Week 01: Introduction

Data Science Bootcamp
Summer, 2021

Instructor: Sagar Patel

About the Bootcamp

- This is a **beginner's** level programme
  - It is okay if you can not program in Python!
- The **advanced** batch of this programme will start in **Fall** *(more updates on that later)*
- The bootcamp has been designed to help you prepare for technical interviews which incline towards these types of opportunities

Timeline

Week 1	Week 2	Week 3	Week 4	Week 5	Week 6
Introduction	SQL	Python libraries	Exploratory Data Analysis	Machine Learning	Project & Wrap!

Communities

- Join the **Slack** community to not miss out on any announcements and updates
  Link: https://join.slack.com/t/nyu-dsbc-s21/shared_invite/zt-reskrjo0-eCwFfCmnVYnTAc82lTVNJw
- Share your **GitHub** Username on **#general** to be added to the NYU Data Science Bootcamp Organization where all the resources and tasks will be available after each session
  - If you do not have a GitHub account, create one!
- You can also email us at **datasciencebootcamp@nyu.edu**

slido

How comfortable are you using Python?

ⓘ Start presenting to display the poll results on this slide.

Python, n.:

The best thing to happen to students and researchers

¯_(ツ)_/¯

History

- Conceived in the late *1980s* by **Guido van Russom**
- Release dates:
  - Version 1.0 -- December 1989
  - Version 2.0 -- October 16, 2000
  - Version 3.0 -- December 3, 2008

As of June 2021, the latest version is
Python 3.9.5

Guido Van Russom

What makes Python so great?

- Easy to read, learn and write
- Very **productive**
  - No need to spend time in understanding the syntax of the language
- Dynamically typed
  - The data type is assigned to the variable during execution
- Vast libraries support
  - Using **Python package manager** (pip) makes it easier to import external packages
- It's **FREE!**

But also...

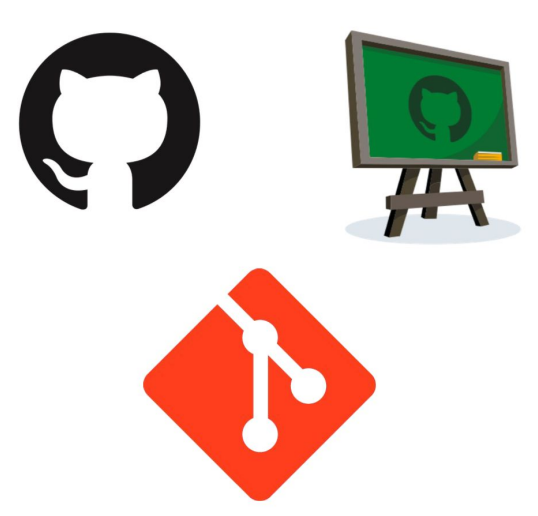
- Python is very **slow**
  - The line by line execution of code often leads to slow execution
- Not memory-efficient
  - A **large** amount of **memory** is consumed during execution
- Runtime errors
  - Since Python is a **dynamically typed** language, the data type of the variable can change anytime
  - A variable containing integer may hold a string in the future

Let’s get started!

Git Basics

What is Git?

- **Git** is a system for creating and updating a distributed source code control repository
- **GitHub** is a website that allows for **free storage** of public repositories (you can also call them “repos”)

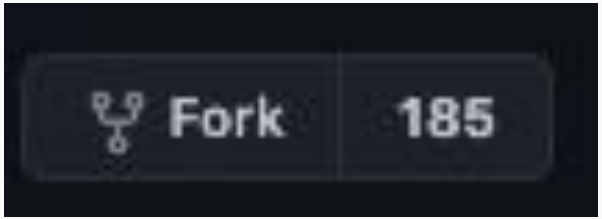


Setting up git

- **Mac** users generally have git installed on their system
- For **Windows** users, follow instructions on this link:
  - Windows Subsystem for Linux: https://docs.microsoft.com/en-us/windows/wsl/install-win10
  - Or, Install git on Windows: https://help.github.com/articles/set-up-git/#setting-up-git

Forking a repository

- A Fork refers to a copy of the repository. Forking a repository helps you to experiment with it however we like without affecting the original repository
- Once can fork a repository by clicking the fork option at the top right corner of a GitHub repository page
- For more information, follow: https://help.github.com/articles/fork-a-repo/



Forking a Repository

- After forking, in order to “clone” or download it to your local machine, navigate to the Code and copy the link to the repository
  - If you are using HTTPS, copy the link
  - If you are using SSH, refer: https://help.github.com/en/github/authenticating-to-github/generating-a-new-ssh-key-and-adding-it-to-the-ssh-agent
- Enter the following command on the terminal (Mac), command line (LINUX/WSL) or Git Bash (Windows):

git clone <paste link here>

Some Basic Commands

- **Pull**ing any latest changes from the repository to your local machine

  git pull origin master
- Checking the **status** to see the files which have been staged and unstaged in the local repository on your local machine

  git status
- **Staging** an unstaged file for a commit

  git add <filename>
  - Can also stage *all the files* by using

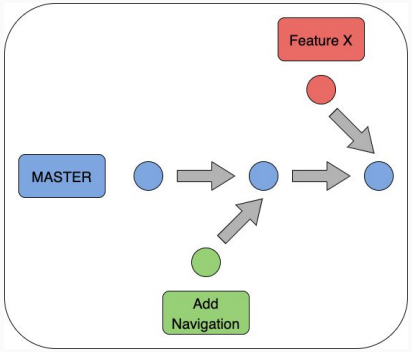
    git add .
- It is important to **commit** your staged files as they will not be pushed otherwise

  git commit -m “<your message here>”
- Finally, to **push** the code with all the changes made locally

  git push origin master

Branching and Merging

- Entering the command **git checkout -b <branch name>** we can **create** a new branch and switch to it at the same time
- Use **git push origin <branch name>** to **push** any changes made to this branch
- In order to **switch** to the master branch, use **git checkout master**
- To **merge** all the changes made on the new branch to the master branch, use **git merge <branch name>**



That’s all Folks!

See you in the next session :)