

# QSS20: Modern Statistical Computing

## Session 03: Catch Up

# Goals for today's session

- ▶ **Housekeeping**
- ▶ Review notebook focused on pandas questions
- ▶ Review notebook covering plotting (relevant for question 2 graphs)
- ▶ Loops versus functions (relevant for problem 2.3.3 bandwidth question)

# Housekeeping

- ▶ **Problem set deadline extended until Sunday 11:59 PM** (canvas announcement yesterday)
- ▶ Heterogeneity in backgrounds
- ▶ Please fill out Slack profile with picture otherwise we will bother you via DM!
- ▶ ggplot2 datacamp assignment - 100% optional and nongraded (optional modules will have March 15th deadline); just if you want plotting practice
- ▶ MLK day and next week

# Goals for today's session

- ▶ Housekeeping
- ▶ **Review notebook focused on pandas questions**
- ▶ **Review notebook covering plotting (relevant for question 2 graphs)**
- ▶ Loops versus functions (relevant for problem 2.3.3 bandwidth question)

## Notebook focused on Pandas questions

Open up the following notebook...

[00\\_classquestions.ipynb](#)

## Notebook focused on plotting code

Open up the following notebook...

[01\\_example\\_plotting.ipynb](#)

# Goals for today's session

- ▶ Housekeeping
- ▶ Review notebook focused on pandas questions
- ▶ Review notebook covering plotting (relevant for question 2 graphs)
- ▶ **Loops versus functions (relevant for problem 2.3.3 bandwidth question)**

# Common task

- ▶ Do something repeatedly to something else (e.g., do something to every row in a dataset; transform every column)
- ▶ Within pandas, the built in methods like `df.mean()`, `df.col.str.contains()` etc only get us so far
- ▶ Often may want to iterate over rows, check for something or do something, and store the result
  - ▶ **Example:** pset question 3.3 on (1) focusing on each defendant, and (2) seeing if we can find any matches who are a different race but the same age and gender



To make more concrete: we have a couple example crime reports

CCN	WARD	OFFENSE	REPORT_DAT
20165648	6	MOTOR VEHICLE THEFT	2020/11/19 21:25:50+00
20123250	2	MOTOR VEHICLE THEFT	2020/08/29 01:00:25+00

For each of these two crimes, we want to see if there are any reported crimes with (1) same ward and (2) reported within 20 minutes of the first crime. We want to pull all those crimes and rank them by time proximity.

For both approaches, define crimes to look for and crimes to look within

```
1 ## two examples
2 CCN_examples = ['20165648', '20123250']
3
4 ## crimes to look 4 matches for
5 crimes_lookfor = dc_crim_2020.loc[dc_crim_2020.CCN.astype(str).isin(
6     (CCN_examples),
7     ['CCN', 'WARD', 'OFFENSE', 'report_dt']).copy()]
8
9 ## crimes to look for matches within
10 other_crimes = dc_crim_2020[~dc_crim_2020.CCN.astype(str).isin(
11     CCN_examples)].copy()
```

# Approach 1: loop through crimes

```
1 ## create empty container to store results
2 store_matches = {}
3 ## loop through two example crimes
4 for i in range(0, crimes_lookfor.shape[0]):
5     ## extract row
6     one_row = crimes_lookfor.iloc[i]
7
8     ## first, subset to crimes in same ward
9     same_wards = other_crimes[other_crimes.WARD == one_row.WARD].
10    copy()
11
12    ## second, with those same-ward crimes, filter to crimes within
13    ## 20 minutes after focal crime
14    cutoff = one_row.report_dt + timedelta(minutes=20)
15    same_wards_sametime = same_wards[
16        (same_wards.report_dt >= one_row.report_dt) &
17        (same_wards.report_dt <= cutoff)].copy()
18
19    ## third, store the results
20    store_matches[str(one_row.CCN)] = same_wards_sametime
21
22 ## finally, rowbind results into one df
23 all_matches = pd.concat(store_matches)
```

How might we transition this to a function?

# General structure of a function

```
1
2 ## what is the function called?
3 ## and what are its inputs?
4 def do_something(search_for: pd.DataFrame,
5                  search_in: pd.DataFrame):
6
7     ## here i'm doing things
8     ## (similar to meat of a for loop)
9     my_output = search_in[search_in.something ==
10                          search_for.something].copy()
11     ## and so on...
12
13     ## here i'm returning things
14     ## (similar to last part of a loop, though don't need
15     ## a within-function container)
16     return(my_output)
```

## Approach 2: first define a function

```
1 def proximate_crimes(search_for ,
2                       search_in):
3
4     ## first , subset to crimes in same ward
5     same_wards = search_in[search_in.WARD ==
6                             search_for.WARD].copy()
7
8     ## second , with those same-ward crimes , construct indicator
9     ## for reported within 20 minutes
10    cutoff = search_for.report_dt + timedelta(minutes=20)
11    same_wards_sametime = same_wards[
12        (same_wards.report_dt >= search_for.report_dt) &
13        (same_wards.report_dt <= cutoff)].copy()
14
15    ## add col for focal match
16    same_wards_sametime['focal_crime'] = search_for.CCN
17
18    ## return
19    return(same_wards_sametime)
```

## Approach 2: applying the function

```
1 ## apply to a single crime
2 one_match = proximate_crimes(search_for = crimes_lookfor.iloc[0],
3                               search_in = other_crimes)
4
5 ## iterate over the crimes, apply, and rowbind results
6 all_matches_list = [proximate_crimes(search_for =
7                                   crimes_lookfor.iloc[i],
8                                   search_in = other_crimes)
9                       for i in
10                          range(0, crimes_lookfor.shape[0])]
11 all_matches = pd.concat(all_matches_list)
```