Objetivos de aprendizaje de esta sesión

-Comprender la historia y evolución de la Inteligencia Artificial (específicamente Machine Learning)

-Entiende el concepto de Aprendizaje supervisado vs Aprendizaje no supervisado

-Conocer el Flujo de Trabajo en ML

-Identificar falacias en los Datos

-Conocer qué Transformaciones se pueden hacer sobre los Datos

-Conocer Scikit-Learn

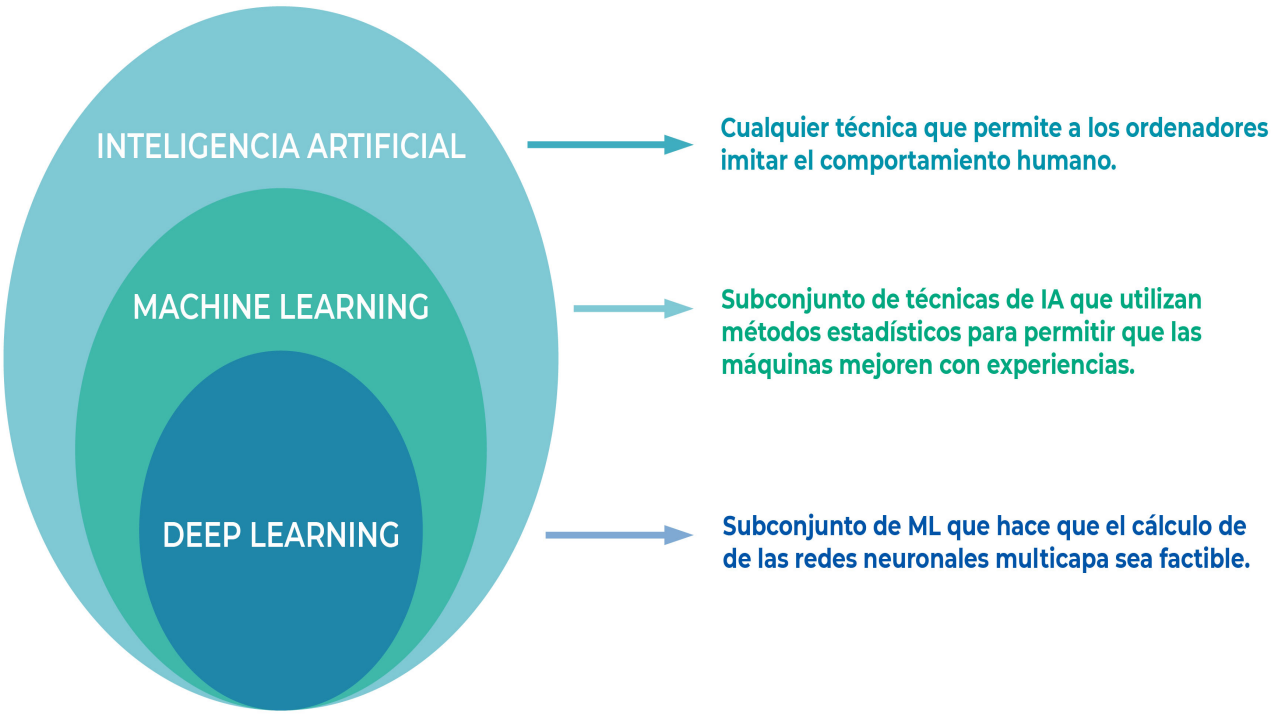
-Comprender las ventajas de Reescalar los Datos

-Comprender el concepto de Correlación

-Comprender el funcionamiento del algoritmo Regresión Lineal

Machine Learning

Introducción



En el transcurso de este módulo haremos un recorrido por cada una de las temáticas esenciales de Machine Learning (en adelante, "ML"). Veremos cuál es su alcance, características, metodología, aplicabilidad y casos de uso. Abordaremos, a su vez, los elementos centrales de campos como `series de tiempo`, `procesamiento de lenguaje natural` y `sistemas de recomendación`.

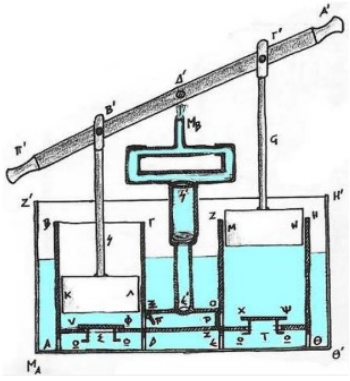
Finalmente, en la última clase repasaremos algunas nociones de `Deep Learning`.

Como podemos observar en la imagen supra, ML se inserta en una disciplina más general, macro, conocida como **Inteligencia Artificial**. Para profundizar un poco en estos conceptos, recomendamos acceder al video que se deja a continuación.

AI, ML & DL

Historia y evolución de la Inteligencia Artificial

- En la Grecia antigua, Aristóteles (384-322 a.C.) fue el primero en describir una parte del funcionamiento de la mente humana en un conjunto de reglas para obtener conclusiones racionales. Ctesibio de Alejandría (250 a.C.) construyó la primera máquina autocontrolada, un regulador del flujo de agua, racional pero sin razonamiento.

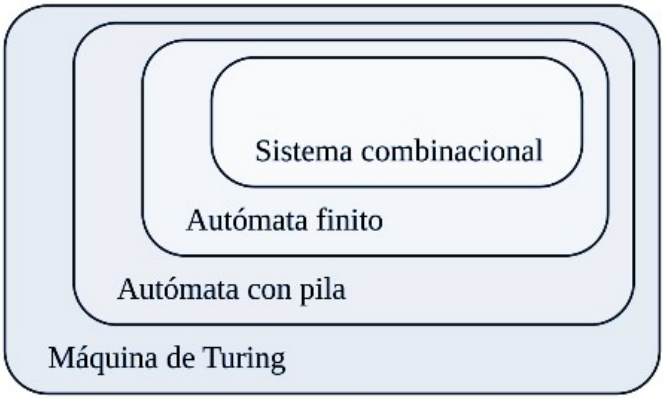


- En su obra Ars Magna (1274), Raimundo Lulio inventó un lenguaje formal basado en la lógica combinatoria para poder hablar de todo aquello relevante a la filosofía y la religión sin la barrera de las lenguas. Es el primer intento de crear una lengua universal e influenció a Leibniz, quien expandió el sistema lo denominó combinatoria.

- En su trabajo "El arte de la combinatoria", escrito en 1666, Leibniz justificó la importancia del simbolismo racional para la lógica y para las conclusiones heurísticas; Argumentó que el conocimiento se reduce a pruebas de afirmaciones, pero para encontrar pruebas es necesario mediante un cierto método.

- En 1854 George Boole publicó “Una investigación sobre las leyes del pensamiento”, donde desarrolló un sistema de reglas que le permitían expresar, manipular y simplificar problemas lógicos y filosóficos cuyos argumentos admiten los estados verdadero o falso por procedimientos matemáticos.

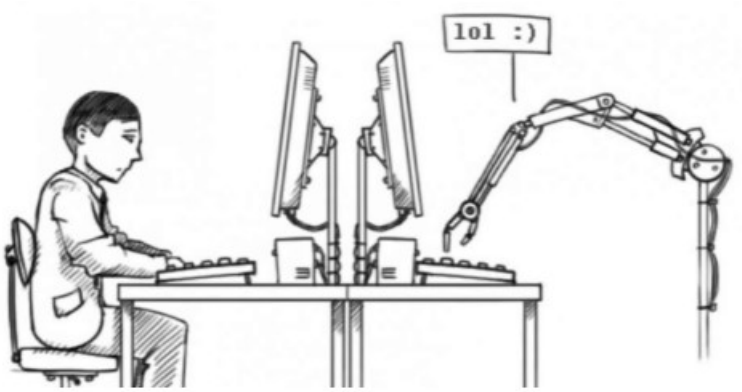
- En 1936, Alan Mathison Turing ideó un modelo formal de computador extremadamente sencillo, la Máquina de Turing, con el que es posible realizar cualquier cómputo que un computador digital sea capaz de realizar. Es el primer modelo teórico de un procesador.



- En 1943, Warren McCulloch y Walter Pitts presentaron su modelo de neuronas artificiales, el cual se considera el primer trabajo del campo, aún cuando todavía no existía el término.

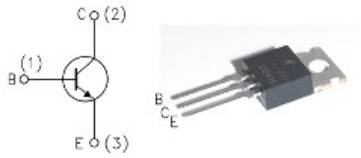
- En 1949, Donald Hebb fundamenta que la mayoría de las funciones de aprendizaje que pueden hallarse en una red neuronal. Su idea fue que el aprendizaje ocurría cuando ciertos cambios en una neurona eran activados.

- En 1950, Alan Turing crea la Prueba de Turing, que consiste en que si una persona, en una conversación con una máquina, no es capaz de distinguirlo de un humano, esta es aprobada como inteligente.

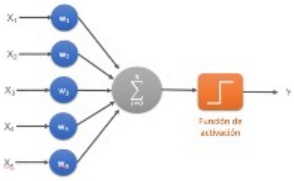


- En 1952, Arthur Samuel escribe el primer programa de ordenador capaz de aprender: un programa que jugaba a las Damas y mejoraba en cada partida.
- En 1956, John McCarthy, Marvin Minsky y Claude Shannon acuñaron el término Inteligencia Artificial durante una conferencia en Dartmouth.

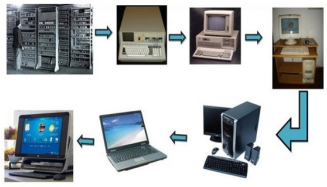
- A mediados de los años 50, con el invento de los transistores, apareció un gran avance con computadoras más poderosas y confiables, y menos costosas. Abriendo paso a la segunda generación de computadoras.



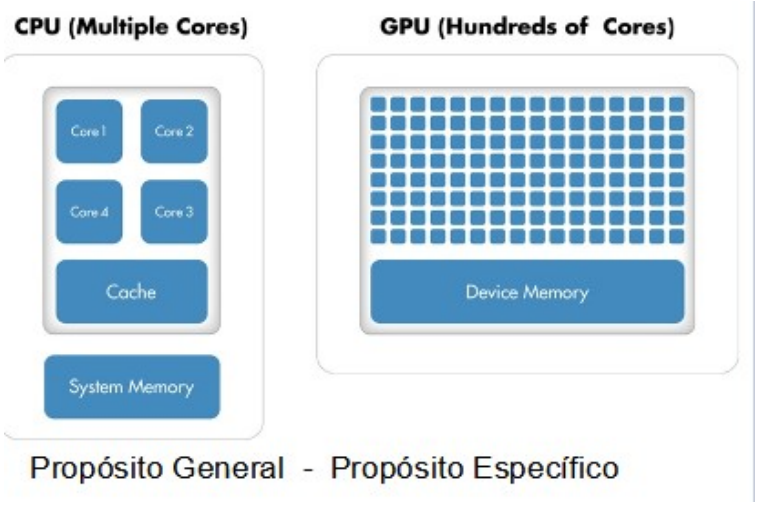
- En 1958 Frank Rosenblatt diseña el Perceptrón, la primera red neuronal artificial basado el modelo de neuronas de McCulloch y Pitts de 1943.



- A mediados de los años 60, aparecen los sistemas expertos, que predicen la probabilidad de una solución bajo un conjunto de condiciones.
- En 1967, la creación del algoritmo conocido como “vecinos más cercanos” permitió a las computadoras utilizar un reconocimiento de patrones muy básico. Incluso tuvo fines comerciales.
- En los años 1970, con John Henry Holland surgió una de las líneas más prometedoras de la inteligencia artificial: los algoritmos genéticos, son llamados así porque se inspiran en la evolución biológica y su base genético-molecular.
- En 1981, Gerald Dejong plantea el concepto «Aprendizaje Basado en Explicación» (EBL), donde la computadora analiza datos de entrenamiento y crea reglas generales que le permiten descartar datos menos importantes.
- En 1982, con la quinta generación de computadoras, el objetivo era el desarrollo de computadoras que utilizarían inteligencia artificial, mejorando hardware como software, sin obtener los resultados esperados: casos en los que es imposible llevar a cabo una paralelización, no se aprecia mejora alguna, o se pierde rendimiento.



- En 1985, Terry Sejnowski inventa NetTalk, un programa que aprende a pronunciar palabras de la misma manera que lo haría un niño.
- En 1986, McClelland y Rumelhart publican Parallel Distributed Processing (Redes Neuronales).
- En 1997, Gari Kaspárov, campeón mundial de ajedrez, pierde ante la computadora autónoma Deep Blue.
- En la década de los 90s, el Machine Learning ganó popularidad gracias a la intersección de la Informática y la Estadística que dio lugar a enfoques probabilísticos en la IA. Esto generó un gran cambió al campo del aprendizaje automático de las computadoras, ya que se trabajaría con más datos.
- En este periodo que se comenzó a utilizar esta tecnología en áreas comerciales para la Minería de Datos, software adaptable y aplicaciones web, aprendizaje de texto y aprendizaje de idiomas.
- La llegada del nuevo milenio trajo consigo una explosión en el uso del Machine Learning. Geoffrey Hinton en 2006 acuña el término "Deep Learning", con el que se explican nuevas arquitecturas de Redes Neuronales profundas que permiten a las computadoras "ver" y distinguir objetos y texto en imágenes y videos.
- También en 2006 se consigue facilitar los cálculos independientes necesarios para renderizar cada píxel en GPUs. Hasta entonces era impensable que los científicos usarán GPUs para su trabajo, pero a partir de ese momento lenguajes de alto nivel como C++ o Python se pueden utilizar para programar complejos cálculos y algoritmos permitiendo programar trabajos en paralelo y con gran cantidad de datos.



- En 2011 IBM desarrolló Watson, la computadora ganó una ronda de tres juegos seguidos de Jeopardy!.
- En 2016 Google DeepMind vence en el juego Go a un jugador profesional por 5 partidas a 1. El algoritmo realizó movimientos creativos que no se habían visto hasta el momento. El Go es considerado uno de los juegos de mesa más complejos.

- Hoy existen personas que al dialogar sin saberlo con un chatbot no se percatan de hablar con un programa, de modo tal que se cumple la prueba de Turing como cuando se formuló: “Existirá Inteligencia Artificial cuando no seamos capaces de distinguir entre un ser humano y un programa de computadora en una conversación a ciegas.”

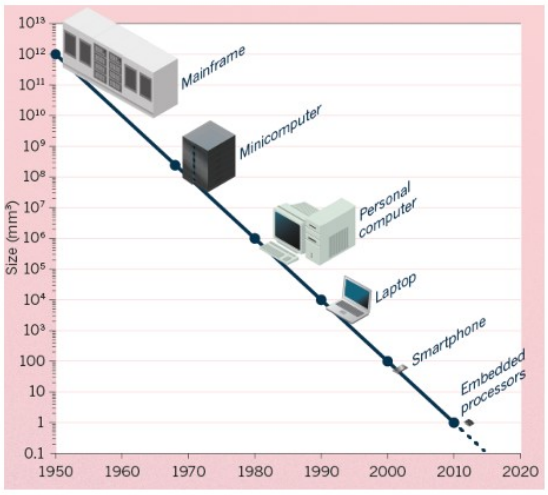
- Las redes neuronales son como una caja negra. Esto quiere decir que cuando la máquina da una solución a un problema, es muy complicado conocer cuáles son sus “razonamientos” para llegar a dicha solución.

Presente y Futuro de la IA

Inicialmente había dos obstáculos:

- La capacidad de cómputo: Superado por medio de la escalabilidad vertical, haciendo las computadoras más potentes siguiendo la Ley de Moore.

- Los datos: Superado por medio de la escalabilidad horizontal, haciendo que varios ordenadores computen como uno solo, por medio de tecnologías como Big Data.



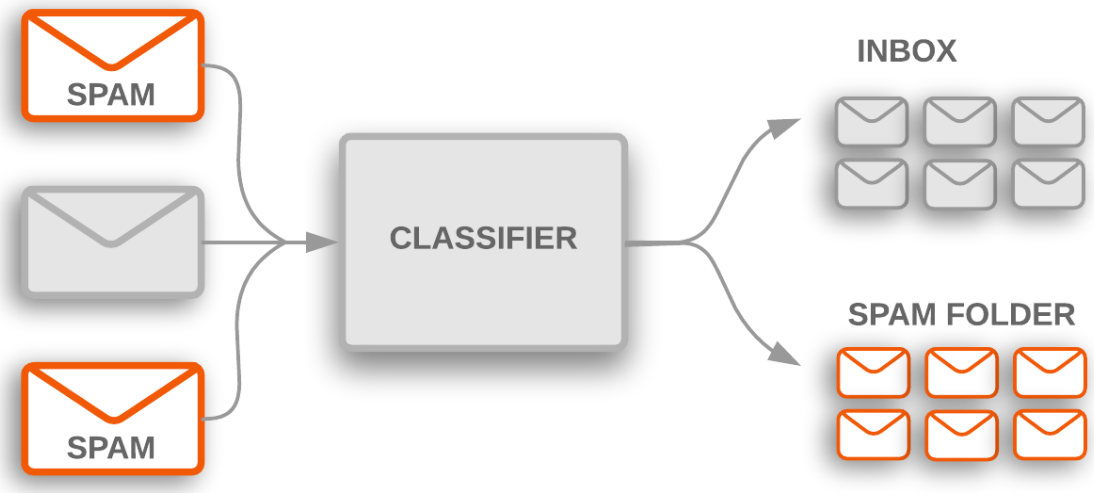
La pregunta clave que buscaremos responder en esta sección es: ¿cómo hacemos para que las computadores aprendan de los datos?

Para ello, debemos comenzar hablando de modelos matemáticos y cómo estos se relacionan con ML.

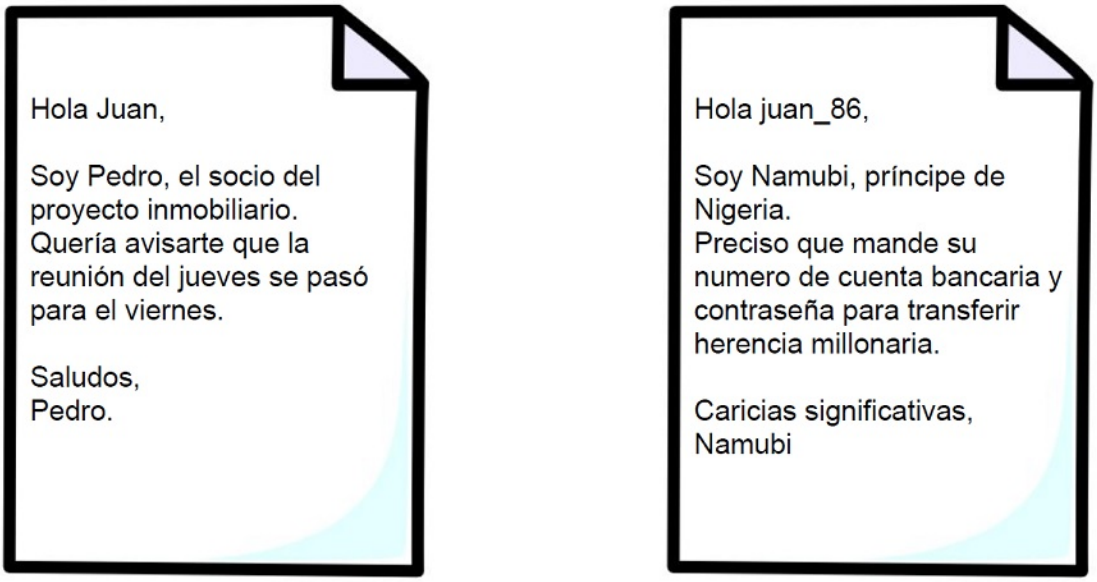
Los modelos matemáticos son descripciones de un sistema que usa conceptos y lenguaje matemático, que pueden ayudar a estudiar los efectos de diferentes componentes y hacer predicciones relacionadas a este comportamiento. Este proceso de agrupar información en una herramienta para explicar y predecir se llama ***modelado*** y se estructura alrededor de una idea: por qué ciertos eventos acontecieron, acontecen y acontecerán. En esta línea, una primera aproximación a un modelado podría armarse con las observaciones y los patrones que se desprenden de eventos pasados, ya que podríamos aseverar que en el futuro podrían repetirse esos componentes y eventos.

Ejemplo clásico

Para avanzar con los contenidos de ML, nos detendremos ahora en un ejemplo que demuestre su funcionalidad. Para este claso, empleamos un ejemplo de Correo Spam.



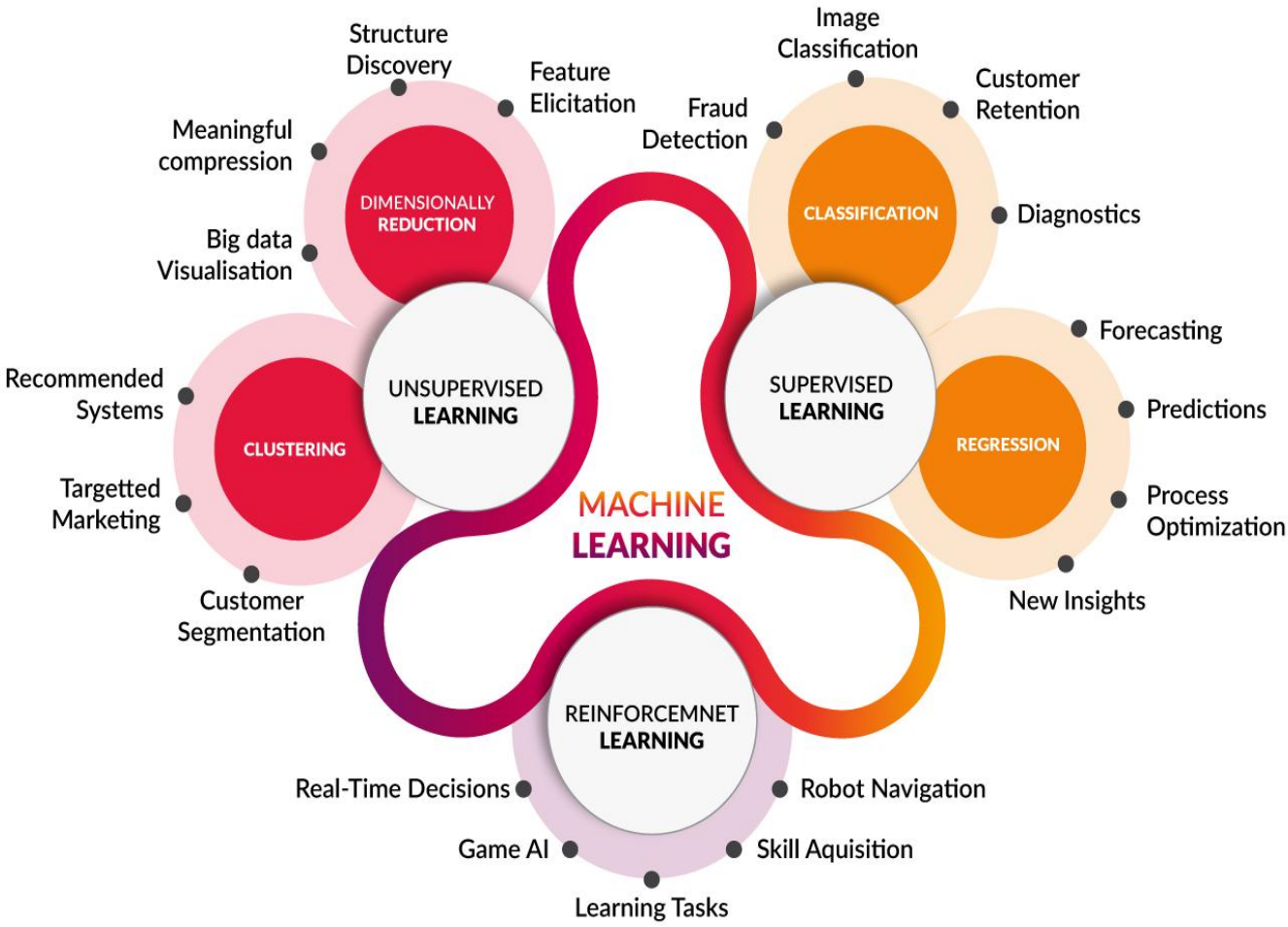
Habitualmente, recibimos correos en nuestra casilla. Algunos tendrán un contenido que es de nuestro interés, pero otros serán correos no deseados. Ahora bien, ¿cómo hace el sistema para distinguir un correo spam de otro que no lo es? En definitiva, ¿cómo lo detecta? En efecto, con un modelo de Machine Learning. El algoritmo clasificador, en este caso, separa los correos deseados de aquellos que detecta como spam.



Para nosotros sería bastante fácil poder detectar un correo spam. Pero para la máquina esto requiere de un nivel de abstracción bastante amplio. Debe entender el texto, relacionar las palabras, buscar patrones, etc. Además, debe buscar palabras clave. A priori, ya sabemos que si en un correo tenemos las palabras "príncipe", "cuenta bancaria" o "herencia" probablemente estemos frente a un correo spam.

Esquema de ML

En el siguiente esquema podremos observar las tres grandes ramas en las que se subdivide Machine Learning:



Respecto al alcance de este módulo, abordaremos **APRENDIZAJE SUPERVISADO** y **APRENDIZAJE NO SUPERVISADO**.

Aprendizaje supervisado

> $f(X) = y$

Buscamos un modelo ***f*** que permita determinar la salida ***y*** a partir de la entrada ***X***.

En esta función, ***X*** son los atributos -generalmente se denota con mayúscula porque incluye más de una variable- e ***y*** es la etiqueta.

El aprendizaje supervisado permite modelar la relación entre las características medidas de los datos y alguna etiqueta asociada con ellos. Es decir, podremos predecir ***y*** para nuevos datos ***X*** de los cuales no conozcamos la salida.

De acuerdo al tipo de etiquetas que asociamos a los datos, el modelo puede realizar dos tipos de tareas:

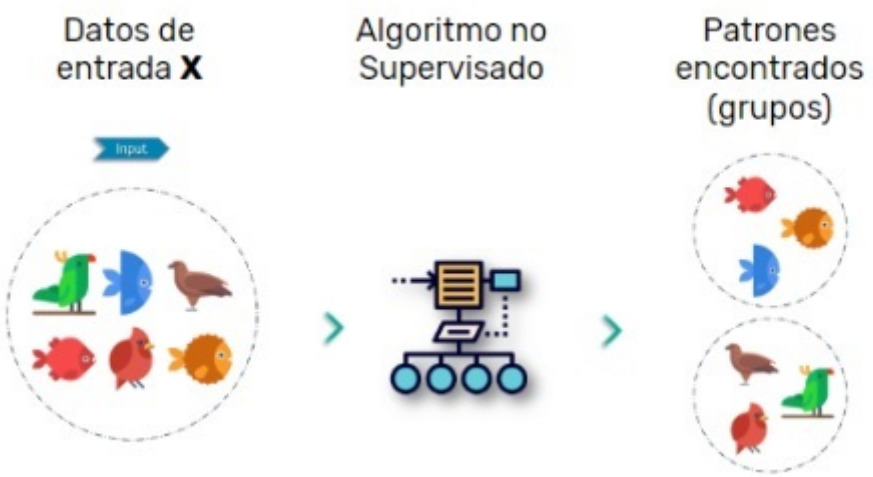
- `Clasificación:` las etiquetas son categorías. Ejemplo: enfermo/sano, gato/perro/pájaro, spam/no spam.
- `Regresión:` la variable de salida es un valor numérico. Ejemplo: precio, cantidad, temperatura.

Aprendizaje no supervisado

En este caso, se deja que el conjunto de datos hable por sí mismo. Este modelo tiene datos de entrada, pero no se busca una salida en particular. Implicar modelar las características de un conjunto de datos sin referencia a ninguna etiqueta.

La función de este tipo de algoritmos es encontrar patrones de similaridad. Por eso, incluyen tareas como `clustering` (agrupación) y `reducción de dimensionalidad`. En este último, el algoritmo busca representaciones más concisas de los datos. En clustering, busca identificar distintos grupos de datos.

Ejemplo de clustering:



Enlace Relacionado:

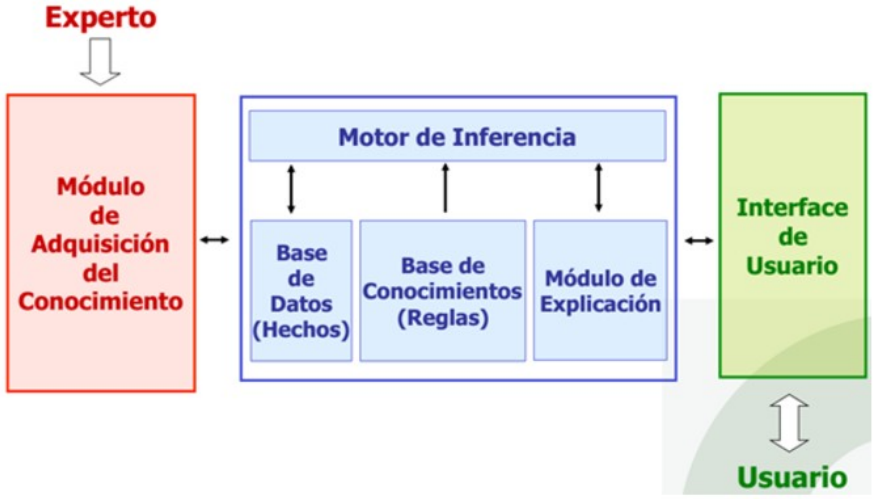
- Roca, Genís: "La sociedad digital", TedxTalks

Resolver problemas diversos de forma automática y simulando el comportamiento humano es una idea que existe desde la antigüedad. Sin embargo, no ha sido hasta mediados del siglo pasado que el avance de las ciencias de la computación permitió plantearse la posibilidad de que las computadoras pudieran dar soluciones "inteligentes" a esos problemas.

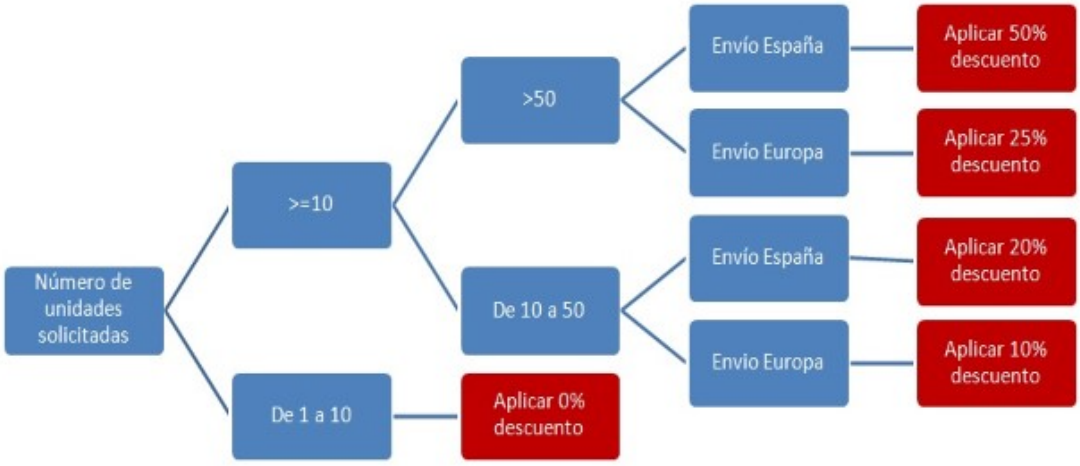
Primeros algoritmos de Machine Learning

- Sistemas Expertos: El experto humano ingresa cada experiencia con su regla de resolución. Esto se traduce en una base de conocimientos con reglas de inferencia que permite luego, a un usuario no

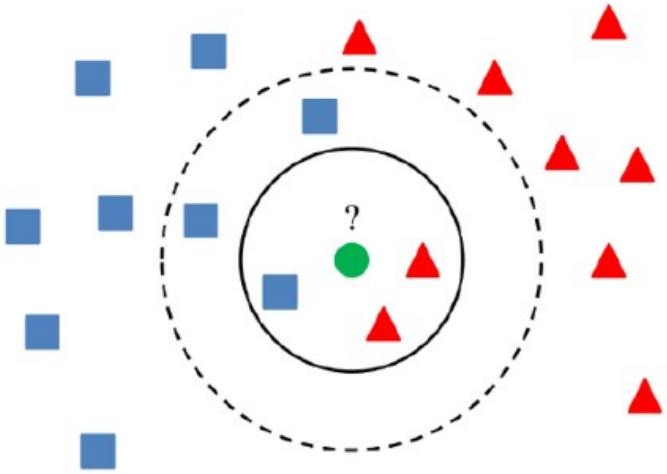
experto, ante la misma problemática (misma experiencia) encontrar una regla de resolución adecuada.



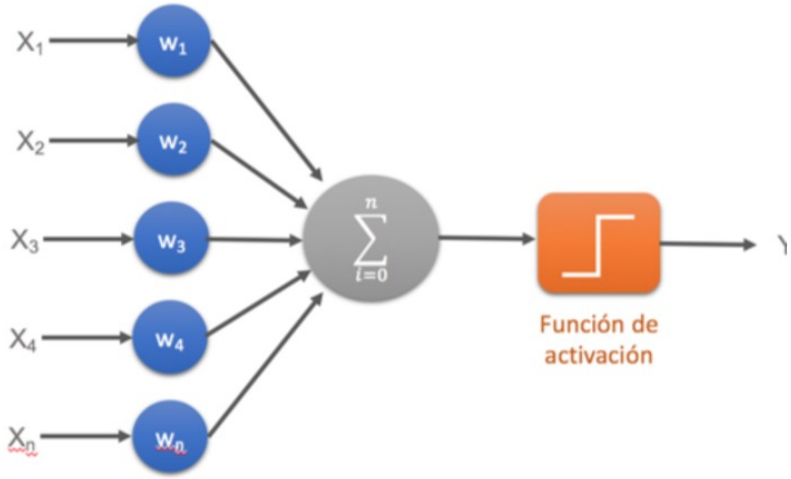
- Árbol de Decisión: Cada nodo puede ser una pregunta a los datos, o una resolución.



- Vecinos más cercanos ó K-Vecinos: Una nueva instancia de categoriza según sea la categoría de los k vecinos más cercanos



- Perceptrón simple: Basado en el modelo de redes neuronales artificiales, consta de n entradas X, que se ponderan con su respectivo valor W, se sumarizan y, por último, esta salida pasa por una función de activación para dar un resultado final.



Flujo de Trabajo

1. Definición: Se definen las preguntas que queremos responder. ¿Qué datos necesitamos para responder esas preguntas?
2. Investigación: Se obtienen los datos, se "limpian" y se procede a explorarlos.
3. Análisis: Los datos obtenidos se analizan con modelos (estadísticos, Machine Learning, etc.). Interpretamos los resultados y transformamos datos en información.
4. Presentación: Presentamos los resultados obtenidos y las conclusiones a las que llegamos. La información se transforma en Conocimiento. Puesta en producción.

Exploración de los datos

Los datos con los que vamos a estar trabajando, son en definitiva la fuente del conocimiento necesario que debemos adquirir para poder resolver las preguntas que nos hacemos, entonces, es preciso conocer todas sus características, algunas de ellas son:

- Variabilidad.
- Estadística.
- Distribución.
- Rangos.

Falencias en los datos

Como primera medida, antes de comenzar a realizar las tareas de análisis, vamos a encontrarnos con ciertas cuestiones que hacen a la calidad y fiabilidad del dato, y debemos resolverlas, entre ellas:

- Faltantes: ¿Qué hacer?
- Rangos de datos numéricos.
- Normalización.
- Errores: Su tratamiento.

Transformacion de Datos

Es el proceso que más tiempo lleva en un flujo de Data Science y resulta muy importante no perder el objetivo de por qué lo hacemos. Por un lado, los modelos de Machine Learning que usemos, que van a "aprender" de nuestros datos, sólo entienden de números. Esto quiere decir que, si en un dataset hay una columna de género con dos valores, 'femenino' y 'masculino', tendremos que llevarlo a una representación que los modelos entiendan. En este caso, asignamos un '0' y un '1' a cada uno de los valores y problema resuelto. En el fondo, hay una hipótesis implícita: el género es un campo muy útil para hacer la tarea de predicción o clasificación que requerimos, por lo que queremos mantener esta información y hacerla comprensible para nuestros modelos. La pregunta que queremos responder nos va a indicar cómo tenemos que trabajar con nuestro dataset.

Tratamiento sobre Variables Cualitativas Ordinales

Sus posibles valores son categorías pero sí hay una relación de orden. A pesar de que pueden ser números, ¡no se deben sumar!

Ejemplos:

- Tamaño de una prenda de ropa: XS, S, M, L, XL
- Tipo de Nafta por octanaje: 95, 98, más de 98
- Rangos etarios: bebé, niño/a, adolescente, adulto/a, anciano/a

Podemos hacer una asignación a número enteros manteniendo el orden:
S→0
M→1
L→2

Este es uno de los tipos de encoding más comunes que vamos a tener que hacer. En el ejemplo, queremos llevar al género, los valores male y female, a 0 y 1. Lo importante es no perder cuál es cuál. Esto, en Pandas, lo podemos hacer con la función map(). Este tipo de encoding se denomina Label_enconding.

```
diccionario = {'male': 0, 'female': 1}

df['Sex_Map'] = df.Sex.map(diccionario)
df.head()
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked	Age_labels	Sex_Map
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S	adulto	0
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C	adulto	1
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S	adulto	1
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S	adulto	1
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S	adulto	0

Tratamiento sobre Variables Cualitativas Nominales

Sus posibles valores pertenecen a una o varias categorías. Las categorías no siguen una relación de orden. Ninguna es mayor o menor que otra.

Ejemplos:

- Nacionalidad
- Tipo de Vino

- Especies de flores
- Color de auto

Se llevan a variables dummies con One-Hot Encoding. La variable dummie será entonces aquella que tome valores 0 o 1, en función de la presencia o no de un atributo. Puede hacer que nuestro dataset crezca mucho.

Obs.	Ciudad
1	Rosario
2	Buenos Aires
3	Rosario
4	Mar del Plata
5	Córdoba

Obs.	D_BA	D_C	D_R
1	0	0	1
2	1	0	0
3	0	0	1
4	0	0	0
5	0	1	0

Una variable dummy toma como valor 0 o 1 para indicar la presencia o ausencia de algún atributo categórico. La función get_dummies() hace automáticamente esto en un dataframe sobre las columnas indicadas.

```
df = pd.concat([df, pd.get_dummies(df['Sex'])], axis=1)
df.head()
```

PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked	Age_labels	Sex_Map	female	male
1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S	adulto	0	0	1
2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C	adulto	1	1	0
3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S	adulto	1	1	0
4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S	adulto	1	1	0

Tratamiento sobre Variables Cuantitativas

Son aquellas variables que se miden o se cuentan. Pueden ser discretas o continuas. Hay una relación de orden entre ellas. Se puede aplicar funciones de agregación.

Ejemplos:

- Edad, Altura y Peso.
- Puntaje, precio de un vino.
- Valor de un pasaje.

En general, ya vienen en un formato “cómodo” para trabajar, pero a veces queremos agruparlas según grupos o rangos, por ej.: agrupar edades en rangos (bebés, niños, adolescentes, adultos, ancianos), esto se denomina Discretización y Binning.

Enlace recomendado:

[All about Categorical Variable Encoding] (https://towardsdatascience.com/all-about-categorical-variable-encoding-305f3361fd02)

Variables Numéricas

En general, las variables numéricas podrían ser usadas sin demasiado preprocesamiento, ya que ya están en un formato que los modelos entienden. Sin embargo, muchas veces no es ese el caso.

Discretización y binning: La función cut() transforma la columna Age en rangos (bins) convirtiendo así el dato de edad en una variable discreta.

```
bins = [0,3,12,18,60,100]
labels = ['bebe', 'niño', 'adolescente', 'adulto', 'anciano']

cats = pd.cut(df.Age, bins, labels = labels)
df['Age_labels'] = cats
df.head()
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked	Age_labels
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S	adulto
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C	adulto
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S	adulto
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S	adulto
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S	adulto

Reescalar es útil sobre todo cuando queremos llevar los datos a una distribución de probabilidad con la que podamos trabajar mejor, como en el caso que vimos de llevar el dato a su logaritmo base 10.

Scikit-Learn

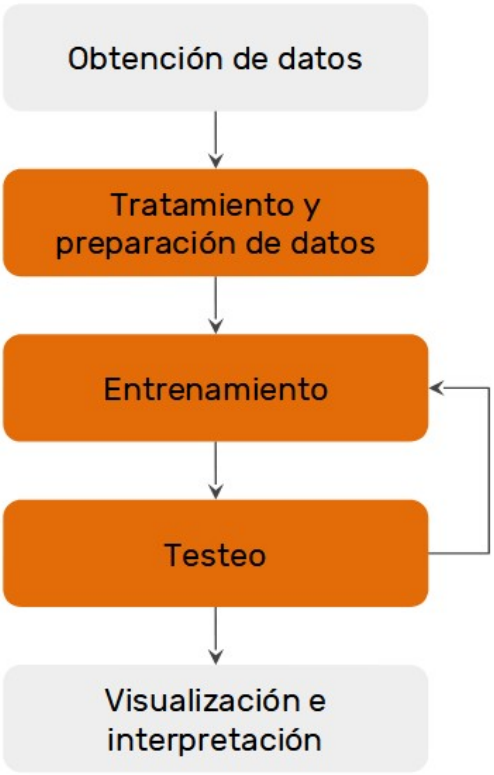
Scikit-Learn es la librería base para Machine Learning en Python.

Se puede usar para

- Procesamiento de los datos
- Modelos de Clasificación y Regresión
- Métricas de Evaluación de algoritmos

Adecuado para seguir un Pipeline: Es un objeto en el que es posible encapsular todo el proceso de los datos.

- [Documentación] (https://scikit-learn.org/stable/index.html)



Vamos a encontrar que Scikit-Learn trabaja con Clases e implementa de manera uniforme los atributos y métodos de sus objetos:

- Estimadores: Todos tienen el método fit()
- Predictores: Todos tienen el método predict()
- Transformadores: Todos tienen el método transform()
- Modelos: Todos tienen el método score()

En general, los pasos a seguir con una biblioteca de Scikit-Learn serán:

1. Crear el objeto con los parámetros correspondientes. objeto = Clase(parametros)
2. Implementar el método fit() que aprende de los datos. objeto.fit(Datos)
3. Implementar el método transform() que transforma los datos. objeto.transform(Datos)

Las siguientes clases son las herramientas disponibles para procesar datos:

- SimpleImputer: Rellena valores faltantes.

```
from sklearn.impute import SimpleImputer
imp = SimpleImputer(strategy='mean')
```

```
clics = usuarios_df.clics_por_hora.values
imp.fit(clics.reshape(-1,1))
print(imp.statistics_)
```

```
[182.66666667]
```

```
clics_imputed = imp.transform(clics.reshape(-1,1))
```

```
usuarios_df['clics_por_hora_imputed'] = clics_imputed
usuarios_df
```

	edad	clics_por_hora	explorador	clics_por_hora_imputed
0	32	156.0	Chrome	156.000000
1	20	NaN	Opera	182.666667
2	41	210.0	Firefox	210.000000
3	20	210.0	Chrome	210.000000
4	35	NaN	Explorer	182.666667
5	42	130.0	Firefox	130.000000
6	24	170.0	Chrome	170.000000
7	28	220.0	Chrome	220.000000

- OneHotEncoder: Pasa de variables categóricas a dummies. Notar que con N instancias, son necesarias solo N-1 nuevas columnas.

```
from sklearn.preprocessing import OneHotEncoder
onehot_encoder = OneHotEncoder(sparse = False)
```

```
exploradores = usuarios_df.explorador.values.reshape(-1,1)
```

```
onehot_encoder.fit(exploradores)
```

```
OneHotEncoder(categories='auto', drop=None, dtype=<class 'numpy.float64'>,
              handle_unknown='error', sparse=False)
```

```
onehot_encoder.categories_
```

```
[array(['Chrome', 'Explorer', 'Firefox', 'Opera'], dtype=object)]
```

```
explorador_encoded = onehot_encoder.transform(exploradores)
print(explorador_encoded)
```

```
[[1. 0. 0. 0.]
 [0. 0. 0. 1.]
 [0. 0. 1. 0.]
 [1. 0. 0. 0.]
 [0. 1. 0. 0.]
 [0. 0. 1. 0.]
 [1. 0. 0. 0.]
 [1. 0. 0. 0.]]
```

```
usuarios_df['chrome_encoded'] = explorador_encoded[:,0]
usuarios_df['explorer_encoded'] = explorador_encoded[:,1]
usuarios_df['firefox_encoded'] = explorador_encoded[:,2]
usuarios_df['opera_encoded'] = explorador_encoded[:,3]
usuarios_df
```

	edad	clics_por_hora	explorador	clics_por_hora_imputed	chrome_encoded	explorer_encoded	firefox_encoded	opera_encoded
0	32	156.0	Chrome	156.000000	1.0	0.0	0.0	0.0
1	20	NaN	Opera	182.666667	0.0	0.0	0.0	1.0
2	41	210.0	Firefox	210.000000	0.0	0.0	1.0	0.0
3	20	210.0	Chrome	210.000000	1.0	0.0	0.0	0.0
4	35	NaN	Explorer	182.666667	0.0	1.0	0.0	0.0
5	42	130.0	Firefox	130.000000	0.0	0.0	1.0	0.0
6	24	170.0	Chrome	170.000000	1.0	0.0	0.0	0.0
7	28	220.0	Chrome	220.000000	1.0	0.0	0.0	0.0

- LabelEncoder: Pasa variables categóricas a valores numéricos.
- KBinsDiscretizer: Para discretización y binning, la principal diferencia con Pandas es que Scikit-Learn decide los límites de los bines de acuerdo a una estrategia que le pasemos de parámetro.
- SelectKBest: Selecciona atributos del dataset en base a diferentes criterios de evaluación. Puede servir como respaldo o referencia del análisis que se está realizando sobre los datos.

```
from sklearn import preprocessing
le = preprocessing.LabelEncoder()
```

```
le.fit(usuarios_df['explorador'])
```

```
LabelEncoder()
```

```
usuarios_df['explorador_le'] = le.transform(usuarios_df['explorador'])
```

```
usuarios_df
```

	edad	clics_por_hora	explorador	clics_por_hora_imputed	explorador_le
0	32	156.0	Chrome	156.000000	0
1	20	NaN	Opera	182.666667	3
2	41	210.0	Firefox	210.000000	2
3	20	210.0	Chrome	210.000000	0
4	35	NaN	Explorer	182.666667	1
5	42	130.0	Firefox	130.000000	2
6	24	170.0	Chrome	170.000000	0
7	28	220.0	Chrome	220.000000	0

Reescalar los datos

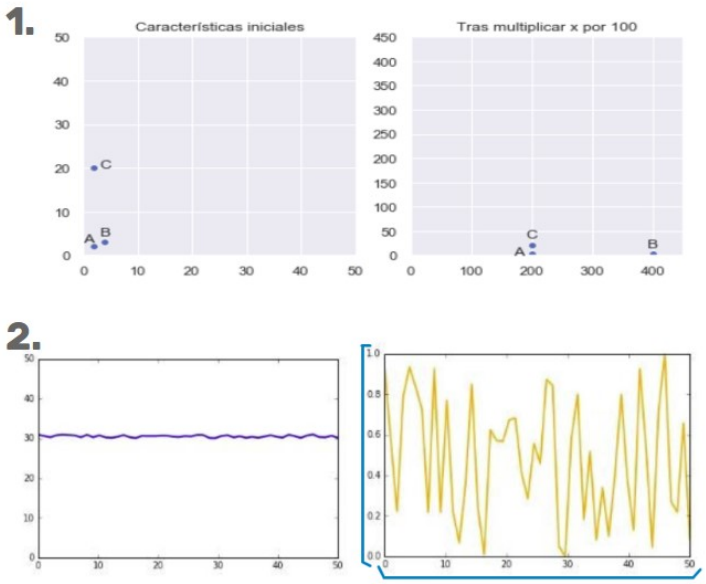
Muchos algoritmos funcionan mejor normalizando sus variables de entrada. Lo que en este caso significa comprimir o extender los valores de la variable para que estén en un rango definido. Sin embargo, una mala aplicación de la normalización o una elección descuidada del método de normalización puede arruinar los datos y, con ello, el análisis.

- MinMax Scaler:

Las entradas se normalizan entre dos límites definidos:

$$X_{normalized} = \frac{X - X_{min}}{X_{max} - X_{min}}$$

Tener en cuenta que si se reescala un atributo, quizás sea conveniente reescalar otro, debido a que estamos rompiendo la proporcionalidad de los datos:



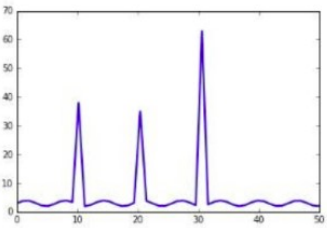
En 1 originalmente, A estaba más cerca de B, al multiplicar por 100, quedó más cerca de C. En 2 el ruido de la señal se hizo más notorio.

- Standard Scaler:

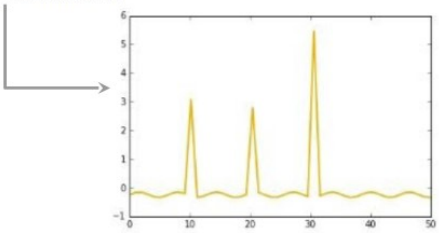
A cada dato se le resta la media de la variable y se le divide por la desviación típica:

$$X_{normalized} = \frac{X - X_{mean}}{X_{stddev}}$$

Si bien puede resultar conveniente en datos que no tienen distribución de probabilidad Gaussiana o Normal debido a que se puede trabajar mejor bajo ese esquema, tanto la media como la desviación típica son muy sensibles a outliers.



Ahora hay valores negativos, cuando antes no. Y los valores pico y valle han quedado muy atenuados debido a las anomalías.



Estadística - Correlación entre variables

Eventualmente vamos a querer conocer si existe una relación entre dos o más variables de un dataset, entendiendo por relación a una "variación conjunta". Si este es el caso, podríamos ver que si una de las variables aumenta o disminuye su valor, que la otra también lo hace. La covarianza es una medida que intenta cuantificar esa relación.

$$Cov(X,Y) = \frac{\sum_1^n (x_i - \bar{x})(y_i - \bar{y})}{n}$$

Con la covarianza también podemos determinar el coeficiente de relación o la recta de regresión, pero tiene el inconveniente de depender de la escala de los datos, motivo por el cual definimos la correlación, que es la covarianza dividida la desviación estándar de cada variable aleatoria obteniendo un valor que va de -1 a 1. Donde:

$$r_{xy} = \frac{s_{xy}}{s_x s_y}$$

- S_xy es la covarianza entre x e y.
- S_x y S_y son la desviación estándar de x e y respectivamente.
- r_xy es el coeficiente de correlación.

Este coeficiente se denomina correlación lineal o de Pearson, y es una cantidad adimensional. Por lo tanto, si cambiamos la unidad de x e y, el valor del coeficiente seguirá siendo el mismo.

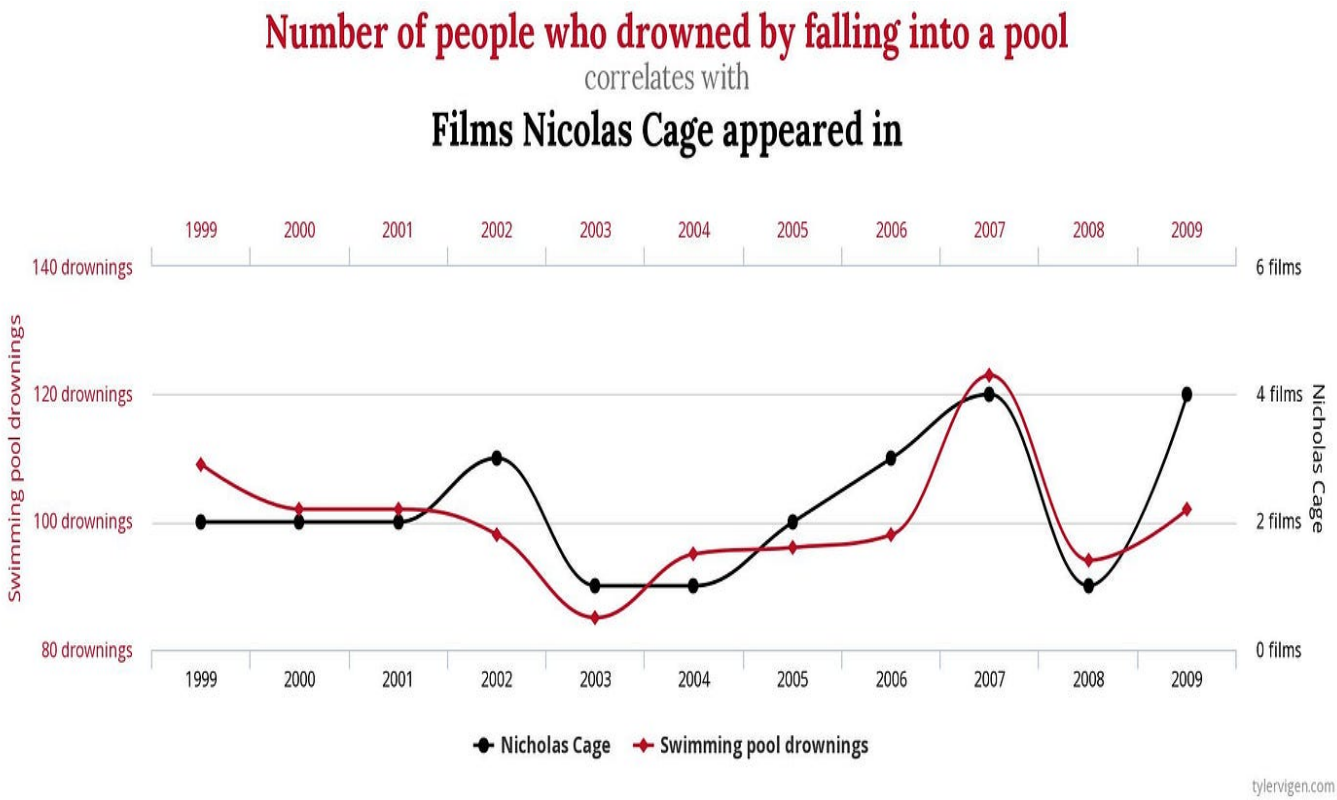
- Correlación no implica causalidad.
- La correlación de Pearson es muy útil para encontrar correlaciones lineales.
- Si la relación entre las variables NO es lineal, existen otras correlaciones que pueden ser útiles: Spearman y Kendall.
- Coeficiente Negativo significa que son inversamente proporcionales entre sí con el valor del factor de coeficiente de correlación.
- Coeficiente Positivo significa que son directamente proporcionales entre sí, la media varía en la misma dirección con el factor del valor del coeficiente de correlación.

- Si el coeficiente de correlación es 0, significa que no existe una relación lineal entre las variables, sin embargo, podría existir otra relación funcional.

- Si no hay ninguna relación entre dos variables, entonces el coeficiente de correlación será ciertamente 0; sin embargo, si es 0, solo podemos decir que no existe una relación lineal, pero podría existir otra relación funcional:

Para profundizar sobre los tipos de correlación `Pearson`, `Spearman` y `Kendall`, sugerimos este artículo.

Cerramos esta clase con la siguiente proposición que representa un escenario al que se enfrentarán en múltiples ocasiones y resulta menester identificar: ***correlación no implica causalidad***.



Práctica

La estructura de este módulo, para cada una de sus clases, se compone de la siguiente manera:

1. Clase teórica

2. Clase práctica

La primera parte se aborda en los respectivos README. La segunda parte, a través de notebooks donde realizamos prácticas guiadas para bajar a tierra algunos de los conceptos vistos en la teoría.

La práctica de hoy es:

- Práctica_01: regresión lineal

Homework

- Práctica_Adicional_Regresión.ipynb