Data Science for Everyone

Week 7

Angela Lai

New York University

Outline

- Logistics
- Concept Review
- Coding Demo
- Open Questions

Logistics

- All grades on assignments up to the midterm available on Gradescope
- Midterm grade and feedback on NYU Classes
- Deadline for regrade requests is end of **today, March 27**

Logistics

- No immediate upcoming deadlines
- Lab 5 is a freebie: everyone gets the 2%
- Homework 3/4 out April 6, due April 27
- Going forward, feedback for coding assignments will be released via email. You'll still use Jupyter Hub to fetch and submit assignments

Logistics

- Project due at 8 p.m. EDT on May 4
- Suggested mini deadlines from lecture 8.1:
  - April 6: Have a dataset decided on
  - April 20: Be at least halfway through the questions
  - April 27: Be finished with all the coding and analysis (only writing, checking left)
  - May 4: Totally done and proofread by this day, upload well before 8p deadline
- Optional: get feedback/approval on a chosen data set by filling out a Google Form (will be emailed after section) by **April 6**

range([start], stop, [step]) list(range(n)) range(0, n, 1)

start = 0

sequence of numbers

list(range(n)) → [0, 1, 2, ..., n-1]

list(range(strt, stp, step)) → [strt, strt+step, strt+2*steps, ...]

list(range(5)) = [0, 1, 2, 3, 4]

```
n = len(list)
for i in range(n):
    mylst[i] = #do this
```

np.arange() gives a np array

Concept Review: range()

Python's built-in range() function is useful for creating a sequence of numbers. We have *`range([start=0], stop, [step=1])`* where the number "stop" is never included in the sequence. By default, our range starts at 0 and takes "steps" of size 1.

So *`range(0,5,1)`* produces the same thing as *`range(5)`*.

If we call *`list(range(5))`*, we get *`[0,1,2,3,4].`*

`np.arange()` essentially does the same thing but creates a numpy array instead of a range object.

Concept Review: range()

Say we have numbers *start, end, step*. Then if we call *list(range(start,end,step))*, we get *[start,start+step,start+step*2,...]*, which goes all the way up to but does not include *end*.

Challenge Question

```
def myfunction(c):
    if c<-2:
        return 4
    elif c>2:
        return 4
    else:
        return np.abs(c) + 2

input = np.arange(-3, 3.1, 1)
output = [myfunction(x) for x in input]
```

What is the value of ***output***?

Open Lab

Questions?