IA Scientific Computing

Dr Damon Wischik
“A mathematician is a machine for turning coffee into theorems” – Erdős / Rényi

A software engineer

A scientist / machine learner / modeller
What is scientific computing?

Jupyter notebooks
Python
numpy + pandas + matplotlib
How to write bad code

Let's next look at police activity by day of the week, as measured by average number of stop-and-search events, and let's also show the variability as measured by the standard deviation. This involves computing two summaries of the data, combining them, and plotting the result.

In this code I'm doing my own arithmetic on timestamps to get day of week. I find it's easier to do a little bit of arithmetic then to wade through library documentation about datetime utility functions.

```python
In [47]:
# Compute the weekday for each row.
# Timestamp is measured in seconds since Thu 2016-01-01 00:00:00, so this computation gives me Python off: weekday = dataframe/(24*3600) - 4
5
weekday_names = ['Mon', 'Tue', 'Wed', 'Thu', 'Fri', 'Sat', 'Sun']
# The previous plot suggests there's a change in policing after 2016-08, so let's restrict attention to that range
gf['stop'] = gf['weekday'] / (24*3600)
plt.title('stop mean/weekday')
t = gf['weekday'].days.plot()
plt.ylabel('time (s)')
plt.xlabel('day

In [48]:
# Compute the two statistics we want to plot: mean and standard deviation of number of stops
x = df2.groupby('weekday')['stops'].sum().reset_index(name='mean_stops')
y = df2.groupby('weekday')['stops'].sum().reset_index(name='sd_stops')
stats = x.merge(y, on='weekday')

# A bar plot with error bars
plt.bar(stats['weekday'], stats['mean_stops'], align='center', yerr=stats['sd_stops'])
plt.xticks(range(7, weekday_names)
plt.ylabel('mean stopped')
plt.xlabel('weekday')
```

first I ran this cell

and now this cell is producing strange answers

Then this cell, I think
What does good code look like?

What Not to Do

- Your ML has doubtless been one big file where you threw together all the functions and value declarations.
- Lots of C programs look like this :-(
  - We could emulate this in OOP by having one class and throwing everything into it.
- We can do (much) better

OOP Concepts

- OOP provides the programmer with a number of important concepts:
  - Modularity
  - Code Re-Use
  - Encapsulation
  - Inheritance (lecture 5)
  - Polymorphism (lecture 5)
- Let's look at these more closely...

Modularity and Code Re-Use

- You've long been taught to break down complex problems into more tractable sub-problems.
- Each class represents a sub-unit of code that (if written well) can be developed, tested and updated independently from the rest of the code.
- Indeed, two classes that achieve the same thing (but perhaps do it in different ways) can be swapped in the code.
- Properly developed classes can be used in other programs without modification.
How to make effective use of notebooks

“Every line of code is written without reason, maintained out of weakness, and orphaned by chance.”
Jean-Paul Sartre

“Look at each line of your code and ask yourself: ‘does this spark joy?’ If not, don’t keep it.”
Marie Kondo

For this course, you should create your own notebooks from scratch.

- A notebook is not a logfile. It’s a document, which you edit and reshape.
- A notebook is not a source file. Text cells are more important than code cells.
- A notebook is for exploring and explaining. One-off tasks. Many of the questions are one- or two-liners.

https://programmingisterrible.com/post/139222674273/write-code-that-is-easy-to-delete-not-easy-to-extend
“Effective Jupyter notebooks” https://news.ycombinator.com/item?id=19860955
“JupyterCon: I don’t like Notebooks” https://news.ycombinator.com/item?id=17856700
Don’t set traps for yourself with disorganized notebooks

- You should accumulate code during a work session, but trim it at the end of each session.
- Your reader should be able to understand your work by reading top-to-bottom. Your code should all work if run top-to-bottom.

while working

- imports
- experiment 1
- debug code
- tweaked experiment 1
- experiment 2
- update to experiment 1
- forgotten import

after you’ve finished

- imports
- utility functions
- run-once setup code
- functions that implement your solutions
- import autograder
- submit solutions to autograder
Submitting your answers to the autograder
Submitting your answers to the autograder

# Code to run in the preamble to the “submit solutions” section of your notebook:

```python
!pip3 install ucamcl
import ucamcl
GRADER = ucamcl.autograder('https://markmy.solutions', course='scicomp').subsection('notes0')
```

# To answer a question:

```python
q = GRADER.fetch_question('q1')
my_ans = ... # using the parameters in q
GRADER.submit_answer(q, my_ans)
```
How is the course structured and examined?

1. Programming in Python
2. Numerical computation
   Assignment marked \{0,1,2\}
3. Working with data
   Assignment marked \{0,1,2\}
   A. Data scraping tips
   B. Example data analyses

- Handout
- Online only

Self-paced work, with online autograder

It's the answers that matter. There's no written exam on the notes.

Ticking session on
28 Jan 2020

Online only

Will be released on
Monday
- It’s your job to write your own unit tests.
- Nature’s datasets are full of quirks, and nature is out to deceive you.
- As a data scientist you need to be ever vigilant, always checking your assumptions.

MY CODE PASSED QUESTION 2. BUT IT WAS BUGGY CODE, AND IT TOOK ME AGES TO DEBUG AND PASS QUESTION 3.

YOUR GRADER SUCKS.
Running Jupyter

WHAT IF I TOLD YOU

THE CLOUD IS JUST SOMEONE ELSE'S COMPUTER

... and it will fail when you need it most.
Backup!
Running Jupyter

The autograder will run wherever you run Jupyter + Python3.

- notebooks.azure.com
- hub.cl.cam.ac.uk
- your own machine

(installation tips on Moodle)
Where do I go for help?
1. StackOverflow
2. Moodle help forum
3. Help sessions at the beginning of Lent term