

# GitHub

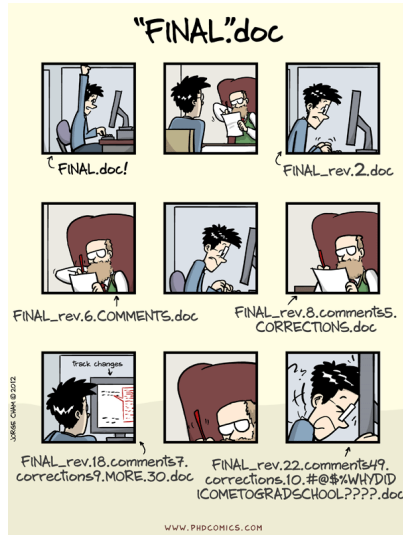
## Big Data y Machine Learning para Economía Aplicada

Ignacio Sarmiento-Barbieri

Universidad de los Andes

# Intro to Git(Hub)

## Motivation



Source: [http://phpcomics.com/comics/archive\\_print.php?comicid=1531](http://phpcomics.com/comics/archive_print.php?comicid=1531)

# Intro to Git(Hub)

## Motivation

### ► Git

- Git is a distributed version control system
- It is optimized for the things that economists and data scientists spend a lot of time working on (e.g. code).

### ► GitHub

- It's important to realize that Git and GitHub are distinct things.
- GitHub is an online hosting platform built on top of the Git system. (Like Bitbucket and GitLab.)
- Just like we don't *need* Rstudio to run R code, we don't *need* GitHub to use Git

# Let's get hands on

- Let's get hands on



Source: [http://phdcomics.com/comics/archive\\_print.php?comicid=1531](http://phdcomics.com/comics/archive_print.php?comicid=1531)

# Collaboration time

- ▶ This is what I want you to do
  - ▶ Partner 1: Invite Partner 2 to join you as a collaborator on the GitHub repo that you created earlier. (See the Settings tab of your repo.)
  - ▶ Partner 2: Clone Partner 1's repo to your local machine. Make some edits to the README (e.g. delete lines of text and add your own). Stage, commit and push these changes. Make sure you change into a new directory first, with a different name so you don't have conflicts with your own
  - ▶ Partner 1: Make your own changes to the README on your local machine. Stage, commit and then try to push them (\*after\* pulling from the GitHub repo first).

# Collaboration time

... and we are back

- ▶ Did Partner 1 encounter a 'merge conflict' error?

# Collaboration time

... and we are back

- ▶ Did Partner 1 encounter a 'merge conflict' error?
- ▶ Check what is going on

```
$ gitstatus
```

Unmerged paths:

(use "git add <file>..." to mark resolution)

```
* both modified:   README.md
```

- ▶ Git is protecting P1 by refusing the merge. It wants to make sure that you don't accidentally overwrite all of your changes by pulling P2's version of the README.

# Collaboration time

```
# README
```

```
Some text here.
```

```
<<<<<< HEAD
```

```
Text added by Partner 2.
```

```
=====
```

```
Text added by Partner 1.
```

```
>>>>>> 814e09178910383c128045ce67a58c9c1df3f558.
```

```
More text here.
```



# Collaboration time

- ▶ Fixing these conflicts is a simple matter of (manually) editing the README file.
  - ▶ Delete the lines of the text that you don't want.
  - ▶ Then, delete the special Git merge conflict symbols.
- ▶ Once that's done, you should be able to stage, commit, pull and finally push your changes to the GitHub repo without any errors.
  - ▶ P1 gets to decide what to keep because they fixed the merge conflict.
  - ▶ The full commit history is preserved, so P2 can always recover their changes if desired.
  - ▶ Another solution is using branches