CS 133 Cheat Sheet of Methods

[Data Science Python Reference](http://data8.org/fa18/python-reference.html)

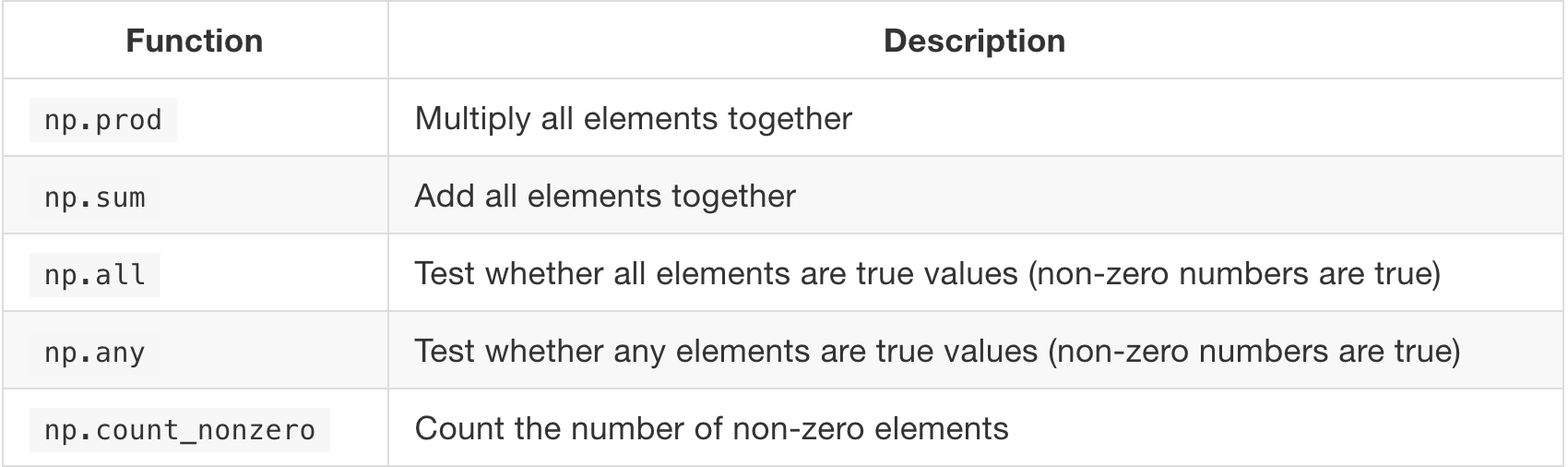
[Python’s Built in Functions](https://docs.python.org/3/library/functions.html)

[Math Module Functions](https://docs.python.org/3/library/math.html)

|  |  |  |
| --- | --- | --- |
| String Methods | What it does | Example |
| *String*.upper() | Casts the letters in the string all capital letters | “loud”.upper()  LOUD |
| *String.lower()* | Casts the letters in the string to lower case letters | “HaPPy”.lower()  Happy |
| *String.replace(‘str’, ‘str’)* | Replaces the first string passed in as a parameter with the second string passed in | ‘hitchhiker’.replace(‘hi’, ‘ma’)  Matchmaker |

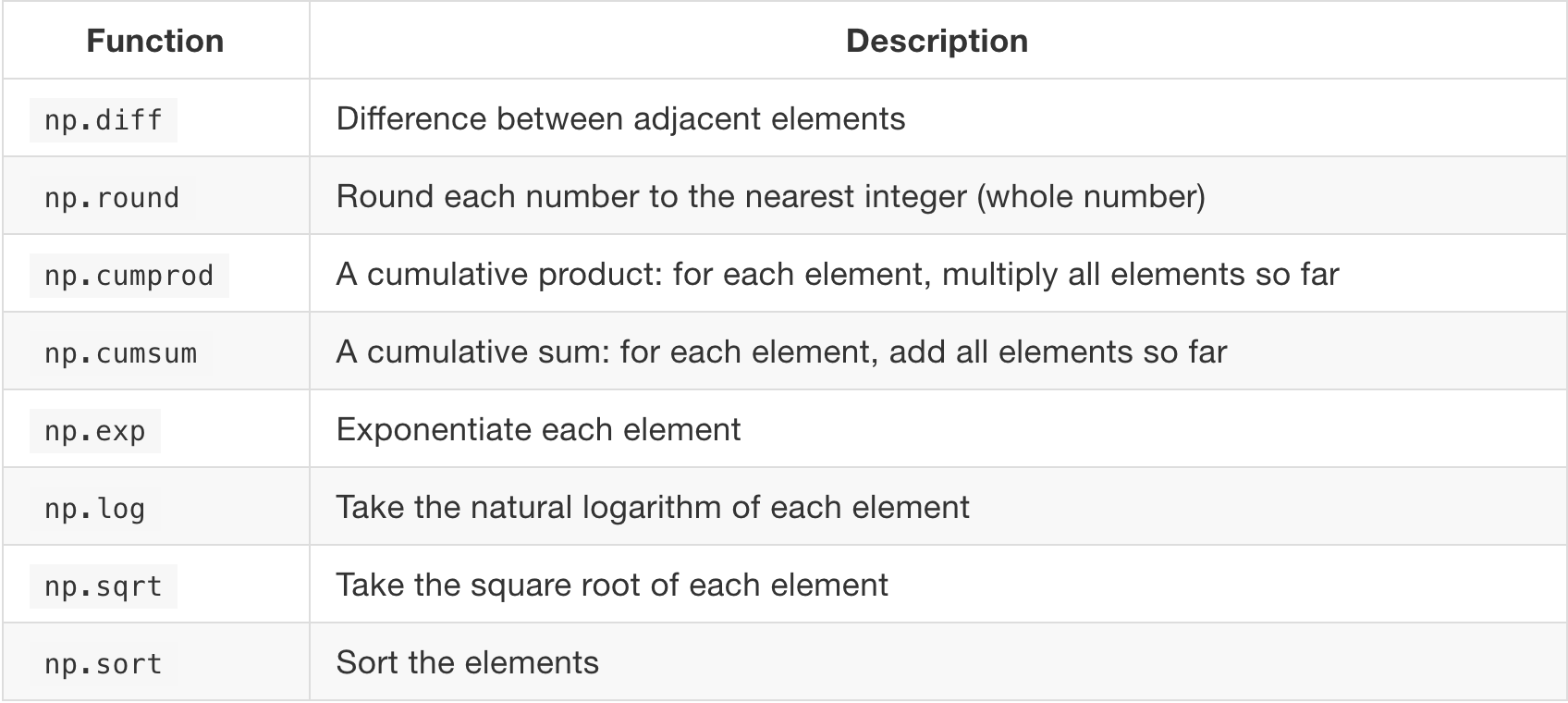
[NumPy Reference](https://docs.scipy.org/doc/numpy/reference/)

Each of these functions takes an array as an argument and returns a single value.

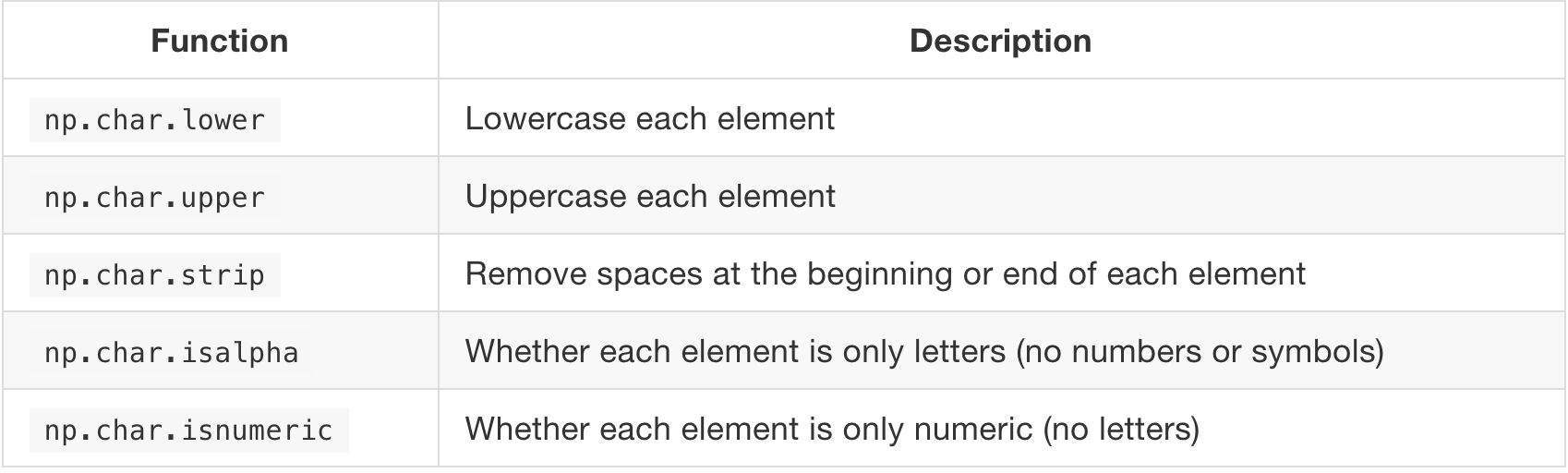


Each of these functions takes an array as an argument and returns an array of

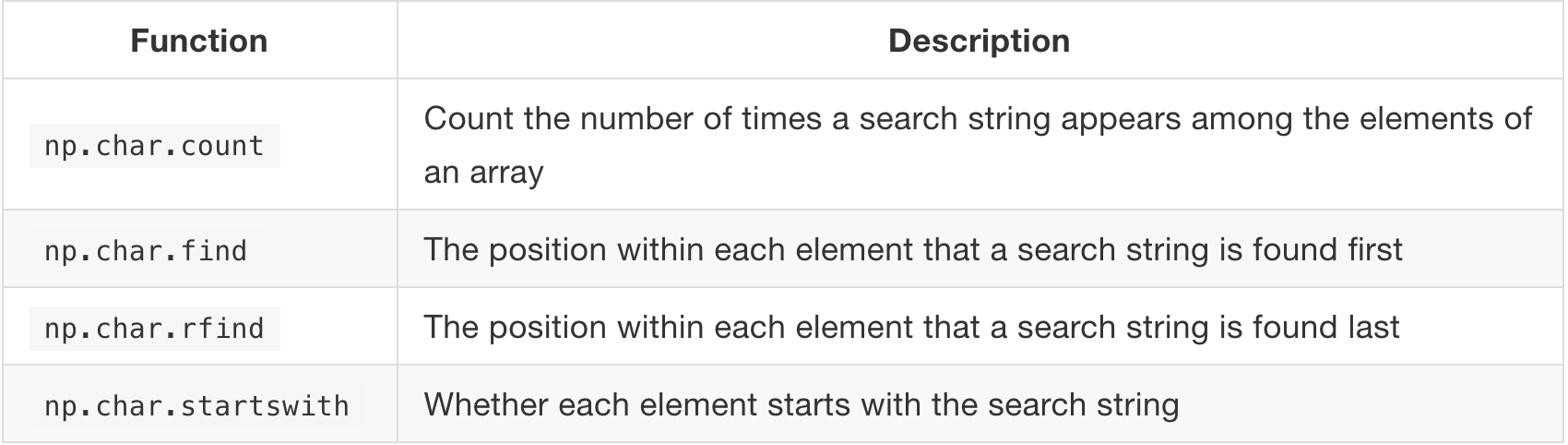
values.



Each of these functions takes an array of strings and returns an array.



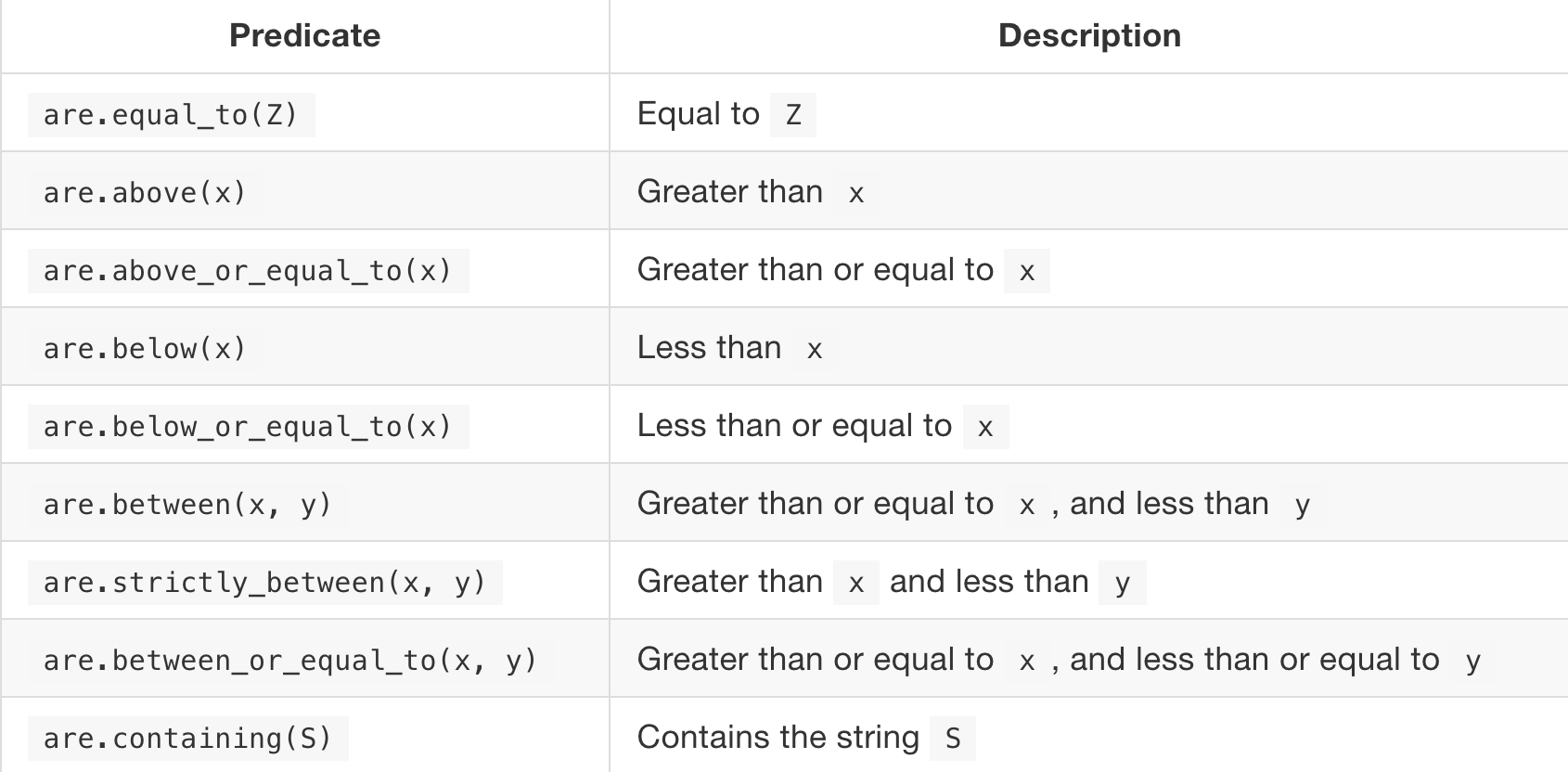
Each of these functions takes both an array of strings and a *search string*; each returns an array.



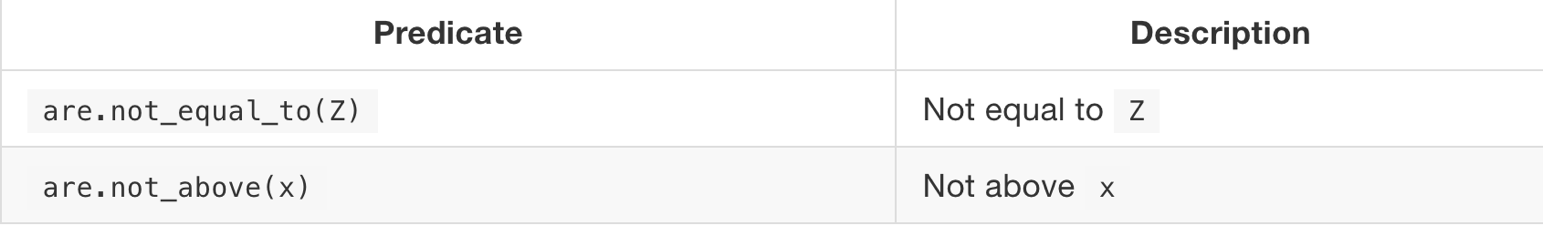
Making a table:

|  |  |
| --- | --- |
| Method/Function Call | What it does |
| Table().with\_columns(*column\_name,* make\_array()) | Creates a table with the columns that are passed in as parameters |
| *table\_name.*with\_columns(*column\_name,* make\_array()) | Adds a column to the already created table with the name *table\_name* |
| Table.read\_table(‘*csv\_file\_name’*) | Reads in a csv file and creates a table with that data |
| *table\_name.*num\_columns | Returns the number of columns in the table with the given *table\_name* |
| *table\_name.*num\_rows | Returns the number of rows in the table with the given *table\_name* |
| *table\_name*.labels | Returns a list of the column names |
| *table\_name*.relabeled(‘str’, ‘str’) | Relabels the column with the first string name with the new string name that is the second parameter |
| *table\_name*.column(‘*column\_name’/index*) | Returns the array that has the values in the designated column |
| .item(index) | Returns the item at the specified index |
| Method/Function Call | What it does |
| *table\_name*.select(‘*column’/index*, ‘*column/index’*) | Selects whatever column you pass in as a parameter and creates a new table |
| *table\_name*.drop(‘column\_name’/index) | You can pass in as many parameters as need be but drops the columns on the table with the wanted request. Returns a new table |
| *table\_name*.sort(‘*column\_name’*)  *table\_name*.sort(‘*column\_name’, descending = True*) | Sorts the specified column numerically or alphabetically |
| *table\_name.*sample(*quantity*) | Randomly selects an element out of table with replacement however many times you specify (quantity). |

Note that x and y are numbers, STRING is a string, and Z is either a number or a string; you have to specify these depending on the feature you want.



You can also specify the negation of any of these conditions, by using .not\_ before the condition:



Creating visual representations:

|  |  |
| --- | --- |
| Method/Function call | What it does |
| *table\_name.*scatter*(“name of column 1”, “name of column 2”)* | Creates a scatter plot with column 1 on x-axis and column 2 on y-axis |
| *table\_name.plot(“name of column 1”, name of column 2”)* | Creates a line plot with column 1 on x-axis and column 2 on the y-axis |
| *table\_name.*barh*(“category”, “numerical data”)* | Creates a horizontal bar graph |
| *table\_name.*bar*(“category”, “numerical data”)* | Creates a vertical bar graph |
| *table\_name.*hist*(‘name of bins’, unit=”unit name”)* | Creates a histogram |