Introduction to Data Programming

CSE 160
University of Washington
Autumn 2020
Ruth Anderson

Slides based on previous versions by Michael Ernst and earlier versions by Bill Howe
Agenda for Today

• What is this course?
• Course logistics
• Python!
Welcome to CSE 160!

CSE 160 teaches **core programming concepts** with an emphasis on **real data manipulation tasks** from science, engineering, and business.

**Goal** by the end of the quarter: Given a data source and a problem description, you can independently write a complete, useful program to solve the problem.
Course staff

• Lecturer:
  – Ruth Anderson

• TAs:
  – Hannah Cheung
  – Niamh Froelich
  – Zoe Kaputa
  – Travis McGaha
  – Joely Nelson
  – Trinh Nguyen
  – Amanda Ong
  – Ece Oz
  – Yuxiao Shi
  – Zoe Steine-Hanson
  – Brian Zhu,

• Ask us for help!
Learning Objectives

• Computational problem-solving
  – Writing a program will become your “go-to” solution for data analysis tasks

• Basic Python proficiency
  – Including experience with relevant libraries for data manipulation, scientific computing, and visualization.

• Experience working with real datasets
  – astronomy, biology, linguistics, oceanography, open government, social networks, and more.
  – You will see that these are easy to process with a program, and that doing so yields insight.
What this course is **not**

• A “skills course” in Python
  – ...though you will become proficient in the basics of the Python programming language
  – ...and you will gain experience with some important Python libraries
• A data analysis / “data science” / data visualization course
  – There will be very little statistics knowledge assumed or taught
• A “project” course
  – the assignments are “real,” but are intended to teach specific programming concepts
• A “big data” course
  – Datasets will all fit comfortably in memory
  – No parallel programming
“It’s a great time to be a data geek.”
-- Roger Barga, Microsoft Research

“The greatest minds of my generation are trying to figure out how to make people click on ads”
-- Jeff Hammerbacher, co-founder, Cloudera
All of science is reducing to computational data manipulation

Old model: “Query the world” (Data acquisition coupled to a specific hypothesis)

New model: “Download the world” (Data acquisition supports many hypotheses)

- Astronomy: High-resolution, high-frequency sky surveys (SDSS, LSST, PanSTARRS)
- Biology: lab automation, high-throughput sequencing,
- Oceanography: high-resolution models, cheap sensors, satellites

Slide from Bill Howe, eScience Institute
Example: Assessing treatment efficacy

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
<th>G</th>
<th>H</th>
<th>I</th>
<th>J</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>fu_2wk</td>
<td>fu_4wk</td>
<td>fu_8wk</td>
<td>fu_12wk</td>
<td>fu_16wk</td>
<td>fu_20wk</td>
<td>fu_24wk</td>
<td>total4type_fu</td>
<td>clinic_zip</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>3</td>
<td>4</td>
<td>7</td>
<td>9</td>
<td>9</td>
<td>9</td>
<td>12</td>
<td>98405</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>4</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>98405</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>8</td>
<td>98405</td>
</tr>
<tr>
<td>5</td>
<td>3</td>
<td>2</td>
<td>2</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>98405</td>
</tr>
<tr>
<td>6</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>8</td>
<td>98405</td>
</tr>
<tr>
<td>7</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>98405</td>
</tr>
<tr>
<td>8</td>
<td>1</td>
<td>2</td>
<td>5</td>
<td>6</td>
<td>8</td>
<td>10</td>
<td>10</td>
<td>14</td>
<td>98405</td>
</tr>
<tr>
<td>9</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>98499</td>
</tr>
<tr>
<td>10</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>98405</td>
</tr>
<tr>
<td>11</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>98405</td>
</tr>
<tr>
<td>12</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>98405</td>
</tr>
<tr>
<td>13</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>98404</td>
</tr>
<tr>
<td>14</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>2</td>
<td>98499</td>
</tr>
<tr>
<td>15</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>6</td>
<td>98499</td>
</tr>
<tr>
<td>16</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>6</td>
<td>98499</td>
</tr>
<tr>
<td>17</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>4</td>
<td>98499</td>
</tr>
<tr>
<td>18</td>
<td>1</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>98499</td>
</tr>
<tr>
<td>19</td>
<td>1</td>
<td>1</td>
<td>4</td>
<td>5</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>98499</td>
</tr>
</tbody>
</table>

**Question:** Does the distance between the patient’s home and clinic influence the number of follow ups, and therefore treatment efficacy?
Python program to assess treatment efficacy

This program reads an Excel spreadsheet whose penultimate and antepenultimate columns are zip codes. It adds a new last column for the distance between those zip codes, and outputs in CSV (comma-separated values) format. Call the program with two numeric values: the first and last row to include. The output contains the column headers and those rows.

Libraries to use
import random
import sys
import xlrd     # library for working with Excel spreadsheets
import time
from gdapi import GoogleDirections

No key needed if few queries
gd = GoogleDirections('dummy-Google-key')

wb = xlrd.open_workbook('mhip_zip_eScience_121611a.xls')
sheet = wb.sheet_by_index(0)

User input: first row to process, first row not to process
first_row = max(int(sys.argv[1]), 2)
row_limit = min(int(sys.argv[2]+1), sheet.nrows)

def comma_separated(lst):
    return ','.join([str(s) for s in lst])

headers = sheet.row_values(0) + ['"distance"']
print comma_separated(headers)

for rownum in range(first_row,row_limit):
    row = sheet.row_values(rownum)
    (zip1, zip2) = row[-3:-1]
    if zip1 and zip2:
        # Clean the data
        zip1 = str(int(zip1))
        zip2 = str(int(zip2))
        row[-3:-1] = [zip1, zip2]
        # Compute the distance via Google Maps
        try:
            distance = gd.query(zip1,zip2).distance
        except:
            print >> sys.stderr, "Error computing distance:", zip1, zip2
        distance = ""
        # Print the row with the distance
        print comma_separated(row + [distance])
# Avoid too many Google queries in rapid succession
time.sleep(random.random()+0.5)

23 lines of executable code!
Course logistics

• Website: http://www.cs.washington.edu/cse160
  – See the website for all administrative details
• Homework 0 - due Monday
  – Two Preliminary Surveys are due Friday
• Questions? rea@cs.washington.edu
How to succeed

• No prerequisites
• **Non**-predictors for success:
  – Past programming experience
  – Enthusiasm for games or computers
• Programming and data analysis are challenging
• Every one of you can succeed
  – There is no such thing as a “born programmer”
  – Work hard
  – Follow directions
  – Be methodical
  – *Think* before you act
  – Try on your own, then ask for help
  – Start early
Me (Ruth Anderson)

- **Grad Student at UW:** in Programming Languages, Compilers, Parallel Computing
- **Taught Computer Science** at the University of Virginia for 5 years
- **PhD at UW:** in Educational Technology, Pen Computing
- **Current Research:** Computing and the Developing World, Computer Science Education
Introductions on Ed Board

- Name
- Major
- Hometown
- Interesting Fact or what I did over break.