

Pedagogía con Python

Sebastián Flores

12 Noviembre 2021






- Who: Sebastián Flores
- Day Job: Chief Data Officer en uPlanner
- Interests: Computación, Matemática y Educación
- Weapons of choice: Python, javascript y html
- Where to find: sebastiandres.xyz
- Username: @sebastiandres



Sobre mi formación:

- Ingeniería Civil Matemática, Universidad Técnica Federico Santa María, Chile.
- Ingénieur Diplômé, Mécanique Numérique, École Polytechnique, Francia.
- Master Computational and Mathematical Engineering, Stanford University, EEUU.

Algunos proyectos:

- Modelamiento de autos híbridos, PSA Peugeot Citröen.
- Modelamiento y simulación numérica de hidrofracturas para minería subterránea, CMM Chile.
- Modelamiento y simulación de micro-sismicidad minera, CMM Chile.
- Desarrollo de software de simulación de tsunamis, Método Volúmenes Finitos para Shallow Water Equations, UTFSM Chile.
- Desarrollo de librería para simulación de ecuaciones de difusión-reacción para catalizadores inmovilizados (pypsdier).
- PyCon: 2020 (, ) y 2021 (, , )

Agenda de la charla:

1. Experiencia transformando un curso con Python.
2. Algunas herramientas recomendadas.
3. Machine Learning, Deep Learning y Data Science.

1. Transformando un curso con python

Lo que aprendí dictando un curso de programación para matemáticos...

1. Transformando un curso con python

En el año 2014, se me ofrece dictar el curso MAT244 Aplicaciones de la Matemática para la Ingeniería. La versión oficial tenía el siguiente programa:

OBJETIVOS:

Al aprobar la asignatura, el alumno será capaz de:

- Comprender y utilizar conceptos matemáticos en algunos modelos específicos de ingeniería.
- Aplicar técnicas y conceptos matemáticos para resolver problemas específicos del ámbito de la ingeniería.

CONTENIDOS:

1. Métodos de aproximación
 - a) Perturbaciones en problemas de valores propios.
 - b) Estimaciones variacionales.
2. Propagadores
 - a) La distribución delta de Dirac.
 - b) Funciones de Green en ecuaciones diferenciales ordinarias y parciales.
 - c) Núcleo de ecuaciones integrales.
3. Aplicaciones. Modelación matemática en áreas tales como:
 - a) Tratamiento de señales.
 - b) Análisis estructural.
 - c) Dinámica de estructuras.
 - d) Sistemas de control.
 - e) Otros.

1. Transformando un curso con python

Mi visión:

Dictar el curso que me hubiera gustado recibir.

1. Transformando un curso con python

Mi primera versión del curso tenía el siguiente programa:

MAT244

Unidad 0: Presentación del curso

- [Reglas aplicables a todas las Lecturas](#)
- [Reglas aplicables a todas las Tareas](#)
- [Lecturas Unidad 0, Viernes 22 Agosto](#)
- Programa:
 - Semana 1: Contenido y motivación del curso, ejemplos de aplicaciones.

Unidad 1: Herramientas transversales en ingeniería.

- [Lecturas Unidad 1, Viernes 5 Septiembre](#)
- [Tarea Unidad 1, Viernes 12 Septiembre](#)
- Taller Python/Numpy, Heat Equation ([zip](#))
- Transparencias ([zip con contraseña](#))
- Programa:
 - Semana 1: Educated guessing, aproximaciones y adimensionalización.
 - Semana 2: Más Adimensionalización
 - Semana 3: Python, numpy, scipy, matplotlib.
 - Semana 4: Visualización y metodologías de trabajo.

Unidad 2: Modelación y error.

- [Lecturas Unidad 2, Viernes 3 Octubre](#)
- [Tarea Unidad 2, Viernes 10 Octubre](#)
- Transparencias ([zip con contraseña](#))
- Programa:
 - Semana 1: Tipos de errores. Error de entrenamiento y error de predicción.
 - Semana 2: Holdout Set y Cross Validation.
 - Semana 3: Normas de error vectorial y matricial, índices de error.

Unidad 3: Machine Learning y Data Science.

- [Lecturas Unidad 3, Viernes 17 Octubre](#)
- [Tarea Unidad 3](#)
- Transparencias ([zip con contraseña](#))
- Programa:
 - Semana 1: Machine Learning: introducción, ejemplos y librerías.
 - Semana 2: Unsupervised learning: k-means
 - Semana 3: Supervised learning: Regresión Lineal y Logística.
 - Semana 4: PCA: aplicaciones de SVD y vectores propios.

Unidad 4: Proyectos en ingeniería

- Programa:
 - Semana 1: Minería subterránea y rajo abierto.
 - Semana 2: Sismicidad y tsunamis.
 - Semana 3: Bioquímica.

Unidad 5: Proyectos

- [Proyectos](#)
- Programa:
 - Semana 1: Presentación de proyectos en clase.

1. Transformando un curso con python

Mi resumen del curso en la primera clase dice mucho:

- Curso atípico: interactivo y énfasis en vinculación a "mundo real"
- Trabajo con datos reales: esencial
- Programaremos en python
- Para aprobar: ¡trabajar!

1. Transformando un curso con python

Algunas "innovaciones":

- Exigir informes y presentaciones en beamer (latex).
- Atrasos planificados.
- Sin prueba. Evaluaciones: Tareas, lecturas y proyecto final.
 - Tareas: código en Python.
 - Lecturas: artículos interesantes.
 - Proyecto final: tema a elección, presentación abierta a la comunidad.
- Presentaciones, lecturas y código en un repositorio bitbucket:
<https://bitbucket.org/sebastiandres/mat244>

1. Transformando un curso con python

HOW TO STUDY MATH



Don't just read it; fight it!

— Paul R. Halmos

Mi sello:

1. Transformando un curso con python

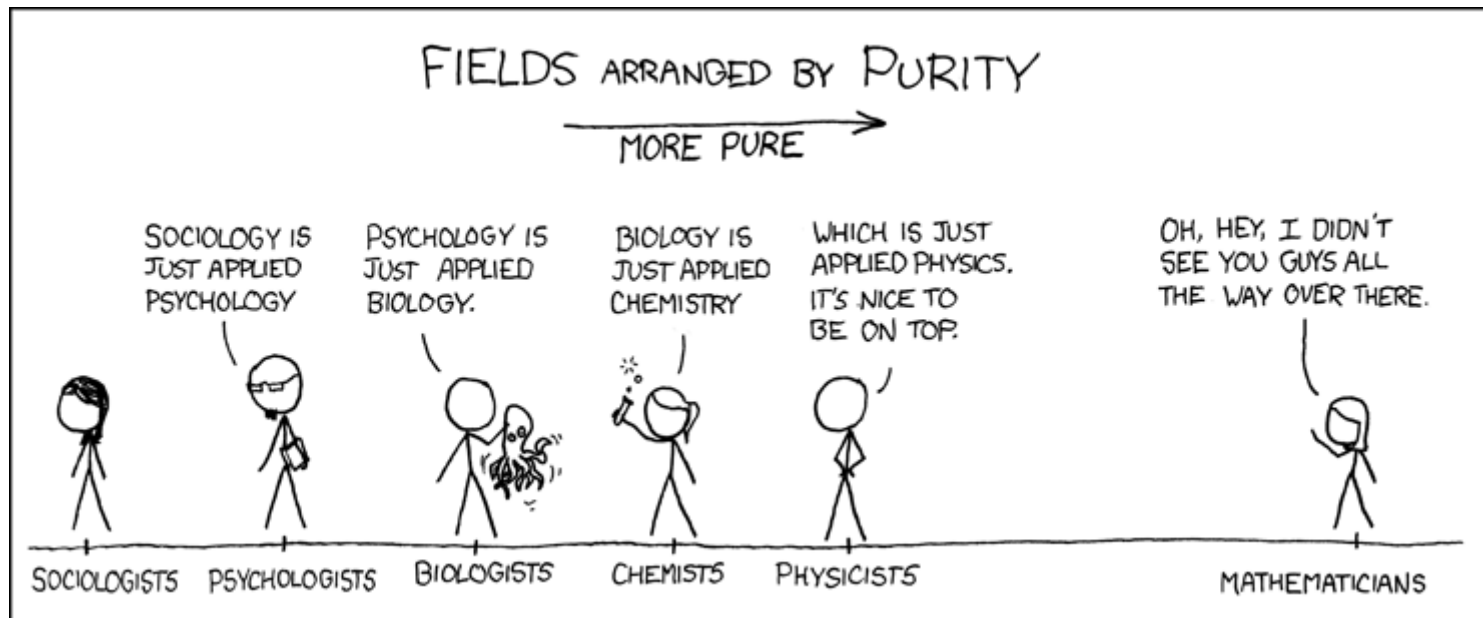
Año 2015.

La segunda versión del curso consideró la retroalimentación de los estudiantes de reducir carga: lecturas son opcionales.

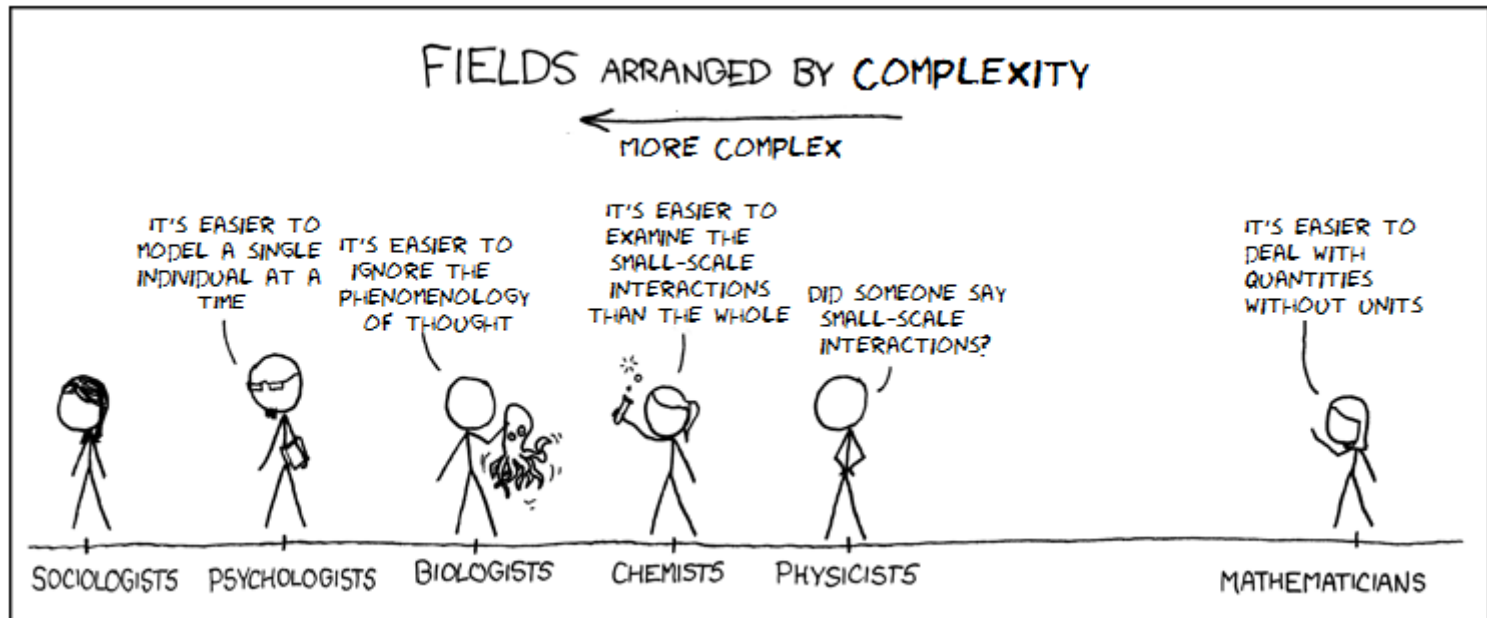
Además, consideró los siguientes cambios:

- Repositorio en github: https://github.com/usantamaria/mat281_2015S2
- Jupyter notebooks y RISE para el contenido
- Clases teóricas, ayudantías y laboratorios.

1. Transformando un curso con python



1. Transformando un curso con python



1. Transformando un curso con python

Año 2018.

La tercera versión del curso consideró mis aprendizajes en la industria:

Otros cambios:

- Repositorio en github: https://github.com/usantamaria/mat281_2018S2
- Mayor foco en data science y Machine Learning, y nuevas librerías (¡pandas!).
- Foco en laboratorios como métodos de enseñanza.

1. Transformando un curso con python

Año 2019.

La cuarta versión del curso ya no la dicté yo, la dictaron mis antiguos alumnos: Alonso Ogueda y Francisco Alfaro.

- Repositorio en github: https://github.com/usantamaria/mat281_2019S2

1. Transformando un curso con python

Año 2021.

¿Qué nuevos temas incluiría en el curso?

- Data Engineering
- Ética
- Publicidad y la economía de la atención
- Comunidad y open source

1. Transformando un curso con python

Data Engineering

El ingeniero de software, ingeniero de datos, científico de datos y roles similares, cada vez se vuelven más "full stack" y tienen que auto-atenderse en el uso de grandes volúmenes de datos.

Estudiantes necesitan saber:

- Nociones básicas de SQL
- Nociones básicas de visualización (gráficos y dashboards)
- Nociones básicas de data pipelines y ETLs.

1. Transformando un curso con python

Ética

- "Create a product right" versus "Create the right product".
- GDPR y protección de datos.

1. Transformando un curso con python

Publicidad y la economía de la atención

En un mundo disperso y bombardeado de distracciones, tener foco es un superpoder

- Nociones de cómo funciona la publicidad y porqué todos los productos quieren nuestra atención 100% del tiempo.
- Nociones de flow state y deep work

1. Transformando un curso con python

Comunidad y open source

- FOSS (Free Open Source Software) está presente en todas las empresas
- Contribuir en un FOSS es un imperativo ético (y skill marketeable)
- Encontrar y crear comunidades de interés
- Participar en eventos de tecnología
- Tareas:
 - Contribuir en la documentación de un proyecto de Open Source
 - Hacer un Pull Request en un proyecto de Open Source
 - Editar un artículo en wikipedia

1. Transformando un curso con python

Importante

Curso fue tremendamente exitoso y muy valorado por los estudiantes.

Presento el caso exitoso. Pero también he dictado otros cursos sin el mismo éxito:

- *Matemáticas 1* (cálculo y álgebra primer año ingeniería)
- *Introducción a la programación* (python primer año ingeniería)

1. Transformando un curso con python

¿Qué aprendí?

Considerar el curso como un producto (de software) donde los estudiantes son los clientes:

- Pedir retroalimentación de manera frecuente.
- Estar en sintonía con las herramientas de la industria.
- Mejorar iterativamente el contenido.

1. Transformando un curso con python

¿Qué aprendí?

- Ponerse en los zapatos de un estudiante: ¡no olvidar lo que es no saber!
- Experimentar. Atreverse a que las cosas no resulten durante la clase.

1. Transformando un curso con python

¿Qué aprendí?

- Enseñar habilidades y competencias, por sobre contenido.
- *"Learn in case" vs "learn in need"*.

1. Transformando un curso con python

¿Qué aprendí?

- Genera impacto a través de transparencia y crear hacia la comunidad.
- Sólo podemos inspirar a que alguien aprenda por sí mismo.
- Que las evaluaciones no sean irrelevantes.

1. Transformando un curso con python

¿Qué aprendí?

Usa ejemplos reales, imágenes y videos atractivos, datasets interesantes.

- <https://archive.ics.uci.edu/ml/datasets.php>
- <https://es.datachile.io/>
- <http://datagramas.cl/courses/infovis/resources/>

¿Qué aprendí?



No dictes un curso, ¡comienza una revolución!

2. Algunas herramientas recomendadas

¿Qué resultados arrojó la encuesta?

¿Qué contenido te gustaría recibir?

Siempre estoy interesado en aprender más sobre machine learning, pero insisto si es posible quisiera ver ejemplos de los cuales los alumnos puedan participar.

Ejemplos prácticos

Me gustaria ideas de proyectos que puedan ser interesantes para motivar a los alumnos

- * automatizacion y scripting

- * web scraping

Desarrollo web y bases de datos con ORM

Aplicaciones de funciones

POO

Machine learning

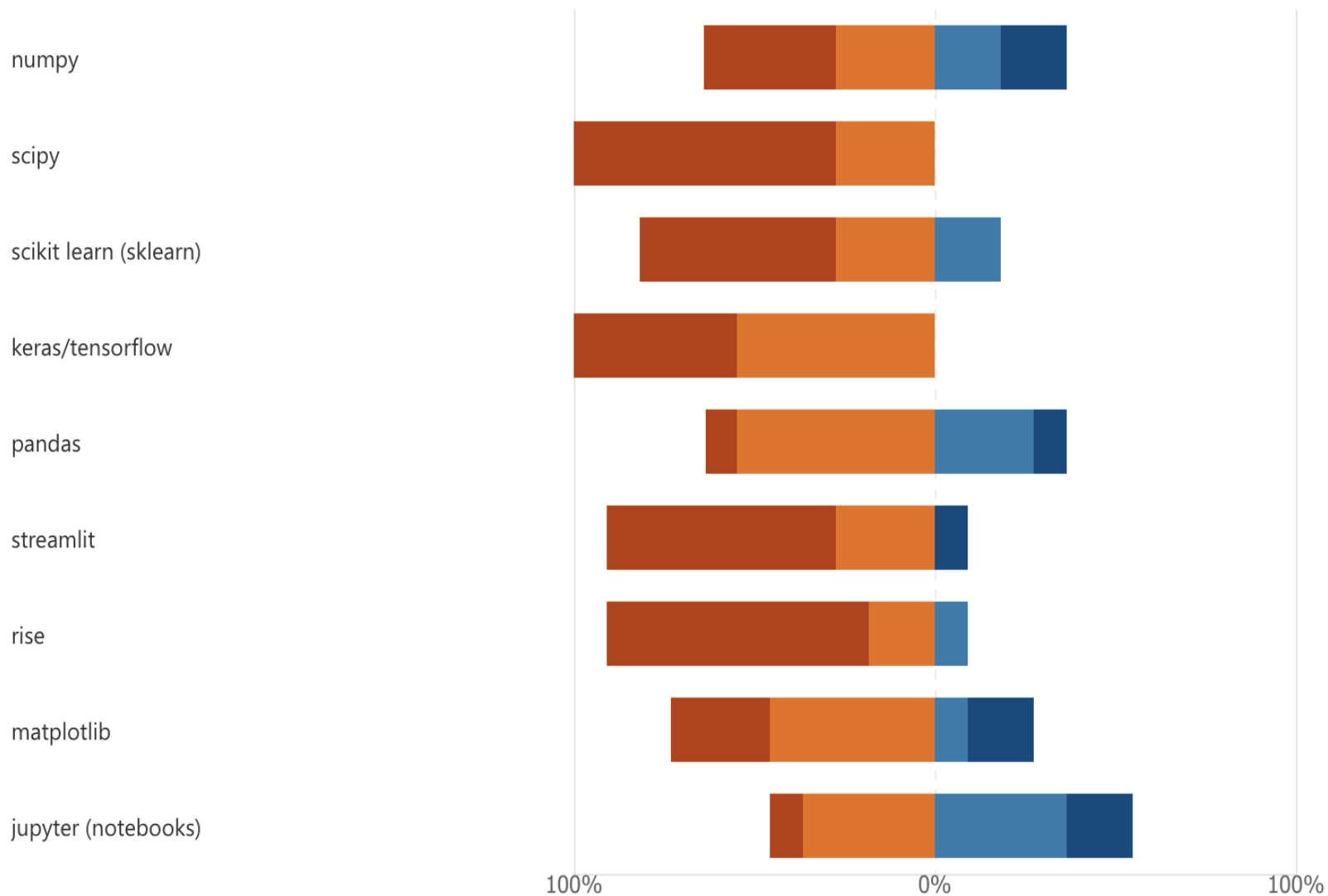
Contenido sobre data science.

Me gustaria comenzar a aprender sobre machine learning

Manejo de errores, objetos

Información sobre multiproceso y paralelismo

■ No lo conozco ■ Lo he probado ■ Lo uso a veces ■ Lo conozco bien



¿Qué contenidos enseñas (o te gustaría enseñar) a tus alumnos mediante Python?

De momento estamos comenzando a ver POO, una introducción esencialmente. Y si tuviera que elegir un tema para enseñar, me gustaría mostrarles a mis alumnos algo con Redes Neuronales, o quizás diseñar algún juego que puedan probar entre ellos. Que sea en si más didáctico que complejo, tal vez reconocimiento de imágenes, o diseñar un juego de cartas. Quiero que ellos participen de alguna manera.

Resolución de problemas, algoritmos.

Algo de Pygame, interfaces Graficas en escritorio con tkinter y procesamiento de imagenes

Por ahora estoy enseñando contenidos basicos

Utilizar IDE's y/o editores de texto, terminal, modularizacion de archivos, estructuras de datos, typing/type hints, pruebas unitarias, test driven development, buenas prácticas, API's.

Aplicaciones con funciones

POO

Me gustaría enseñar una introducción de GUI en python

De todo un poco. Sea desarrollo web, desktop de propósito general, o data science.

Python basico

El uso correcto de clases y objetos en python

Enseño Python para principiantes

2. Herramientas recomendadas

- jupyter notebook + rise
- streamlit
- scipy

2.1 rise

En lugar de mostrar una larga página web con las celdas, las celdas **se agrupan** en diapositivas.

Las distintas celdas pueden ser editadas y ejecutadas *DURANTE LA PRESENTACIÓN*.

Se usa la librería de javascript [reveal.js](#).

2.1 rise

En lugar de mostrar una larga página web con las celdas, las celdas **se agrupan** en diapositivas.

Las distintas celdas pueden ser editadas y ejecutadas *DURANTE LA PRESENTACIÓN*.

Se usa la librería de javascript [reveal.js](#).

Con **herramientas** de *markdown*, **html** y latex: $e^{i\pi} + 1 = 0$

2.1 rise

En lugar de mostrar una larga página web con las celdas, las celdas **se agrupan** en diapositivas.

Las distintas celdas pueden ser editadas y ejecutadas *DURANTE LA PRESENTACIÓN*.

Se usa la librería de javascript [reveal.js](#).

Con **herramientas** de *markdown*, **html** y latex: $e^{i\pi} + 1 = 0$

```
In [ ]: # kernels, en particular python
        print("Hola \U0001F30E")
```

2.1 rise

2.1 rise

Ventajas

- Simplificar la **generación** de material.
- Simplificar la **distribución** del material.

2.1 rise

Ventajas

- Simplificar la **generación** de material.
- Simplificar la **distribución** del material.

Desventajas

- RISE funciona sólo con Jupyter Notebook. No funciona con Jupyter Lab ni con Google Colaboratory.
- Es completamente interactivo, con todos los riesgos y beneficios que eso representa.

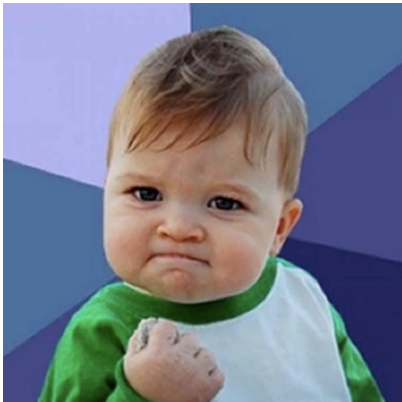
2.1 rise: ¿Qué contenido incluir?

Todo lo que ya incluyes en Jupyter Notebook: puedes mezclar contenido usando las celdas de markdown y código, según necesites:

- **Markdown:** texto, latex, imágenes, tablas, etc.
- **Código:** código, gráficos simples o interactivos, videos, sonido, iframes, javascript, entre otros.

2.1 rise: fotos y GIFs Animados

Incluir gifs animados es igual de fácil que incluir imágenes, y suele ayudar a tener apoyos visuales atractivos.



2.1 rise: gráficos

Para mostrar gráficos resulta práctico que no se genere una ventana adicional, sino que se agreguen a la celda de resultados. Esta es una práctica común en jupyter notebook/lab, pero es más importante aún al pensar en las diapositivas.

```
In [ ]: %matplotlib inline
        from matplotlib import pyplot as plt
        import numpy as np

        with plt.xkcd():
            fig = plt.figure(figsize=(14,6))
            x = np.linspace(-5,5,num=1000)
            y = np.abs(np.abs(np.sin(5*x)/x))
            plt.plot(x,y)
```

In []:

```
import numpy as np
import matplotlib.pyplot as plt
dt = 0.01
t = np.arange(0.0, 2.0, dt)
s = np.sin(2*np.pi*t)

plt.figure(figsize=(12,6))
plt.plot(t, s)
plt.title(r'$\alpha_i > \beta_i$', fontsize=20)
plt.text(1, -0.6, r'$\sum_{i=0}^{\infty} x_i$', fontsize=20)
plt.text(0.6, 0.6, r'$\mathcal{A}\mathrm{sin}(2 \omega t)$',
         fontsize=20)
plt.xlabel('time (s)')
plt.ylabel('volts (mV)')
plt.show()
```


2.1 rise: IPython.display

```
In [ ]: from IPython.display import Audio, Image, display
a = Audio("http://www.w3schools.com/html/horse.ogg")
i = Image(filename="images/Python.png", width=100)
display(a,i)
```

```
In [ ]: from IPython.display import Javascript
Javascript('alert("¡¡Tenemos javascript!!");')
```

2.1 rise: widgets

In []:

```
%matplotlib inline
import ipywidgets as widgets
import numpy as np
import matplotlib.pyplot as plt

@widgets.interact(grado=(0, 9), N_puntos=(5, 35))
def mi_plot(grado=3, N_puntos=5):
    x = np.linspace(-10, 10, N_puntos)
    y = x**grado
    plt.figure(figsize=(12, 8))
    plt.plot(x, y, 'ro-')
    plt.show()
```

2.1 rise: clases

Al empaquetar tu código con clases, asegúrate de incluir una función `_repr_html_` para poder interactuar nativamente con jupyter notebook (y RISE):

```
In [ ]: class Alerta(object):  
  
    def __init__(self, text):  
        self.text = "\U000026A0 - " + text.upper()  
  
    def _repr_html_(self):  
        return "<h1 style='color:white;background:red;padding:30px'>" + self.text + "</h1>"
```

```
In [ ]: Alerta("Comenzar a hablar de RISE")
```

2.1 rise

- ¿Para qué podría usarse?
 - Centralizar el material en un único repositorio.
- Más información:
 - Página oficial: <https://rise.readthedocs.io/>
 - Charla Pycon pylatam:
https://github.com/sebastiandres/talk_2021_08_pylatam
 - Tutorial: <https://sebastiandres.github.io/blog/tutorial-rise/>

2.2 Streamlit

La librería streamlit permite crear aplicaciones web de manera extremadamente simple.

In []:

```
from IPython.display import IFrame  
IFrame("https://share.streamlit.io/sebastiandres/talk_2021_11_pyconcl/main", width=600, height=400)
```

2.2 Streamlit

- ¿Para qué podría usarse?
 - Crear tareas y proyectos con impacto real en las comunidades: desde python básico a avanzado.
 - Ayudar a los estudiantes a crear un portafolio de proyectos.
- Más información:
 - Página oficial: <https://streamlit.io/>
 - Charla en Pycon Chile:
https://github.com/sebastiandres/talk_2021_11_pyconcl

2.3 scipy

La librería scipy complementa numpy, disponibilizando una gran cantidad de algoritmos científicos.

En scipy existen muchos submódulos específicos para ciencia:

- `scipy.cluster` Vector quantization / Kmeans
- `scipy.constants` Constantes físicas y matemáticas
- `scipy.fftpack` Transformadas de Fourier
- `scipy.integrate` Rutinas para integración
- `scipy.interpolate` Interpolación
- `scipy.io` Entrada y salida
- `scipy.linalg` Rutinas para algebra lineal

- `scipy.ndimage` n-dimensional image package
- `scipy.odr` Orthogonal distance regression
- `scipy.optimize` Optimización
- `scipy.signal` Procesamiento de señales
- `scipy.sparse` Matrices dispersas
- `scipy.spatial` estructuras de datos espaciales y algoritmos
- `scipy.special` Funciones matemáticas especiales
- `scipy.stats` Estadística

Ecuaciones

Para resolver un sistema de ecuaciones usamos numpy y scipy:

$$x + y + z = 1$$

$$x - y + z = 2$$

$$x + y - z = 3$$

```
In [ ]: # Importar las librerías
import numpy as np
from scipy.linalg import solve

# define matrix A using Numpy arrays
A = np.array([[1, 1, 1],
              [1,-1, 1],
              [1, 1,-1]])
#define matrix B
b = np.array([1, 2, 3])
# linalg.solve is the function of NumPy to solve a system of linear scalar equations
x = solve(A, b)
print("Solución:\n", x)
```

```
In [ ]: # Verifiquemos
print("A*x=", A@x)
```

También es posible calcular la matriz inversa y multiplicar, pero para sistemas de ecuaciones grandes es muy ineficiente.

```
In [ ]: from scipy.linalg import inv
        # Invertir explícitamente una matrix
        A_inv = inv(A)
        print("A^{-1}=\n", A_inv)
        print("A^{-1} * x =\n", A_inv @ b)
```

Ejemplo para clases:

$$\text{Apple} + \text{Apple} + \text{Apple} = 30$$

$$\text{Apple} + \text{Banana} + \text{Banana} = 18$$

$$\text{Banana} - \text{Coconut} = 2$$

$$\text{Coconut} + \text{Apple} + \text{Banana} = ?$$

In []:

```
# Solución
import numpy as np
from scipy.linalg import solve

# Buscaremos resolver x = [manzana, banana, coco]

# define matrix A using Numpy arrays
A = np.array([[3, 0, 0],
              [1, 8, 0],
              [0, 4, -2]])

#define matrix B
b = np.array([30, 18, 2])

# linalg.solve is the function of NumPy to solve a system of linear scalar equations
x = np.linalg.solve(A, b)
print("Valores individuales:\n", x)
print("Solución:\n", np.dot(x, np.array([1,3,1])))
```

3. Aprendiendo Machine Learning, Deep Learning y/o Data Science

3. Aprendiendo Machine Learning, Deep Learning y/o Data Science

Breve y modesta **motivación...**

3. Aprendiendo Machine Learning, Deep Learning y/o Data Science

Breve y modesta **motivación**...

Machine learning algorithms can figure out how to perform important tasks by generalizing from examples. This is often feasible and cost-effective where manual programming is not. As more data becomes available, more ambitious problems can be tackled

Pedro Domingos: [A Few Useful Things to Know about Machine Learning](#)

Machine Learning

- Estudia y construye sistemas que pueden aprender de los datos, más que seguir instrucciones explícitamente programadas.
- Machine Learning es un conjunto de técnicas y modelos que permiten el modelamiento predictivo de datos, reunidas a partir de la intersección de elementos de probabilidad, estadística e inteligencia artificial.
- Pregunta fundamental: ¿Qué conocimiento emerge a partir de los datos? ¿Qué modelo/técnica otorga la mejor predicción para estos datos?

Data Science

- Se preocupa de la practicidad de resolver problemas complejos utilizando datos.
- Data Science es la aplicación de diversas técnicas de modelamiento con un fin específico. También se conoce como eScience.
- La base de datos a utilizar no ha sido necesariamente creada.
- Típicamente, un data scientist se encuentra en la industria, buscando automatizar un análisis complejo para un cliente (interno o externo).
- Pregunta fundamental: ¿Qué puedo decir de **X** a partir de los datos?

Inteligencia Artificial

- Tiene por objetivo hacer que el computador ejecute tareas para las cuales el hombre, en un contexto dado, es actualmente mejor que la máquina.
- Es un concepto más amplio que Machine Learning, en el sentido de definir el concepto de *inteligencia*.
- Algunos enfoques de inteligencia consideran que tiene que ver principalmente con acciones racionales. Que un agente inteligente es capaz de percibir y actuar, realizando la mejor acción posible en una situación dada.
- Pregunta fundamental: ¿Puedo modelar el problema pero que su resolución no es razonablemente factible por algoritmos existentes?

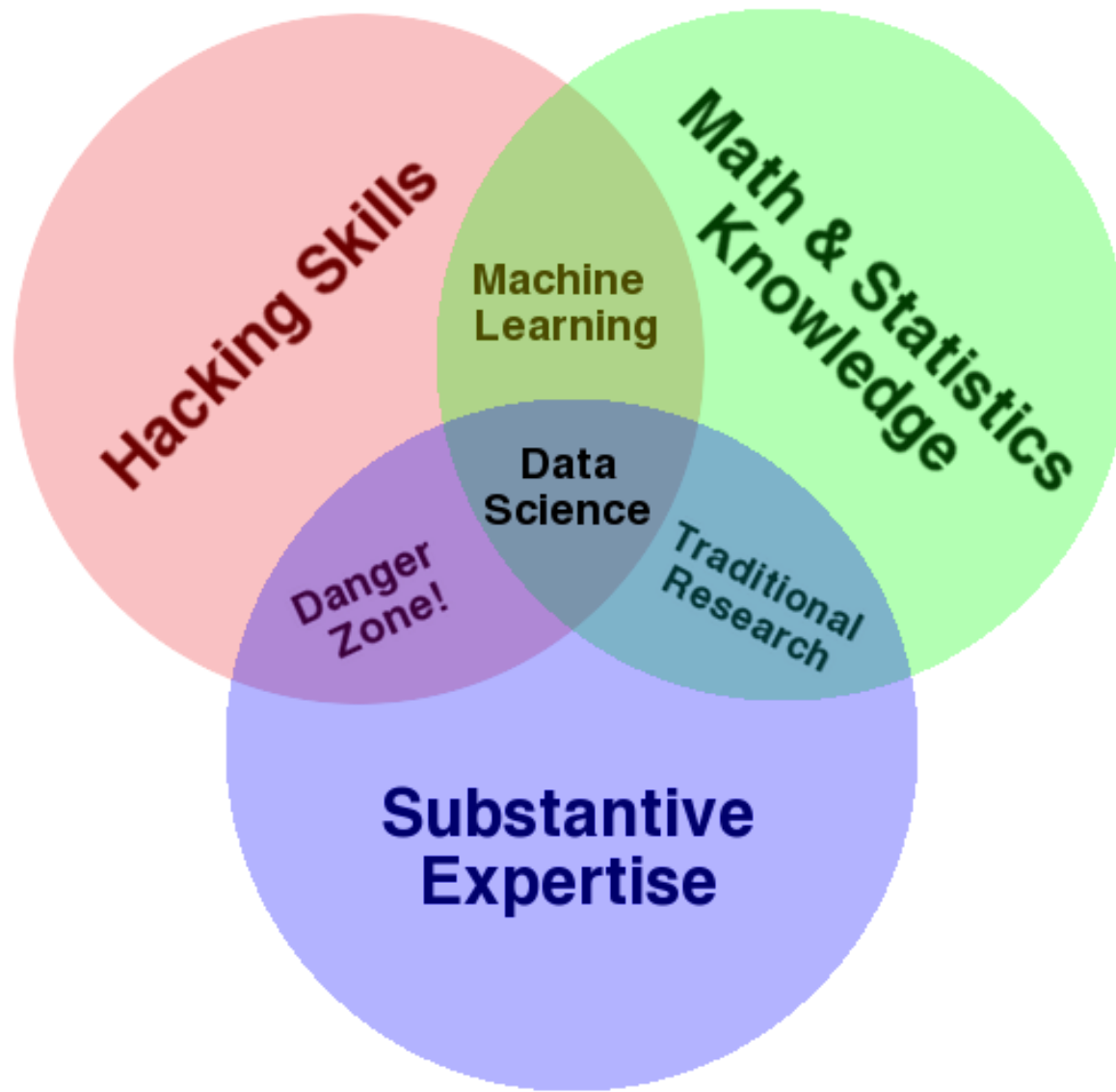
Big Data

- Caracterizado por 4 V:
 - **Volumen:** ¿Cuántos datos se requiere procesar? ¿Terabytes, Exabytes, Zettabytes?
 - **Variedad:** ¿Qué patrón siguen los datos? ¿SQL o noSQL?
 - **Velocidad:** ¿Con que velocidad podemos procesar los datos? ¿A qué velocidad se generan?
 - **Veracidad:** ¿Qué tan confiables son los datos?
- Pregunta fundamental: ¿Cómo proceso esta monstruosidad de datos?

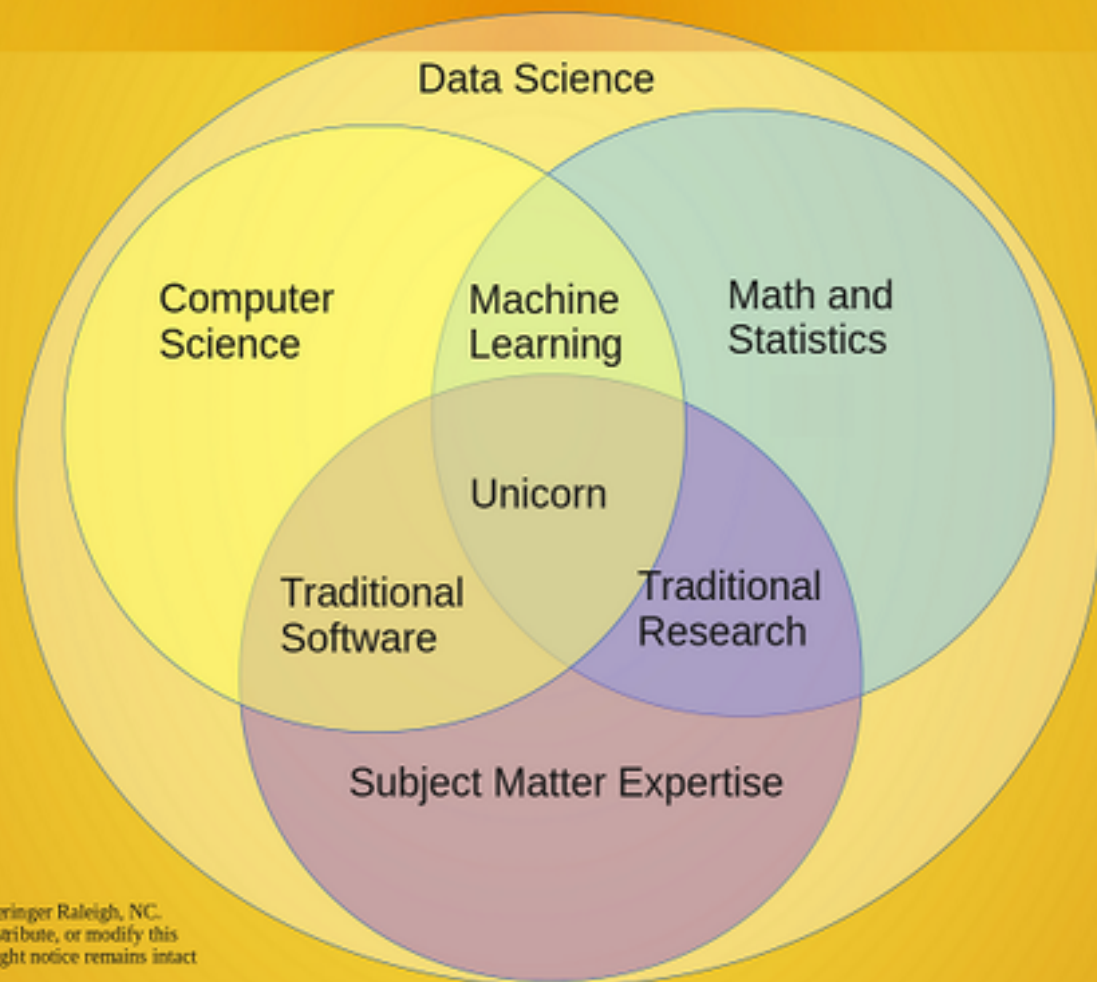
Es usual ver en LinkedIn, foros y/o blogs una serie de infografías tratando de definir o separar conceptos. Al no existir límites fijos, estas infografías suelen variar entre autores.

No creas todo lo que ves.

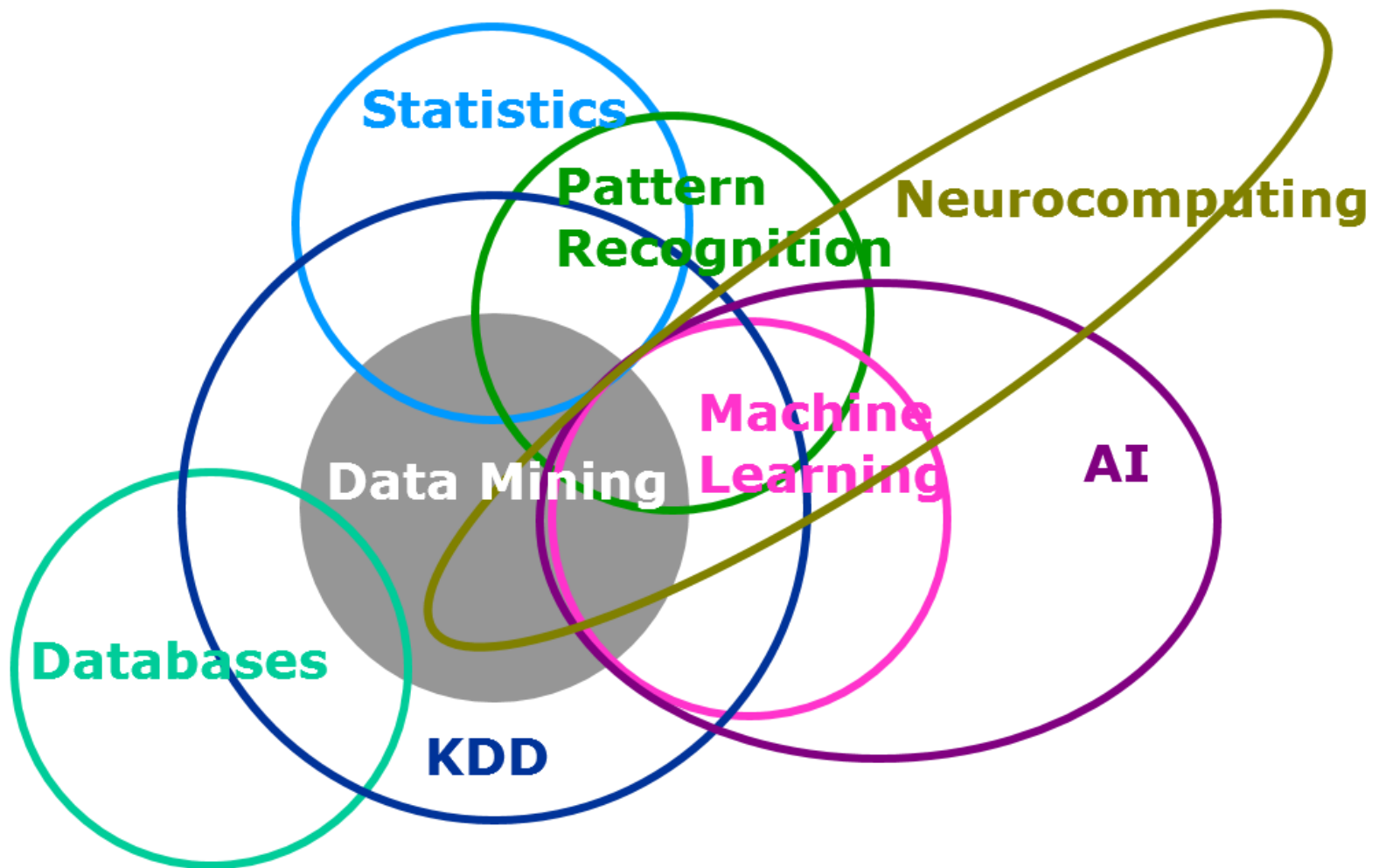
My opinión personal: No todo lo que brilla es oro. No soy fan de estas infografías o diagramas.



Data Science Venn Diagram v2.0



Copyright © 2014 by Steven Geringer Raleigh, NC.
Permission is granted to use, distribute, or modify this
image, provided that this copyright notice remains intact



Modelamiento

¿Qué es un modelo?

- Un modelo es
 - Una representación abstracta y conveniente de un sistema.
 - Una simplificación del mundo real.
 - Un medio de exploración y de explicación para nuestro entendimiento de la realidad.
- Un modelo NO es
 - Igual al mundo real.
 - Un sustituto para mediciones o experimentos.

Modelamiento

¿Por qué utilizar modelos?

- Permiten reproducir escalas temporales o espaciales donde las mediciones o los experimentos son costosos, difíciles o imposibles.
 - Ejemplo: deriva continental, impacto de asteroides, evolución de enfermedades, etc.
- Es mejor que no hacer nada o confiar en juicios dudosos de terceros.
 - Ejemplo: Predicción meteorológica versus danza de la lluvia.

"All models are wrong,
some are useful" - George Box"

Modelamiento

Objetivos del modelamiento

- Ayudar al diseño de experimentos y mediciones.
- Condensar entendimiento al forzar la abstracción, integración y formalización de ideas científicas.
- Permitir la realización de experimentos virtuales.
- Divulgar conocimiento científico a no-expertos.
- Realizar predicciones.

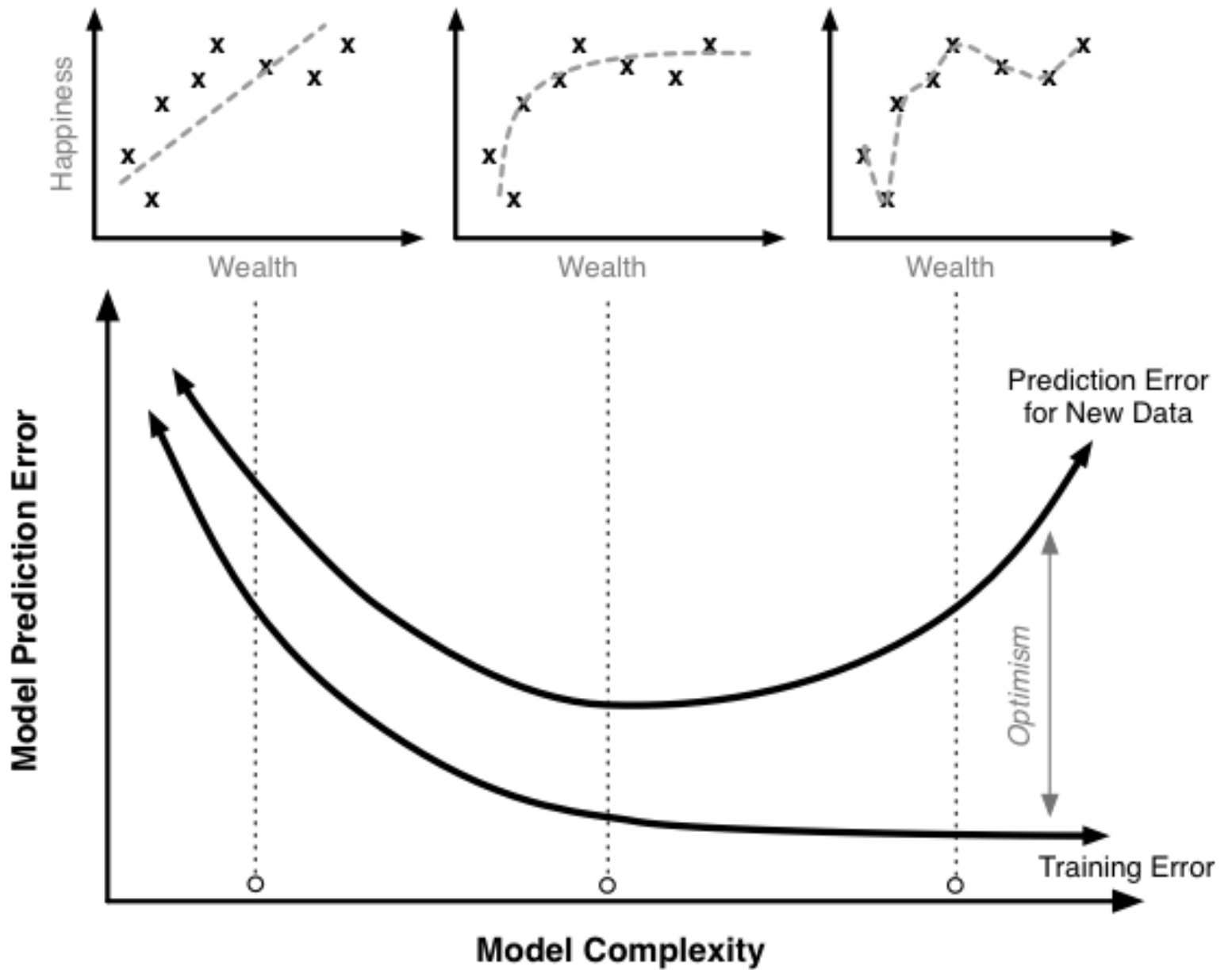
El modelamiento nunca es un objetivo en sí mismo.

Modelamiento

Algunos principios básicos

1. Un mismo fenómeno físico puede tener distintos modelos, cada uno apropiado para el estudio de un cierto aspecto del fenómeno.
2. Modelamiento requiere tiempo: es necesario determinar el nivel de complejidad en función los resultados deseados y del tiempo disponible.
3. GIGO: Garbage In = Garbage Out: Un buen modelo no compensa malos datos de entrada.
4. Todo modelo es una simplificación de la realidad.
5. Que algo se pueda simular, no significa que se corresponda con la realidad.
6. Validación completa del un modelo es imposible, al igual que cualquier teoría.

"Everyone believes the recorded data,
except the scientist that record it.
No one believes the model,
except for the scientist that created it."
- Unknown



Actividad Práctica

Realizaremos una actividad práctica de aprendizaje:

Objetivo:

- Aprender diferencia entre parámetros y metaparámetros de un modelo.
- Aprender la importancia de la cantidad de datos de entrenamiento.
- Aprender la diferencia entre datos de entrenamiento, testeo y validación.

Enlace: https://share.streamlit.io/sebastiandres/streamlit_ml_edu/main

Consejos

Excelente artículo de Pedro Domingos: [A Few Useful Things to Know about Machine Learning](#).

1. Learning = Representation + Evaluation + Optimization
2. La meta fundamental es generalizar a partir de los ejemplos.
3. Sólo tener datos no es suficiente.
4. Cuidado con overfitting en el modelo.
5. La intuición es difícil en grandes dimensiones.

3. Machine Learning

¿Cómo seguir?

- ¡Existe un sitio con una animación increíble!: <http://www.r2d3.us/>
- Kaggle: datasets, comunidad, muchos problemas (sintéticos y reales).
- Muchos cursos en línea, libros, etc. El problema no es acceso, sino seleccionar buenas fuentes y dedicarle tiempo.