bin width of 10. So, for example, x_hist[0] represents the frequency of numbers between 1-10, x_hist[1] is the frequency of numbers between 11-20, and so on. Be sure to print x_hist at the end.

In [15]:
```
x_hist = [0, 0, 0, 0, 0, 0, 0, 0, 0]
for i in x_list:
    if i <= 10:
        x_hist[0] += 1
    elif 10 < i <= 20:
        x_hist[1] += 1
    elif 20 < i <= 30:
        x_hist[2] += 1
    elif 30 < i <= 40:
        x_hist[3] += 1
    elif 40 < i <= 50:
        x_hist[4] += 1
    elif 50 < i <= 60:
        x_hist[5] += 1
    elif 70 < i <= 80:
        x_hist[6] += 1
    elif 80 < i <= 90:
        x_hist[7] += 1
    elif i > 90:
        x_hist[8] += 1

print(x_hist)
```

```
[982, 974, 929, 1021, 1050, 1023, 976, 1039, 1020]
```

8) What is numpy? What are its strengths? Does it have any weaknesses?

NumPy is a library which adds support to python for large data frames and related algorithms. NumPy uses arrays to store data, which makes it great for data science. One weakness is that NumPy uses "nan" to represent "not a number" which can sometimes pose issues with data handling.

9) Import the numpy package as np and print the numpy version

In [13]:
```
import numpy as np
print(np.__version__)
```

```
1.23.5
```

10) What is the primary object type in numpy? Can it store data of different types? Discuss.

NumPy arrays are the primary object type. These arrays can store numerical data, and are much more efficient than python's built-in data structures.

11) Discuss the types of data available in numpy. You need not list every type. Just generalize, and discuss how the type system is different than the built-in types int and float in Python.

The types of data available in numpy are all numerical: booleans, integers, unsigned integers, floating point and complex. These types are not size-flexible like python's are, so they are much more efficient to use than the built-in types.

12) Research how to use numpy to compute the same result as problem 8 above.

In [14]:
```
print(np.histogram(x_list))
```

```
(array([ 982,  974,  929, 1021, 1050, 1023,  986,  976, 1039, 1020],
      dtype=int64), array([  1. ,  10.9,  20.8,  30.7,  40.6,  50.5,  60.4,  70.3,  8
0.2,
        90.1, 100. ]))
```

13) Create a numpy array from x_list. Reassign it as x_list. Show the contents of the first 20 entries.

In [17]:
```
x_list = np.array(x_list)

print(x_list[:20])
```

```
[94 18 50  2 18 85 42 69 45 41 99 13 83 61 26 32 62 24 99 49]
```

14) Redo the previous exercise, but set the data type of each value to 'float32'.

In [18]:
```
x_list = np.array(x_list, dtype=np.float32)

print(x_list[:20])
```

```
[94. 18. 50.  2. 18. 85. 42. 69. 45. 41. 99. 13. 83. 61. 26. 32. 62. 24.
 99. 49.]
```

15) Create a length 10 integer array filled with zeros.

In [22]:
```
np.zeros(10)
```

Out[22]:
```
array([0., 0., 0., 0., 0., 0., 0., 0., 0., 0.])
```

16) Create a float array of 3 rows and 4 columns, all initialized to one.

In [27]:
```
np.ones((3, 4))
```

Out[27]:
```
array([[1., 1., 1., 1.],
       [1., 1., 1., 1.],
       [1., 1., 1., 1.]])
```

17) Create an array of 20 values, evenly spaced between 1 and 3.

In [28]:
```
np.linspace(1, 3, num = 20)
```

Out[28]:
```
array([1.        , 1.10526316, 1.21052632, 1.31578947, 1.42105263,
       1.52631579, 1.63157895, 1.73684211, 1.84210526, 1.94736842,
       2.05263158, 2.15789474, 2.26315789, 2.36842105, 2.47368421,
       2.57894737, 2.68421053, 2.78947368, 2.89473684, 3.        ])
```

18) Set the numpy random seed to the value 12345, then create a 10 x 5 array of random integers on the interval [10, 50), assigned to x_mat.

In [35]:
```
np.random.seed(12345)
x_mat = np.random.randint(10, 49, size=(10, 5))
print(x_mat)
```

```
[[44 47 39 11 46]
 [47 44 39 11 24]
 [37 26 19 21 23]
 [20 27 28 48 17]
 [33 39 41 43 37]
 [33 46 10 13 15]
 [25 21 24 27 15]
 [44 18 15 44 15]
 [29 22 24 26 37]
 [48 34 42 15 20]]
```

19) Use slicing to select and show the first and last row of x_mat.

In [39]:
```
print(x_mat[0:10:9])
```

```
[[44 47 39 11 46]
 [48 34 42 15 20]]
```

20) Now, show the first, third and fifth column of x_mat.

In [41]:
```
print(x_mat[0:5:2])
```

```
[[44 47 39 11 46]
 [37 26 19 21 23]
 [33 39 41 43 37]]
```

21) Show the sum of each row.

In [44]:
```
np.sum(x_mat, axis = 1)
```

Out[44]:
```
array([187, 165, 126, 140, 193, 117, 112, 136, 138, 159])
```