CSX415
Data Science Principals and Practice
02 Data Science Environment

Christopher Brown
U.C. Berkeley / Decision Patterns LLC
Spring 2018

Berkeley
UNIVERSITY OF CALIFORNIA

Data Science Stack

Tools and Technologies

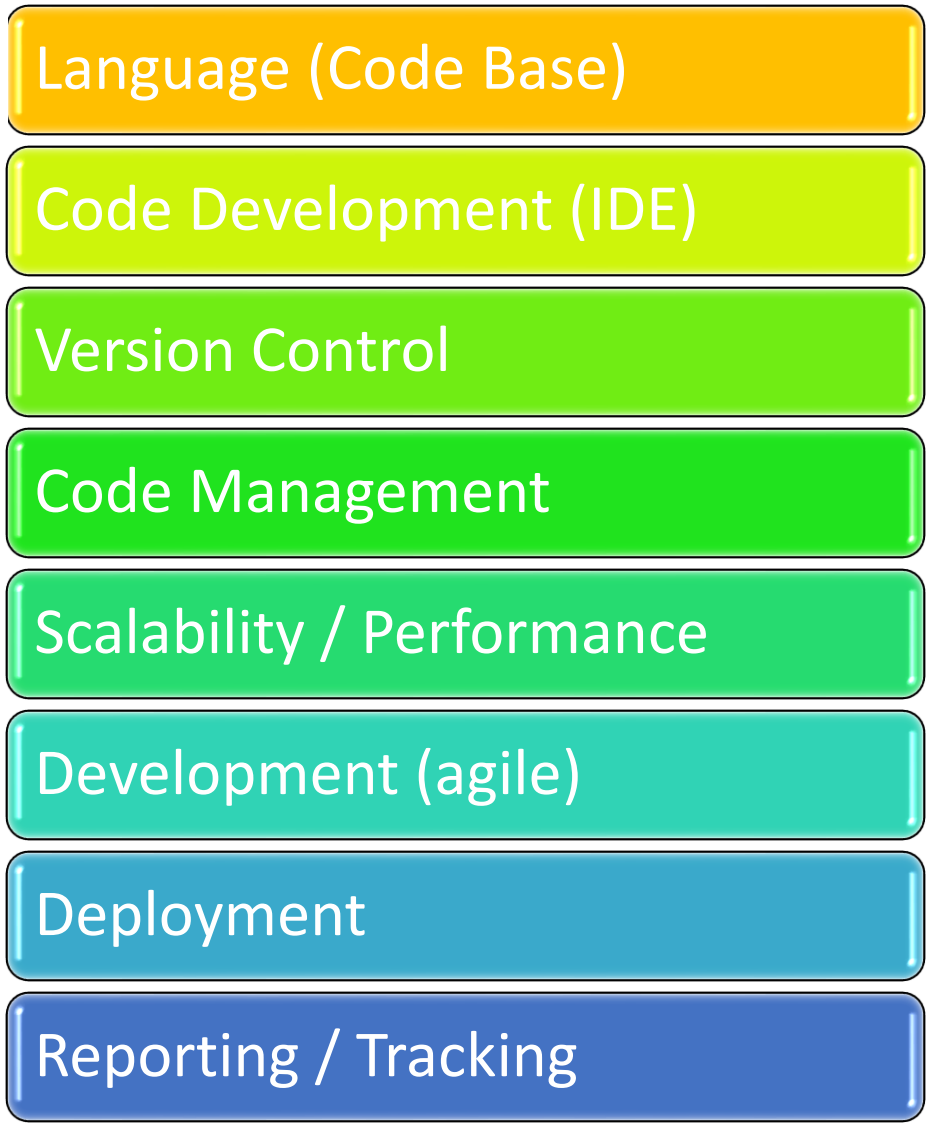
There is no single ***Data Science Application*** but

a ***Collage of Technologies and Application*** to manage lifecycle tasks

… the following is an ***opinionated*** list

Applications Stack

- Organizations choose balance of applications for each of these tasks
- Limited good choices for each
- Most of the stack is usually *Open Source*



Language (Code Base)

Others: Spark/Scala, Julia, SAS

- Language (Code Base)
- Code Development (IDE)
- Version Control
- Code Management
- Scalability / Performance
- Development (agile)
- Deployment
- Reporting / Tracking

Code Development (IDE)

- Language (Code Base)
- **Code Development (IDE)**
- Version Control
- Code Management
- Scalability / Performance
- Development (agile)
- Deployment
- Reporting / Tracking

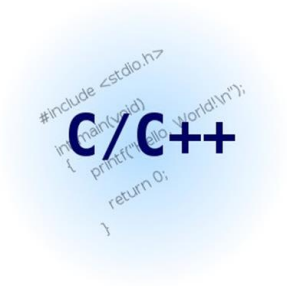
Version Control

- Language (Code Base)
- Code Development (IDE)
- **Version Control**
- Code Management
- Scalability / Performance
- Development (agile)
- Deployment
- Reporting / Tracking

Code Storage / Management

- Language (Code Base)
- Code Development (IDE)
- Version Control
- **Code Management**
- Scalability / Performance
- Development (agile)
- Deployment
- Reporting / Tracking

Scalability / Performance



On Prem / Native

SaaS

- Language (Code Base)
- Code Development (IDE)
- Version Control
- Code Management
- **Scalability / Performance**
- Development (agile)
- Deployment
- Reporting / Tracking

Development (agile)

- Language (Code Base)
- Code Development (IDE)
- Version Control
- Code Management
- Scalability / Performance
- **Development (agile)**
- Deployment
- Reporting / Tracking

Deployment

Varies widely on requirements, but

Bokeh

- Language (Code Base)
- Code Development (IDE)
- Version Control
- Code Management
- Scalability / Performance
- Development (agile)
- **Deployment**
- Reporting / Tracking

Reporting / Tracking / Reproducibility / Literate Programming

Varies widely, but

- Language (Code Base)
- Code Development (IDE)
- Version Control
- Code Management
- Scalability / Performance
- Development (agile)
- Deployment
- **Reporting / Tracking**


git

Git

What is it?

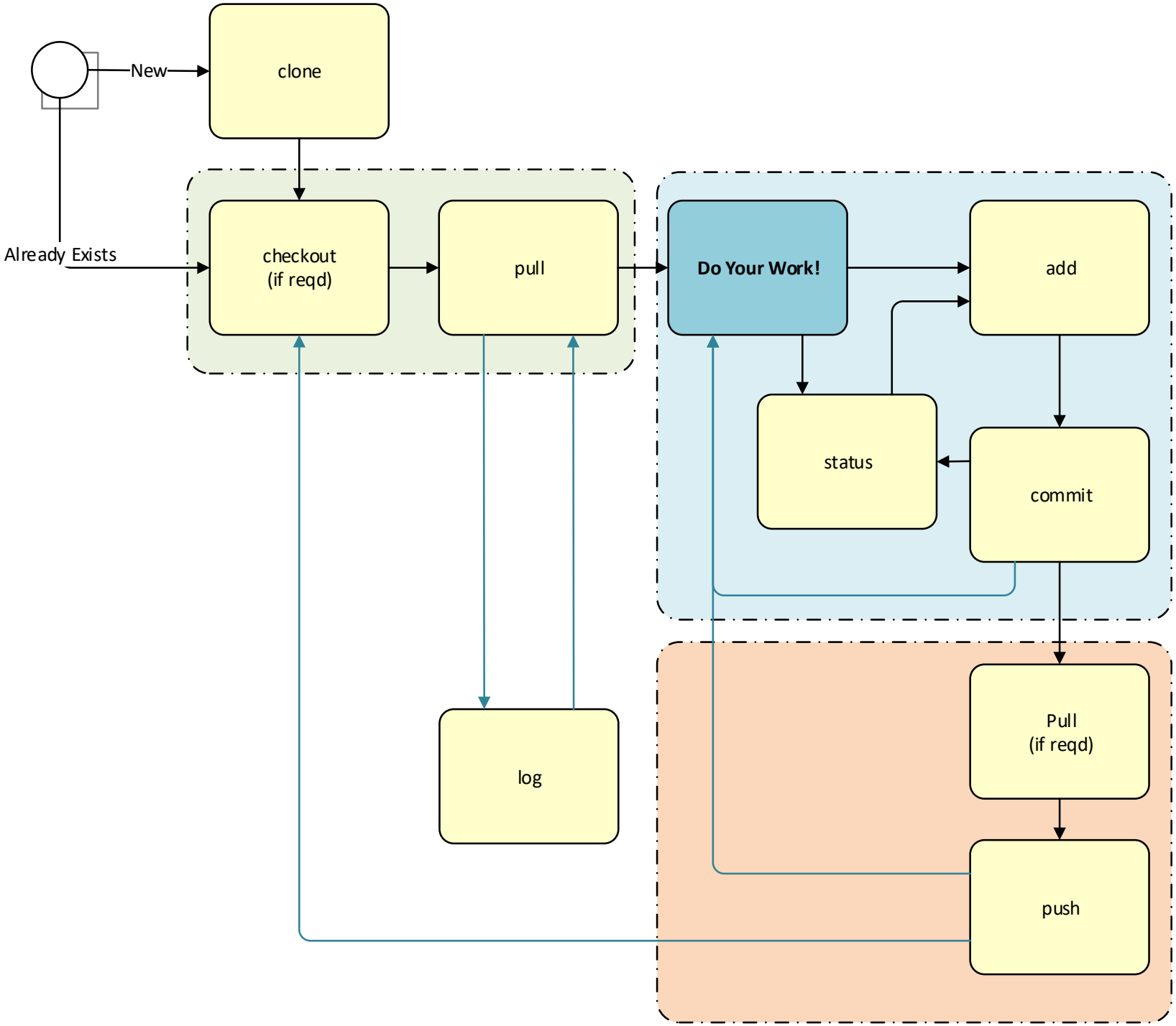
A source control tool (and ***process***) to promote **collaborative** development

Features

- Distributed
- Each clone contains complete history
- Ability to return to
- Branch in and merging

Interfaces

- Github (directly)
- Source Tree™
- *R Studio*
- ***command-line***



source: Decision Patterns

Concepts

- **Repo(sitory)**
  location where files are stored. If different from original source: “fork”
- **Remote**
  The **named** storage **location** of a repository
- **Branch**
  Copy of code that can be independently worked on (without affecting other branches).
  - **Master Branch (**usually) the place most stable and accepted collections of code

- **Staged**
  Changes to code that are deemed “ready to be accepted” as part of the project.
- **Commit**
  Changes that have been accepted as part of the project

Git Commands

Git commands all have the same format

```
git verb [options] [target]
```

Verbs types

- Navigation

GIT VERBS for NAVIGATION :

- **clone** [remote]
  create a local, indep. copy of repo.
- **branch** [name]
  create a local branch named [name]
- **checkout** [branch]
  switch to [branch]

Git Commands

Git commands all have the same format

```
git verb [options] [target]
```

Verbs types

- Navigation
- Editing

GIT VERBS for EDITING

- **add** [files]
  Tell which changed files to “stage” (accept) commit. Done on a per-file basis. Each file must be added, wildcards (*) are accepted.

- **commit** [files] [-m’comment’]
  Accept changes. This permanently fixes added files into the working branch.

Git Commands

Git commands all have the same format

```
git verb [options] [target]
```

Verbs types

- Navigation
- Editing
- Synchronization

GIT VERBS for SYNCHRONIZATION:

- **pull** [remote] [branch]
  retrieve changes from the [branch] of a [remote] (repository)
- **push** [remote] [branch]
  send committed changes to [branch] of a remote repository.
- **clone** [remote]
  create a local, indep. copy of repo.

Git Commands

Git commands all have the same format

```
git verb [options] [target]
```

Verbs types

- Navigation
- Editing
- Synchronization
- Informational

GIT VERBS for INFORMATION:

- **log**
  review history of commits
- **status**
  review “staged” status. See what files have changed. Useful in conjunction with **add/commit.**