Data cleansing y otras cosas

Documentación

Alejandro Manuel Arranz López

datahack
BIG DATA FAMILY

Una definición (o mejor dos)

Machine Learning es la **ciencia** (y el **arte**) de programar **máquinas** de manera que estas puedan **aprender** a partir de **datos**.

Se habla de que un programa ha aprendido de la experiencia (**E**) con respecto a una determinada tarea (**T**) y algún tipo de rendimiento medible (**R**); cuando su rendimiento en T, medido por **R**, **mejora** a medida que aumenta **E**.

Best Buy Viagra Generic Online

Viagra 100mg x 100 Pills $125, Free Pills & Reorder Discount. We accept VISA & E-Check Payments, 90000+ Satisfied Customers!

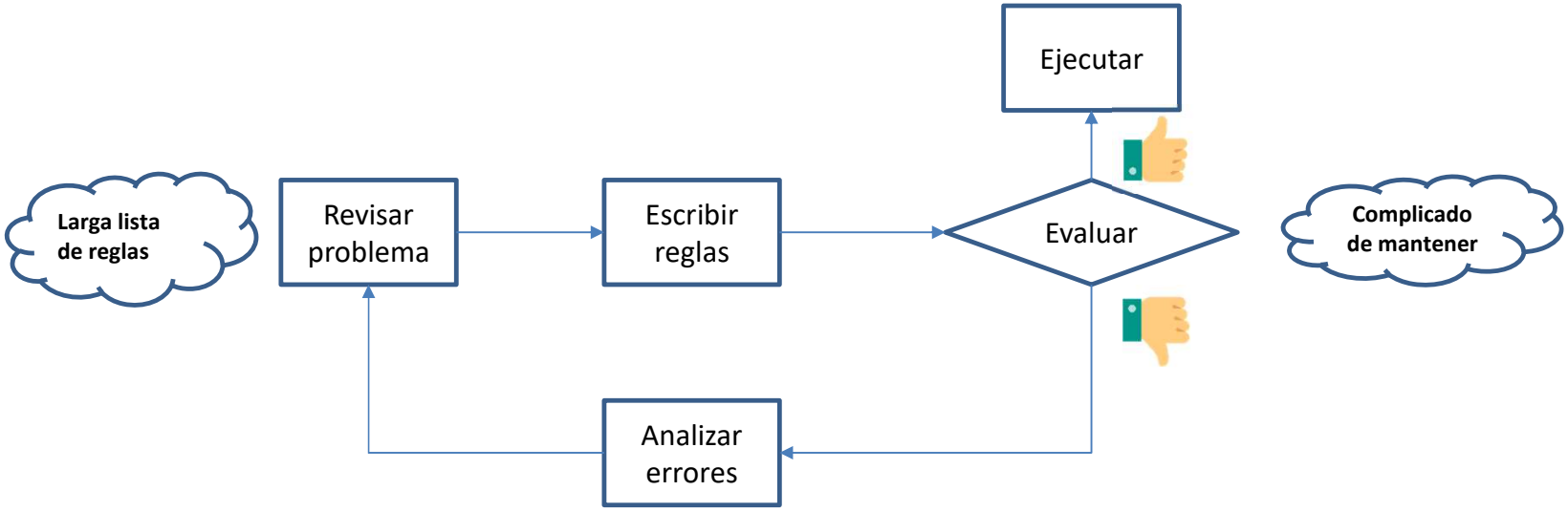
Top Selling 100% Quality & Satisfaction guaranteed!

Un ejemplo (algo clásico y omnipresente)

- El filtro de spam que incorpora nuestro correo no es sino Machine Learning que aprende a distinguir lo que es Spam de lo que no lo es a partir de ejemplos de mails "buenos" y de spam.

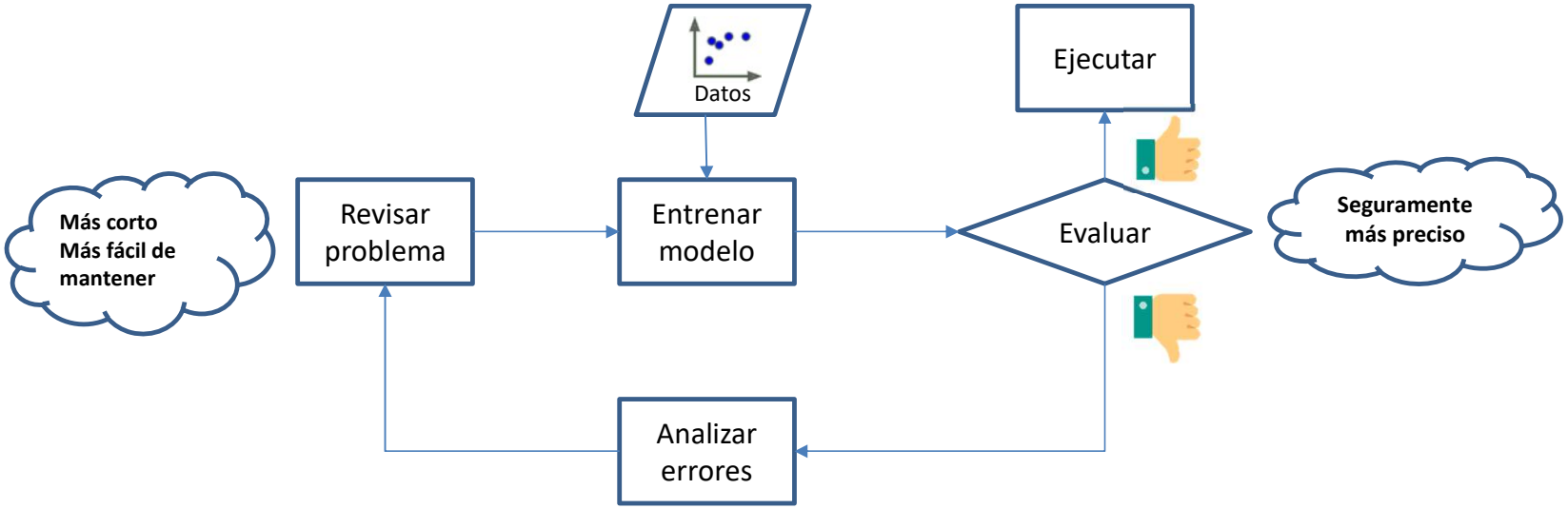
SPAM: Aproximación tradicional

1. Detección de palabras y expresiones comunes: “Free”, “Amazing”, “Viagra”...
2. Codificar algoritmo de detección para todos los patrones detectados,
   - Si se detecta cierta cantidad de estos patrones, el programa marcará el correo como Spam.
3. Probar el programa y repetir los dos pasos anteriores hasta que el resultado sea bueno.



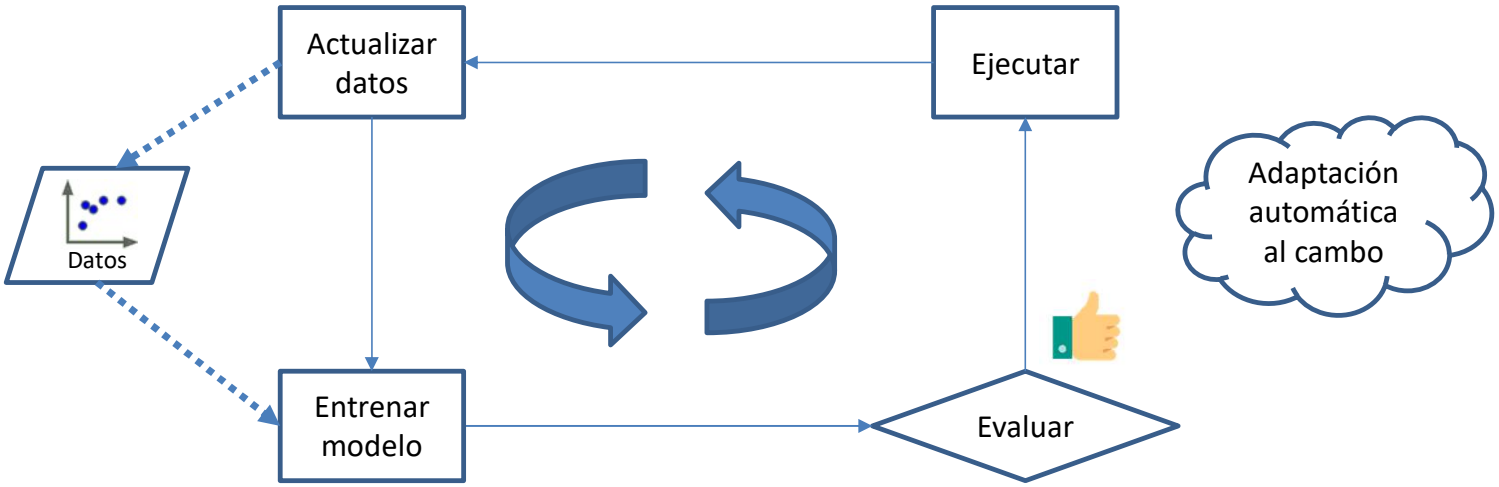
SPAM: Aproximación ML,

1. El modelo automáticamente aprende qué palabras o frases son buenos indicadores de spam.
2. Esto se consigue comparando ejemplos de spam, con ejemplos de correos válidos.
3. Probar el programa y repetir los dos pasos anteriores hasta que el resultado sea bueno.



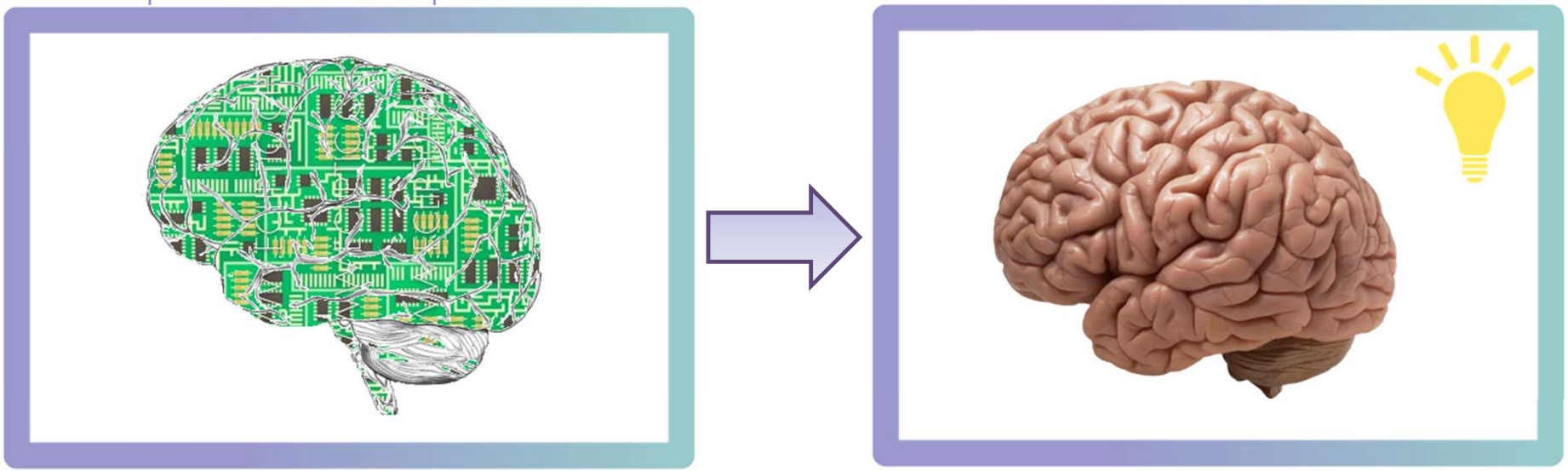
SPAM: Adaptación al cambio,

- En el más que probable caso de que los Spammers espabilen y cambien su estrategia,
  - Nuestra aproximación tradicional haría aguas y habría que replantear las reglas.
  - La aproximación con Machine Learning detectaría nuevos patrones en el spam marcado como tal por los usuarios.



APRENDIZAJE HUMANO

- Aunque en algunos casos puede ser arduo, es posible inspeccionar un modelo de ML entrenado, para ver que patrones ha detectado; en ocasiones esto puede llevarnos a descubrir tendencias o relaciones que no sospechábamos siquiera.



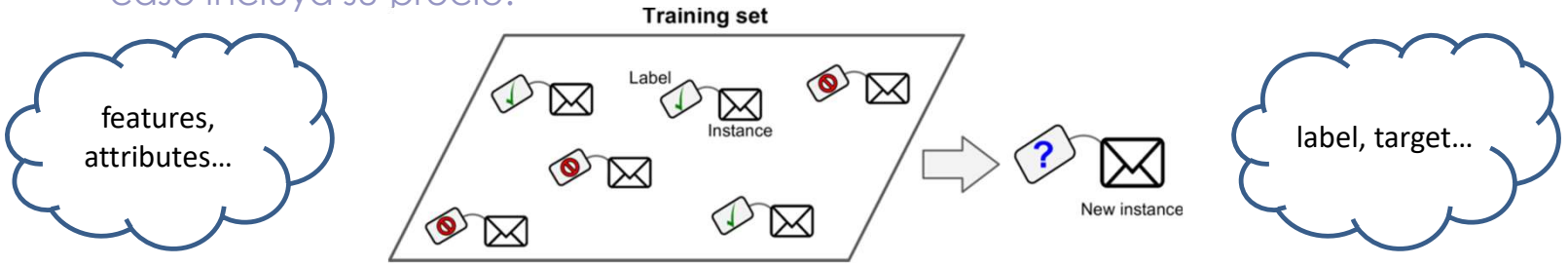
- Problemas cuya solución pasa por muchas manualidades y largas listas de reglas.
- Problemas cambiantes, en los que un modelo de Machine Learning se puede adaptar a los nuevos datos.
- Obtener información sobre problemas complejos que involucran muchos datos.
- Problemas complejos para los cuales no hay solución mediante las aproximaciones tradicionales.

Pero no todos los problemas son iguales…

- El aprendizaje supervisado se distingue porque los datos de entrenamiento que se le pasan al algoritmo incluyen la solución correcta para cada uno (lo que se conoce como etiqueta, **label** o **target**)

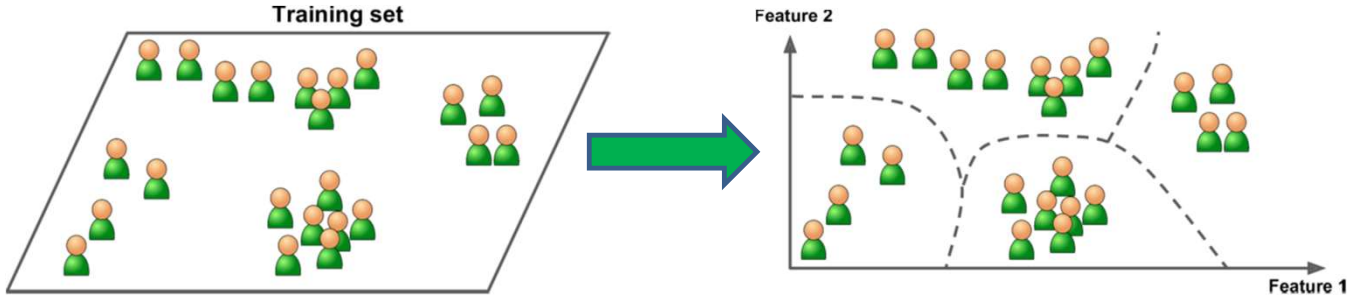
- Dos problemas típicos:
  - **Clasificación**: siguiendo con el ejemplo del spam, el modelo es entrenado con muchos ejemplos de mails acompañados de su etiqueta (spam o ham). El objetivo es que el modelo sea capaz de clasificar los nuevos mails que le lleguen.

  - **Regresión**: el objetivo en este caso es predecir un valor numérico como por ejemplo el precio de una casa a partir de una serie de características o **features** (superficie, antigüedad, distrito…). También será necesario que cada caso incluya su precio.

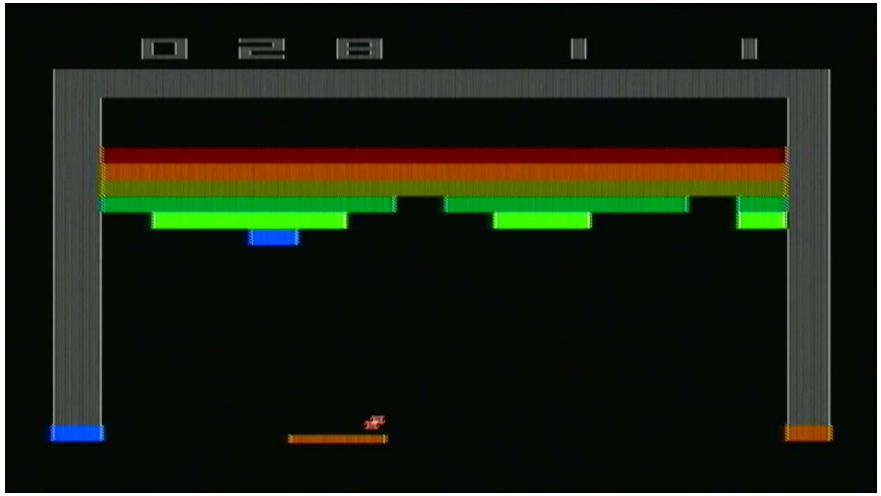


- El aprendizaje no supervisado se distingue porque los datos de entrenamiento que se le pasan al algoritmo **NO** están etiquetados, es decir, no se proporciona lo que se conoce como etiqueta, **label** o target. El sistema trata de **aprender sin maestro**.

- Posible situación:
    - Tenemos un blog que genera datos sobre los lectores del mismo: queremos generar un modelo de clustering que permita **discernir grupos de lectores con un perfil similar**. En ningún momento se le dice al modelo los grupos que esperamos encontrar: ha de hallar las **conexiones sin ayuda**. Podría detectar por ejemplo que el 40% de lectores son mujeres de entre 20 y 30 años que les encantan los animales y que leen el blog por las noches y el 20% son hombres mayores de 60 años aficionados a las películas de terror que leen el blog por las mañanas.



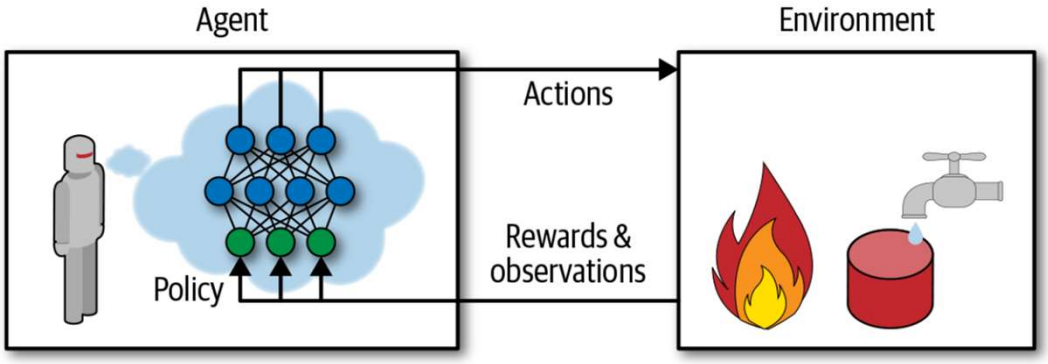
- Disciplina que lleva entre nosotros desde los años 50 del siglo pasado.
- Boom en 2013: DeepMind consigue que un sistema "aprenda" a jugar juegos de Atari (por ejemplo el Breakout) sin conocimiento previo de las reglas, utilizando píxeles como entrada.
- 2016 AlphaGo, 2019 AlphaStar
- Google compró DeepMind en 2014 por 500 millones de USD



- En Reinforcement Learning, existe un agente que se encuentra en un determinado entorno, dentro del cual realiza observaciones y ejecuta acciones. Como resultado de esas acciones recibe una recompensa (positiva o negativa). El objetivo es maximizar la recompensa acumulada a lo largo del tiempo.

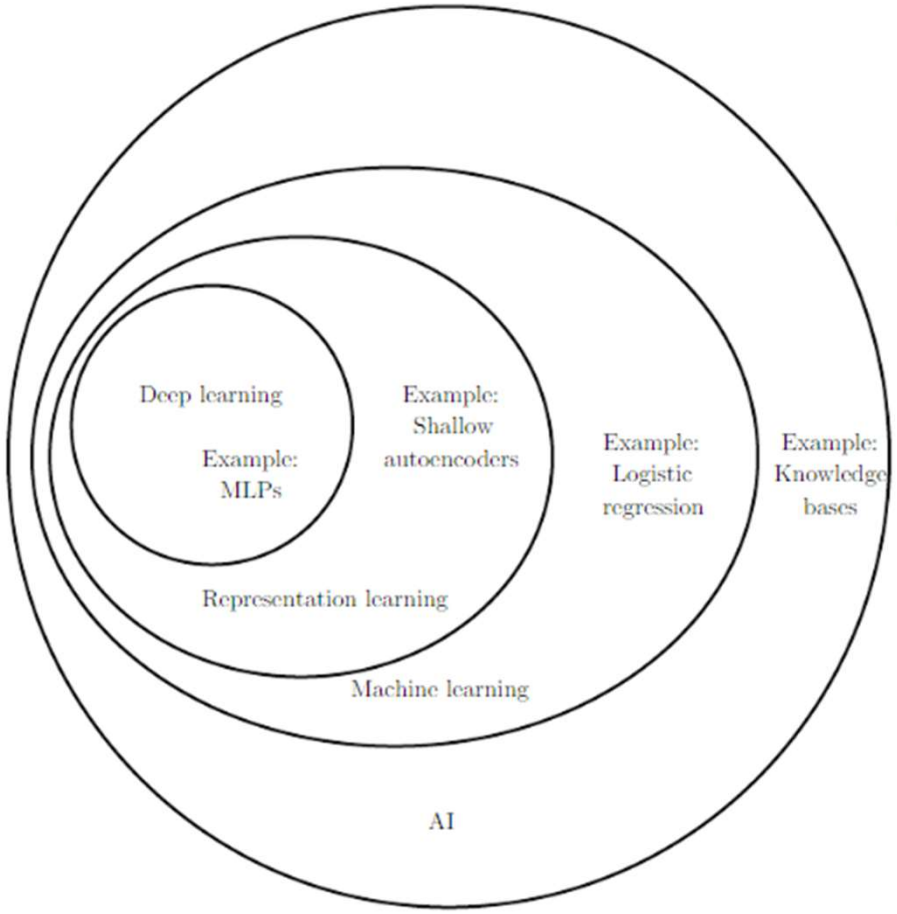
- Ejemplos:
  - Agente: programa encargado de controlar a un **robot**. Entorno, el mundo real. Observaciones: a través de los sensores del robot. Acciones: movimiento por parte del robot. Recompensa: positiva cuando se acerca al destino, negativa cuando da rodeos o se aleja.
  - Agente: programa que controla a **Pac-Man**. Entorno: simulador del juego. Observaciones: pantallazos del juego. Acciones: las del joystick. Recompensa: los puntos del juego.
  - Agente: un **termostato**. Entorno: nuestra casa. Observaciones: a través de sensores de temperatura. Acciones: modificar la temperatura para que sea la adecuada. Recompensa: negativa siempre que tengamos que tocar el termostato para configurar manualmente la temperatura.

- ¿Cómo se maximiza la recompensa? Encontrando la **policy** óptima.
- Una **policy** es cualquier algoritmo que determine como nuestro agente se comportará bajo cualquier circunstancia que pueda darse en nuestro entorno.
- DeepMind, planteó como **policy** un modelo basado en Deep Learning, que recibía como entrada las observaciones del entorno y devolvía como salida las acciones a tomar.



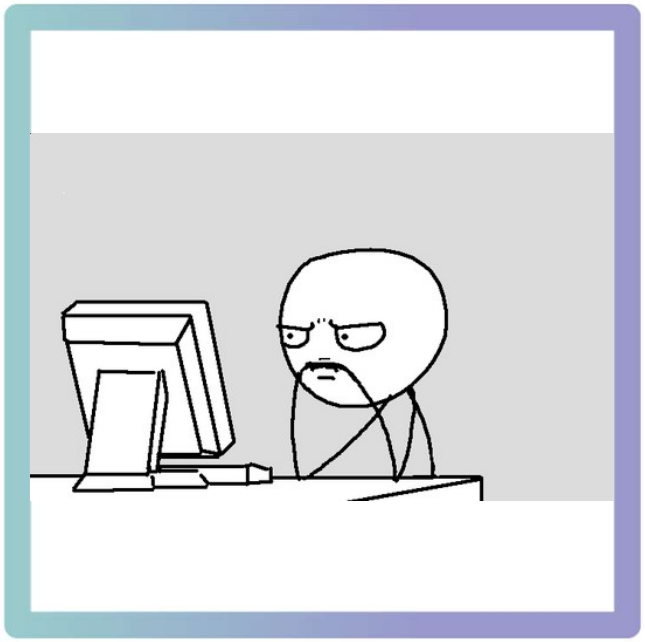
- OpenAI Gym, es una buena manera de empezar a trastear con esta disciplina.

APLICANDO MACHINE LEARNING



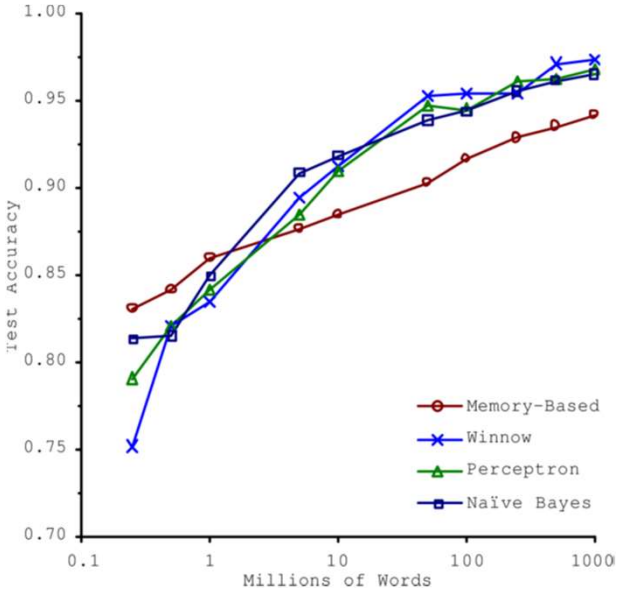
1. Knowledge bases (En**CyC**lopedia).
2. Machine Learning: Logistic Regression.
3. Representation Learning: Autoencoders.
4. Aplicación del modelo en la predicción de nuevos casos.

¿QUÉ PUEDE IR MAL?



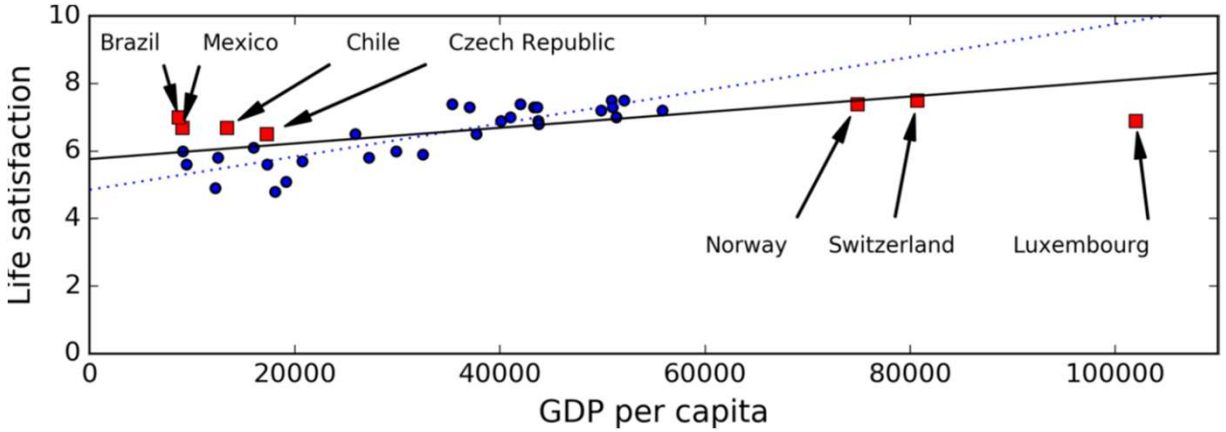
1. Problemas con los datos.
2. Problemas con algoritmo.

Incluso para aquellos problemas que puedan parecer más simples o acotados, un algoritmo de Machine Learning puede requerir miles de datos para comportarse decentemente. Problemas complejos como reconocimiento de voz o imágenes requieren millones de ejemplos.



The Unreasonable Effectiveness of Data Halevy, Norvig and Pereira
http://static.googleusercontent.com/media/research.google.com/fr//pubs/archive/35179.pdf
Scaling to Very Very Large Corpora for Natural Language Disambiguation Banko and Brill
http://www.aclweb.org/anthology/P01-1005

- Queremos un modelo que pueda ser aplicado a nuevos casos, por lo tanto necesitamos que generalice bien y para esto hacen falta datos representativos.



- Si tomamos una muestra demasiado pequeña, la probabilidad de que los datos no sean suficientemente representativos es mayor (**sampling noise**)

- Si la muestra es grande pero el método de muestreo no es adecuado, tendremos también un problema con la representatividad de los datos (**sampling bias**)

MÁS PROBLEMAS... | Datos no representativos...

- El científico de datos, dedica buena parte de su tiempo a la limpieza de datos y no por capricho; unos datos de entrenamiento llenos de errores, outliers y ruido reducirán la posibilidad de que nuestro modelo se comporte bien. Ejemplos de decisiones en la limpieza de datos,

    - **Outliers**, podría ser viable realizar ajustes sobre ellos manualmente o podría darse la circunstancia de que no fueran un problema.

    - **Missing values**, por ejemplo usuarios que no especificaron su edad. Se puede optar por ignorar el atributo edad, ignorar solo las instancias vacías o rellenarlas con algún estadístico…Incluso entrenar un modelo incluyendo esta feature y otro sin ella…

- Otro aspecto crucial: si a nuestro modelo le metemos basura, nos devolverá basura (garbage in, garbage out); por tanto es conveniente ser escrupuloso en lo que se conoce como Feature Engineering.

- El Feature Engineering tiene como máxima que el sistema aprenderá si se le pasan abundantes features relevantes y el menor número posible de features irrelevantes. Modalidades,

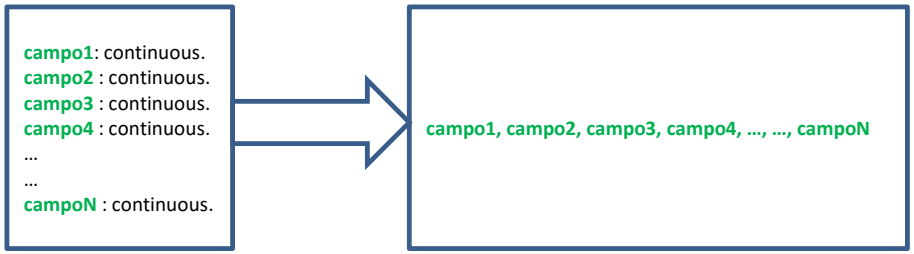
    - **Feature selection,** consiste en determinar cuales son las features más útiles para entrenar de entre todas las que tenemos.
    - **Feature extraction,** se trata de combinar varias features para obtener una nueva más relevante.

Manos a la obra

- Empecemos con el notebook *00_project_Flow.ipynb*

cal_housing.domain y cal_housing.data

- cal_housing.domain, esto es lo que vamos a hacer:

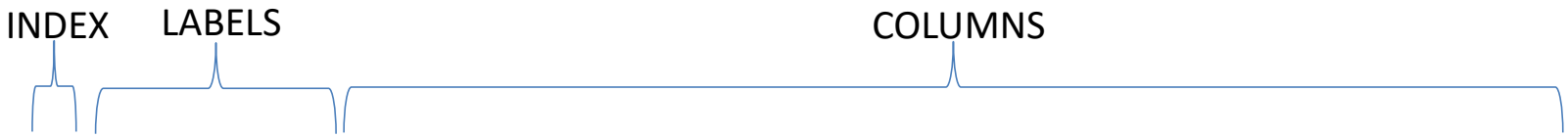


- cal_housing.data, no lo vamos a tocar:

```
-122.230000,37.880000,41.000000,880.000000,129.000000,322.000000,126.000000,8.325200,452600.000000-
122.220000,37.860000,21.000000,7099.000000,1106.000000,2401.000000,1138.000000,8.301400,358500.000000
-122.240000,37.850000,52.000000,1467.000000,190.000000,496.000000,177.000000,7.257400,352100.000000
.....
.....
```

- housing.csv (se concatena cal_housing.domain transformado con cal_housing.data),

```
campo1, campo2, campo3, campo4.....campoN
-122.230000,37.880000,41.000000,880.000000,129.000000,322.000000,126.000000,8.325200,452600.000000-
122.220000,37.860000,21.000000,7099.000000,1106.000000,2401.000000,1138.000000,8.301400,358500.00000
0-122.240000,37.850000,52.000000,1467.000000,190.000000,496.000000,177.000000,7.257400,352100.000000
.....
.....
.....
```

		feature1	feature2	feature3	feature4	feature5
0	07162534J	...	...	...	...	...
1	27361591K	...	...	...	...	...
2	12849299L	...	...	...	...	...
...	....	....	....	....	....	....

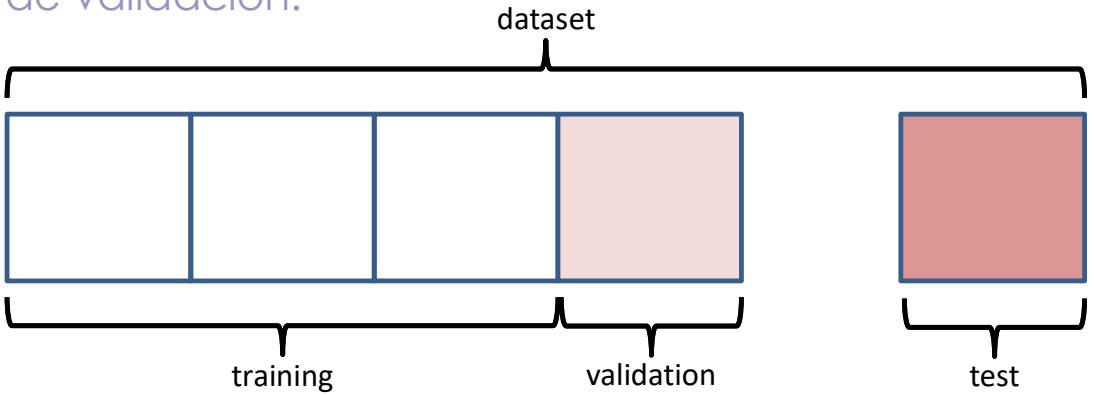
- Pandas proporciona dos métodos para acceder a los registros del dataframe mediante su índice o mediante su label:
  - *dataframe.iloc[...] a través del índice.*
  - *dataframe.loc[...] a través del label.*

dataframe.mask(...)

feature1	boolean mask	feature1 resultante
A	0 (False)	A
B	0 (False)	B
C	1 (True)	NaN
B	0 (False)	B

- El método mask, funciona como un IF ...: <algo> else: <otra cosa>:
  - *Por defecto <algo> es NaN, así que si el valor de la máscara es 1, el valor original se convertirá en NaN.*
  - *En el caso del else, si el valor de la máscara es 0, el valor original se devolverá intacto.*

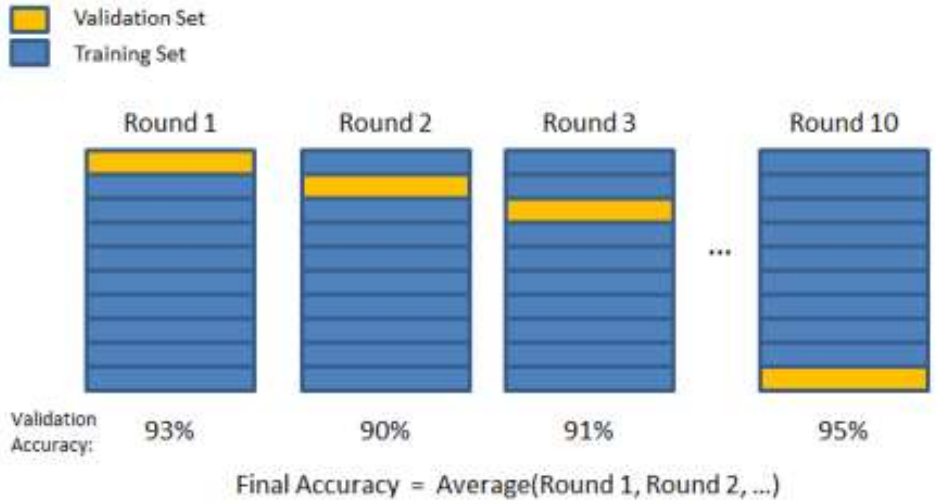
- El conjunto de training será la porción de dataset utilizada para entrenar los modelos.
- El conjunto de validación sería otra porción de datos que nos guardamos para probar los modelos que vayamos obteniendo a partir de los distintos hiperparámetros. El modelo que escojamos será el que lo haga mejor contra el conjunto de validación.



- Es importante recordar que los datos de test se deben reservar para el modelo escogido; es decir, no podemos utilizarlo para escoger nuestros modelos o sus hiperparámetros…para eso tenemos el conjunto de validación.

https://www.memecenter.com/fun/1198090/choose-wisely

- Un paso más allá de la estrategia de validación es la validación cruzada o Cross Validation. Su versión más extendida es la K-fold Cross Validation.
- Los datos de entrenamiento son divididos en **K subconjuntos** (OJO, seguimos manteniendo un conjunto de test).
    - Entrenaremos cada combinación de hiperparámetros K veces
        1. Se entrenará usando K-1 subconjuntos.
        2. Se evaluará contra el subconjunto restante.
        3. Se anota la puntuación y se "enjuaga" el modelo.
    - La puntuación final de cada modelo será la media de las puntuaciones obtenidas.

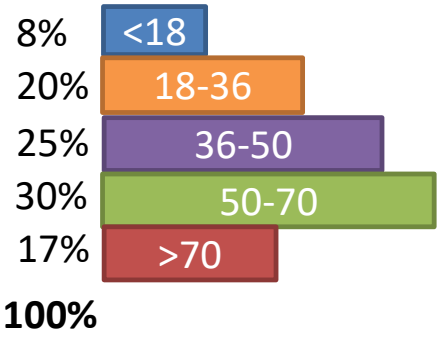


Cortesía de Neerav Basant

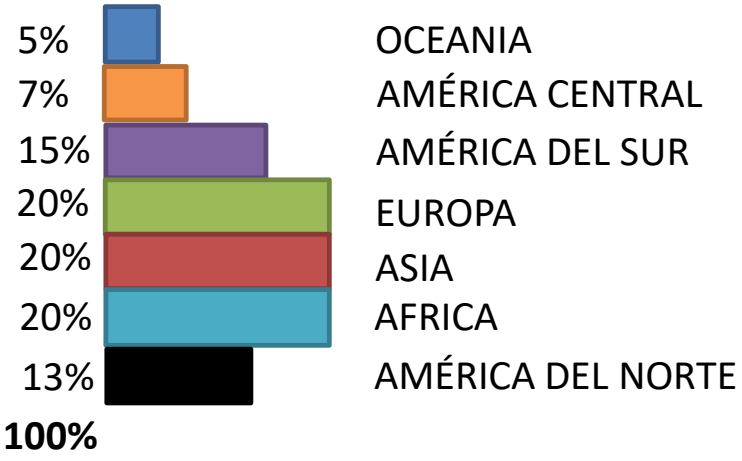
Stratified sampling

- Considerando posibles estratos en nuestra población (entendiendo como población el conjunto de registros que conforman nuestro dataset):

AGE STRATA



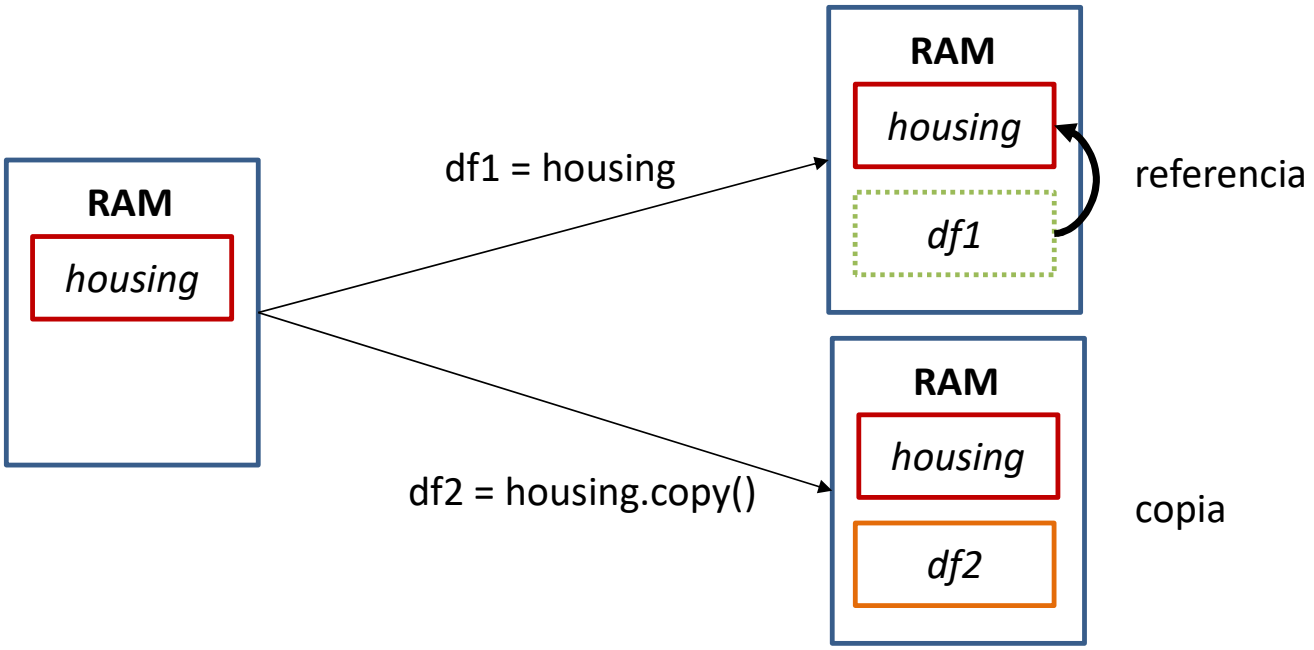
CONTINENT STRATA



LABEL STRATA

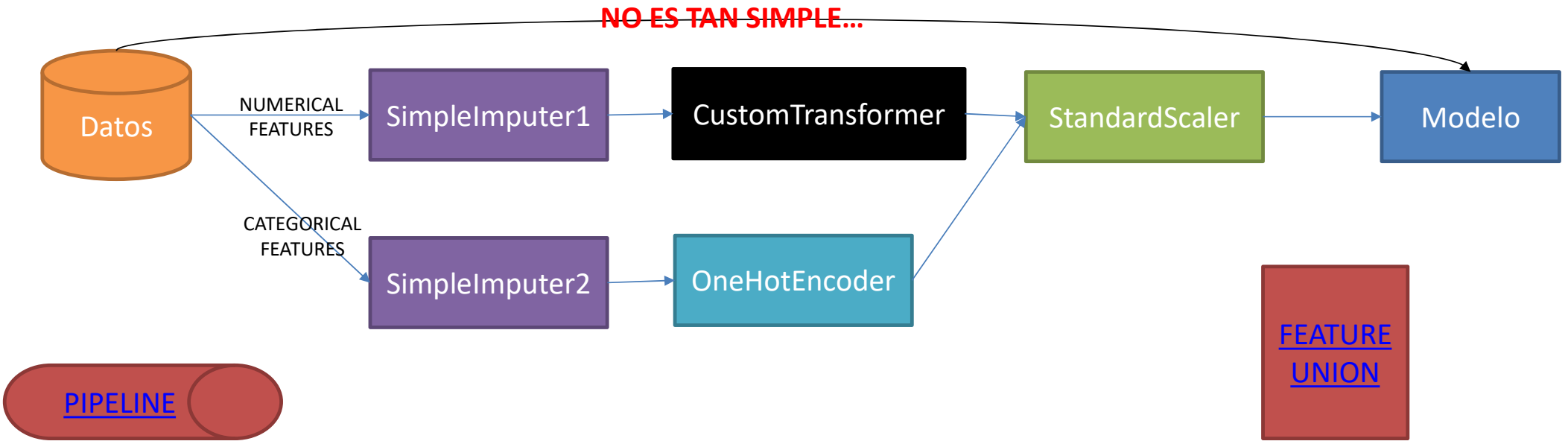
Copias y referencias

- A veces asignar el valor de un objeto Python a una variable mediante un simple igual (=) no tiene el efecto que nosotros pensamos.



Lego y Power Rangers

- A estas alturas ya sabemos que la cosa no es tan simple como pasarle unos datos a un modelo para que este busque patrones.



Categórica y nominal

- Supongamos que tenemos una feature llamada TIPO que puede tomar los valores A, B y C.

USUARIO	TIPO
1	A
2	C
3	C
4	B
5	A

USUARIO	is_A	is_B	is_C
1	1	0	0
2	0	0	1
3	0	0	1
4	0	1	0
5	1	0	0

ENLACES “DE INTERÉS”

- FAILS:
  - Google & Gorillas: https://www.theverge.com/2018/1/12/16882408/google-racist-gorillas-photo-recognition-algorithm-ai
  - AWS & Alexa: https://www.fastcompany.com/90163588/why-alexas-laughter-creeps-us-out
  - Microsoft & Tay: https://www.theverge.com/2016/3/24/11297050/tay-microsoft-chatbot-racist

- Skynet Today:
  - Elon Musk & Robotaxis: https://www.skynettoday.com/briefs/elon-musk-1mil-robotaxis

- Pipelines & Featureunions: http://zacstewart.com/2014/08/05/pipelines-of-featureunions-of-pipelines.html