Data Science Foundations

Lesson #2 - Data Science Platforms

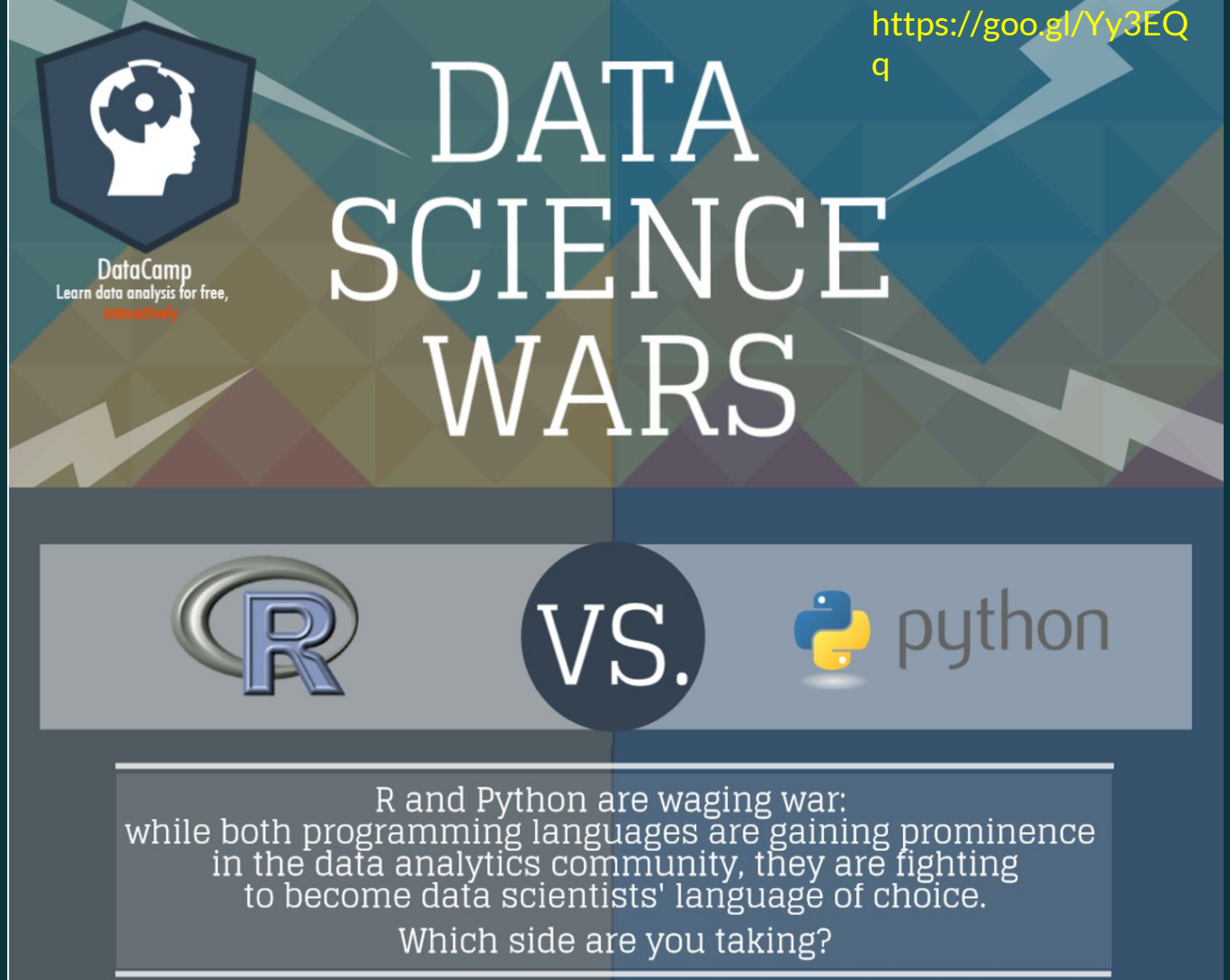
Ivanovitch Silva

August, 2017

Agenda

- Data Science War
- Anaconda
- My First Notebook
- Version Control System
- Intro to Python for Data Science

https://goo.gl/Yy3EQq



https://goo.gl/2mQm2K

If you come from a C.S./developer background, you'll probably feel more comfortable with Python. On the other hand, if you come from a statistics/analyst background, R will likely be more intuitive

Language Rank	Types	Spectrum Ranking
1. C		100.0
2. Java		98.1
3. Python		98.0
4. C++		95.9
5. R		87.9
6. C#		86.7
7. PHP		82.8
8. JavaScript		82.2
9. Ruby		74.5
10. Go		71.9

IEEE Spectrum - Jul 2016 https://goo.gl/BqrkDI

Language Rank	Types	Spectrum Ranking
1. Python		100.0
2. C		99.7
3. Java		99.5
4. C++		97.1
5. C#		87.7
6. R		87.7
7. JavaScript		85.6
8. PHP		81.2
9. Go		75.1
10. Swift		73.7

IEEE Spectrum - Jul 2017 https://goo.gl/HSPLWe

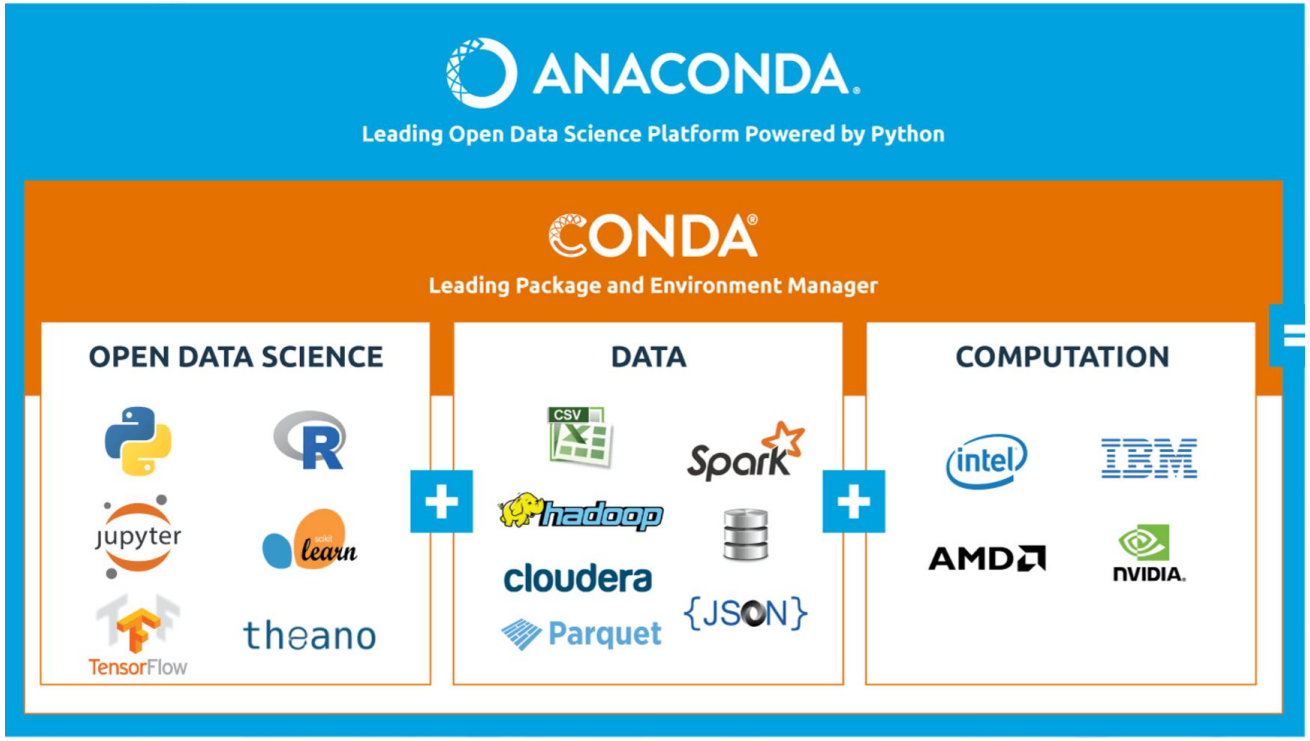
Version 3.x (https://www.python.org/downloads/)

ANACONDA

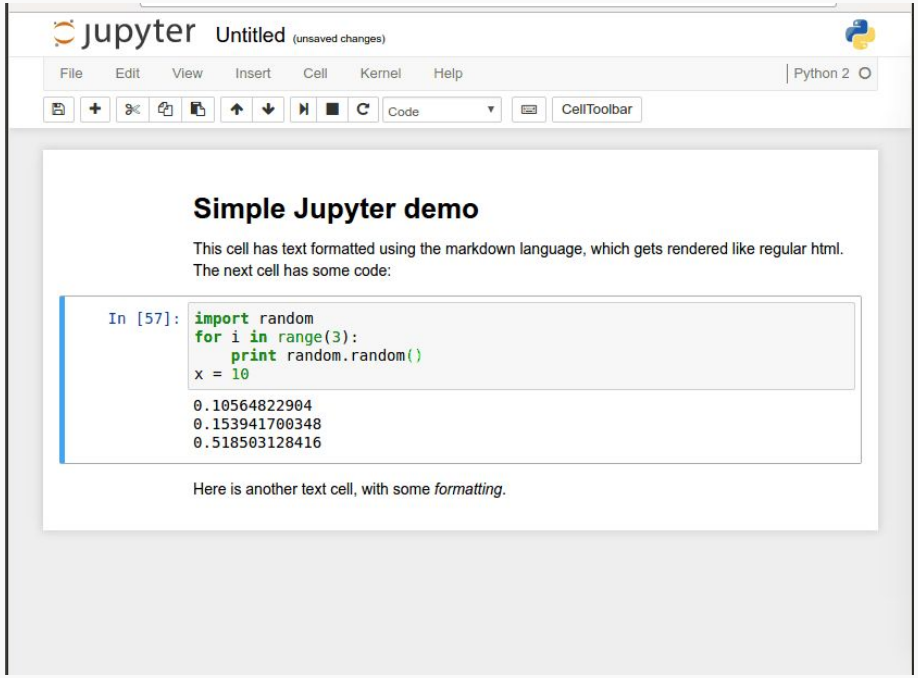
Modern open source analytics platform
powered by Python

https://www.continuum.io/downloads

Why Anaconda?



What is a Jupyter Notebook?



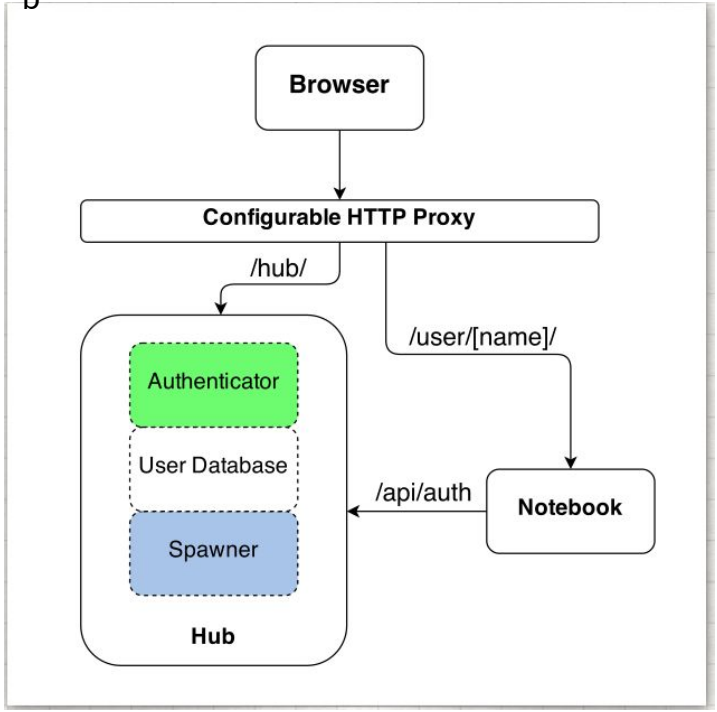
Mix of code and rich elements (text, figures, links, equations, etc)

Aside from JUlia, PYThon and R (JUPYTER) notebook technology also supports many other languages. https://github.com/jupyter/jupyter/wiki/Jupyter-kernels

https://goo.gl/LfEx5r

JupyterHub

https://github.com/jupyterhub/jupyterhub



JupyterHub can be used to serve notebooks to a class of students, a corporate data science group, or a scientific research group.

CONTINUUM ANALYTICS

Gallery About Pricing Anaconda Help Download Anaconda Sign In

Search Anaconda Cloud

Where packages, notebooks, and environments are shared.

Powerful collaboration and package management for open source and private projects.

Public projects and notebooks are always free.

Private plans start at $7/month.

Sign Up Sign In

New to Anaconda Cloud? Sign up!

Pick a username

Your email

Use at least one lowercase letter, one numeral, and seven characters.

Create a password

Confirm password

I accept the Terms & Conditions

Sign up!

By clicking "Sign up!" you agree to our privacy policy and terms of service. We will send you account related emails occasionally.

ANACONDA NAVIGATOR

Upgrade Now

Sign in to Anaconda Cloud

Home

Environments

Projects (beta)

Learning

Community

Documentation

Developer Blog

Feedback

Applications on root

Channels

Refresh

glueviz

0.9.1

Multidimensional data visualization across files. Explore relationships within and among related datasets.

Launch

notebook

4.3.1

Web-based, interactive computing notebook environment. Edit and run human-readable docs while describing the data analysis.

Launch

qtconsole

4.2.1

PyQt GUI that supports inline figures, proper multiline editing with syntax highlighting, graphical calltips, and more.

Launch

spyder

3.1.2

Scientific PYthon Development EnviRonment. Powerful Python IDE with advanced editing, interactive testing, debugging and introspection features

Launch

orange3

3.4.1

Install

rstudio

1.0.136

A set of integrated tools designed to help you be more productive with R. Includes R essentials and notebooks.

Install

GitHub

An extremely brief tutorial

Introduction to Git

script.py

```
if __name__ == "__main__":
    print("Welcome to a script!")
```

you

```
import math
print(10 + 10)
if __name__ == "__main__":
    print("Welcome to a script!")
```

coworker

```
if __name__ == "__main__":
    print("Welcome to a script!")
    print("Here's my amazing contribution to this project!")
```

merge

```
import math
print(10 + 10)
if __name__ == "__main__":
    print("Welcome to a script!")
    print("Here's my amazing contribution to this project!")
```

Installing Git

Downloads

Mac OS X Windows

Linux Solaris

Older releases are available and the Git source repository is on GitHub.

GUI Clients

Git comes with built-in GUI tools (**git-gui**, **gitk**), but there are several third-party tools for users looking for a platform-specific experience.

View GUI Clients →

Logos

Various Git logos in PNG (bitmap) and EPS (vector) formats are available for use in online and print projects.

View Logos →

https://git-scm.com/downloads

First step: create a repository (repo)

1. Create a folder named DataScience.
2. Navigate into this folder and initialize a Git repository (git init)
3. Run ls -la to check the contents of the DataScience folder

Creating files in the repo

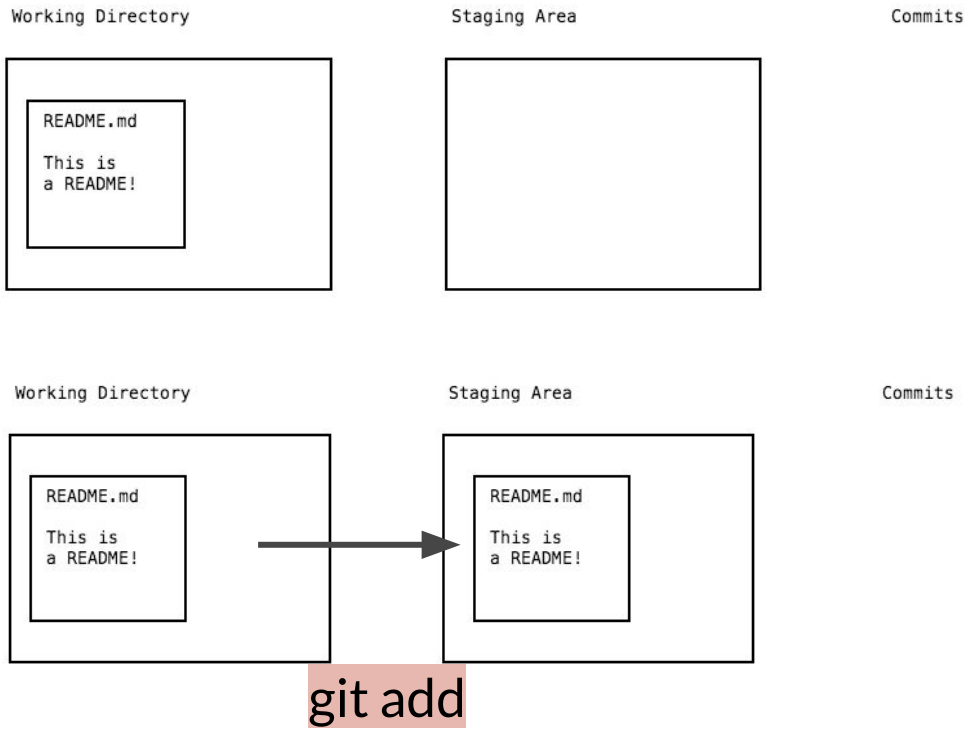
1. Create a file named `README.md` with the following content:

> ## My first git project

2. Create a file named `script.py` with this content:

```
if __name__ == "__main__":
    print("10")
```

Checking file status



Verify the status of files: git status

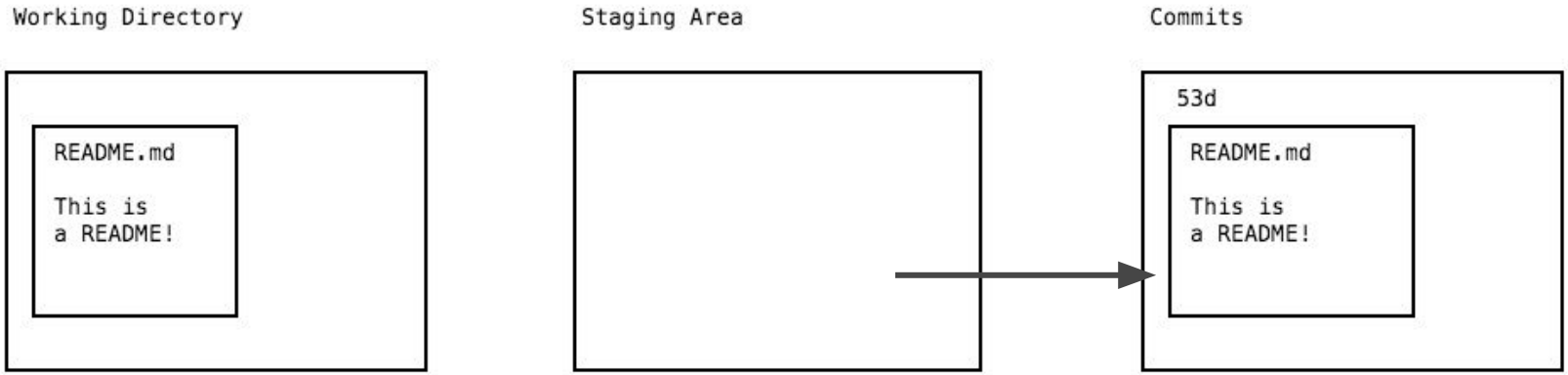
1. Check the status of the repo.
2. Add script.py to the staging area.
3. Add README.md to the staging area.

Configuring identity in Git

```
git config --global user.email "your.email@domain.com"
```

```
git config --global user.name "Your name"
```

Committing changes



git commit

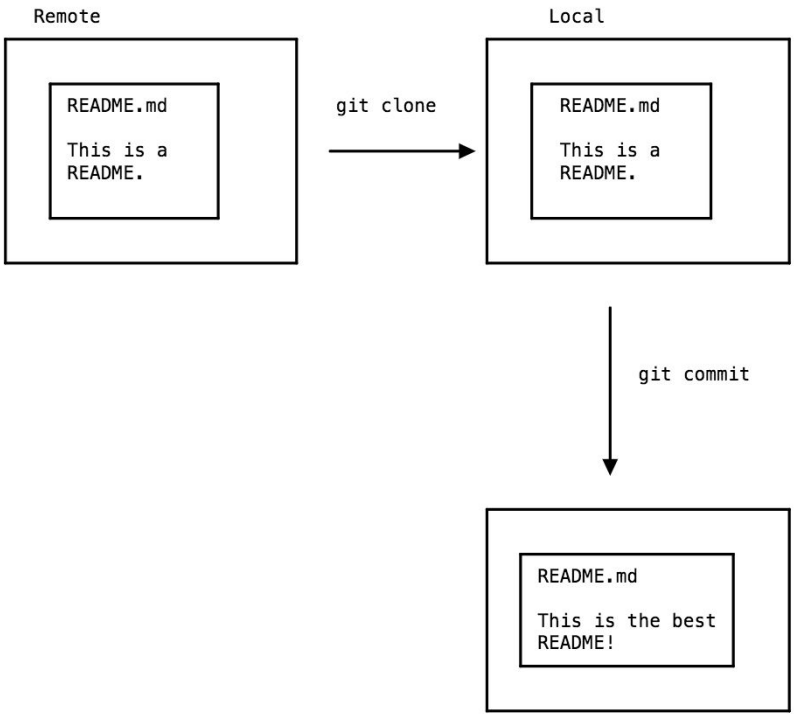
Type `git commit -m "Initial commit. Added script.py and README.md"` to make the first commit to the repository with an informative message.

Reviewing the commit history

Description:

1. Run git log to explore the commit history of the repository.

Remote repositories



- Share our code with others and build a portfolio
- Collaborate with others on a project and build code together.
- Download and use code others have created

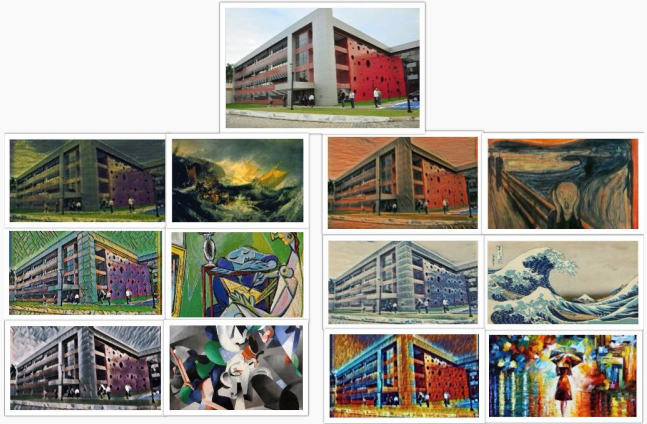
Remote repositories

Here's how we'd typically clone the Amazon Deep Learning repo from GitHub:

- git clone https://github.com/amznlabs/amazon-dsstne.git

Remote repositories [exercise]

1. Clone the "fast style transfer" project from Github to your local repository.
2. https://github.com/lengstrom/fast-style-transfer
3. Show history from git log
4. Clone the material of course



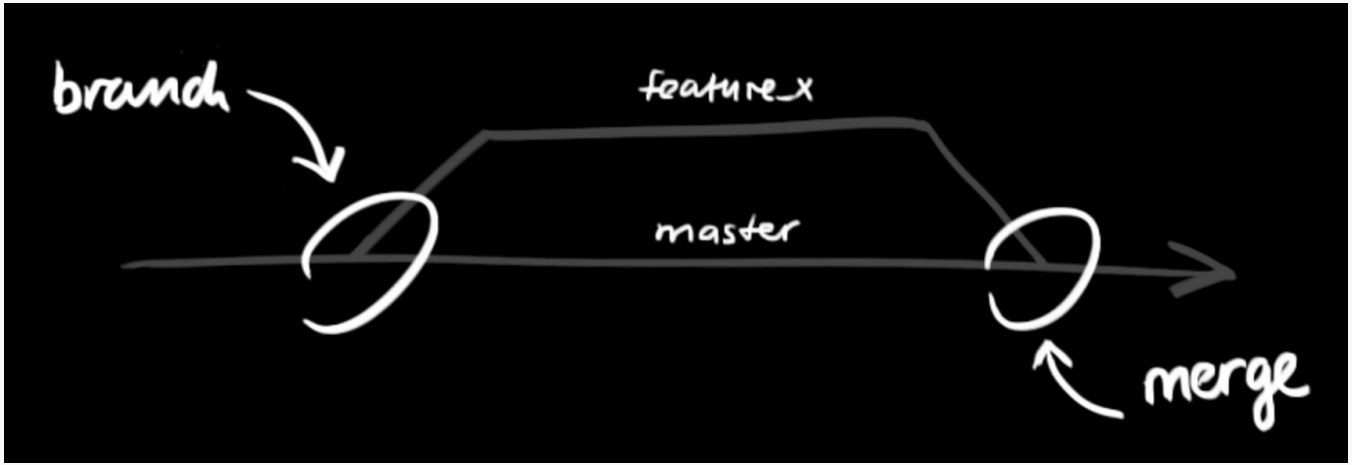
… go back to "DataScience" repo!!!

Github integration

Create a Github account

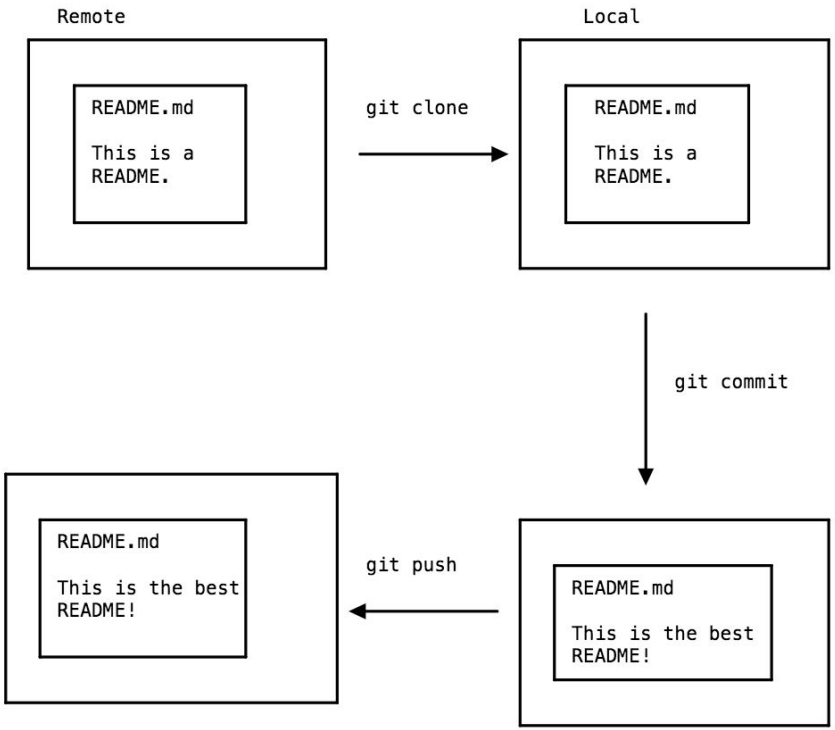
- Create a personal account. Select a unique username and password and enter your email.
- Choose a plan. If you select the free plan, all of your code (which is organized in repositories) will be public. Select the free plan for now. You can always upgrade to a paid plan later on, which would allow you to have private repositories.
- Read the GitHub Hello World guide.

Branch on repository



- Every Git repository consists of one or more branches.
- The main branch of a Git repo is typically called master.
- Use the git branch command to visualize the current branch of project

Pushing repo to Github



Pushing repo to Github [Exercise]

- Use the git remote command to visualize information about the repo.
- Create a repository in Github

```
git remote add origin https://github.com/<your_github_user>/hello-world.git
git push -u origin master
```

See the following notebooks for additional info

Git and a Version Control - Introduction to Git.ipynb
Git and a Version Control - Git Remotes.ipynb
Git and a Version Control - Git Branches.ipynb


MacBook Pro

Warming up

- Python versions
- Basic data types
- List
- Files and Loops
- If statements
- Dictionaries
- Functions and Packages

Notebook: "Warming_up.ipynb"

Lists

```
# Create the areas list
areas = ["hallway", 11.25, "kitchen", 18.0, "living room",
         20.0, "bedroom", 10.75, "bathroom", 9.50]

# Print out areas
print(areas)

# Print out the type of areas
print(type(areas))

# Print out second element from areas
print(areas[1])

# Print out last element from areas
print(areas[-1])

# Print out the area of the living room
print(areas[-5])

# Add two new elements to the end of the list
areas.append("laundry")
areas.append(8.75)
```

Files and Loops

```
#create an empty list
int_crime_rates=[]

#print the rate of crimes for each city using a list(int)
for i in rows:
    int_crime_rates.append(int(i.split(",")[1]))
```

Dictionaries

(key,value)

```
# From string in countries and capitals, create dictionary europe
europe = {'spain':'madrid','france':'paris','germany':'berlin','norway':'oslo'}

# Print europe
print(europe)

# Print out the keys in europe
print(europe.keys())

# Print out value that belongs to key 'norway'
print(europe['norway'])
```

Dictionaries of Dictionaries

```
# Dictionary of dictionaries
europe = { 'spain': { 'capital':'madrid', 'population':46.77 },
           'france': { 'capital':'paris', 'population':66.03 },
           'germany': { 'capital':'berlin', 'population':80.62 },
           'norway': { 'capital':'oslo', 'population':5.084 } }


# Print out the capital of France
print(europe['france']['capital'])

# Create sub-dictionary data
data = {'capital':'rome', 'population':59.83}

# Add data to europe under key 'italy'
europe['italy'] = data
```

Loop over dictionaries

```
world = { "afghanistan":30.55,
          "albania":2.77,
          "algeria":39.21 }
for key, value in world.items() :
    print(key + " -- " + str(value))
```

Modules

```
#Import the math module as m
import math as m

#Use the sqrt() function from the math module
root = m.sqrt(33)

print(root)
```


MacBook Pro

Introduction to Python for Data Science

- Set
- Missing Values
- Try/Except blocks
- Enumerate
- List comprehensions

Notebook "Introduction to Python for Data Science I.ipynb"

Set

When exploring data, it's often useful to extract the unique elements in a *list*

```
["Dog", "Cat", "Hippo", "Dog", "Cat", "Dog", "Dog", "Cat"]
```

```
["Dog", "Cat", "Hippo"]
```

Set

```
unique_animals = set(["Dog", "Cat", "Hippo", "Dog", "Cat", "Dog", "Cat"])
print(unique_animals)
```

```
unique_animals.add("Tiger")

unique_animals.remove("Dog")

unique_animals.add("Tiger")
```

Missing values

Missing values are very common in real world data analysis, since the people compiling the datasets often don't have full information. What to do when it occurs?

```
rows = [
    ["Bassett", "Richard", "1745-04-02", "M", "sen", "DE", "Anti-Administrat
ion"],
    ["Bland", "Theodorick", "1742-03-21", "", "rep", "VA", ""]
]
for row in rows:
    if row[6] == "":
        row[6] = "No Party"
```

Try/Excpet blocks

```
numbers = [1,2,3,4,5,6,7,8,9,10]
for i in numbers:
    try:
        int('')
    except Exception:
        print("There was an error")
```

Enumerate

```
animals = ["Dog", "Tiger", "SuperLion", "Cow", "Panda"]
viciousness = [1, 5, 10, 10, 1]
for animal in animals:
    print("Animal")
    print(animal)
    print("Viciousness")
```

???? how to print *viciousness*??

Enumerate

```
animals = ["Dog", "Tiger", "SuperLion", "Cow", "Panda"]
viciousness = [1, 5, 10, 10, 1]
for i, animal in enumerate(animals):
    print("Animal")
    print(animal)
    print("Viciousness")
    print(viciousness[i])
```

List Comprehensions

```
animals = ["Dog", "Tiger", "SuperLion", "Cow", "Panda"]
animal_lengths = []
for animal in animals:
    animal_lengths.append(len(animal))
```



```
animal_lengths = [len(animal) for animal in animals]
```

References

- http://rogerdudler.github.io/git-guide/
- http://product.hubspot.com/blog/git-and-github-tutorial-for-beginners
- http://www.dataquest.io/
- http://www.datacamp.com/
- https://www.datacamp.com/community/tutorials/tutorial-jupyter-notebook#gs.2H4cgDM
- http://nbviewer.jupyter.org/github/jakevdp/PythonDataScienceHandbook/blob/master/notebooks/Index.ipynb

References

- http://www.kdnuggets.com/2017/06/7-steps-mastering-data-preparation-python.html
- https://blog.codecentric.de/en/2017/07/combining-social-network-analysis-topic-modeling-characterize-codecentrics-twitter-friends-followers/
- https://medium.com/intuitionmachine/why-teaching-will-be-the-sexiest-job-of-the-future-a-i-economy-b8e1c2ee413e

Lesson #2