Advanced Data Management (CSCI 640/490)

Python

Dr. David Koop
Supporting Data Science

In order to study the movement patterns for airports and train stations, we select the regions in their vicinity and examine trips starting at the two major airports in NYC: JFK and La Guardia. The figure also shows that most people go to Midtown (the darkest region), followed by the Upper West Side.

The choropleth (Fig. 8) shows that there are many more pickups around the train stations. Another interesting question is where passengers go. The choropleth (Fig. 8) shows that most people go to Midtown (the darkest region), followed by the Upper West Side.

The heat map also shows that while people take taxis in the number of trips to/from Harlem compared to other more affluent neighborhoods (see Fig. 7), we examine trips starting at the two major airports in NYC: JFK and La Guardia, we select the regions in their vicinity and examine trips starting at the two major airports in NYC: JFK and La Guardia.

The heat map also shows that while people take taxis to/from Harlem compared to other more affluent neighborhoods, the nightlife on weekends is very lively in Downtown. The heat map also shows that while people take taxis to/from Harlem compared to other more affluent neighborhoods, the nightlife on weekends is very lively in Downtown.

Another interesting question is where passengers go. The choropleth (Fig. 8) shows that most people go to Midtown (the darkest region), followed by the Upper West Side.

The heat map also shows that while people take taxis to/from Harlem compared to other more affluent neighborhoods, the nightlife on weekends is very lively in Downtown. The heat map also shows that while people take taxis to/from Harlem compared to other more affluent neighborhoods, the nightlife on weekends is very lively in Downtown.
**FINDINGS**

We got about the future of the data science, the most salient takeaway was how excited our respondents were about the evolution of the field. They cited things in their own practice, how they saw their jobs getting more interesting and less repetitive, all while expressing a real and broad enthusiasm about the value of the work in their organization.

As data science becomes more commonplace and simultaneously a bit demystified, we expect this trend to continue as well. After all, last year's respondents were just as excited about their work (about 79% were “satisfied” or better).

**How a Data Scientist Spends Their Day**

Here’s where the popular view of data scientists diverges pretty significantly from reality. Generally, we think of data scientists building algorithms, exploring data, and doing predictive analysis. That’s actually not what they spend most of their time doing, however.

As you can see from the chart above, 3 out of every 5 data scientists we surveyed actually spend the most time cleaning and organizing data. You may have heard this referred to as “data wrangling” or compared to digital janitor work. Everything from list verification to removing commas to debugging databases— that time adds up and it adds up immensely. Messy data is by far the more time-consuming aspect of the typical data scientist’s work flow. And nearly 60% said they simply spent too much time doing it.

![Chart showing data scientist job satisfaction and time spent on various tasks](image-url)

**What data scientists spend the most time doing**

- 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]
Data Wrangling

<table>
<thead>
<tr>
<th>Transaction Date</th>
<th>Customer Name</th>
<th>Phone Numbers</th>
<th>Address</th>
</tr>
</thead>
<tbody>
<tr>
<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>
</tr>
<tr>
<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>
</tr>
<tr>
<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>
</tr>
<tr>
<td>6/1 Jan-11 02:27:21</td>
<td>Smith, Jane</td>
<td>tel: 4250013981</td>
<td>4740 N 132nd St Apt 417, Omaha, 68164</td>
</tr>
<tr>
<td>2011-Jan-12</td>
<td>Anthony R Von Fange II</td>
<td>650-384-9911</td>
<td>10508 Prairie Ln, Oklahoma City</td>
</tr>
<tr>
<td>2010-Dec-24</td>
<td>Mr. Peter Tyson</td>
<td>(405)-123-3981</td>
<td>525 1st St, Marysville, WA 95901</td>
</tr>
<tr>
<td>9/2/2011</td>
<td>Dan E. Williams</td>
<td>1-650-1234183</td>
<td>211 W Ridge Dr, Waukon,52172</td>
</tr>
<tr>
<td>7/11/2012</td>
<td>James Davis Sr.</td>
<td>+1-425-736-9999</td>
<td>13120 Five Mile Rd, Brainerd</td>
</tr>
<tr>
<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>
</tr>
<tr>
<td>3/31/2013</td>
<td>Donald Edward Miller</td>
<td>(206) 309-8381</td>
<td>840 W Star St, Greenville, 27834</td>
</tr>
<tr>
<td>6/1/2009 12:01</td>
<td>Miller, Donald</td>
<td>206 309 8381</td>
<td>25571 Elba, Redford, 48239</td>
</tr>
<tr>
<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>
</tr>
<tr>
<td>1/4/2011 14:33</td>
<td>Daniel Chen</td>
<td>425 960 3566</td>
<td>1008 Whitlock Ave NW, Marietta, 30064</td>
</tr>
</tbody>
</table>

API is returned as the top result.

and mapping tables that we believe are important environment without switching.

from code libraries crawled at GitHub, and over 2M code snippets existed for decades – developers traditionally build custom code li-
new programs whose output are consistent with the given examples. Within a few seconds, a ranked list of programs are

ow. In a recent crawl, we extracted over 1.8M functions

ce Store for Excel.

can synthesize desired programs using relevant transformation . The
rst page. Copyrights for components of this work owned by others than the

y source code, DLLs, web services, and mapping
tions by examples, a paradigm known as program-by-example

TDE

TDE

TDE

TDE

TDE

TDE

TDE

TDE

TDE

© ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3183713.3193539

hallucinated

address, date time, phone number, etc., before they can perform

Demonstrations

Transform-Data-by-Example

2018. Transform-Data-by-Example

SIGMOD'18: 2018 Interna-

1

TDE

puter is just like a search engine “indexing” a new document.

this data in order to perform analysis. Suppose she wants to
cally extracted, and made immediately search-able. The way

for data transformation. (More details of the system can be

us-address, url, unit-conversion, etc. Many of these
del many head and tail domains, such as date-time, person-name,
target output, all within just a few seconds. Expert-users have the
programs, sometimes with multiple functions, to exactly match
exact output speci

c source code, DLLs, web services, and mapping
transformations cannot be handled by any existing system.

Since existing functions rarely produce the

tions by examples, a paradigm known as program-by-example

that

rst three values in column-D are provided as output examples. The desired

collected from di

same columns are highly heterogeneous, which can often happen

have no built-in support as it has not encountered and

websites is just like a search engine “indexing” a new document.

6

3

5

14

[209x202]SIGMOD'18, June 10–15, 2018, Houston, TX, USA

when data is collected from di

Figure 2:

(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'

[NLP Publishing Stats, M. Rei & R. Allen]
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."
Data Storage

SQL DATABASES

NoSQL DATABASES

Relational

Column

Graph

Key-Value

Document
Scaling Dataframes

[Diagram showing a process flow with stages: New Data Source, New spec, New requirements, Prototyping, Exploring, Testing, Production, Laptop/Workstation, Small Cluster, Large Cluster, MODIN, Feedback.]

[D. Petersohn]
Provenance and Reproducibility

Data Management

Computation

Visualization

DATA

Paper
Provenance and Reproducibility

Data Management

Provenance

Visualization

Paper

DATA

Computation

DATA

Visualization

Data Management

Provenance

Paper
About this course

• Course web page is authoritative:
  - faculty.cs.niu.edu/~dakoop/cs640-2023sp/
  - Schedule, Readings, Assignments will be posted online
  - Check the web site before emailing me

• 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:
  - Python for Data Analysis, W. McKinney
  - Effective Pandas, M. Harrison
  - Intro to Python, Deitel & Deitel
  - Python Data Science Handbook, J. VanderPlas

- Research papers
- Many websites
Syllabus Questions?
Class Roster Check
In this Notebook we explore the Lorenz system of differential equations:

\[
\begin{align*}
\dot{x} &= \sigma(y - x) \\
\dot{y} &= x \rho - y - xz \\
\dot{z} &= -xy + r
\end{align*}
\]

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, y, z = solve_lorenz(N=10)
```

```python
def solve_lorenz(N=10, max_time=4.0, sigma=10.0, beta=8.0/3, rho=28.0):
    """Solve the Lorenz differential equations."""
    fig = plt.figure()
    ax = fig.add_axes([0, 0, 1, 1], projection='3d')
    ax.set_xlim([-25, 25])
    ax.set_ylim([-35, 35])
    ax.set_zlim([5, 55])
    fig.add_subplot(111, projection='3d')

    # Prepare the axes limits
    x, y, z = x, y, z
    return [sigma * (y - x), x * (rho - z) - y, x * y - beta * z]

def lorenz_deriv(x_y_z, t, sigma=sigma, beta=beta, rho=rho):
    """Compute the time-derivative of a Lorenz system."""
    x, y, z = x_y_z
    return [sigma * (y - x), x * (rho - z) - y, x * y - beta * z]

# Choose random starting points, uniformly distributed from -15 to 15
x0 = -15 + 30 * np.random.random((N, 3))
```
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 or 3.10 version
- 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
JupyterLab Output Types

```python
[2]: a = 12
    for i in range(3):
        print("Some output")
    plt.bar([1,2,3,4],[20,30,15,40])
    plt.show()
    a + 3

stdout

Some output
Some output
Some output

output

[2]: 15

[3]: 1 / 0

ZeroDivisionError Traceback (most recent call last)
<ipython-input-3-bc757c3fda29> in <module>
    ----> 1 1 / 0

ZeroDivisionError: division by zero
```
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
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
• https://wiki.python.org/moin/BeginnersGuide
• https://wiki.python.org/moin/IntroductoryBooks
• 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
    
    ```python
    z = 20
    if x > 0:
      if y > 0:
        z = 100
      else:
        z = 10
    ```
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
  ```python
  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