Advanced Data Management (CSCI 680/490)

Python

Dr. David Koop
Data to Knowledge and Beyond

[D. Somerville, based on H. McLeod’s original]
Data to Knowledge and Beyond

[D. Somerville, based on H. McLeod’s original]
Data to Knowledge and Beyond

[D. Somerville, based on H. McLeod’s original]
Data to Knowledge

Can computers do this for us?

[D. Somerville, based on H. McLeod’s original]
How do data scientists spend their time?

- Building training sets: 3%
- Cleaning and organizing data: 60%
- Collecting data sets: 19%
- Mining data for patterns: 9%
- Refining algorithms: 4%
- Other: 5%

[CrowdFlower Data Science Report, 2016]
Finding & Discovering Data (even data you already have!)

DATA WAREHOUSE VS DATA LAKE

- Data is processed and organized into a single schema before being put into the warehouse.
- The analysis is done on the cleansed data in the warehouse.

- Raw and unstructured data goes into a data lake.
- Data is selected and organized as and when needed.

[S. Dewan]
Data Wrangling

<table>
<thead>
<tr>
<th>A</th>
<th>Transaction Date</th>
<th>B</th>
<th>Customer Name</th>
<th>C</th>
<th>Phone Numbers</th>
<th>D</th>
<th>Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>Wed, 12 Jan 2011</td>
<td>John K. Doe Jr.</td>
<td>(609)-993-3001</td>
<td>2196 184th Ave. NE, Redmond, 98052</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Thu, 15 Sep 2011</td>
<td>Mr. Doe, John</td>
<td>609.993.3001 ext 2001</td>
<td>4297 148th Avenue NE, Bellevue, 98007</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Mon, 17 Sep 2012</td>
<td>Jane A. Smith</td>
<td>+(1)-4250013981</td>
<td>2720 N Mesa St, El Paso, 79902, USA</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>2011-Jan-11 02:27:21</td>
<td>Smith, Jane</td>
<td>tel: 4250013981</td>
<td>4740 N 132nd St Apt 417, Omaha, 68164</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>2011-Jan-12</td>
<td>Anthony R Von Fange II</td>
<td>650-384-9911</td>
<td>10508 Prairie Ln, Oklahoma City</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>2010-Dec-24</td>
<td>Mr. Peter Tyson</td>
<td>(405)123-3981</td>
<td>525 1st St, Marysville, WA 95901</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>9/22/2011</td>
<td>Dan E. Williams</td>
<td>1-650-1234183</td>
<td>211 W Ridge Dr, Waukon, 52172</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>7/11/2012</td>
<td>James Davis Sr.</td>
<td>+(1)-425-736-9999</td>
<td>13120 Five Mile Rd, Brainerd</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>2/12/2012</td>
<td>Mr. James J. Davis</td>
<td>425.736.9999 x 9</td>
<td>602 Highland Ave, Shinnston, 26431</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>3/31/2013</td>
<td>Donald Edward Miller</td>
<td>(206) 309-8381</td>
<td>840 W Star St, Greenville, 27834</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>6/1/2009 12:01</td>
<td>Miller, Donald</td>
<td>206 309 8381</td>
<td>25571 Elba, Redford, 48239</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>2/26/2007 18:37</td>
<td>Rajesh Krishnan</td>
<td>206 456 8500 extension 1</td>
<td>539 Co Hwy 48, Sikeston, USA</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>1/4/2011 14:33</td>
<td>Daniel Chen</td>
<td>425 960 3566</td>
<td>1008 Whitlock Ave NW, Marietta, 30064</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Table C**

<table>
<thead>
<tr>
<th>D</th>
<th>Customer Name</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>John K. Doe Jr.</td>
<td>Doe, John</td>
<td></td>
</tr>
<tr>
<td>Mr. Doe, John</td>
<td>Doe, John</td>
<td></td>
</tr>
<tr>
<td>Jane A. Smith</td>
<td>Smith, Jane</td>
<td></td>
</tr>
<tr>
<td>Smith, Jane</td>
<td>Smith, Jane</td>
<td></td>
</tr>
<tr>
<td>Dr Anthony R Von Fange II</td>
<td>Von Fange, Anthony</td>
<td></td>
</tr>
<tr>
<td>Peter Tyson</td>
<td>Tyson, Peter</td>
<td></td>
</tr>
<tr>
<td>Dan E. Williams</td>
<td>Williams, Dan</td>
<td></td>
</tr>
<tr>
<td>James Davis Sr.</td>
<td>Davis, James</td>
<td></td>
</tr>
<tr>
<td>James J. Davis</td>
<td>Davis, James</td>
<td></td>
</tr>
<tr>
<td>Mr. Donald Edward Miller</td>
<td>Miller, Donald</td>
<td></td>
</tr>
</tbody>
</table>

**Table D**

<table>
<thead>
<tr>
<th>D</th>
<th>Address</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>2196 184th Ave. NE Apt 417, Redmond, 98052</td>
<td>Redmond, WA, 98052</td>
<td></td>
</tr>
<tr>
<td>4297 148th Avenue NE L105, Bellevue, WA 98007</td>
<td>Bellevue, WA, 98007</td>
<td></td>
</tr>
<tr>
<td>2720 N Mesa St, El Paso, 79902, USA</td>
<td>El Paso, TX, 79902</td>
<td></td>
</tr>
<tr>
<td>3524 W Shore Rd APT 1002, Warwick,02886</td>
<td>Warwick, RI, 02886</td>
<td></td>
</tr>
<tr>
<td>4740 N 132nd St, Omaha, 68164</td>
<td>Omaha, NE, 68164</td>
<td></td>
</tr>
<tr>
<td>10508 Prairie Ln, Oklahoma City</td>
<td>Oklahoma City, OK, 73162</td>
<td></td>
</tr>
<tr>
<td>525 1st St, Marysville, WA 95901</td>
<td>Marysville, CA, 95901</td>
<td></td>
</tr>
<tr>
<td>211 W Ridge Dr, Waukon,52172</td>
<td>Waukon, IA, 52172</td>
<td></td>
</tr>
<tr>
<td>602 Highland Ave, Shinnston, 26431</td>
<td>Shinnston, WV, 26431</td>
<td></td>
</tr>
<tr>
<td>840 W Star St, Greenville, 27834</td>
<td>Greenville, NC, 27834</td>
<td></td>
</tr>
</tbody>
</table>

[Y. He et al., 2018]
Data Cleaning/Standardization (Aliases)

'google brain resident': 'google',
'google brain': 'google',
'google inc.': 'google',
'google inc.': 'google',
'google research nyc': 'google',
'google research': 'google',
'google, inc.': 'google',
'deepmind @ google': 'deepmind',
'deepmind technologies': 'deepmind',
'google deepmind': 'deepmind',

'ibm research - china': 'ibm',
'ibm research': 'ibm',
'ibm research, ny': 'ibm',
'ibm research, usa': 'ibm',
'ibm t. j. watson research center': 'ibm',
'ibm t. j. watson research': 'ibm',
'ibm t. j. watson research center': 'ibm',
'ibm t. j. watson research center': 'ibm',
'ibm thomas j. watson research center': 'ibm',
'ibm tj. watson research center': 'ibm',

'microsoft research cambridge': 'microsoft',
'microsoft research india': 'microsoft',
'microsoft research maluuba': 'microsoft',
'microsoft research new england': 'microsoft',
'microsoft research': 'microsoft',
'microsoft research, redmond, wa': 'microsoft',
'microsoft research, redmond, wa': 'microsoft',
'microsoft research': 'microsoft',

[Carnegie Mellon University, Microsoft, Stanford University, University of Cambridge, University of Edinburgh, Georgia Institute of Technology, Chinese Academy of Sciences, Harbin Institute of Technology, Peking University, Tsinghua University, Cornell University, University College London, University of Toronto, University of Maryland, INRIA, University of Pennsylvania, Princeton University, UT Austin, Johns Hopkins University, University of Cambridge, University of Washington, Massachusetts Institute of Technology, UC Berkeley, Columbia University, Google, Microsoft, Carnegie Mellon University]
Data Integration

- **Google Thinks I’m Dead** (I know otherwise.) [R. Abrams, NYTimes, 2017]

- Not only Google, but also Alexa:
  - "Alexa replies that Rachel Abrams is a sprinter from the Northern Mariana Islands (which is true of someone else)."
  - "He asks if Rachel Abrams is deceased, and Alexa responds yes, citing information in the Knowledge Graph panel."

---

D. Koop, CSCI 680/490, Spring 2022
Data Storage

**SQL DATABASES**

- Relational

**NoSQL DATABASES**

- Column
- Graph
- Key-Value
- Document
Provenance and Reproducibility

DATA

Data Management

Visualization

Computation

Paper

DATA

(compilation)
Provenance and Reproducibility

DATA

Data Management

Visualization

Provenance

Computation

Paper

Provenance
Towards provenance enabling paraview pages fifiz–fiz5v flzz6x
http://www.stccmop.org


About this course

• Course web page is authoritative:
  - Schedule, Readings, Assignments will be posted online
  - Check the web site before emailing me

• Lectures planned for in-person, plans can change

• Course is meant to be more "cutting edge"
  - Still focus on building skills related to data management
  - Tune into current research and tools

• Requires student participation: readings and discussions
Course Material

• Helpful Books:
  - Effective Pandas, M. Harrison
  - Python for Data Analysis, W. McKinney
    • 3rd edition updated for pandas 1.4
    • https://wesmckinney.com/book/
  - Intro to Python, Deitel & Deitel
  - Python Data Science Handbook, J. VanderPlas
• Research papers
• Many websites
References

- McKinney has a 3rd ed. preview available:
  - First 6 chapters currently available
In this Notebook we explore the Lorenz system of differential equations:
\[
\begin{align*}
\dot{x} &= \sigma(y - x) \\
\dot{y} &= px - y - xz \\
\dot{z} &= -\rho z + xy
\end{align*}
\]

Let's call the function once to view the solutions. For this set of parameters, we see the trajectories swirling around two points, called attractors.

```python
In [4]: from lorenz import solve_lorenz
t, x, z = solve_lorenz(N=10)
```
JupyterLab

- An interactive, configurable programming environment
- Supports many activities including notebooks
- Runs in your web browser
- Notebooks:
  - Originally designed for Python
  - Supports other languages, too
  - Displays results (even interactive maps) inline
  - You decide how to divide code into executable cells
  - Shift+Enter to execute a cell
Installing Python & JupyterLab

- www.anaconda.com/download/
- Anaconda has Jupyter Lab
- Use Python 3.9 version (**not** 2.7)
- Anaconda Navigator
  - GUI application for managing Python environment
  - Can install packages
  - Can start JupyterLab
- Can also use the shell to do this:
  - `$ jupyter lab`
  - `$ conda install <pkg_name>"
JupyterLab Notebook Tips

• Starts with a directory view
• Create new notebooks using the Launcher (+ icon on the left)
  - New notebooks have the name "Untitled"
  - File → Rename Notebook… (or right-click) to change the name
• Save a notebook using the command under the File menu
• Shutting down the notebook requires quitting the kernel
  - Web browser is **interface** to display code and results
  - **Kernel** runs the code: may see messages in a console/terminal window
  - Closing the browser window does not stop Jupyter
  - Use File → Shut Down to shut down everything
JupyterLab Notebooks

• Open a notebook using the left panel like you would in a desktop view
• Past results are displayed—does not mean they are loaded in memory
• Use "Run All" or "Run All Above" to re-execute past work
  - If you shut down the kernel, all of the data and variables you defined need to be redefined (so you need to re-run all)
  - **Watch Out—Order Matters:** If you went back and re-executed cells in a different order than they are shown, doing "Run All" may not produce the same results!
• Edit mode (green) versus Command mode (blue == Be Careful)
JupyterLab Notebooks

• Can write code or plain text (can be styled Markdown)
  - Choose the type of cell using the dropdown menu
• Cells break up your code, but all data is **global**
  - Defining a variable `a` in one cell means it is available in any other cell
  - This includes cells **above** the cell `a` was defined in!
• Remember **Shift+Enter** to execute
• Enter just adds a new line
• Use `?<function_name>` for help
• Use Tab for **auto-complete** or suggestions
• Tab also indents, and Shift+Tab unindents
JupyterLab Outputs

• stdout: where print commands go
• stderr: where error messages go
• display: special output channel used to show rich outputs
• output: same as display but used to display the value of the last line of a cell
Other JupyterLab Features

• Terminal
  - Similar to what you see on turing/hopper but for your local machine

• File Viewers
  - CSV
  - Plugins available

• Console
  - Can be linked to notebooks
JupyterLab Documentation

- JupyterLab Tutorial Video
- JupyterLab Documentation
Jupyter Notebook

- Original Notebook Interface
- Just notebooks
- Same rich representations and text
- Same cell structure
- Same notebook files .ipynb
- Web-based
Python

• Started in December 1989 by Guido van Rossum
• “Python has surpassed Java as the top language used to introduce U.S. students to programming…” (ComputerWorld, 2014)
• Python and R are the two top languages for data science
• High-level, interpreted language
• Supports multiple paradigms (OOP, procedural, imperative)
• Help programmers write readable code, Use less code to do more
• Lots of libraries for python
  - Designed to be extensible
  - Easy to wrap code from other languages like C/C++
• Open-source with a large, passionate community
Learning Python Resources

- Python for Programmers
- http://www.pythontutor.com
- https://www.python-course.eu
- https://software-carpentry.org/lessons/
Python Compared to C++ and Java

• Dynamic Typing
  - A variable does not have a fixed type
  - Example: `a = 1; a = "abc"

• Indentation
  - Braces define blocks in Java, good style is to indent but not required
  - Indentation is **critical** in Python
    
    ```
    z = 20
    if x > 0:
      if y > 0:
        z = 100
      else:
        z = 10
    ```
In-Class Notebook

- Try out the examples from the following slides:
  - Download
  - View
Print function

- `print("Hello World")`
- Can also print variables:
  ```python
  name = "Jane"
  print("Hello,", name)
  ```
Python Variables and Types

• No type declaration necessary
• Variables are names, not memory locations
  
  a = 0
  a = "abc"
  a = 3.14159

• Don't worry about types, but think about types
• Strings are a type
• Integers are as big as you want them
• Floats can hold large numbers, too (double-precision)
Python Math and String "Math"

- **Standard Operators:** +, -, *, /, %
- **Division** "does what you want" (new in v3)
  - 5 / 2 = 2.5
  - 5 // 2 = 2 # use // for integer division
- **Shortcuts:** +=, -=, *=
- **No** ++, --
- **Exponentiation (Power):** **
- **Order of operations and parentheses:** (4 - 3 - 1 vs. 4 - (3 - 1))
  - "abc" + "def"
  - "abc" * 3
Python Strings

• Strings can be delimited by single or double quotes
  - "abc" and 'abc' are exactly the same thing
  - Easier use of quotes in strings: "Joe's" or 'He said "Stop!"'

• String concatenation: "abc" + "def"

• Repetition: "abc" * 3

• Special characters: \n \t like Java/C++
Python Strings

• Indexing:
  
a = "abcdef"
a[0]
• Slicing: a[1:3]
• Format:
  
  name = "Jane"
  print("Hello, {}".format(name))
  - or
  
  print(f"Hello, {name}")
Loops

• while <condition>:
  <indented block>
  # end of while block (indentation done)

• Remember the colon!

• a = 5
  while a > 0:
    print(a)
    a -= 2

• a > 0 is the condition

• Python has standard boolean operators (<, >, <=, >=, ==, !=)
  - What does a boolean operation return?
  - Linking boolean comparisons (and, or)
Conditionals

• if, else
  - Again, indentation is required
• elif
  - Shorthand for else: if:
• Same type of boolean expressions (and or)
break and continue

- **break** stops the execution of the loop
- **continue** skips the rest of the loop and goes to the next iteration

```python
• a = 7
  while a > 0:
    a -= 2
    if a < 4:
      break
    print(a)
```

```python
• a = 7
  while a > 0:
    a -= 2
    if a < 4 and a > 2:
      continue
    print(a)
```
True and False

• True and False (captialized) are defined values in Python
• \( v == 0 \) will evaluate to either True or False
Functions

• Calling functions is as expected:
  \[
  \text{mul}(2,3) \quad \# \text{ computes } 2*3 \quad (\text{mul from operator package})
  \]
  - Values passed to the function are parameters
  - May be variables!
    
    \[
    a = 5 \\
    b = 7 \\
    \text{mul}(a,b)
    \]

• \text{print} is a function
  
  \[
  \text{print("This line doesn't end.", end=" ")}
  \]
  \[
  \text{print("See it continues")}
  \]
  - \text{end} is also a parameter, but this has a different syntax (keyword argument!)
Defining Functions

• **def** keyword

• Arguments have names but **no types**
  
  ```
  def hello(name):
      print(f"Hello {name}"
  ```

• Can have defaults:
  
  ```
  def hello(name="Jane Doe"):
      print(f"Hello {name}"
  ```

• With defaults, we can skip the parameter: `hello()` or `hello("John")`

• Also can pick and choose arguments:
  
  ```
  def hello(name1="Joe", name2="Jane"):
      print(f"Hello {name1} and {name2}"
  hello(name2="Mary")
  ```
Return statement

• Return statement gives back a value:
  ```python
def mul(a, b):
    return a * b
  ```

• Variables changed in the function won’t be updated:
  ```python
def increment(a):
    a += 1
    return a

b = 12

print(b, increment(b))
```
Python Containers

• Container: store more than one value
• Mutable versus immutable: Can we update the container?
  - Yes → mutable
  - No → immutable
  - Lists are mutable, tuples are immutable
• Lists and tuples may contain values of different types:
  • List: [1,"abc",12.34]
  • Tuple: (1, "abc", 12.34)
• You can also put functions in containers!
• len function: number of items: len(l)