INSTRUCTIVO: Desafío de Machine Learning en Economía de la Salud

Cómo Obtener los Materiales del Curso

Materia: Aplicaciones en Ciencia de Datos
Docente: Francisco Fernández
Institución: Universidad Nacional del Oeste (UNO)
Año: 2025

Objetivo de este Instructivo

Este documento te va a guiar paso a paso para:

1. **Crear tu cuenta** en GitHub (si no la tenés)
2. **Hacer un Fork** del repositorio del desafío (crear la copia del grupo)
3. **Clonar** el Fork del grupo a tu computadora
4. **Trabajar colaborativamente** con tus compañeros
5. **Entender** qué son Git y GitHub (herramientas esenciales en ciencia de datos)

Tiempo estimado: 20-25 minutos (primera vez)

PARTE 1: ¿Qué es todo esto? (Contexto Teórico)

¿Qué es Git?

Git es un sistema de **control de versiones** creado por Linus Torvalds (el creador de Linux) en 2005.

¿Para qué sirve?

- Mantener un **historial completo** de todos los cambios en tu código
- **Volver atrás** a versiones anteriores si algo se rompe
- **Colaborar** con otras personas en el mismo proyecto sin pisarse
- Crear **ramas** para experimentar sin afectar el código principal

Analogía: Imaginá que estás escribiendo tu tesis. Git es como tener un sistema que automáticamente guarda:

- `Tesis_v1.docx`
- `Tesis_v2_con_correciones.docx`
- `Tesis_v3_final.docx`
- `Tesis_v4_final_AHORA_SI.docx`

...pero **mucho más inteligente**: Git sabe exactamente QUÉ cambió, CUÁNDO, y QUIÉN lo cambió. Además, solo guarda las diferencias (no copia todo el archivo cada vez), ahorrando espacio.

¿Qué es GitHub?

GitHub (propiedad de Microsoft desde 2018) es una **plataforma web** que:

- 🌐 Hospeda repositorios Git en la nube (como un "Google Drive para código")
- 🤝 Permite colaboración: varios programadores trabajando juntos
- 📊 Ofrece herramientas de gestión de proyectos, revisión de código, y más
- 🌍 Es la red social de los programadores (más de 100 millones de usuarios)

Alternativas a GitHub: GitLab, Bitbucket, Gitea (pero GitHub es el más popular)

¿Por qué lo usamos en este curso?

- ☑ Distribución fácil de materiales actualizados
- ☑ Ves cómo trabajan los profesionales de datos
- ☑ Podés llevar este conocimiento a tus futuros trabajos
- ☑ Podés ver el historial de cambios si actualizamos el desafío

---

Conceptos Clave de Git/GitHub

Concepto	Definición	Analogía
Repositorio (repo)	Una carpeta con todo el proyecto y su historial	Tu carpeta de la materia, pero con superpoderes
Fork	Tu propia copia de un repositorio de otra persona	Fotocopiar un libro entero para hacer anotaciones sin arruinar el original
Commit	Una "foto" del estado del proyecto en un momento	Punto de guardado en un videojuego
Clone	Descargar una copia completa del repositorio	Descargar un proyecto de Google Drive
Pull	Actualizar tu copia local con cambios nuevos	Sincronizar con la nube
Push	Subir tus cambios a GitHub	Guardar tus cambios en la nube
Branch	Una línea alternativa de desarrollo	Trabajar en un borrador sin tocar el original
Remote	La versión del repositorio en la nube (ej: GitHub)	El servidor donde está el "original"

¿Por qué hacer un FORK? (MUY IMPORTANTE)

Situación:

- El profesor tiene el repositorio original: `github.com/panchtox/health_economics_challenge`
- **Trabajan en GRUPOS de 3 personas**
- Cada grupo necesita su propia copia para trabajar

Opciones:

❌ **Clonar directamente el repo del profesor:**

- Solo podrían descargar
- No podrían subir SUS cambios a GitHub
- No tendrían su propia versión del proyecto

☑ **UN integrante del grupo hace el Fork:**

- Crea la copia del grupo: `github.com/USUARIO_DEL_GRUPO/health_economics_challenge`
- Los TRES trabajan en la rama `main` de ese Fork
- Pueden subir sus cambios colaborativamente
- Mantienen una conexión con el original (por si hay actualizaciones)

Flujo correcto para trabajar EN GRUPO:

```
Repo del Profesor (panchtox)
        ↓ FORK (lo hace UN integrante)
Repo del Grupo en GitHub (usuario_del_grupo) - rama main
        ↓ CLONE (cada integrante clona en su PC)
Computadora de cada integrante ← Todos trabajan en main
```

Roles en el grupo:

- 👤 **Uno hace el Fork** (crea el repositorio del grupo)
- 👥 **Los tres clonan** el mismo Fork
- 🌿 **Los tres trabajan** en la rama `main`
- 📤 **Los tres pueden hacer Push** a esa rama

🆕 PARTE 2: Crear tu Cuenta en GitHub (OBLIGATORIO)

Paso 1: Registrarse en GitHub

1. Andá a: **https://github.com/**
2. Hacé clic en **"Sign up"** (arriba a la derecha)
3. Completá el formulario:
   - **Email:** Usá tu email universitario (puede darte beneficios educativos)
   - **Username:** Elegí un nombre profesional
     - ☑ Bueno: `maria_gonzalez`, `jperez_data`, `lrodriguez23`
     - ❌ Evitá: `xXsuper_hacker99Xx`, `el_mas_capo`
     - **Tip:** Este username puede aparecer en tu CV, elegilo con cabeza
   - **Password:** Contraseña segura (mínimo 15 caracteres o 8 con combinaciones)
4. Resolvé el captcha (verificación anti-robots)
5. GitHub te va a enviar un código de verificación a tu email

- Abrí tu correo
- Copiá el código
- Pegalo en GitHub

6. **Personalizá tu experiencia (opcional):**
   - "What would you like to do with GitHub?" → Podés elegir "Collaborate on projects"
   - "How would you describe your level?" → Elegí tu nivel de experiencia
   - Podés saltear esto haciendo clic en "Skip personalization"
7. **¡Listo!** Ya tenés tu cuenta de GitHub

Paso 2: Configurar tu Perfil (Recomendado)

1. Hacé clic en tu foto de perfil (arriba a la derecha)
2. Seleccioná **"Settings"**
3. En **"Public profile":**
   - **Name:** Tu nombre completo (ej: María González)
   - **Bio:** Algo simple (ej: "Estudiante de Ciencia de Datos - UNO")
   - **Location:** Argentina (opcional)
   - **Company:** Universidad Nacional del Oeste (opcional)
4. Guardá los cambios

¿Por qué hacer esto?

- GitHub puede servir como portfolio profesional
- Los reclutadores buscan perfiles de GitHub
- Es gratis y te diferencia

Paso 3: Beneficios para Estudiantes (Opcional pero MUY Recomendado)

GitHub ofrece el **GitHub Student Developer Pack** que incluye:

- GitHub Pro gratis (repositorios privados ilimitados)
- Créditos en Azure, AWS, DigitalOcean
- Acceso a herramientas pagas de desarrollo
- Y mucho más (valuado en +$200.000 USD en herramientas)

Cómo obtenerlo:

1. Andá a: **https://education.github.com/pack**
2. Hacé clic en **"Get your pack"**
3. Verificá tu condición de estudiante con tu email universitario
4. Subí una foto de tu certificado de alumno regular o analítico

Nota: Esto es opcional para el desafío, pero vale la pena.

🍴 PARTE 3: Hacer un FORK del Repositorio (CRÍTICO)

⚠ IMPORTANTE: Trabajo en Grupo

Este desafío se trabaja en GRUPOS de 3 personas.

Organización recomendada:

1. **UN integrante** hace el Fork del repositorio del profesor
2. **Los TRES integrantes** del grupo:
    - Son agregados como colaboradores al Fork
    - Clonan el mismo repositorio
    - Trabajan en la rama `main` del Fork
    - Pueden hacer Push de sus cambios

Beneficios de trabajar así:

- ☑ Un solo repositorio por grupo (más organizado)
- ☑ Todos ven los cambios de los demás
- ☑ Aprenden a trabajar colaborativamente (como en la vida real)
- ☑ El profesor puede ver el trabajo de cada grupo claramente

---

¿Qué es un Fork?

Un **Fork** es tu propia copia completa de un repositorio. Es como sacar una fotocopia de todo el proyecto para que el grupo pueda:

- Modificarlo como quiera
- Experimentar sin miedo a romper nada
- Subir sus cambios a SU versión
- Trabajar colaborativamente

Importante: El Fork vive en la cuenta de GitHub de UN integrante del grupo, no en las computadoras todavía.

Paso 1: Elegir quién hace el Fork

En el grupo, pónganse de acuerdo:

- ¿Quién va a hacer el Fork? (UN solo integrante)
- Sugerencia: El que tenga más experiencia con Git/GitHub o el que tenga mejor conexión a internet

El resto del grupo: Esperen a que el compañero haga el Fork antes de continuar.

Paso 2: Ir al Repositorio Original del Desafío (Solo quien hace el Fork)

1. Abrí tu navegador
2. Andá a: **https://github.com/panchtox/health_economics_challenge**
3. Asegurate de estar logueado en GitHub (arriba a la derecha debería aparecer tu foto)

Paso 3: Hacer el Fork (Solo quien hace el Fork)

1. Arriba a la derecha del repositorio, vas a ver un botón que dice **"Fork"** con un ícono de tenedor 🍴
2. Hacé clic en **"Fork"**
3. Te va a aparecer una pantalla: **"Create a new fork"**
   - **Owner:** Tu usuario (debería estar seleccionado automáticamente)
   - **Repository name:** Dejá el nombre como está: `health_economics_challenge`
   - **Description:** Poné algo descriptivo: "Desafío ML Salud - Grupo X" (reemplazá X con tu número de grupo)
   - ☑ **Copy the main branch only** → Dejá esto MARCADO
4. Hacé clic en **"Create fork"**
5. **Esperá unos segundos** mientras GitHub copia todo el repositorio
6. **¡Listo!** Ahora el grupo tiene su propia copia del repositorio

Vas a notar que:

- La URL cambió de `github.com/panchtox/...` a `github.com/TU_USUARIO/...`
- Arriba del nombre del repo dice: "forked from panchtox/health_economics_challenge"

Paso 4: Agregar a tus Compañeros como Colaboradores (Solo quien hizo el Fork)

Para que tus compañeros puedan hacer Push:

1. En tu Fork, andá a: **Settings** (arriba a la derecha, en el menú del repositorio)
2. En el menú de la izquierda, hacé clic en: **Collaborators**
3. Te va a pedir la contraseña de GitHub → ponela
4. Hacé clic en: **"Add people"**
5. Buscá a cada uno de tus compañeros por:
   - Username de GitHub
   - Email de GitHub
6. Hacé clic en **"Add [nombre] to this repository"**
7. **Repetí** para cada compañero del grupo (los otros dos integrantes)
8. **Tus compañeros van a recibir un email de invitación**
   - Deben aceptar la invitación para poder colaborar

Paso 5: Aceptar la Invitación (Resto del grupo)

Si sos uno de los otros dos integrantes:

1. Revisá tu email (el de GitHub)
2. Vas a tener un mail de GitHub que dice: **"[Usuario] has invited you to collaborate..."**
3. Hacé clic en **"View invitation"** o andá a: https://github.com/notifications
4. Hacé clic en **"Accept invitation"**
5. **¡Listo!** Ahora podés colaborar en el repositorio del grupo

---

Paso 6: Verificar el Fork del Grupo

Todos los integrantes deberían verificar:

- ☑ El Fork está en: `https://github.com/USUARIO_DEL_GRUPO/health_economics_challenge`
- ☑ Dice "forked from panchtox/health_economics_challenge"
- ☑ Todos fueron agregados como colaboradores (aparecen en Settings → Collaborators)
- ☑ Tiene todos los archivos del original (README.md, carpetas dataset/, codigo_base/, etc.)

Importante: A partir de ahora, los TRES van a trabajar con:

- El MISMO repositorio (el Fork del grupo)
- La MISMA rama (`main`)
- Cada uno en su PROPIA computadora

💻 PARTE 4: Clonar el Fork del Grupo (TODOS los integrantes)

Ahora sí, **cada integrante del grupo** va a descargar el Fork a su computadora. Tenés **dos opciones**:

1. **GitHub Desktop** → Interfaz gráfica (más fácil para empezar) ⭐ **Recomendado**
2. **Línea de comandos** → Para los más técnicos

IMPORTANTE: Los TRES integrantes van a:

- Clonar el MISMO repositorio (el Fork que hizo un compañero)
- Trabajar en la rama `main`
- Hacer cambios en sus PROPIAS computadoras
- Subir sus cambios colaborativamente

🖥️ MÉTODO 1: GitHub Desktop (Recomendado)

Paso 1: Instalar GitHub Desktop

1. Andá a: **https://desktop.github.com/**
2. Descargá el instalador para Windows/Mac/Linux
3. Ejecutá el instalador (siguiente, siguiente, finalizar)
4. Abrí GitHub Desktop

Requisitos:

- Windows 10 o superior / macOS 10.13+ / Ubuntu 20.04+
- 500 MB de espacio disponible
- Conexión a Internet

Paso 2: Iniciar Sesión en GitHub Desktop

Cuando abras GitHub Desktop por primera vez:

1. **Sign in to GitHub.com**
    - Te va a pedir que inicies sesión
    - Hacé clic en **"Sign in to GitHub.com"**
    - Se va a abrir tu navegador
    - Autorizá a GitHub Desktop
    - Volvé a GitHub Desktop
2. **Configure Git**
    - **Name:** Tu nombre completo (ej: "María González")
    - **Email:** Tu email de GitHub (el mismo con el que te registraste)
    - Hacé clic en **"Continue"**
3. **¡Listo!** Ya estás configurado

Paso 3: Clonar el Fork del Grupo (TODOS los integrantes)

MUY IMPORTANTE: Vas a clonar el Fork DEL GRUPO (el que hizo un compañero).

1. En GitHub Desktop, andá a: **File** → **Clone Repository**
2. Vas a ver tres pestañas: seleccioná **"GitHub.com"**
3. **Si sos el que hizo el Fork:**
    - En la lista, deberías ver: `TU_USUARIO/health_economics_challenge`
    - Seleccionalo
4. **Si sos colaborador:**
    - En la lista, deberías ver: `USUARIO_DEL_COMPAÑERO/health_economics_challenge`
    - Si no lo ves, hacé clic en **"Refresh"** arriba a la derecha
    - Si aún no aparece, andá a la pestaña **"URL"** y seguí el paso alternativo
5. **Paso Alternativo (si no aparece en la lista):**
    - Pestaña **"URL"**
    - Pedile a tu compañero que te pase la URL del Fork del grupo
    - Debería ser: `https://github.com/USUARIO_DEL_GRUPO/health_economics_challenge`
    - Pegala en el campo URL

6. **Local Path:** Elegí dónde guardar el proyecto
    - Recomendado: `C:\Users\TuUsuario\Documents\health_economics_challenge\`
    - O la carpeta que prefieras (acordate dónde la pusiste)
7. Hacé clic en **"Clone"**
8. **Esperá** a que descargue (puede tardar 1-2 minutos dependiendo de tu conexión)
9. Te va a preguntar: **"How are you planning to use this fork?"**
    - **Si sos el que hizo el Fork:** Seleccioná **"For my own purposes"**
    - **Si sos colaborador:** Seleccioná **"To contribute to the parent project"**
    - Hacé clic en **"Continue"**

Paso 4: Abrir la Carpeta del Proyecto

Una vez clonado:

1. En GitHub Desktop, hacé clic derecho en el repositorio
2. Seleccioná: **"Show in Explorer"** (Windows) o **"Reveal in Finder"** (Mac)
3. Vas a ver toda la estructura del desafío:

```
health_economics_challenge/
├── README.md                          ← Empezá leyendo esto
├── dataset/
│   ├── dataset_desafio.csv            ← Datos para trabajar
│   └── diccionario_variables.md       ← Qué significa cada variable
├── codigo_base/
│   ├── CONFIG_minimo.yml              ← Configuración
│   ├── 0_HEALTH_EXE.R                 ← Script principal
│   ├── 01_FE_health.R                 ← Acá van a crear variables
│   ├── 02_TS_health.R                 ← Training Strategy
│   └── 03_HT_health.R                 ← Hyperparameter Tuning
└── exp/                               ← Resultados (se crea automáticamente)
```

Paso 5: Workflow Colaborativo del Grupo

Cuando **cualquier integrante** modifique archivos y quiera guardar cambios:

1. Hacé cambios en los archivos

- Ejemplo: modificás `01_FE_health.R`

2. ANTES de hacer Commit: Fetch (traé cambios de tus compañeros)

- Arriba a la derecha, hacé clic en: **"Fetch origin"**
- Si hay cambios de tus compañeros, va a aparecer: **"Pull origin"**
- Hacé clic en **"Pull origin"** para descargar sus cambios

- **¿Por qué?** Para no crear conflictos con el trabajo de tus compañeros

3. GitHub Desktop detecta TUS cambios automáticamente

- Los vas a ver en la columna izquierda

4. Commit (guardá una "foto" de TUS cambios):

- Abajo a la izquierda, en "Summary", poné un mensaje descriptivo:
  - ☑ Bueno: "Agregué 5 variables nuevas de ratios de eficiencia - Juan"
  - ☑ Bueno: "Configuré presente=2021 y orden_lead=1 - María"
  - ❌ Malo: "cambios" o "asdf"
- **Tip:** Poné tu nombre en el commit para que el grupo sepa quién hizo qué
- Hacé clic en **"Commit to main"**

5. Push (subí TUS cambios a GitHub):

- Arriba, hacé clic en **"Push origin"**
- Esto sube tus cambios al repositorio del grupo
- **Tus compañeros ahora pueden ver lo que hiciste**

6. Verificá en el navegador:

- Andá a `https://github.com/USUARIO_DEL_GRUPO/health_economics_challenge`
- Deberías ver tus cambios reflejados

🎯 Buenas Prácticas del Grupo

Para evitar conflictos:

- 📢 **Comunicación:** Avisenle al grupo cuándo van a trabajar
- 🔄 **Fetch primero:** Siempre hacé Fetch/Pull ANTES de empezar a trabajar
- 💬 **Commits descriptivos:** Pongan mensajes claros y su nombre
- 📦 **Commits frecuentes:** No esperen a tener todo perfecto, commiteen seguido
- 🗣️ **División del trabajo:**
  - Persona A: Variables de eficiencia (ratios, indicadores)
  - Persona B: Variables de tendencias temporales
  - Persona C: Variables de contexto económico
  - Los tres: Analizar resultados juntos

Si hay conflictos:

- GitHub Desktop te va a avisar
- Te va a mostrar qué archivos tienen conflicto
- Tenés que decidir en grupo qué versión mantener
- O combinar manualmente los cambios

Paso 6: Actualizar desde el Repositorio del Profesor

Si el profesor actualiza el repositorio original (corrige errores, agrega datos, etc.):

1. En GitHub Desktop, andá a: **Branch → Merge into current branch**
2. Hacé clic en **"Choose a branch"**
3. Buscá **"upstream/main"** (el repositorio del profesor)
    - Si no aparece, primero tenés que configurar upstream (ver más abajo)
4. Seleccionalo y hacé clic en **"Merge upstream/main into main"**
5. Si hay conflictos, GitHub Desktop te va a ayudar a resolverlos

Nota: Esto es avanzado, probablemente no lo necesiten para el desafío. Si el profesor actualiza algo importante, les va a avisar.

💻 MÉTODO 2: Línea de Comandos (Avanzado)

Paso 1: Instalar Git

Windows:

1. Andá a: **https://git-scm.com/download/win**
2. Descargá el instalador (64-bit recomendado)
3. Ejecutá el instalador con configuraciones por defecto
4. Durante la instalación, asegurate de seleccionar:
    - ☑ "Git from the command line and also from 3rd-party software"
    - ☑ "Use Windows' default console window"

Mac:

```
# Si tenés Homebrew instalado:
brew install git

# O descargá desde: https://git-scm.com/download/mac
```

Linux (Ubuntu/Debian):

```
sudo apt update
sudo apt install git
```

Paso 2: Configurar Git (Primera Vez)

Abrí la terminal (Windows: Git Bash / Mac: Terminal / Linux: Terminal) y ejecutá:

```
# Configurá tu nombre (va a aparecer en tus commits)
git config --global user.name "Tu Nombre Completo"
```

```
# Configurá tu email (el mismo de GitHub)
git config --global user.email "tu.email@ejemplo.com"

# Verificá la configuración
git config --list
```

Paso 3: Clonar el Fork del Grupo

MUY IMPORTANTE: Vas a clonar el Fork DEL GRUPO (el que hizo un compañero).

1. Andá al Fork del grupo en el navegador:
   - **Si sos el que hizo el Fork:** URL: `https://github.com/TU_USUARIO/health_economics_challenge`
   - **Si sos colaborador:** Pedile la URL a tu compañero: `https://github.com/USUARIO_DEL_GRUPO/health_economics_challenge`
2. Hacé clic en el botón verde **"Code"**
3. Copiá la URL que aparece (HTTPS):
   - Debería ser: `https://github.com/USUARIO_DEL_GRUPO/health_economics_challenge.git`
4. Abrí la terminal / Git Bash
5. Navegá a la carpeta donde querés guardar el proyecto:

```
# Ejemplo: ir a la carpeta Documentos
cd ~/Documents/

# O en Windows:
cd C:\Users\TuUsuario\Documents\
```

6. Cloná el Fork del grupo:

```
git clone
https://github.com/USUARIO_DEL_GRUPO/health_economics_challenge.git
```

   Reemplazá USUARIO_DEL_GRUPO con el username real del compañero que hizo el Fork

7. Entrá a la carpeta del proyecto:

```
cd health_economics_challenge
```

8. Verificá que esté todo:

```
ls -la    # Mac/Linux
dir       # Windows
```

9. Verificá que estés en la rama `main`:

```
git branch
# Debería mostrar: * main
```

Paso 4: Configurar el Repositorio Remoto (Upstream)

Para poder recibir actualizaciones del profesor, tenés que agregar su repositorio como "upstream":

```
# Agregá el repositorio original del profesor
git remote add upstream https://github.com/panchtox/health_economics_challenge.git

# Verificá que esté configurado
git remote -v

# Deberías ver:
# origin    https://github.com/USUARIO_DEL_GRUPO/health_economics_challenge.git (fetch)
# origin    https://github.com/USUARIO_DEL_GRUPO/health_economics_challenge.git (push)
# upstream  https://github.com/panchtox/health_economics_challenge.git (fetch)
# upstream  https://github.com/panchtox/health_economics_challenge.git (push)
```

Paso 5: Workflow Colaborativo con Git (Comandos)

Antes de empezar a trabajar (SIEMPRE):

```
# 1. Asegurate de estar en la rama main
git branch
# Debería mostrar: * main

# 2. Traé los cambios de tus compañeros
git pull origin main
```

Cuando hacés cambios:

```
# 1. Ver qué archivos modificaste
git status

# 2. Agregar los archivos modificados
git add 01_FE_health.R
# O agregar todos:
git add .

# 3. Hacer commit con mensaje descriptivo (incluí tu nombre)
git commit -m "Agregué 5 variables de eficiencia - Juan"

# 4. Subir tus cambios a la rama main del grupo
git push origin main
```

Ver qué hicieron tus compañeros:

```
# Ver el historial de commits
git log --oneline --graph

# Ver quién modificó qué
git log --oneline --author="María"
```

Si hay conflictos:

```
# Cuando hagas pull y haya conflictos:
git pull origin main

# Git te va a avisar qué archivos tienen conflicto
# Abrí esos archivos y vas a ver:
# <<<<<<< HEAD
# Tu versión
# =======
# Versión de tu compañero
# >>>>>>>

# Editá manualmente para quedarte con lo que querés
# Guardá el archivo

# Después:
git add archivo_con_conflicto.R
git commit -m "Resolví conflicto en archivo_con_conflicto.R"
git push origin main
```

Paso 6: Actualizar desde el Repositorio del Profesor

Si el profesor actualiza el repositorio original:

```
# 1. Descargá los cambios del profesor
git fetch upstream

# 2. Asegurate de estar en la rama main
git checkout main

# 3. Fusioná los cambios del profesor con tu versión
git merge upstream/main

# 4. Subí los cambios actualizados al repositorio del grupo
git push origin main
```

Si hay conflictos, Git te va a avisar y vas a tener que resolverlos manualmente.

🛠️ PARTE 5: Configurar el Entorno de Trabajo

Paso 1: Instalar R y RStudio

Si aún no los tenés instalados:

1. **R (lenguaje):**
   - Andá a: **https://cran.r-project.org/**
   - Descargá la versión para tu sistema operativo
   - Instalá con configuraciones por defecto
2. **RStudio (IDE):**
   - Andá a: **https://posit.co/download/rstudio-desktop/**
   - Descargá RStudio Desktop (versión gratuita)
   - Instalá con configuraciones por defecto

Paso 2: Instalar Librerías de R Necesarias

Abrí RStudio y ejecutá:

```
# Instalar librerías necesarias (solo la primera vez)
install.packages(c(
  "data.table",
  "lightgbm",
  "yaml",
  "mlrMBO",
  "ggplot2",
  "dplyr",
  "stringr",
  "lubridate"
```

```
))

# Si lightgbm te da problemas, probá:
# install.packages("lightgbm", repos = "https://cran.r-project.org")
```

Nota: Esto puede tardar 5-10 minutos dependiendo de tu conexión.

Paso 3: Abrir el Proyecto en RStudio

1. En RStudio: **File** → **Open Project...**
2. Navegá a la carpeta `health_economics_challenge/`
3. Si hay un archivo `.Rproj`, abrilo
4. Si no, simplemente abrí: **File** → **Open File...** y seleccioná `0_HEALTH_EXE.R`

☑ PARTE 6: Verificación de Instalación

Checklist de Verificación

Verificá que todo esté funcionando:

Instalación y Configuración GitHub

- ☐ Cuenta en GitHub creada
- ☐ Perfil de GitHub configurado (nombre, bio)
- ☐ **UN integrante** hizo el Fork del repositorio del profesor
- ☐ **El que hizo el Fork** agregó a los otros dos como colaboradores
- ☐ **Los otros dos** aceptaron la invitación
- ☐ El Fork del grupo está en: `github.com/USUARIO_DEL_GRUPO/health_economics_challenge`

Instalación y Configuración Git

- ☐ Git instalado y configurado
- ☐ GitHub Desktop funcionando (o línea de comandos configurada)
- ☐ **TODOS** clonaron el repositorio del grupo exitosamente
- ☐ **TODOS** están trabajando en la rama `main`
- ☐ Puedo hacer commits y push sin errores

Instalación R

- ☐ R y RStudio instalados
- ☐ Todas las librerías de R instaladas
- ☐ Script de test ejecuta correctamente

Comprensión del Proyecto

- ☐ Leí el README.md completo
- ☐ Entiendo la estructura de carpetas
- ☐ Revisé el diccionario de variables

- [ ] Entiendo el objetivo del desafío (predecir hf3_ppp_pc para 2022)
- [ ] Leí la rúbrica de evaluación

Organización del Grupo

- [ ] Nos dividimos el trabajo de Feature Engineering
- [ ] Decidimos quién va a trabajar en qué variables
- [ ] Sabemos cómo comunicarnos para evitar conflictos
- [ ] Entendemos el flujo: Fetch/Pull → Modificar → Commit → Push

Preparación del Trabajo

- [ ] Creamos una carpeta `mi_trabajo/` para experimentos individuales
- [ ] Probamos ejecutar el pipeline base (sin modificaciones)
- [ ] Entendemos qué es la función AgregarVariables()
- [ ] Entendemos el dilema COVID (presente, orden_lead)

Test Rápido en R

Ejecutá este código en RStudio para verificar que todo esté OK:

```
# Test de librerías
librerias <- c("data.table", "lightgbm", "yaml",
               "mlrMBO", "ggplot2", "dplyr")

for (lib in librerias) {
  if (require(lib, character.only = TRUE)) {
    cat("✓", lib, "instalado correctamente\n")
  } else {
    cat("✗", lib, "NO ENCONTRADO - instalar con install.packages()\n")
  }
}

# Test de lectura de datos
# IMPORTANTE: Ajustá la ruta a donde clonaste el repositorio
setwd("C:/Users/TuUsuario/Documents/health_economics_challenge/")
datos <- fread("dataset/dataset_desafio.csv")
cat("\n✓ Dataset cargado:", nrow(datos), "filas,", ncol(datos), "columnas\n")
```

Si todo imprime "✓" → **¡Están listos para empezar!**

🆘 PARTE 7: Solución de Problemas Comunes

Problema 1: "Git no reconocido como comando"

Windows:

- Git no está en el PATH

- **Solución:** Reinstalá Git y marcá la opción "Git from the command line..."

Mac/Linux:

- Puede que necesites reiniciar la terminal después de instalar

---

Problema 2: No veo el Fork del grupo en GitHub Desktop

Solución:

1. Hacé clic en **"Refresh"** en la lista de repositorios
2. Si no aparece, usá la opción **"URL"** y pegá la URL del Fork del grupo
3. Asegurate de estar logueado en GitHub Desktop con tu cuenta

---

Problema 3: "Permission denied" al hacer Push

Causa: Problemas de autenticación con GitHub

Solución:

1. Si usás GitHub Desktop: asegurate de haber iniciado sesión
2. Si usás línea de comandos:
    - GitHub ahora usa tokens en lugar de contraseñas
    - Creá un Personal Access Token: https://github.com/settings/tokens
    - Usá el token en lugar de tu contraseña cuando te lo pida

---

Problema 4: No me agregaron como colaborador

Solución:

1. El integrante que hizo el Fork debe ir a:
    - **Settings** → **Collaborators**
    - **Add people** → Buscar tu username o email
    - Enviarte la invitación
2. Vos revisá tu email y aceptá la invitación
3. También podés ir a: https://github.com/notifications y aceptar desde ahí

Problema 5: Conflictos entre cambios del grupo

Situación: Dos personas modificaron el mismo archivo al mismo tiempo

Prevención (lo mejor):

- Comunicá en el grupo cuando vas a trabajar
- Dividí responsabilidades claramente
- Hacé Fetch/Pull ANTES de empezar

Solución si ya hay conflicto:

GitHub Desktop:

1. Va a aparecer un warning de conflicto
2. Hacé clic en "View conflicts"
3. GitHub Desktop te va a mostrar los archivos en conflicto
4. Abrí el archivo y vas a ver algo así:

```
<<<<<<< HEAD
Código de uno
=======
Código del otro
>>>>>>> origin/main
```

5. Editá manualmente para quedarte con lo correcto
6. Borrá las líneas con `<<<<<<<`, `=======`, `>>>>>>>`
7. Guardá el archivo
8. En GitHub Desktop, marcá el conflicto como resuelto
9. Commit y Push

Línea de comandos:

```
# Editá el archivo manualmente
# Guardá los cambios
git add archivo_con_conflicto.R
git commit -m "Resolví conflicto entre Juan y María"
git push origin main
```

Problema 6: Un compañero no puede hacer Push

Causa: No fue agregado como colaborador o no aceptó la invitación

Solución:

1. Verificar que esté en la lista de colaboradores
2. Que revise su email y acepte la invitación
3. Que cierre sesión y vuelva a abrir GitHub Desktop
4. Que intente de nuevo

Problema 7: RStudio no encuentra los archivos

Causa: El Working Directory no está configurado

Solución:

```
# Verificá dónde estás
getwd()

# Cambiá a la carpeta del proyecto
setwd("C:/ruta/completa/a/health_economics_challenge/")

# Mejor aún: usá RStudio Projects (abrí el .Rproj)
```

Problema 8: Error al instalar `lightgbm`

Causa: Requiere compiladores C++ en algunos sistemas

Solución Windows:

```
# Instalá Rtools primero
# https://cran.r-project.org/bin/windows/Rtools/

# Luego instalá lightgbm
install.packages("lightgbm", type = "source")
```

Solución Mac:

```
# Instalá Xcode Command Line Tools
xcode-select --install

# Luego en R:
install.packages("lightgbm")
```

Problema 9: Cloné el repo del profesor por error

No pasa nada, simplemente:

1. Borrá la carpeta que clonaste
2. Esperá a que un compañero haga el Fork del repositorio del profesor
3. Que ese compañero te agregue como colaborador
4. Aceptá la invitación
5. Cloná el Fork DEL GRUPO (no el del profesor)

🎓 PARTE 8: Workflow Recomendado para el Desafío

Flujo de Trabajo Ideal del Grupo

```
ORGANIZACIÓN INICIAL (Una sola vez)
1. ☑ Los tres crean cuenta en GitHub
2. ☑ UN integrante hace Fork del repo del profesor
3. ☑ Ese integrante agrega a los otros dos como colaboradores
4. ☑ Los otros dos aceptan invitación
   ↓

CONFIGURACIÓN (Cada integrante en su PC)
5. Los TRES clonan el Fork del grupo
6. Los TRES trabajan en la rama `main`
7. Los TRES configuran el entorno (R, librerías)
   ↓

TRABAJO ITERATIVO (Ciclo que se repite)
8. 🔄 ANTES DE EMPEZAR: Fetch/Pull (traer cambios del grupo)
9. 📖 Leer documentación y datos
10. 💻 Trabajar en tu parte asignada:
    - Persona A: Variables de eficiencia
    - Persona B: Variables de tendencias
    - Persona C: Variables de contexto
11. 💾 Commit con mensaje descriptivo + tu nombre
12. ⬆ Push a main
13. 💬 Avisar al grupo que subiste cambios
    ↓

ANÁLISIS CONJUNTO
14. 🔄 Los tres hacen Pull para tener todo sincronizado
15. ▶ Ejecutan 0_HEALTH_EXE.R (pipeline completo)
16. 📊 Analizan resultados juntos
17. 🎯 Deciden siguientes mejoras
    ↓

ENTREGA FINAL
18. Preparar informe grupal
19. Verificar que todo esté en el Fork del grupo
20. Hacer commit final con toda la entrega
```

División del Trabajo Recomendada

Reunión Inicial del Grupo (30 min):

1. Decidir quién hace el Fork
2. Dividir responsabilidades de Feature Engineering
3. Decidir estrategia COVID (presente, orden_lead)
4. Establecer horarios de trabajo para evitar conflictos

Persona A - Variables de Eficiencia:

```
# Ejemplo: Ratios de gasto/expectativa de vida
dataset[, health_efficiency := SP.DYN.LE00.IN / SH.XPD.CHEX.PC.CD]
dataset[, health_gdp_ratio := SH.XPD.CHEX.GD.ZS / NY.GDP.PCAP.PP.CD]
```

Persona B - Variables de Tendencias:

```
# Ejemplo: Tendencias temporales
dataset[, life_exp_change := c(NA, diff(SP.DYN.LE00.IN)), by = `Country Code`]
dataset[, gdp_growth_volatility := rollapply(NY.GDP.MKTP.KD.ZG, 3, sd, fill = NA)]
```

Persona C - Variables de Contexto:

```
# Ejemplo: Dummies y contexto
dataset[, crisis_2008 := ifelse(year %in% 2008:2009, 1, 0)]
dataset[, high_income := ifelse(income == "High", 1, 0)]
```

Todos Juntos:

- Analizar importancia de variables
- Interpretar resultados económicamente
- Escribir informe ejecutivo
- Revisar y mejorar código

Comunicación del Grupo

Slack/WhatsApp/Discord - Mensajes Importantes:

```
❌ Malo: "hice cambios"

☑ Bueno:
"Subí 3 variables de eficiencia en salud.
Modifiqué: 01_FE_health.R (líneas 45-67)
Pueden hacer pull ahora. - Juan"

☑ Bueno:
"Voy a modificar CONFIG_minimo.yml para probar presente=2020.
NO toquen el archivo por 30 min. - María"

☑ Bueno:
"Ejecuté el pipeline. RMSE: 0.89
Variables importantes: ver exp/03_HT/tb_importancia.txt
¿Probamos aumentar learning_rate? - Pedro"
```

Estructura de Carpetas Recomendada

```
health_economics_challenge/          ← Fork del grupo clonado (SÍ pueden
modificar)
├── README.md
├── dataset/
├── codigo_base/
│   ├── 01_FE_health.R               ← Modificá ESTE archivo
│   └── CONFIG_minimo.yml            ← Modificá ESTE archivo
└── mi_trabajo/                      ← Creá esta carpeta para experimentos
individuales
    ├── notas_experimentos.md
    ├── graficos/
    └── informe_final/
```

¿Por qué pueden modificar el Fork?

- Es el repositorio del GRUPO, no del profesor
- Pueden experimentar libremente
- Sus cambios no afectan al original
- Pueden subir todo a GitHub

PARTE 9: Recursos Adicionales

Tutoriales de Git/GitHub

Principiantes:

- GitHub Skills - Tutoriales interactivos oficiales
- Git Handbook - Guía básica
- Visualizing Git - Ver cómo funciona Git

Videos:

- Git y GitHub para Principiantes (YouTube, español)
- GitHub Desktop Tutorial

Avanzado:

- Pro Git Book - Libro completo gratuito

Documentación del Desafío

Una vez clonado el repositorio, lean en orden:

1. **README.md** - Overview del desafío
2. **documentacion/01_guia_instalacion.md** - Setup detallado
3. **documentacion/02_guia_estrategia_covid.md** - Decisión estratégica clave

4. **documentacion/03_guia_feature_engineering.md** - Cómo crear variables
5. **evaluacion/rubrica_evaluacion.md** - Cómo los vamos a evaluar

---

Comunidad y Soporte

Para dudas sobre Git/GitHub:

- Stack Overflow (español)
- Foro de GitHub

Para dudas del desafío:

- Foro del campus virtual (prioridad)
- Horarios de consulta con el docente
- Grupos de estudio con compañeros

🎉 ¡Están Listos!

Si llegaron hasta acá y completaron todos los pasos, **¡felicitaciones!** Ya tienen:

- ☑ Las tres cuentas de GitHub configuradas
- ☑ El Fork del grupo creado
- ☑ Los tres son colaboradores
- ☑ Cada uno tiene el proyecto clonado en su computadora
- ☑ Todos están trabajando en la rama `main`
- ☑ Todos los materiales del desafío
- ☑ El entorno configurado correctamente
- ☑ Las herramientas necesarias para trabajar colaborativamente
- ☑ Conocimiento básico de Git/GitHub (útil para su carrera)

Ahora pueden empezar a trabajar EN GRUPO de forma profesional.

📞 Contacto y Ayuda

Docente: Francisco Fernández
Email: [Email del campus]
Horario de consultas: [Horario definido]

Repositorio ORIGINAL del desafío (del profesor):
https://github.com/panchtox/health_economics_challenge

FORK del GRUPO (su versión):
https://github.com/USUARIO_DEL_GRUPO/health_economics_challenge
(Reemplazá USUARIO_DEL_GRUPO con el username real del compañero que hizo el Fork)

Importante: Si encuentran errores en los materiales o tienen sugerencias, pueden:

1. Comentar en el foro del campus

2. Enviar un email al docente
3. (Avanzado) Crear un "Issue" en el Fork del grupo en GitHub

---

Licencia y Uso de Materiales

Los materiales de este desafío son para **uso educativo exclusivo** de los alumnos de la materia Aplicaciones en Ciencia de Datos de la Universidad Nacional del Oeste.

Pueden:

- ☑ Hacer Fork y usar el repositorio para el desafío
- ☑ Modificar el código en su Fork para sus experimentos
- ☑ Compartir conocimientos con sus compañeros de grupo

No pueden:

- ❌ Redistribuir los materiales fuera del curso
- ❌ Usar los datos para fines comerciales
- ❌ Publicar las soluciones del desafío públicamente

Historial de Versiones de este Instructivo

Versión	Fecha	Cambios
1.0	Nov 2025	Versión inicial

Última actualización: Noviembre 2025
Documento creado por: Francisco Fernández (Fran)
Formato: Markdown (convertible a PDF)

Glosario de Términos

Término	Definición
Branch	Rama de desarrollo paralela al código principal
Clone	Copiar un repositorio completo a tu computadora
Commit	Punto de guardado en el historial de Git
Fork	Tu propia copia de un repositorio de otra persona en GitHub
Git	Sistema de control de versiones distribuido
GitHub	Plataforma web para hospedar repositorios Git
IDE	Integrated Development Environment (como RStudio)
Merge	Unir cambios de una rama a otra

Término	Definición
Origin	Tu repositorio remoto (el Fork del grupo en GitHub)
Pull	Descargar cambios del repositorio remoto
Push	Subir tus cambios al repositorio remoto
Remote	Repositorio en la nube (ej: en GitHub)
Repository (Repo)	Carpeta de proyecto con historial de Git
Upstream	El repositorio original del cual hicieron Fork
Working Directory	Carpeta donde Git está rastreando cambios

🔑 Diferencias Clave: Fork vs Clone

Concepto	¿Dónde está?	¿Para qué sirve?
Repositorio Original	GitHub del profesor	El proyecto original que creó el profesor
Fork (Copia del grupo)	GitHub en cuenta de un integrante	La versión del grupo en la nube
Clone (Copia local)	Computadora de cada integrante	Para trabajar offline y programar

Flujo completo:

```
github.com/panchtox/health_economics_challenge (Profesor)
              ↓ FORK
github.com/USUARIO_DEL_GRUPO/health_economics_challenge (GitHub del grupo)
              ↓ CLONE
C:\...\health_economics_challenge\ (PC de cada integrante)
```

🌿 APÉNDICE: ¿Qué son las Ramas? (Opcional - Para Curiosos)

Concepto de Ramas (Branches)

Una **rama** es como una línea de tiempo alternativa de tu proyecto. Te permite:

- Experimentar sin tocar el código "oficial"
- Trabajar en una funcionalidad nueva sin romper lo que funciona
- Volver a la versión estable cuando quieras

Analogía: Es como hacer una copia de tu tesis en otra carpeta para probar un enfoque distinto, sin borrar la versión original.

¿Cuándo son útiles las ramas?

Para este desafío NO son necesarias, pero podrían ser útiles si:

- Querés probar una configuración radical sin romper lo que funciona
- Un integrante quiere experimentar solo antes de compartir con el grupo
- Querés mantener separadas distintas estrategias (ej: una rama con COVID, otra sin COVID)

Comandos Básicos de Ramas (GitHub Desktop)

Crear una rama nueva:

1. Arriba, hacé clic en **"Current branch: main"**
2. Hacé clic en **"New Branch"**
3. Poné un nombre descriptivo: `experimento_juan` o `estrategia_sin_covid`
4. Hacé clic en **"Create Branch"**

Cambiar entre ramas:

1. Hacé clic en **"Current branch"**
2. Seleccioná la rama a la que querés ir

Fusionar una rama con main:

1. Cambiá a la rama `main`
2. Andá a: **Branch → Merge into current branch**
3. Seleccioná la rama que querés fusionar
4. Hacé clic en **"Merge"**

Comandos Básicos de Ramas (Línea de Comandos)

```
# Ver qué ramas existen
git branch

# Crear una rama nueva
git branch experimento_juan

# Cambiar a esa rama
git checkout experimento_juan

# Crear Y cambiar en un solo comando
git checkout -b experimento_juan

# Volver a main
git checkout main

# Fusionar una rama con main
git checkout main
git merge experimento_juan

# Borrar una rama (después de fusionarla)
git branch -d experimento_juan
```

Ejemplo de Uso de Ramas

Situación: Juan quiere probar usar `presente=2018` sin afectar el trabajo del grupo que está usando `presente=2021`.

```
# 1. Juan crea su rama
git checkout -b juan_presente_2018

# 2. Juan modifica CONFIG_minimo.yml
# presente: 2018

# 3. Juan hace commit
git commit -am "Probé con presente=2018 - RMSE: 0.95"

# 4. Juan vuelve a main (el trabajo del grupo)
git checkout main

# 5. Si el experimento funcionó, Juan puede fusionar
git merge juan_presente_2018

# 6. O simplemente dejar el experimento ahí y no fusionarlo
```

Importante: Si usan ramas, asegúrense de que todos entiendan qué está pasando. Puede confundirse rápido.

💡 Consejo Final

Para este desafío: Trabajen todos en `main`, mantengan buena comunicación, y hagan Pull antes de empezar a trabajar. Es simple, funciona, y es lo que se usa en el 80% de los proyectos reales.

Las ramas son útiles, pero agregan complejidad. Si nunca usaron Git antes, mejor empiecen simple. Ya van a tener tiempo de aprender ramas en futuros proyectos.

¡Mucha suerte con el desafío! 🚀📊

Cualquier duda, pregunten en el campus o en las consultas. ¡No se queden con dudas!
