**Notas generales**

Use python, principalmento con una herramienta que se llama [ipython notebook](http://ipython.org/notebook.html) (cuaderno ipython). Es analogo a RStudio pero lo prefiero. Tambien se lo puede usar con R. Mira el projecto de fuente libre, [Jupyter](http://jupyter.org), con su interpretacion para R, [IRKernel](https://github.com/IRkernel/IRkernel),

Hice un notebook diferente por cada paso abajo (sino #1). Es importante separar tus codigos en filas logicas, para mantenerlos comprensibles para ti y para otros.

Tambien hice un repositorio git (de [GitHub](http://www.github.com)) en que guardo mis codigos frecuentemente. Este tiene dos usos. Primero, guarda todos los versiones pasados de mis codigos, y si borro algo acidentalmente, puedo cargar la version mas reciente que guarde. Seguno, puedo compartir mis codigos y resultados facilmente con otra gente.

**Se puede usar todos estos metodos con R y Rstudio tambien. Se puede usar git, filas diferentes para varios pasos, etc.**

**Pasos**

1. **Inspecciono los datos visualmente**, en la fila excel para entender la estructura de los datos y buscar cosas extranas. Muchas veces los datos son demasiado grandes hacer esto, y tambien no es posible descubrir todos los errores visualmente. Por eso, es importante explorar y averiguar los datos con codigos y summarios tambien (abajo).
   * Para entender los datos intuitivamente, **pregunto a un “experto del dominion”** -- en este caso, una person que sabe bien la agricultura, quien me puede explicar los columnos y como entender los datos. Regreso preguntarle muchas veces durante el proyecto.
2. **Hago un CSV**: Cargo los datos en un DataFrame en python (usando el paquete “pandas”), casi igual como un data.frame en R. Uso un ipython notebook. Convierto el excel a un CSV. [convertir\_xls\_ascii\_csv.ipynb]
   * Tenia dificultades con las letras internacionales. Por eso, converti todos a ascii.
   * Converti los datos de Excel a CSV (se puede hacer totalmente con pandas).
   * Ahora tengo un CSV funcional para usar.
3. **Limpio los datos**: [limpiar.ipynb]
   * Borro valores/filas/columnos invalidos, columnos innecessarios, corrijo valores y nobres de columnos, etc.
   * Es posible descubrir estos problemas en el paso 1 (inspeccionar la fila excel) y el paso 4 (sumarios), y a veces siguentes. Es necesari regresar a este paso varias veces.
4. **Hago sumarios y busco errores**. Uso DataFrame.summary(), y tambien, hago graficos sumarios (histogram, figuras bar). Mirarlos cada uno para encontrar errores. [crear\_figuras\_sumarias.ipynb]
   * Descubri valores falsos. Por ejemplo, en la figura de fenologia\_emerg\_cosecha\_dias, habia una espiga al valor -40000. Obviamente falso -- es cierto que es un valor “centinela” que significa algo especial -- que no hay data. Por eso, edite a mi notebook de Limpiar a cambiar estos valores a NA.
   * Sigo asi, descubriendo errores, cambiando los codigos de limpiar, hasta que me parece bien. Puedo hacer circulos asi, saltando de paso a paso y atras, durante todo el projecto. Como dijo Sylvain, es proceso iterativo.
5. Exploro correlaciones simples [explorar]
   * Visualizo unos variables contro otros en scatterplot, y tambien con bar, kde, etc.
   * Es facil con pandas calcular correlaciones lineales rapidamenta. La function .corr() hace todos los correlaciones entre dos variables de en data frame, y es facil visualizarlos para entender las influencias obvias. Tambien el paquete seaborn tiene muchas funciones de visualizacion, por ejemplo jointplot.
   * Con numpy, puedo hacer unos lineas polinomias, y guarde varias figuras con lineas de regresion del segundo orden.
6. Busco otros senales interesantes [tempuraturas\_criticas]
   * En este caso, cargue y use los datos de clima. Por falto de tiempo, contrui una sola idea, de buscar un alcance importante para el arroz.