Math221_Visualisation

January 21, 2018

Math 221 - Applied Statistics (Data Analysis)
Dr. Kamal Dingle (Spring 2018)
VISUALISING DATA
We start by importing the **numpy** library as "np", as a shorthand

```
In [2]: import numpy as np
```

numpy has very many numerical packages in it - we will use it all the time.
Let's test everything is working. Define some vector x

```
In [3]: x = np.array([1.5,2.5,3.1,1.0,3.4,5.4,5.2,4.9,2.0,1.1,1.2,5,1.0])
```

Let's add up all the values in x, just for fun

```
In [4]: np.sum(x)
```

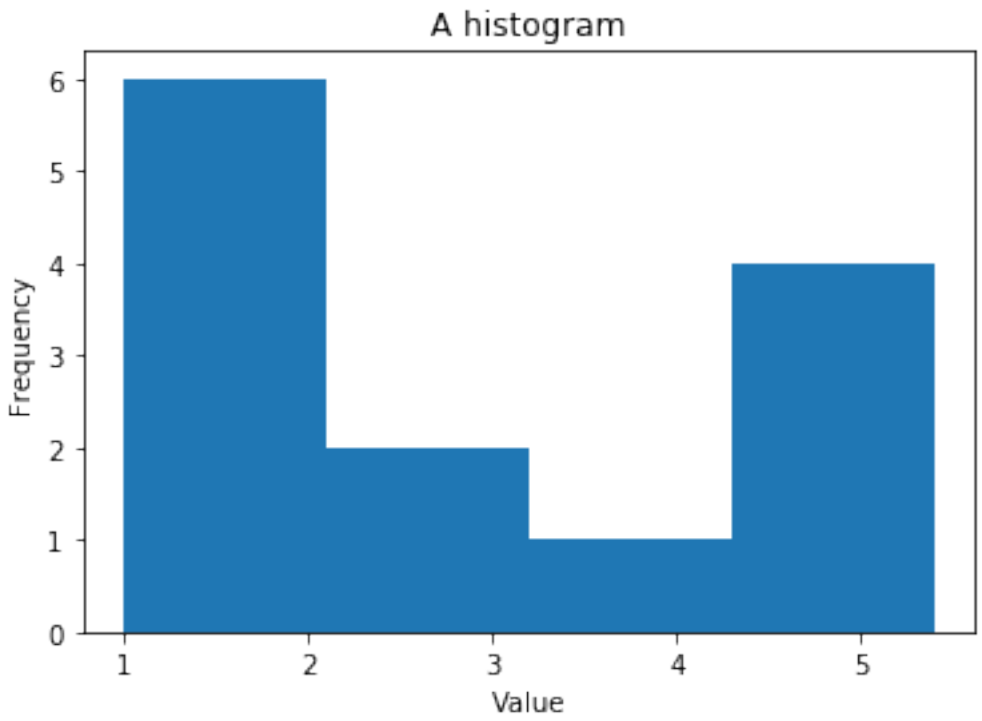
```
Out[4]: 37.299999999999997
```

Now we want to plot some graphs. We will use the following **matplotlib** library

```
In [5]: import matplotlib.pyplot as plt
```

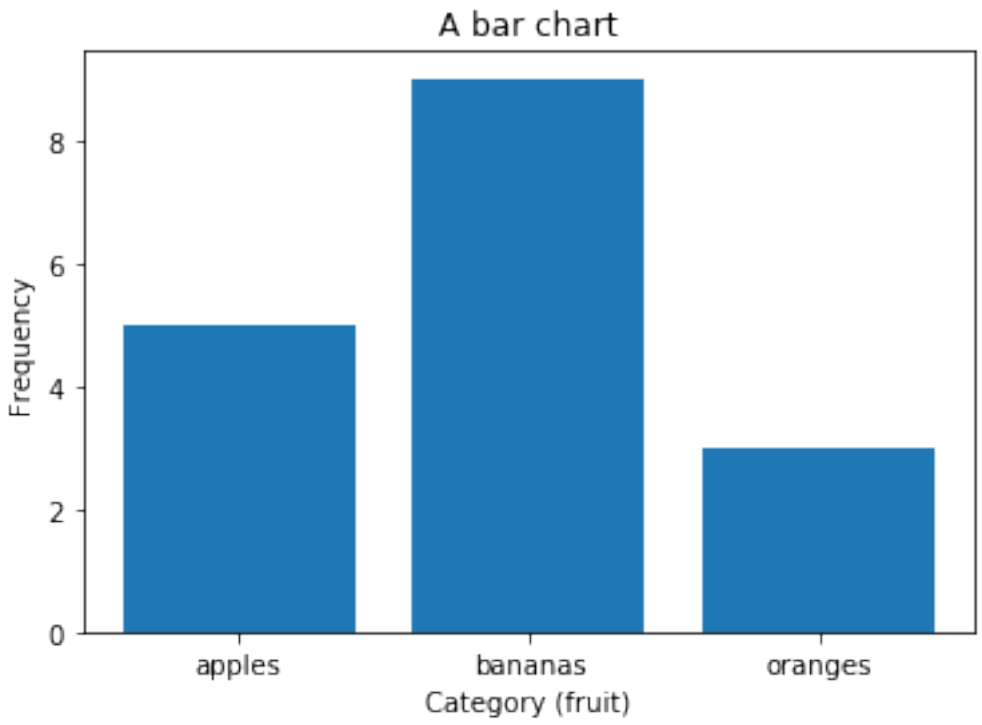
We can now draw a **historgram** of the data in x, speciyfing 4 **bins** (or classes). The ranges are automatically defined by Python.

```
In [6]: plt.figure()
        plt.hist(x,bins=4)
        plt.xlabel('Value')
        plt.ylabel('Frequency')
        plt.title('A histogram')
        plt.show()
```



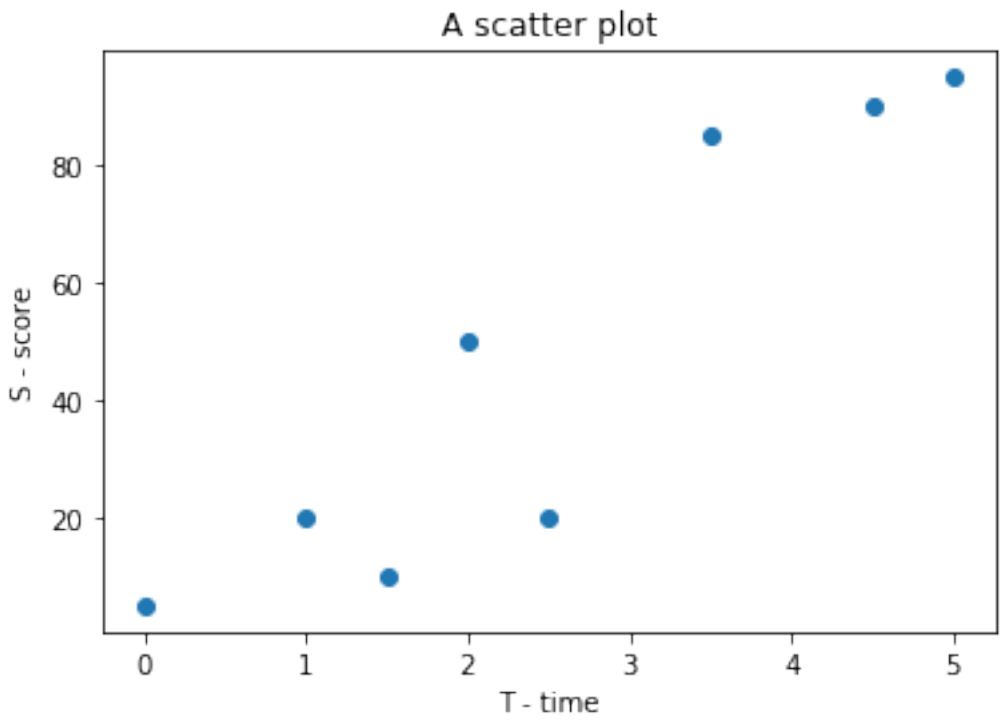
Now let's make a **bar chart**. Assume we asked 17 children about their favourite fruits.

```
In [7]: plt.figure()
        plt.bar(['apples','oranges','bananas'],np.array([5,3,9]))
        plt.ylabel('Frequency')
        plt.xlabel('Category (fruit)')
        plt.title('A bar chart')
        plt.show()
```



Now let's look at a **scatter plot**. Is the time spent studying for a test related to the score on the test? Imagine we asked some students, and we got values for time (T) in hours they spent studying, and the score (S) they achived on the test.

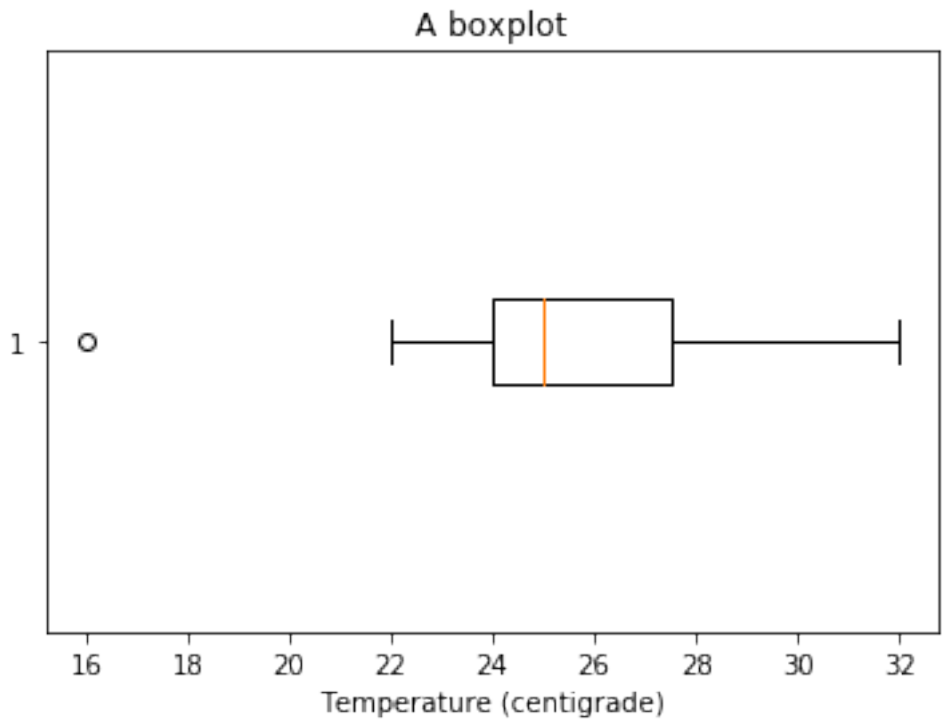
```
In [8]: T = [0.0,1.5,1.0,5.0,2.5,4.5,2.0,3.5]
        # Note: We do not need to have an array for a scatter plot, a list works just as well.
        S = [5, 10, 20, 95, 20, 90, 50,85]
        plt.figure()
        plt.scatter(T,S)
        plt.xlabel('T - time')
        plt.ylabel('S - score')
        plt.title('A scatter plot')
        plt.show()
```



So it looks like spending more time studying results in higher scores on the test (a **positive correlation**)

A **Boxplot** (also known as a *box and whisker plot*) is another common way to represent continuous numerical data. Suppose we measure the temperature on several days, and find

```
In [9]: t = [25,26,24,30,25,32,28,26,24,28,16,22,23,25] # temperatures
        plt.figure()
        plt.boxplot(t,vert=False)
        # vert[ical]=False makes the plot horizontal, instead of vertical
        plt.xlabel('Temperature (centigrade)')
        plt.title('A boxplot')
        plt.show()
```



One point has been marked by itself as a circle - this is because it is an outlier (extreme value). Possibly it was a mistaken recording.

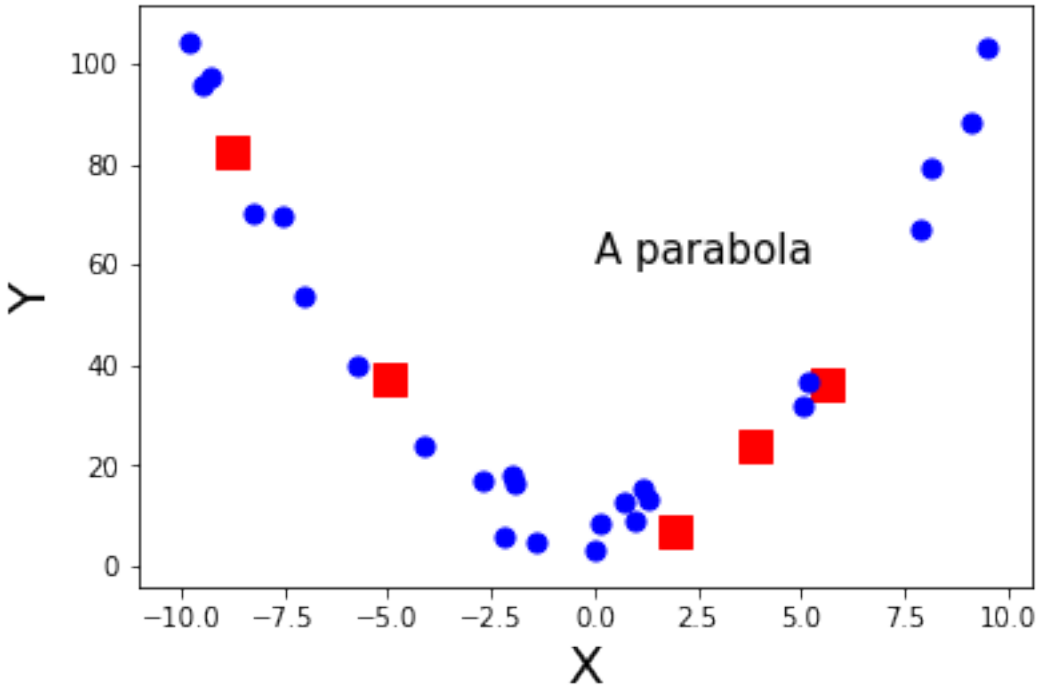
Finally, let's do a more **complex plotting example**, using a for loop. Specifically, we want:

(1) A scatter plot, which shows two variables as a parabola with random noise (30 data points);

(2) The samples with indexes [0,4,6,7,20] should be big red squares

```
In [38]: X = -10 + 20*np.random.rand(30)# uniform random numbers from -10 to 10
         Y = X**2 + 15*np.random.rand(30)# parabola + noise
         SpecialPoints = [0,4,6,7,20] # special points we want to have big, red, and square

         plt.figure()
         for j in range(len(X)):
             if j in SpecialPoints:
                 plt.scatter(X[j],Y[j],c='red',s=150,marker='s')
             else:
                 plt.scatter(X[j],Y[j],c='blue',s=50,marker='o')
         plt.xlabel('X',fontsize=20)
         plt.ylabel('Y',fontsize=20)
         plt.text(x=0,y=60,s='A parabola', fontsize=15)
         plt.show()
```


A parabola
Y
X
100
80
60
40
20
0
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5
10.0