Claire Acke

HWRS 501

Pandas and Numpy Cheatsheet

**Numpy**

**Overall purpose of the numpy library**: it is super helpful when needing to do mathematical calculations, whether it be calculating statistics, vectors, and of course scalar math.

**What are numpy arrays and what things can they and can’t they contain**: arrays are essentially vectors and matrices depending on the dimensions. It must always contain numbers. For example, np.array(2,4) would create an array with two rows and four columns.

**Various approaches for how to slice a numpy array (e.g., grabbing out a row, column, range of values)**: In order to slice an array, you typically need to use different variations of the arr[] function. For a specific value, something like arr[2] would bring the value of the array at index position 2. To return values for a column, arr[2:5, 3] would bring back the values of the array in rows 3-6 in the 3rd column.

**How to create a numpy array from scratch (show 2 options)**: One way that you can create a numpy array is mentioned above, but you can create an an array with np.array(2,4), which creates an array that has two rows and four columns. If we wanted to make another array with only zeros, which can be helpful if you need to manipulate an array further on in the code, you can use np.zero().

**List 5-6 helpful numpy functions and what they do**:

arr.shape() = shows the dimensions of an array

np.copy() = makes a copy of an array, useful for if you need to manipulate an array but don’t want to mess with any properties of the original array

np.append(arr, values) = adds values to the end of the array

arr.astype(dtype) = converts the contents of an array to a different type

np.mean(arr, axis=0) = calculates a mean along a specific axis

**Pandas**

**Overall purpose of the Pandas library**: Pandas provides a way to put data into a “table” and perform a lot of functions, whether it be mathematical calculations or manipulating values (I love dataframes).

**What makes a Pandas** **dataframe different from a numpy array**: Dataframes are able to hold more than just integers and can hold strings.

**Explanation of what the ‘index’ of a dataframe is and why it’s different from other columns**: Indexes don’t necessarily have to be numbers. It can be based on a “label” as well. For example, an index could be a list of months.

**How to setup a Pandas dataframe by reading a file**: the easiest way to import a file into a dataframe is when it comes in the form of a CSV file or table, etc. If we use pd.read\_csv(filename), it will read in the CSV file into a dataframe.

**How to see the index of a Pandas dataframe**: you can use df.info() to see the index, memory information, and datatype.

**How to slice a Pandas dataframe**:

* **Using loc and iloc to get rows**

s.loc[0] = selects by an index

s.iloc[0] = selects by the position

* **Grabbing out columns by name or number**

df[col] = this returns the column values as a series

**5-6 helpful things about Pandas and what they do**:

df.describe() = performs summary statistics for columns with numerical values

pd.isnull() = checks for nan numbers and returns it in an array with Boolean values

df.to\_csv(filename) = export your dataframe and download it into its own CSV files

df.dropna() = drops every row that has nan values

df.max() = returns the maximum values in the entire dataframe