Jupyter Organization Tips

Introduction

All of these slides are **recommendations**. There is no one correct way.

The goal is not just to get to the correct answer, but to do so in a way that is **readable** and **understandable** to you and to anyone else looking at your code!

On the job, you will be handing your code off to other people (and working with code that others wrote). Your code will not be worth anything if no one else can figure out what you did, or, even worse, if you can’t decipher your own work.

General Guidelines for Python Code

PEP 8 - Style Guide for Python Code

Guidelines to improve readability and consistency of code.

Recommendations

- When creating a new notebook, give it a meaningful name. Otherwise, you'll end up with *Untitled.ipynb*, *Untitled1.ipynb*, … and have no idea where any of your work is.
- Take advantage of the fact that you can do Markdown, but also include comments.
  - Comments: code-specific
  - Markdown: annotations and explanations

Recommendations

- Keep imports in the first cell
- Remove old/unnecessary code - eg. you don't need to have multiple *.head()* calls in your final notebook
- You should be able to *Reset and Run All* at any point. Check this periodically, especially if you are cleaning up and removing old code.
- Use logical and human-readable variable names (not *df* and especially not *df2* for DataFrames).
    - Don't worry about making names too long. Tab-completion is your friend.
- Be as consistent as you can with capitalization. A lot of people use *all_lowercase_with_underscores*

Recommendations

- Combine together multiple cells if they don’t produce output
- Take advantage of method chaining

```
In [4]: crashes = crashes.drop(columns = ['Weather Code', 'Illumination Code', 'Harmful Code', 'Mapped Location'])

In [5]: crashes = crashes.rename(columns = lambda col: col.lower().replace(' ', '_'))

In [6]: crashes = crashes.dropna(subset = ['latitude', 'longitude'])
```

```
In [4]: crashes = crashes.drop(columns = ['Weather Code', 'Illumination Code', 'Harmful Code', 'Mapped Location'])
        crashes = crashes.rename(columns = lambda col: col.lower().replace(' ', '_'))
        crashes = crashes.dropna(subset = ['latitude', 'longitude'])
```

Recommendations

- Combine together multiple cells if they don’t produce output
- Take advantage of method chaining

```
In [4]: crashes = crashes.drop(columns = ['Weather Code', 'Illumination Code', 'Harmful Code', 'Mapped Location'])

In [5]: crashes = crashes.rename(columns = lambda col: col.lower().replace(' ', '_'))

In [6]: crashes = crashes.dropna(subset = ['latitude', 'longitude'])
```

```
In [4]: crashes = (crashes
                   .drop(columns = ['Weather Code', 'Illumination Code', 'Harmful Code', 'Mapped Location'])
                   .rename(columns = lambda col: col.lower().replace(' ', '_'))
                   .dropna(subset = ['latitude', 'longitude'])
                  )
```

Recommendations

- Think carefully about **exploration** vs **explanation**.
  - Exploration:
    - Usually just for you.
    - Getting to know the dataset
    - Testing hypotheses, looking for patterns, etc.
  - Explanation: What you want to show to other people.
- As we work on later projects, it can be useful to have multiple notebooks for the various stages of the data science process:
  - Initial cleaning and preparation
  - Exploratory Analysis
  - Model Building
  - Presentation

Recommendations

- Covert repetitive code blocks into reusable functions.
- If you find yourself copying and pasting code, you are probably better off to write a function.
- **DRY**: Don’t Repeat Yourself
- With functions, you only need to make changes in one place.
- For simple tasks, you can also consider using *for* loops or list comprehensions.

Other Recommendations

This notebook on creating reproducible research in Jupyter:
https://www.kaggle.com/rtatman/reproducible-research-best-practices-jupytercon

Cookiecutter Data Science:
https://drivendata.github.io/cookiecutter-data-science/

- A template for data science/analytics work
- Probably overkill for us, but there are some useful principles that I recommend adopting (eg. a folder for data and a folder for notebooks, using sensible naming conventions for your notebooks).