Introduction - What is Data Science?

Data science is a term with many working definitions, but it generally refers to the practice of analyzing (typically large) data sets to reveal hidden relationships. In this course, we will take a very quick romp through some of the practice and tools commonly used by practitioners of data science.

Data Science Tools

Below are some of the common "tools" used:

- "R" programming language
- Spreadsheet software
- Jupyter (usually with Python, but R is also supported)
- Git (in concert with Github)
- The Internet (more of a resource, really)

For this course, we will use Jupyter (JupyterLab to be specific), with Python libraries for numeric and data analysis work, such as matplotlib (graphing), numpy (math libraries beyond the standard math package), pandas (for working with dataframes in an R-like manner). We may use spreadsheet software for comparison. We will use the Internet as a source of raw data and Git with Github as a way to archive, share, and publish our results.

Data Science Practice

Data science is often practiced as a process with (at least) the following steps:

1. Get or generate data
2. Clean the data to remove faulty information or outliers
3. Explore the data
4. Analyze the data
5. Present the data analysis in a way that allows it to be reproduced and critiqued

An Example: Projectile Range Analysis

For the first example, we will generate data and do some simple analysis using both a spreadsheet and Jupyter/Python. Specifically, we will look at how the range of a projectile is affected by its launch angle, using the range formula[1]: $d = \frac{v^2}{g}\sin 2\theta$

In this equation, *d* is the range in meters, *v* is the launch velocity in m/s, *g* is the gravitational constant (9.81 m/s^2), and $\theta$is the launch angle (with respect to horizon).

Analyze Using Google Sheets

First, use Google Sheets to build a table of values for at least 100 different angles, ranging from 0 to 90 degrees. For each angle, compute the projectile range, assuming an initial launch

[1] https://en.wikipedia.org/wiki/Range_of_a_projectile

velocity of 100 meters/second. Use the table to produce a graph that displays range as a function of angle.

I have a simple example of this Google Sheet[2].

What are the problems with this analysis?

- How sure are you that the graph is generated from the data?
- How sure are you that the computed range uses the correct formula?
- How sure are you that all of the formulas used to compute the range are the same?
- If you make modifications to it, how can you be sure that you haven't broken something important?
- If someone makes a copy of this analysis and changes it, how would you know what changed?
- This data set has fewer than 200 rows, so it's not hard to update the formula. How hard would it be if the data set had 1000 rows? 10000 rows?
- It's kind of ugly, though that's probably more to do with my inexperience at spreadsheet design.

What are the advantages of this analysis?

- It is easy to change the formula or initial conditions and see the graph update immediately.
- It was very quick to create and has user-friendly sharing features.
- No other websites or programming knowledge is required. Point 'n click rules!

Analyze Using Jupyter and Python

There is a big reason everyone isn't using Jupyter and Python to do things like this: availability and accessibility. Fortunately for you, Jupyter and Python are **available** and **accessible**.

Get Jupyter

If you have your own Windows or Mac or Linux computer, you can install Jupyter by installing the free version of Anaconda[3]. This is a very large install and I cannot provide support for it, however, it is probably the most popular way to use Jupyter today (as of 2019).

The **preferred** method of accessing Jupyter is **online** at the HHS Jupyter server[4]. This server provides a free user account for any visitor who has a valid hanovernorwichschools.org Google account. When you visit the site you must give it permission to access your Google account in order to use Jupyter. Your Google login is only used to verify that you are a Dresden user and to give you access to your files on Google Drive.

[2] http://bit.ly/2JTVpQe
[3] https://www.anaconda.com/distribution/
[4] https://hhscp.org

The following process also assumes that you have a Github[5] account. If you don't have a Github account, make one now (for free) using your hanovernorwichschools.org alias e-mail account.

Create a Notebook and Synchronize to a Git Repository

Follow these initial steps to set up a synchronized repository on Jupyter and Github:

- In Jupyter, make a new folder called **datascience**. Double-click on it to navigate into it.
- Use the Jupyter launcher to open a terminal. Use a bash command to navigate into your new datascience folder:

  $ cd datascience

- Initialize git in the datascience folder:

  $ git init

- Although nothing changes in the folder view pane on the left, you now have a hidden git folder. You can see it by typing the following in bash:

  $ ls -a

  This does a directory listing, showing **hidden** files. In Unix/Linux, file names that begin with a period are hidden from view by default. The -a option with ls forces bash to show you **a**ll files, including hidden ones. You should see a hidden file (a directory name, actually) called .git.
- Back in the Jupyter launcher, add a Python3 notebook. This should create a new file in your folder. Rename the file as **ballistics**.
- Back in bash, issue the following command:

  $ touch gitignore

- Click the refresh button on the file pane. You should see the gitignore file appear alongside your ballistics.ipynb file. Double click on it to open it in the Jupyter editor. In the file pane, rename gitignore to be .gitignore (insert the period before the name).
- Back in bash, issue the following command:

  $ git status

  The status should show several red files: ballistics.ipynb, .gitignore, and .ipynb_checkpoints/. Use your mouse to copy the .ipynb_checkpoints/ name and paste it into your .gitignore editor window. Save the file. Now return to bash and type:

[5] https://github.com/

$ git status

This should only show ballistics.ipynb and .gitignore now.

- Back in the ballistics pane, let's play around with the markdown editor. Figure out what the following navigation commands do: j, k, b, a, x, c, v, dd, y.
- In the first cell, set it to be in markdown mode and enter the following:

```
# Ballistics and the Range Equation
This is an investigation of the so-called *range equation*: $d=\frac {v^2} g
\sin 2\theta$.

You can write text with *italics*.
You can write text with **bold**.

You can begin new paragraphs by inserting a blank space.

You can
* write text
* with bullets

You can insert
1. Numbered
2. Bullets

This also supports several levels of subheading

## Sub-heading 1
### Sub-heading 2
#### Sub-heading 3

And you can add ***bold and italic*** text!

Do you want to add a hyperlink? How about the documentation for markdown at
[Daring Fireball](https://daringfireball.net/projects/markdown/syntax). See
the reference for more challenging situations.
```

- Press shift-enter to display your markdown cell. This is NOT what we want in our final ballistics analysis, but it is a good demonstration of what Jupyter can do in the way of producing good looking text alongside your analysis. SAVE your notebook.

- Let's get your notebook synchronized with Github. Back in bash, type:

  $ git status

- You should see your notebook file and the .gitignore file that you also created, but not the checkpoints folder. Now let's get it all committed:

  $ git add .
  $ git commit -m "new notebook"

- Visit your Github account and make a new repository called datascience. Do NOT create it with a Readme or .gitignore file. Next, you should see an unfamiliar page with lots of instructions for what to do next. Find the section that says ".. push an existing repo from the command line" and copy the bash commands shown there. Paste them into your bash shell and follow the prompts.
- Refresh your Github repository page. It should show your new ballistics.ipynb file and the .gitignore file. From here on, only edit your repository in Jupyter. Use the (now familiar) git add/commit/push sequence of commands to push your Jupyter changes up to Github.
- Open your ballistics.ipynb file on Github. Looks just like your view in Jupyter, right? If you click the "raw" button in Github, you can see what the file really looks like. Github has built-in Jupyter notebook support! In fact, Github is possibly the single most popular method for archiving and sharing Jupyter notebooks on the Internet. Welcome to the future!

Do the Ballistics Analysis in Jupyter

First of all, feel free to clean up the text in the first cell so that it only talks about the ballistics problem.

Add a code cell with:

```
from math import radians, sin, pi
from numpy import linspace
import matplotlib.pyplot as plt
```

Don't forget to press shift-enter to execute each cell after you add it!

Add a code cell below that and write a function that computes range from initial velocity and angle. Start with the following, then fill in the rest:

```
def rangeformula(v, theta):
    """
```

```
    Inputs:
    v - velocity in m/s
    theta - launch angle with respect to horizon in radians
    Returns:
    Impact distance in m
    """
```

Open another code cell and test your function.

Open another cell and define a list of input angles with:

```
angledegrees = range(0,91)
```

Then use a list comprehension to produce a list of range values (assuming a velocity of 100 m/s) and assign it to a variable called rangefromdegrees. When you call your rangeformula function you will need to convert the angle from degrees to radians using the radians() function.

To plot your results, open another cell and enter:

```
plt.plot(angledegrees, rangefromdegrees)
plt.title('Range as a function of angle at 100 m/s')
plt.xlabel('Angle (degrees)')
plt.ylabel('Range (meters)')
plt.show()
```

This worked pretty well but didn't we use more angles in the spreadsheet version? The range function only produces integer values of angles. The way to fix this is to use the linspace function, which works like this:

```
angledegrees = linspace(0, 90, num=200)
```

Notice that by default, the second argument is the last value in the range (different from how range function works, right?). The number of numbers in the range is given by the num keyword argument. Go to the Jupyter run menu and select "run all cells" to incorporate this change and to make sure all the cells are executed in the correct order. How does it look?

Multiple Data Sets in One Plot

You can incorporate multiple curves in one plot. For example:

```
rangeforangle100 = [rangeformula(100, radians(angle)) for angle in angledegrees]
rangeforangle200 = [rangeformula(200, radians(angle)) for angle in angledegrees]
```

```
plt.plot(angledegrees, rangeforangle100, label="100 m/s")
plt.legend()
plt.plot(angledegrees, rangeforangle200, label="200 m/s")
plt.legend()
plt.title('Range as a function of angle at various velocities')
plt.xlabel('Angle (degrees)')
plt.ylabel('Range (meters)')
plt.show()
```

Publish Your Work

Once you have finished with your notebook (and the exercises below), publish commit and push your notebook up to Github.

You have several additional options for sharing your Jupyter notebooks:

- Export as HTML, open in a browser and print your notebook. Looks pretty good.
- Export as HTML, save as static web pages on a website. I have an example of this[6] that I am working on in my spare time.
- Copy the Github link to your notebook and submit it on the official Jupyter website notebook viewer page[7].

Exercises

- Extend your notebook to graph range as a function of angle in radians.
- Extend your notebook to graph range as a function of initial velocity, for an angle of 45 degrees (or pi/4 radians).
- Extend your notebook to graph range as a function of initial velocity, for 15, 30, 45, 60, 75 degrees, all on one plot.
- Extend your notebook with a markdown section that discusses the advantages and disadvantages of the Jupyter notebook, when compared with a spreadsheet.
- Publish your notebook with the Jupyter nbviewer and submit the link to your teacher!

[6] https://tiggerntatie.github.io/emagnet-py/index.html
[7] https://nbviewer.jupyter.org/