

Lec1: Intro & Git

Isidoro Garcia Urquieta

2021

Objetivos del curso

- ▶ Este es un curso de Data Science/Advanced Analytics. El objetivo es proveerlos de las herramientas estadísticas y de programación desde el enfoque de economistas.

Objetivos del curso

- ▶ Este es un curso de Data Science/Advanced Analytics. El objetivo es proveerlos de las herramientas estadísticas y de programación desde el enfoque de economistas.
- ▶ Al final de curso espero que puedan:

Objetivos del curso

- ▶ Este es un curso de Data Science/Advanced Analytics. El objetivo es proveerlos de las herramientas estadísticas y de programación desde el enfoque de economistas.
- ▶ Al final de curso espero que puedan:
- ▶ Construir modelos predictivos de Regresión o Clasificación (Random Forests, LASSO, XgBoosting)

Objetivos del curso

- ▶ Este es un curso de Data Science/Advanced Analytics. El objetivo es proveerlos de las herramientas estadísticas y de programación desde el enfoque de economistas.
- ▶ Al final de curso espero que puedan:
- ▶ Construir modelos predictivos de Regresión o Clasificación (Random Forests, LASSO, XgBoosting)
- ▶ Construir modelos de inferencia causal combinados con *Machine Learning*

Objetivos del curso

- ▶ Este es un curso de Data Science/Advanced Analytics. El objetivo es proveerlos de las herramientas estadísticas y de programación desde el enfoque de economistas.
- ▶ Al final de curso espero que puedan:
- ▶ Construir modelos predictivos de Regresión o Clasificación (Random Forests, LASSO, XgBoosting)
- ▶ Construir modelos de inferencia causal combinados con *Machine Learning*
- ▶ Usar técnicas de reducción de dimensionalidad (K-means, PCA)

Objetivos del curso

- ▶ Este es un curso de Data Science/Advanced Analytics. El objetivo es proveerlos de las herramientas estadísticas y de programación desde el enfoque de economistas.
- ▶ Al final de curso espero que puedan:
- ▶ Construir modelos predictivos de Regresión o Clasificación (Random Forests, LASSO, XgBoosting)
- ▶ Construir modelos de inferencia causal combinados con *Machine Learning*
- ▶ Usar técnicas de reducción de dimensionalidad (K-means, PCA)
- ▶ Hacer uso de las herramientas más utilizadas en la industria de Data Science (R, Github)

Objetivos del curso

- ▶ Este es un curso de Data Science/Advanced Analytics. El objetivo es proveerlos de las herramientas estadísticas y de programación desde el enfoque de economistas.
- ▶ Al final de curso espero que puedan:
- ▶ Construir modelos predictivos de Regresión o Clasificación (Random Forests, LASSO, XgBoosting)
- ▶ Construir modelos de inferencia causal combinados con *Machine Learning*
- ▶ Usar técnicas de reducción de dimensionalidad (K-means, PCA)
- ▶ Hacer uso de las herramientas más utilizadas en la industria de Data Science (R, Github)
- ▶ Entender como combinar teoría económica con Data Science.

Un poco sobre mi

- ▶ Soy Economista del ITAM y Master en Uchicago.
- ▶ Ahora trabajo como Head of Data Science en Bitso.
- ▶ Antes de eso fui Director de Analytics, Modeling and Experimentation en Banorte. En esta área fundé el equipo de Experimentación basada en Economía Conductual (Behavioral Economics) y el área de Machine Learning relacionado a la Econometría. Pasé por el camino tradicional de Banxico (Inveco), Research Assistant y Banco Mundial.
- ▶ En este tiempo conocí muchos data scientists que o: tienen las habilidades computacionales pero no saben construir relaciones entre variables (no son economistas...) o saben teoría económica pero no saben programar ni conocen algoritmos que involucren Big Data.
 - ▶ Los economistas que sepan combinar teoría (costo de oportunidad, marginalidad, etc) con herramientas estadísticas y de programación pueden conquistar el mundo de Data Science.

Diferencias entre BD, Econometría, Data Science, ML y AI

Big data, Econometría, Data Science, Machine Learning, Artificial Intelligence son varias de las palabras que escuchamos sobre cosas similares.

- ▶ La **econometría** se enfoca primordialmente en la inferencia de **parámetros**. A partir de ahí, se utilizan estos modelos para:
 - ▶ Inferir relaciones causales entre variables,
 - ▶ Predecir la variable endógena del modelo.
 - ▶ De cualquier manera, la econometría hace un énfasis fuerte en basarse en teoría para entender el fenómeno e construir intuición sobre él.
- ▶ **Big Data** se refiere a la extracción de valor (insight y/o predicciones) de bases de datos gigantes (i.e. muchas columnas y/o muchas filas) que no caben en una computadora. Es mucho más pragmático.

Diferencias entre BD, Econometría, Data Science, ML y AI

- ▶ **Data Science** se define como el uso de métodos científicos para extraer conocimiento y valor de los datos (estructurados y no estructurados).
 - ▶ Esto involucra los métodos para analizar los datos: Estadística, Econometría y Machine Learning; y
 - ▶ Computer Science: Los métodos para recibir, guardar y estructurar los datos (Data Engineering)
- ▶ **Machine Learning** son una colección de algoritmos ‘tontos’ que aprenden cosas solos. Esto es, algoritmos de estadística y econometría + esteroides (Re-muestreo y mucha capacidad de cómputo).
- ▶ Finalmente, **Artificial Intelligence** es una colección de algoritmos que aprenden con más autonomía que los algoritmos de machine learning.

De que es este curso entonces?

Los algoritmos que mencionen cubren casi todo el espectro de Data Science menos la ingeniería de datos. Estos cubren:

- ▶ **Métodos Supervisados:** Tienes una variable éndogena y que gobierna el aprendizaje estadístico.
- ▶ **Métodos No Supervisados:** Sólo tienes X' s y no variable endógena.
- ▶ **Métodos Preescriptivos:** Queremos inferir relaciones causal. (Hint: Mucho más allá de ATE)

Finalmente, haré énfasis en el **Exploratory Data Analysis (EDA)** en todo el curso. El uso de gráficas y tablas es CRUCIAL para lograr comunicar efectivamente a audiencias no técnicas.

Temario

Tendremos 17 sesiones. En ellas vamos a tener ~12 sesiones teóricas, 3 sesiones con invitados de la industria y 2 donde se presenten sus proyectos finales. Vamos al temario

- ▶ El repositorio contiene el temario, los pdfs de las clases, tareas y exámenes.
- ▶ Ahí se harán las entregas y subiré todo lo relacionado al curso.

Qué es Git

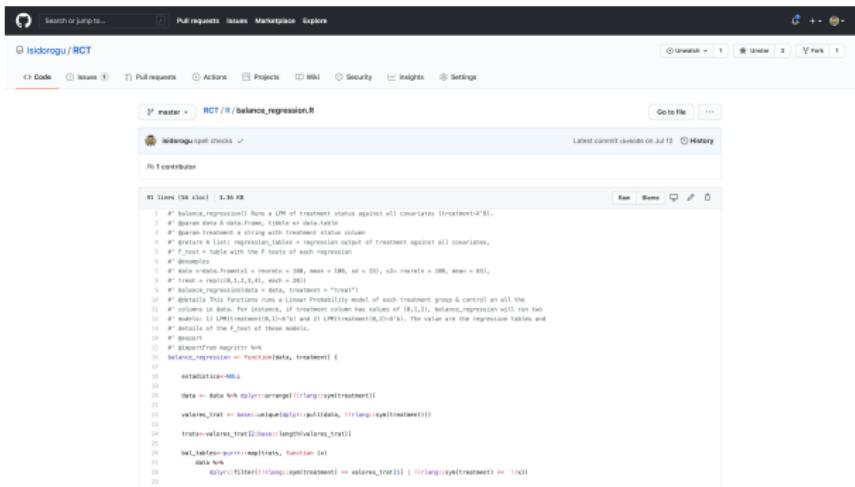
Es un sistema distribución open source para el control de versiones de código & trackeo del mismo durante el desarollo de software.

- ▶ Es como un Google Drive + Control de cambios (pero más cool).



Qué es Github

Github es una plataforma en línea para guardar repositorios (folders) e interactuar con Git de forma mas amigable.



The screenshot shows a GitHub repository interface. The top navigation bar includes links for Search or jump to..., Pull requests, Issues, Marketplace, and Explore. Below the bar, the repository name is shown as isidorogu/RCT. The main content area displays the file RCT/R/balance_regression.R. The code in the file is as follows:

```

1 #`t` balance_regressions() runs a LPM of treatment status against all covariates (treatment~*R).
2 #`t` passes data & data.frames, table or data.table
3 #`t` generates a string with treatment status column
4 #`t` prints a t-test, which compares the F statistic of treatment against all covariates,
5 #`t` F_stat = table with the F tests of each regression
6 #`t` descriptives
7 #`t` dependent variable = outcome ~ 100, mean = 100, sd = 10; obs = outcome ~ 100, mean = 85;
8 #`t` treat = replicate(1,1,t,k,q), each = 200
9 #`t` balance_regressions(data, treatment = "treat")
10 #`t` takes a data.table and creates a model of each treatment group & control on all the
11 #`t` columns of data. For instance, if treatment column has values of (1,2,1), balance_regressions will run two
12 #`t` models: 1) LPM(treatment~0,1)*t(0) & 2) LPM(treatment~0,2)*t(1). The value are the regression tables and
13 #`t` p-values of the F-test of these models.
14 #`t` depends on: dplyr, rlang
15 #`t` disappears from namespace
16 balance_regression = function(data, treatment) {
17   #`t` depends on: dplyr, rlang
18   #`t` extractTables NULL
19   #`t` data ~ data.table[, .by = treatment]
20   #`t` 
21   #`t` values_treat ~ base::unique(data[, treatment])
22   #`t` values_ctrl ~ base::unique(data[, -treatment])
23   #`t` 
24   #`t` treat=values_treat[1] * length(values_ctrl)
25   #`t` 
26   #`t` left_join(data, filter(data, treatment %in% values_treat))
27   #`t` data ~ data[filter(data, treatment == values_treat) | (rlang::syms(treatment) == 1)]
28   #`t` 
29 }

```

Vayan todos a abrir su cuenta en Github. Recuerden su mail y user.name

Instalación de Git

Pasos:

1. Ir a <https://git-scm.com>,
2. Descargar Git para tu sistema operativo (Windows, Mac, Linux),
3. Una vez instalado, Git va a mostrarte Git Bash en Windows. Para Mac, Git va a estar listo desde la Terminal.
 - Desde ambos puedes empezar a usar Git

Despues en Git Bash/Terminal:

```
git config --global user.name 'YOUR NAME' #El usuario  
git config --global user.email 'email@yemail.com'  
git config -l # Ver el resultado
```

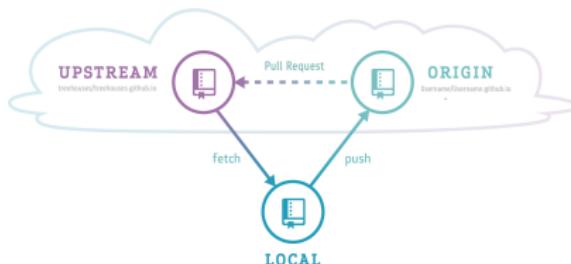
Repo local y remoto

Usar git te va a ayudar a controlar versiones en tus proyectos. Git se organiza en repositorios:

Repositorio: Es un “folder” donde vas a guardar todos los archivos de tu proyecto. Normalmente vas a tener un **Repositorio local (master)** y un **Repositorio Remoto (origin/upstream)** que se comunican entre sí.

Con esto tienes:

- ▶ Acceso al repositorio desde donde sea (como la nube en Google Drive)
- ▶ Acceso a cualquier versión del repositorio (versión por cambio)
- ▶ Una interfaz de control de cambios/versiones muy completa



SSH keys

Vamos a configurar la comunicacion entre local y remote. Para Windows, abran Git Bash. Para Mac abran Terminal. Corran lo siguiente:

```
cd ~/ #ve a home  
ssh-keygen -t rsa #Crea un par de keys
```

Esto genera un par de contraseñas (pública y privada) que se comunican entre sí de manera encriptada. La pública se guarda en Github y la privada se queda en tu computadora.

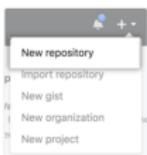
Despues ve a Github -> Settings -> SSH and GPG keys -> New SSH key. Sube la key publica. Nombrala algo como 'mi Mac'.

Con esto, puedes subir/bajar archivos entre tus repositorios de manera trivial

Crear remote desde un local existente

1. Creamos un repo 'bare' vacío en Github

1 Create a new repository on GitHub. To avoid errors, do not initialize the new repository with README, license, or .gitignore files. You can add these files after your project has been pushed to GitHub.



2. Copiamos la información SSH del repo remoto

A screenshot of a GitHub repository page for 'crossfit'. The page shows a single file, 'README.md', with the content 'crossfit'. On the right side, there is a 'Code' dropdown menu open, showing four options: 'Clone' (with links for HTTPS, SSH, GitHub CLI, and a copy icon), 'Open with GitHub Desktop', and 'Download ZIP'. The 'SSH' link is highlighted with a blue border.

3. En Terminal/Git Bash

```
cd './ITAM_Eco_Computacional/' # Cambias a donde estan los archivos
git init # Conviertes a git repository
git add . # Añadir todo '.' lo que esta a stage
git commit -m "Subo archivos"
#La opcion m es 'message'
git remote add origin git@github.com:username/repo.git
# Añade repo remoto al local
git remote -v # Verifica que esten conectados
git push -u origin main
# push 'empuja' los archivos al remoto '-u'
# 'u' es upstream
```

Listo! Tu repo local y remoto están sincronizados

Crear un repo local desde un remoto existente

```
cd './ITAM_Eco_Computacional/'  
# Cambias al directorio donde quieres que esten los archivos  
git clone git@github.com:username/repo.git  
# Clonas al repo remoto al local
```

Listo!

Como trabajar despues del set up

Ya tienes el repo local y remoto configurados. Ahora qué?

- ▶ Trabaja normal en tus archivos en R (Stata, Python, Word, ...)
- ▶ Ve a Terminal/Git Bash y cambia al directorio con cd
- ▶ Checa como se ve todo con git status
- ▶ Haz git add, git commit y git push

Lista de comandos

Veamos la lista de comandos básicos

comando	descripcion
status	Status del repo
checkout	Moverme a alguna version de cambio o branch
add	Añadir archivos al stage
commit	Comprometer los cambios en add
push	Empujar los cambios al repo remoto
remote	Hacer algun cambio del repo remoto al que se conecta el local
fetch	Buscar los cambios hechos en el repo remoto
merge	Combinar tu repo local con los cambios que jalaste en fetch
pull	git fetch + git merge
rebase	Mueve el repo local a un punto particular. cambios secuenciales
branch	Crea copia del repo remoto para trabajo paralelo

git status

Te muestra el estado de tu repo local vs el remoto

Hay cambios que no hiciste git add

```
@Harish KINGM64 /e/ToolsQA/First Project (master)
$ git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)
    modified: ABC.txt
no changes added to commit (use "git add" and/or "git commit -a")
$ | @Harish KINGM64 /e/ToolsQA/First Project (master)
```

Ya que hiciste git add, te dice que falta que hagas git commit. De cualquier manera, te muestra lo que cambio. Muy bueno para llevar control!

```
@Harish KINGM64 /e/ToolsQA/First Project (master)
$ git add ABC.txt
$ git status
@Harish KINGM64 /e/ToolsQA/First Project (master)
$ git status
On branch master
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)
    new file: ABC.txt
$ | @Harish KINGM64 /e/ToolsQA/First Project (master)
```