1 Applying user-defined functions to NumPy and Pandas

Both NumPy and Pandas allow user to functions to applied to all rows and columns (and other axes in NumPy, if multidimensional arrays are used)

1.1 Numpy

In NumPy we will use the *apply_along_axis* method to apply a user-defined function to each row and column.

Let's first set up a array and define a function. We will use a simple user-defined function for illustrative purposes - one that returns the position of the highest value in the slice passed to the function. In NumPy we use the argmax for finding the position of the highest values.

```
import numpy as np
import pandas as pd

my_array = np.array([[10,2,13],
                     [21,22,23],
                     [31,32,33],
                     [10,57,20],
                     [20,20,20],
                     [101,91,10]])


def my_function(x):
    position = np.argmax(x)
    return position
```

Using *axis=0* we can apply that function to all columns:

```
print (np.apply_along_axis(my_function, axis=0, arr=my_array))

OUT:
[5 5 2]
```

Using *axis=1* we can apply that function to all rows:

```
print (np.apply_along_axis(my_function, axis=1, arr=my_array))

OUT:
[2 2 2 1 0 0]
```

1.2 Pandas

Pandas has a similar method, the apply method for applying a user function by either row or column. The Pandas method for determining the position of the highest value is *idxmax*.

We will convert our NumPy array to a Pandas dataframe, define our function, and then apply it to all columns. Notice that because we are working in Pandas the returned value is a Pandas series (equivalent to a DataFrame, but with one one axis) with an index value.

```
import pandas as pd

df = pd.DataFrame(my_array)

def my_function(x):
    z= x.idxmax()
    return z
```

```
print(df.apply(my_function, axis=0))

OUT:
0    5
1    5
2    2
dtype: int64
```

And applying it to all rows:

```
print(df.apply(my_function, axis=1))

OUT:
0    2
1    2
2    2
3    1
4    0
5    0
dtype: int64
```